HAROLD J. MCELHINNY (CA SBN 66781)
HMcElhinny@mofo.com
MICHAEL A. JACOBS (CA SBN 111664)
MJacobs@mofo.com
JENNIFER LEE TAYLOR (CA SBN 161368)
JTaylor@mofo.com
JASON R. BARTLETT (CA SBN 214530)
JasonBartlett@mofo.com
MORRISON & FOERSTER LLP
425 Market Street
San Francisco, California 94105-2482
Telephone: 415.268.7000
Facsimile: 415.268.7522

Attorneys for Plaintiff
APPLE INC.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

APPLE INC., a California corporation, Plaintiff, v. SAMSUNG ELECTRONICS CO., LTD., a Korean corporation; SAMSUNG ELECTRONICS AMERICA, INC., a New York corporation; SAMSUNG TELECOMMUNICATIONS AMERICA, LLC, a Delaware limited liability company. Defendants.	Case No. **JURY TRIAL DEMAND** **COMPLAINT FOR PATENT INFRINGEMENT, FEDERAL FALSE DESIGNATION OF ORIGIN AND UNFAIR COMPETITION, FEDERAL TRADEMARK INFRINGEMENT, STATE UNFAIR COMPETITION, COMMON LAW TRADEMARK INFRINGEMENT, AND UNJUST ENRICHMENT**

Plaintiff Apple Inc. ("Apple") complains and alleges as follows against Defendants Samsung Electronics Co., Ltd., Samsung Electronics America, Inc., and Samsung Telecommunications America, LLC (collectively "Samsung").

THE NATURE OF THE ACTION

1. Apple revolutionized the telecommunications industry in 2007 when it introduced the wildly popular iPhone, a product that dramatically changed the way people view mobile phones. Reviewers, analysts and consumers immediately recognized the iPhone as a "game changer." Before the iPhone, cell phones were utilitarian devices with key pads for dialing and small, passive display screens that did not allow for touch control. The iPhone was radically different. In one small and lightweight handheld device, it offered sophisticated mobile phone functions, a multi-touch screen that allows users to control the phone with their fingers, music storage and playback, a mobile computing platform for handheld applications, and full access to the Internet. These features were combined in an elegantly designed product with a distinctive user interface, icons, and eye-catching displays that gave the iPhone an unmistakable look.

2. Those design features were carried over to the iPod touch, another product that Apple introduced in 2007. The iPod touch has a product configuration and physical appearance that is virtually identical to the iPhone. Moreover, the iPod touch utilizes the same user interface icons and screen layout as the iPhone, displaying the unmistakable iPhone appearance.

3. Apple introduced another revolutionary product, the iPad, in 2010. The iPad is an elegantly designed computer tablet with a color touch screen, a user interface reminiscent of the iPhone's user interface, and robust functionality that spans both mobile computing and media storage and playback. Because of its innovative technology and distinctive design, the iPad achieved instant success.

4. Apple's creative achievements have resulted in broad intellectual property protection for Apple's innovations, including utility and design patents, trademarks, and trade dress protection. Nevertheless, Apple's innovations have been the subject of widespread emulation by its competitors, who have attempted to capitalize on Apple's success by imitating Apple's innovative technology, distinctive user interfaces, and elegant and distinctive product

design. One of the principal imitators is Samsung, which recently introduced the Galaxy line of mobile phones and Galaxy Tab computer tablet, all of which use the Google Android operating system, to compete with the iPhone and iPad. Instead of pursuing independent product development, Samsung has chosen to slavishly copy Apple's innovative technology, distinctive user interfaces, and elegant and distinctive product and packaging design, in violation of Apple's valuable intellectual property rights. As alleged below in detail, Samsung has made its Galaxy phones and computer tablet work and look like Apple's products through widespread patent and trade dress infringement. Samsung has even misappropriated Apple's distinctive product packaging.

5. By this action, Apple seeks to put a stop to Samsung's illegal conduct and obtain compensation for the violations that have occurred thus far.

THE PARTIES

6. Apple is a California corporation having its principal place of business at 1 Infinite Loop, Cupertino, California 95014.

7. Samsung Electronics Co., Ltd. (referred to individually herein as "SEC") is a Korean corporation with its principal offices at 250, 2-ga, Taepyong-ro, Jung-gu, Seoul, 100-742, South Korea. On information and belief, SEC is South Korea's largest company and one of Asia's largest electronics companies. SEC designs, manufactures, and provides to the U.S. and world markets a wide range of products, including consumer electronics, computer components and myriad mobile and entertainment products.

8. Samsung Electronics America, Inc. (referred to individually herein as "SEA") is a New York corporation with its principal place of business at 105 Challenger Road, Ridgefield Park, New Jersey 07660. On information and belief, SEA was formed in 1977 as a subsidiary of SEC, and markets, sells, or offers for sale a variety of consumer electronics, including TVs, VCRs, DVD and MP3 players, and video cameras, as well as memory chips and computer accessories, such as printers, monitors, hard disk drives, and DVD/CD-ROM drives. On information and belief, SEA also manages the North American operations of Samsung Telecommunications America, Samsung Electronics Canada, and Samsung Electronics Mexico.

9. Samsung Telecommunications America, LLC (referred to individually herein as "STA") is a Delaware limited liability company with its principal place of business at 1301 East Lookout Drive, Richardson, Texas 75081. On information and belief, STA was founded in 1996 as a subsidiary of SEC, and markets, sells, or offers for sale a variety of personal and business communications devices in the United States, including cell phones.

JURISDICTION

10. This Court has subject matter jurisdiction under 15 U.S.C. § 1121 (action arising under the Lanham Act); 28 U.S.C. § 1331 (federal question); 28 U.S.C. § 1338(a) (any Act of Congress relating to patents or trademarks); 28 U.S.C. § 1338(b) (action asserting claim of unfair competition joined with a substantial and related claim under the trademark laws); and 28 U.S.C. § 1367 (supplemental jurisdiction).

11. This Court has personal jurisdiction over SEC, SEA and STA because each of these Samsung entities has committed and continues to commit acts of infringement in violation of 35 U.S.C. § 271 and 15 U.S.C. § 1114 and 1125, and places infringing products into the stream of commerce, with the knowledge or understanding that such products are sold in the State of California, including in this District. The acts by SEC, SEA and STA cause injury to Apple within this District. Upon information and belief, SEC, SEA and STA derive substantial revenue from the sale of infringing products within this District, expect their actions to have consequences within this District, and derive substantial revenue from interstate and international commerce.

VENUE AND INTRADISTRICT ASSIGNMENT

12. Venue is proper within this District under 28 U.S.C. §§ 1391(b) and (c) because Samsung transacts business within this district and offers for sale in this district products that infringe the Apple patents, trade dress, and trademarks. In addition, venue is proper because Apple's principal place of business is in this district and Apple suffered harm in this district. Moreover, a substantial part of the events giving rise to the claim occurred in this district. Pursuant to Local Rule 3-2(c), Intellectual Property Actions are assigned on a district-wide basis.

BACKGROUND

APPLE'S INNOVATIONS

13. Apple is a leading designer and manufacturer of mobile communication devices, personal computers, and portable digital media players. As a result of its significant investment in research and development, Apple has developed innovative technologies that have changed the face of the computer and telecommunications industries. One such pioneering technology is Apple's Multi-Touch™ user interface, which allows users to navigate their iPhone, iPod touch, and iPad devices by tapping and swiping their fingers on the screen.

14. In 2007, Apple revolutionized the telecommunications industry when it introduced the iPhone. The iPhone combined in one small and lightweight handheld device sophisticated mobile phone functions, media storage and playback, a tactile user interface that allows users to control the phone with their fingers, mobile computing power to run diverse pre-installed and downloadable applications, and functionality to gain full access to the Internet. These features were combined in an elegant glass and stainless steel case with a distinctive user interface that gave the iPhone an immediately recognizable look.

15. As a direct result of its innovative and distinctive design and its cutting edge technological features, the iPhone was an instant success, and it immediately became uniquely associated with Apple as its source. Reviewers and analysts universally praised the iPhone for its "game changing" features. *Time Magazine* listed the iPhone number one on its List of Top Ten Gadgets for 2007, noting that "[t]he iPhone changed the way we think about how mobile media devices should look, feel and perform." *The New York Times* called it "revolutionary." As of March 2011, more than 108 million iPhones had been sold worldwide.

16. Also in 2007, Apple launched the iPod touch, a digital music player. The iPod touch incorporated the distinct style of the iPhone and also became an immediate success. By March 2011, Apple had sold over 60 million units.

17. After introducing the iPhone, Apple continued to innovate and achieve success with a series of pioneering designs—more sophisticated, advanced versions of the iPhone, and then, in 2010, the iPad. The iPad is a computer tablet with a color 9.7-inch touch screen that

allows users to manipulate icons and data with their fingers in the same fashion as the iPhone and iPod touch screens. Reviewers and analysts immediately recognized the iPad as a revolutionary product, describing it as a "winner" and a "new category of device" that would "replace laptops for many people."

18. No computer product that preceded the iPad looked like the iPad, but its design did resemble other Apple products—namely, the iPhone and the iPod touch, thereby extending the unique and innovative Apple design and trade dress to a new product—tablet computers. In its first 80 days on the market, Apple sold 3 million iPad units. By March 2011, Apple sold over 19 million iPads.

19. Apple's iPhone, iPad and iPod touch products have been extensively advertised throughout the United States in virtually every media outlet, including network and syndicated television, the Internet, billboards, magazines and newspapers—with the vast majority of the advertisements featuring photographs of the distinctive design of these products. Apple's advertising expenditures for these products for fiscal years 2007-2010 were in excess of $2 billion.

20. In addition, Apple's iPhone, iPod touch, and iPad products have received unsolicited comment and attention in print and broadcast media throughout the world. Each new generation of these Apple products is the subject of positive commentary and receives unsolicited praise from independent media commentators. Frequently, these unsolicited commentaries are accompanied by images of the iPhone, iPad, and iPod touch products, including their unique packaging.

21. The Apple iPhone, iPod touch, and iPad product design has come to represent and symbolize the superb quality of Apple's products and enjoys substantial goodwill among consumers. The iPad, iPhone, and iPod touch have garnered widespread acclaim for their unique product design and outstanding performance. *Time Magazine* named the iPad one of the 50 Best Inventions of the Year 2010, *Popular Science* heralded it as the Top Tablet in its Best of What's New 2010 feature, and the popular technology blog Engadget selected the device as both the 2010 Editors' Choice Gadget of the Year and Tablet of the Year. In addition, the iPad received a 2010

Red Dot Award for Product Design and was nominated for the 2010 People's Design Award. The iPhone has received several awards over the years, including a 2008 Design and Art Direction (D&AD) "Black Pencil" award, a 2008 International Forum (iF) Product Design Award, and the 2008 International Design Excellence Award (IDEA) Best in Show. More recently, Engadget named the iPhone 4 the 2010 Editors' Choice Phone of the Year, and the device received the Best Mobile Device award at the Mobile World Congress in February 2011. Engadget also included the iPhone in its feature on the 10 Gadgets That Defined the Decade. The iPod touch won the 2008 D&AD "Yellow Pencil" award as well as the 2008 iF Product Design Award.

APPLE'S INTELLECTUAL PROPERTY RIGHTS

Apple's Utility Patents

22. Apple has protected its innovative designs and cutting-edge technologies through a broad range of intellectual property rights. Among those rights are the utility patents listed below. Apple's utility patents cover many of the elements that the world has come to associate with Apple's mobile devices. These include patents covering fundamental features of the Multi-Touch™ user interface that enable Apple's devices to understand user gestures and to respond by performing a wide variety of functions, such as selecting, scrolling, pinching, and zooming.

23. In addition, Apple has patented many of the individual features that together add up to the high-quality experience that users have come to associate with Apple products. Apple's innovations ranging from the arrangement of text messages on the screen, to the way images and documents appear to "bounce back" when the user scrolls too far, down to movement of the buttons have been recognized by the United States Patent and Trademark Office as patent-worthy contributions to the art.

24. Among the patents that Apple has been awarded are the patents listed below, attached as Exhibits 1-7, to which Apple owns all rights, title, and interest.

Patent Number	Title
7,812,828 (the "'828 patent")	Ellipse Fitting For Multi-Touch Surfaces
7,669,134 (the "'134 patent")	Method and Apparatus For Displaying Information During An Instant Messaging Session
6,493,002 (the "'002 patent")	Method and Apparatus for Displaying and Accessing Control and Status Information in a Computer System
7,469,381 (the "'381 patent")	List Scrolling and Document Translation, Scaling and Rotation on a Touch-Screen Display
7,844,915 (the "'915 patent")	Application Programming Interfaces for Scrolling Operations
7,853,891 (the "'891 patent")	Method and Apparatus for Displaying a Window for a User Interface
7,863,533 (the "'533 patent")	Cantilevered Push Button Having Multiple Contacts and Fulcrums

Apple's Design Patents

25. Apple also has protected its innovative designs through design patents issued by the United States Patent and Trademark Office. The Apple design patents cover the many famous ornamental features of Apple's devices, such as the flat black face, metallic bezel, and the distinctive matrix of application icons. Apple owns all right, title, and interest in and to each of the asserted design patents listed below, copies of which are attached as Exhibits 8-10.

Patent Number	Title
D627,790 (the "'D790 patent")	Graphical User Interface For a Display Screen or Portion Thereof
D602,016 (the "'D016 patent")	Electronic Device
D618,677 (the "'D677 patent")	Electronic Device

Apple's Trade Dress

26. Apple holds trade dress protection in the design and appearance of the iPhone, the iPod touch, and the iPad, together with their distinctive user interfaces and product packaging.

iPhone Trade Dress

27. The iPhone is radically different from the devices that preceded it. It has a distinctive shape and appearance—a flat rectangular shape with rounded corners, a metallic edge, a large display screen bordered at the top and bottom with substantial black segments, and a selection of colorful square icons with rounded corners that mirror the rounded corners of the iPhone itself, and which are the embodiment of Apple's innovative iPhone user interface. As shown below, the end result is an elegant product that is more accessible, easier to use, and much less technically intimidating than previously available smart phones and PDAs. The iPhone product design immediately became closely associated with Apple.

28. Each of these elements of the iPhone product configuration is distinctive and serves to identify Apple as the source of the iPhone products. Moreover, none of these elements is functional.

29. Extending its innovative style to the packaging, Apple created an equally elegant and distinctive packaging for the iPhone products. The packaging features a compact black or black-and-white box with eye-catching metallic silver lettering on a matte black surface, with the

sides of the top of the box extending down to cover the bottom portion of the box completely. The outside of the box has a clean style—with minimal wording and a simple, prominent, nearly full-size photograph of the iPhone product itself. The style carries over within the box—with the iPhone cradled within a specially designed black display so that the iPhone, and nothing else, is immediately visible when the box is opened. The accessories and instructional materials are hidden from view underneath the iPhone tray—emphasizing the accessible nature of the iPhone itself. The design entices purchasers to pick up the iPhone and try it out, without worrying that it is complicated.

30. As with the product configuration itself, each of these elements of the iPhone packaging is distinctive and serves to identify Apple as the source of the iPhone products.

Moreover, none of these elements is functional—and there are a plethora of alternative packaging options available to Apple's competitors.

iPod touch Trade Dress

31. The iPod touch has a product configuration and physical appearance that is virtually identical to the iPhone. It has a flat rectangular shape with rounded corners, a silver edge, a large display screen bordered at the top and bottom with substantial black segments, and a selection of colorful square icons with rounded corners that mirror the rounded corners of the iPod touch (and the iPhone), and which are the embodiment of Apple's innovative iPod touch user interface. As shown below, the end result is an elegant product that invites use. Like the iPhone, the iPod touch immediately became closely associated with Apple.

32. Each of these elements of the iPod touch product configuration is distinctive and serves to identify Apple as the source of the iPod touch products. Moreover, none of these elements is functional.

iPad Trade Dress

33. Because it embodies the same trade dress elements as the iPhone, the iPad resembles a "grown-up iPhone." It has a flat rectangular shape with rounded corners, a silver edge, a large display screen with a substantial black border, and a selection of Apple's colorful

square icons—all with rounded corners that mirror the rounded corners of the iPad, iPhone and iPod touch.

34. Each of these elements of the iPad product configuration is distinctive and serves to identify Apple as the source of the iPad products. Moreover, none of these elements is functional.

35. The packaging for the iPad is similarly innovative and, like the iPhone, utilizes a box that, when opened, prominently displays the product so that it is immediately visible, with all other accessories and materials layered beneath it. Also similar to the iPhone, the outside of the iPad box has a clean style—with minimal silver metallic wording and a simple, prominent, nearly full-size photograph of the iPad product on a white background.

36. Each of these elements of the iPad packaging is distinctive and serves to identify Apple as the source of the products. Moreover, none of these elements is functional.

Trade Dress Registrations

37. Apple owns three registrations for the design and configuration of the iPhone.

38. U.S. Registration No. 3,470,983 is for the overall design of the product, including the rectangular shape, the rounded corners, the silver edges, the black face, and the display of sixteen colorful icons. Attached hereto as Exhibit 11 is a true and correct copy of U.S. Registration No. 3,470,983.

39. U.S. Registration No. 3,457,218 is for the configuration of a rectangular handheld mobile digital electronic device with rounded corners. Attached hereto as Exhibit 12 is a true and correct copy of U.S. Registration No. 3,457,218.

40. U.S. Registration No. 3,475,327 is for a rectangular handheld mobile digital electronic device with a gray rectangular portion in the center, a black band above and below the gray rectangle and on the curved corners, and a silver outer border and side. Attached hereto as Exhibit 13 is a true and correct copy of U.S. Registration No. 3,475,327.

Trade Dress at Issue

41. The following non-functional elements of Apple's product designs comprise the product configuration trade dress at issue in this case (the "Apple Product Configuration Trade Dress"):

- a rectangular product shape with all four corners uniformly rounded;

- the front surface of the product dominated by a screen surface with black borders;
- as to the iPhone and iPod touch products, substantial black borders above and below the screen having roughly equal width and narrower black borders on either side of the screen having roughly equal width;
- as to the iPad product, substantial black borders on all sides being roughly equal in width;
- a metallic surround framing the perimeter of the top surface;
- a display of a grid of colorful square icons with uniformly rounded corners; and
- a bottom row of square icons (the "Springboard") set off from the other icons and that do not change as the other pages of the user interface are viewed.

42. The following non-functional elements of Apple's packaging designs comprise the product packaging trade dress at issue in this case (the "Apple Product Packaging Trade Dress"):

- a rectangular box with minimal metallic silver lettering and a large front-view picture of the product prominently on the top surface of the box;
- a two-piece box wherein the bottom piece is completely nested in the top piece; and
- use of a tray that cradles products to make them immediately visible upon opening the box.

43. Collectively, the Apple Product Configuration Trade Dress and the Apple Product Packaging Trade Dress are referred to herein as the "Apple Product Trade Dress."

Apple's Trademarks

44. Apple has protectable trademark rights in various icons used in the user interface in the iPhone, iPod touch, and iPad products—icons that are consistently used in the Apple family of mobile products.

45. For example, U.S. Registration No. 3,886,196 covers an icon that is green in color with a white silhouette of a phone handset arranged at a 45 degree angle and centered on the icon that represents the application for making telephone calls:

Attached hereto as Exhibit 14 is a true and correct copy of U.S. Registration No. 3,866,196.

46. U.S. Registration No. 3,889,642 covers an icon that is green in color with a white silhouette of a speech bubble centered on the icon that represents the application for messaging:

Attached hereto as Exhibit 15 is a true and correct copy of U.S. Registration No. 3,889,642.

47. U.S. Registration No. 3,886,200 covers an icon featuring a yellow and green sunflower against a light-blue background that represents the application for photos:

Attached hereto as Exhibit 16 is a true and correct copy of U.S. Registration No. 3,866,200.

48. U.S. Registration No. 3,889,685 covers an icon that features gears against a gray background that represents the application for settings:

Attached hereto as Exhibit 17 is a true and correct copy of U.S. Registration No. 3,889,685.

49. U.S. Registration No. 3,886,169 covers an icon that features a yellow note pad that represents the application for notes:

Attached hereto as Exhibit 18 is a true and correct copy of U.S. Registration No. 3,886,169.

50. U.S. Registration No. 3,886,197 is for the silhouette of a man on a spiral bound address book that represents the icon for contacts:

Attached hereto as Exhibit 19 is a true and correct copy of U.S. Registration No. 3,886,197.

51. Collectively, the application icons displayed in Paragraphs 45-50 represent Apple's "Registered Icon Trademarks." Apple uses these Registered Icon Trademarks in connection with its iPhone, iPod touch, and iPad products.

52. To represent the iTunes application, Apple uses an icon that is purple in color with a white circular band and a silhouette of two eighth-notes superimposed on the white circular band:

Pending U.S. Application Serial No. 85/041,463 covers this icon (the "Purple iTunes Store Trademark"). Attached hereto as Exhibit 20 is a true and correct copy of the TARR status report for U.S. Application Serial No. 85/041,463.

53. Moreover, Apple also owns a federal trademark registration for a logo for its iTunes on-line music service, U.S. Registration No. 2,935,038, (the "iTunes Eighth Note and CD Design Trademark"):

U.S. Registration No. 2,935,038 issued on March 22, 2005. Apple filed an Affidavit under Section 15 of the Lanham Act on March 24, 2010, rendering the registration incontestable. Attached hereto as Exhibit 21 is a true and correct copy of U.S. Registration No. 2,935,038.

SAMSUNG'S INFRINGING PRODUCTS

54. Samsung has imported into or sold in the United States the following products, each of which infringes one or more of Apple's Intellectual Property Rights: the Samsung Captivate, Continuum, Vibrant, Galaxy S 4G, Epic 4G, Indulge, Mesmerize, Showcase,

Fascinate, Nexus S, Gem, Transform, Intercept, and Acclaim smart phones and the Samsung Galaxy Tab tablet.

55. Rather than innovate and develop its own technology and a unique Samsung style for its smart phone products and computer tablets, Samsung chose to copy Apple's technology, user interface and innovative style in these infringing products.

56. Samsung's Galaxy family of mobile products, introduced in 2010, is exemplary. The copying is so pervasive, that the Samsung Galaxy products appear to be actual Apple products—with the same rectangular shape with rounded corners, silver edging, a flat surface face with substantial top and bottom black borders, gently curving edges on the back, and a display of colorful square icons with rounded corners. When a Samsung Galaxy phone is used in public, there can be little doubt that it would be viewed as an Apple product based upon the design alone.

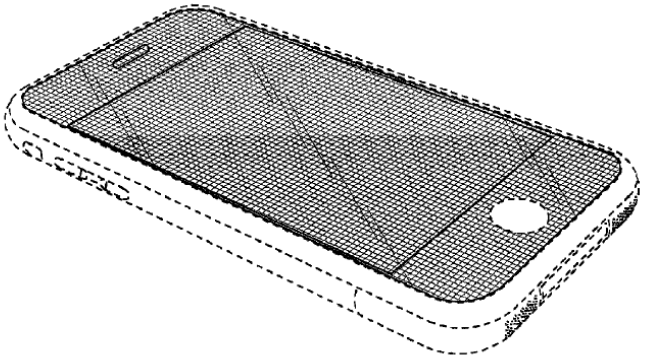
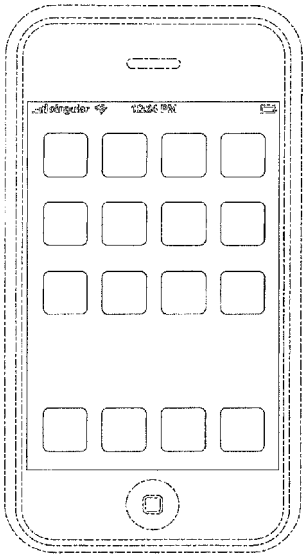
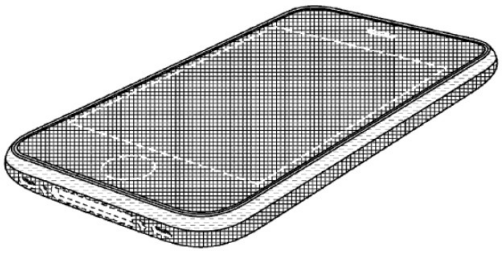
57. Samsung had many options in developing its smart phones. Indeed, earlier versions of Samsung smart phones did not embody the same combination of elements of Apple's trade dress. Even the icons in earlier versions of the Samsung smart phones looked different because they had a variety of shapes—and did not appear as a field of square icons with rounded corners.

58. Samsung it chose to infringe Apple's patents, trade dress, and trademark rights through the design, packaging and promotion of its Galaxy mobile phones and the Galaxy Tab computer tablet, and similar products, and it did so willfully to trade upon the goodwill that Apple has developed in connection with its Apple family of mobile products.

Infringement of Apple's Patents

59. Samsung's infringement of the Apple utility patents identified in this Complaint provides Samsung with unique functionality for its products that was the result of Apple's innovation, not Samsung's. Samsung has not obtained permission from Apple to use its inventions in the identified utility patents.

60. Moreover, as the side-by-side comparisons shown below reveal, Samsung has misappropriated Apple's patented mobile phone design in the accused products, including the Samsung Galaxy mobile phone depicted below.

’D677 Patent 	
’D790 Patent  FIGURE	
’D016 Patent 	

Infringement of Apple's Trade Dress

61. Samsung announced its Galaxy line of Android-based smart phones in March of 2010 in South Korea. The original model, Galaxy S i9000, is shown below side by side with an iPhone 3GS.

Apple iPhone 3GS

Galaxy S i9000

62. Each of Samsung's Galaxy phones embodies a combination of several elements of the Apple Product Configuration Trade Dress identified above, namely, a product configuration with

- a rectangular product shape with all four corners uniformly rounded;
- the front surface of the product dominated by a screen surface with black borders;
- substantial black borders above and below the screen having roughly equal width and narrower black borders on either side of the screen having roughly equal width;
- a metallic surround framing the perimeter of the top surface;
- a display of a grid of colorful square icons with uniformly rounded corners; and
- a bottom row of icons set off from the other icons and that do not change as the other pages of the user interface are viewed.

63. Samsung also imitated Apple's Product Packaging Trade Dress:

Samsung's packaging includes:

- a rectangular box with metallic silver lettering and a large front-view picture of the product prominently on the top surface of the box;
- a two-piece box wherein the bottom piece is completely nested in the top piece; and
- use of a design that cradles products to make them immediately visible upon opening the box.

64. As shown below, Samsung's Galaxy Tab computer tablet also slavishly copies a combination of several elements of the Apple Product Configuration Trade Dress.

Samsung Tab

Specifically, the Samsung computer tablet has:

- rectangular product shape with all four corners uniformly rounded;
- front surface of the product dominated by a screen surface with black borders;
- substantial black borders on all sides being roughly equal in width; and
- a display of a grid of colorful square icons with uniformly rounded corners.

65. Samsung's Galaxy Tab computer tablet packaging also features key elements of the Apple Product Packaging Trade Dress for Apple's iPad products.

The following Apple Product Packaging Trade Dress is incorporated in the Samsung Galaxy Tab tablet computer products:

- a rectangular box with metallic silver lettering and a large front-view picture of the product prominently on the top surface of the box;
- a two-piece box wherein the bottom piece is completely nested in the top piece; and
- use of a design that cradles products to make them immediately visible upon opening the box.

Infringement of Apple's Trademarks

66. In addition to copying Apple's Product Trade Dress, Samsung has also copied numerous application icons in which Apple had valid trademark rights, as shown below:

Apple Icons	Samsung Icons
	
	

Apple Icons	Samsung Icons
	
	
	
	
	

67. Moreover, the icon that Samsung uses for its music application is virtually identical to the iTunes "Eighth Note and CD" logo that Apple has registered with the United States Patent and Trademark Office:

Apple Icon

Samsung Icon

68. Samsung's adoption of a trade dress that slavishly copies the Apple Product Trade Dress and its use of various icons that infringe Apple's trademark rights is likely to cause confusion or mistake, or to deceive consumers, purchasers, and others into thinking that Samsung products are Apple products, or that they are sponsored by or affiliated with Apple, when they are not. The copying is particularly problematic because the Samsung Galaxy products are the type of products that will be used in public—on the bus, in cafes, in stores, or at school, where third parties, who were not present when the products were purchased, will associate them with Apple because they have the unmistakable Apple look that is created from the various elements of the Apple Trade Dress.

69. Of significant concern for Apple is that Apple devotes a tremendous amount of resources—technical research and development *and* design resources—to develop its cutting edge products. Part of the cachet of Apple products is the very fact that they consistently stand-out from all of the other products on the market. Apple's goodwill among consumers is closely tied to its position as an outlier in technology and communications products, which causes each release of a new product to be highly anticipated among consumers who want to be among the early adopters of the newest Apple product. Samsung's flagrant and relentless copying of Apple's intellectual property rights in its Galaxy family of products not only allows Samsung to

reap benefits from Apple's investment, it also threatens to diminish the very important goodwill that Apple has cultivated with its products.

70. On information and belief Samsung's marketing has played up the similarities between its Galaxy family of phones and the Apple iPhone. On information and belief, Samsung's mobile phones were marketed as the phone that is the closest to the iPhone—for consumers who wanted a product with the distinctive Apple look, but who did not want to pay for the real product.

71. Apple's efforts to address Samsung's pervasive copying of Apple's innovations and intellectual property directly with Samsung have been unsuccessful. Apple is left with no choice but to file this lawsuit in order to protect one of its most valuable assets—the technology used in and the designs of the iPhone, iPod touch, and the iPad.

FIRST CLAIM FOR RELIEF

(Trade Dress Infringement)

(Lanham Act Section 43(a), 15 U.S.C. § 1125(a))

72. Apple incorporates and realleges paragraphs 1 through 71 of this Complaint.

73. Apple is the owner of all right and title to the distinctive iPhone, iPod touch, and iPad trade dress. The Apple Product Configuration Trade Dress, as embodied in the Apple iPhone, iPod touch, and iPad products, has acquired secondary meaning, and is not functional. In addition, the Apple Product Packaging Trade Dress, embodied in the packaging for the Apple iPhone and iPad devices, is inherently distinctive and not functional.

74. In addition, based on extensive and consistent advertising, promotion and sales throughout the United States, the Apple Product Trade Dress has acquired distinctiveness and enjoys secondary meaning among consumers, identifying Apple as the source of these products.

75. Apple's extensive promotion of the distinctive Apple Product Trade Dress has resulted in Apple's acquisition of valuable, legally protected rights in the Apple Product Trade Dress as well as considerable customer goodwill.

76. The Samsung Galaxy line of products has misappropriated the Apple Product Trade Dress by mimicking a combination of several elements of that trade dress.

77. Samsung's manufacture and distribution of the Samsung Galaxy products with packaging, product design, and product user interface features that mimic a combination of several elements of the Apple Product Trade Dress is likely to cause confusion, or to cause mistake, or to deceive the consumer as to the affiliation, connection or association of Samsung with Apple, or as to the origin, sponsorship, or approval by Apple of Samsung's goods, services or commercial activities.

78. Samsung's manufacture and distribution of the Samsung Galaxy line of products with packaging, product design and product user interface features that mimic a combination of several elements of the Apple Product Trade Dress enables Samsung to benefit unfairly from Apple's reputation and success, thereby giving Samsung's infringing products sales and commercial value they would not have otherwise.

79. Samsung's actions constitute unfair competition and false designation or origin in violation of Section 43(a) of the Lanham Act, 15 U.S.C. §1125(a).

80. Samsung knew of Apple's Product Trade Dress when it designed its Galaxy line of products, and has refused to change its product or packaging design in response to Apple's repeated objections. Accordingly, Samsung's infringement has been and continues to be intentional, willful and without regard to Apple's Product Trade Dress.

81. Apple has been and will continue to be irreparably harmed and damaged by Samsung's conduct, and Apple lacks an adequate remedy at law to compensate for this harm and damage.

82. Apple is informed and believes, and on that basis alleges, that Samsung has gained profits by virtue of its infringement of the Apple Product Trade Dress.

83. Apple also has sustained damages as a direct and proximate result of Samsung's infringement of the Apple Product Trade Dress in an amount to be proven at trial.

84. Because Samsung's actions have been willful, Apple is entitled to treble its actual damages or Samsung's profits, whichever is greater, and to an award of costs, and, this being an exceptional case, reasonable attorneys' fees pursuant to 15 U.S.C. § 1117(a).

SECOND CLAIM FOR RELIEF

(Federal Trade Dress Infringement)

(15 U.S.C. § 1114)

85. Apple incorporates and realleges paragraphs 1 through 84 of this Complaint.

86. Apple owns three registrations for the design and configuration of the iPhone.

87. U.S. Registration No. 3,470,983 is for the overall design of the product, including the rectangular shape, the rounded corners, the silver edges, the black face, and the display of sixteen colorful icons.

88. U.S. Registration No. 3,457,218 is for the configuration of a rectangular handheld mobile digital electronic device with rounded corners.

89. U.S. Registration No. 3,475,327 is for a rectangular handheld mobile digital electronic device with a gray rectangular portion in the center, a black band above and below the gray rectangle and on the curved corners, and a silver outer border and side.

90. The Samsung Galaxy line of products copy and infringe these three trade dress registrations (collectively, "Apple's Registered Trade Dress").

91. Samsung's manufacture and distribution of the Samsung Galaxy products with packaging, product design, and product user interface features that copy a combination of several elements of Apple's Registered Trade Dress is likely to cause confusion, or to cause mistake, or to deceive the consumer as to the affiliation, connection or association of Samsung with Apple, or as to the origin, sponsorship, or approval by Apple of Samsung's goods, services or commercial activities.

92. Samsung's manufacture and distribution of the Samsung Galaxy line of products with packaging, product design and product user interface features that copy a combination of several elements of Apple's Registered Trade Dress enables Samsung to benefit unfairly from Apple's reputation and success, thereby giving Samsung's infringing products sales and commercial value they would not have otherwise.

93. Prior to Samsung's first use of Apple's Registered Trade Dress, Samsung was aware of Apple's business and had either actual notice and knowledge, or constructive notice of

Apple's Registered Trade Dress, and has refused to change its product or packaging design in response to Apple's repeated objections.

94. Samsung's unauthorized use of a trade dress for its Galaxy product line that infringes Apple's Registered Trade Dress is likely, if not certain, to deceive or to cause confusion or mistake among consumers as to the origin, sponsorship or approval of the Samsung Galaxy line of products and/or to cause confusion or mistake as to any affiliation, connection or association between Apple and Samsung, in violation of 15 U.S.C. § 1114(a).

95. Apple is informed and believes, and on that basis alleges, that Samsung's infringement of Apple's Registered Trade Dress as described herein has been and continues to be intentional, willful and without regard to Apple's Registered Trade Dress.

96. Apple is informed and believes, and on that basis alleges, that Samsung has gained profits by virtue of its infringement of Apple's Registered Trade Dress.

97. Apple will suffer and is suffering irreparable harm from Samsung's infringement of Apple's Registered Trade Dress insofar as Apple's invaluable goodwill is being eroded by Samsung's continuing infringement. Apple has no adequate remedy at law to compensate it for the loss of business reputation, customers, market position, confusion of potential customers and good will flowing from the Samsung's infringing activities. Pursuant to 15 U.S.C. § 1116, Apple is entitled to an injunction against Samsung's continuing infringement of Apple's Registered Trade Dress. Unless enjoined, Samsung will continue its infringing conduct.

98. Because Samsung's actions have been committed with intent to damage Apple and to confuse and deceive the public, Apple is entitled to treble its actual damages or Samsung's profits, whichever is greater, and to an award of costs and, this being an exceptional case, reasonable attorneys' fees pursuant to 15 U.S.C. § 1117(a) and 1117(b).

THIRD CLAIM FOR RELIEF

(Federal Trademark Infringement)

(15 U.S.C. § 1114)

99. Apple incorporates and realleges paragraphs 1 through 98 of this Complaint.

100. Apple owns seven federal trademark registrations for the distinctive and colorful square application icons used in the user interface for the iPhone, iPod touch, and iPad Registered Icon Trademarks, i.e., the Registered Icon Trademarks.

101. The Samsung Galaxy line of products has infringed the Registered Icon Trademarks by using variations of those application icons in Samsung's products. The Samsung Galaxy line of products have also infringed Apple's registered iTunes Eighth Note and CD Logo.

102. Samsung's use of its infringing application icons is likely to cause confusion, or to cause mistake, or to deceive the consumer as to the affiliation, connection or association of Samsung with Apple, or as to the origin, sponsorship, or approval by Apple of Samsung's goods, services or commercial activities.

103. Samsung's use of the infringing application icons enables Samsung to benefit unfairly from Apple's reputation and success, thereby giving Samsung's infringing products sales and commercial value they would not have otherwise.

104. Prior to Samsung's first use of the infringing application icons, Samsung was aware of Apple's business and had either actual notice and knowledge, or constructive notice of, Apple's Registered Icon Trademarks.

105. Samsung's unauthorized use of the infringing application icons is likely, if not certain, to deceive or to cause confusion or mistake among consumers as to the origin, sponsorship or approval of the Samsung Galaxy line of products and/or to cause confusion or mistake as to any affiliation, connection or association between Apple and Samsung, in violation of 15 U.S.C. § 1114(a).

106. Apple is informed and believes, and on that basis alleges, that Samsung's infringement of Apple's Registered Icon Trademarks as described herein has been and continues to be intentional, willful and without regard to Apple's rights.

107. Apple is informed and believes, and on that basis alleges, that Samsung has gained profits by virtue of its infringement of Apple's Registered Icon Trademarks.

108. Apple will suffer and is suffering irreparable harm from Samsung's infringement of Registered Icon Trademarks insofar as Apple's invaluable good will is being eroded by

Samsung's continuing infringement. Apple has no adequate remedy at law to compensate it for the loss of business reputation, customers, market position, confusion of potential customers and good will flowing from the Samsung's infringing activities. Pursuant to 15 U.S.C. § 1116, Apple is entitled to an injunction against Samsung's continuing infringement of Apple's Registered Icon Trademarks. Unless enjoined, Samsung will continue its infringing conduct.

109. Because Samsung's actions have been committed with intent to damage Apple and to confuse and deceive the public, Apple is entitled to treble its actual damages or Samsung's profits, whichever is greater, and to an award of costs and, this being an exceptional case, reasonable attorneys' fees pursuant to 15 U.S.C. § 1117(a) and 1117(b).

FOURTH CLAIM FOR RELIEF

(Common Law Trademark Infringement)

110. Apple incorporates and realleges paragraphs 1 through 109 of this Complaint.

111. Apple has prior rights in Apple's Registered Icon Trademarks and the Purple iTunes Store Trademark.

112. The Samsung Galaxy line of products have infringed Apple's Registered Icon Trademarks and the Purple iTunes Store Trademark by using identical or similar application icons in Samsung's products.

113. Samsung's use of its infringing application icons is likely to cause confusion, or to cause mistake, or to deceive the consumer as to the affiliation, connection or association of Samsung with Apple, or as to the origin, sponsorship, or approval by Apple of Samsung's goods, services or commercial activities.

114. Samsung's use of the infringing application icons enables Samsung to benefit unfairly from Apple's reputation and success, thereby giving Samsung's infringing products sales and commercial value they would not have otherwise.

115. Prior to Samsung's first use of the infringing application icons, Samsung was aware of Apple's business and had either actual notice and knowledge, or constructive notice of Apple's Registered Icon Trademarks and the Purple iTunes Store Trademark.

116. Samsung's unauthorized use of the infringing application icons is likely, if not certain, to deceive or to cause confusion or mistake among consumers as to the origin, sponsorship or approval of the Samsung Galaxy line of products and/or to cause confusion or mistake as to any affiliation, connection or association between Apple and Samsung, in violation of 15 U.S.C. § 1114(a).

117. Apple is informed and believes, and on that basis alleges, that Samsung's infringement of Apple's Registered Icon Trademarks and the Purple iTunes Store Trademark as described herein has been and continues to be intentional, willful and without regard to Apple's rights in its Registered Icon Trademarks and the Purple iTunes Store Trademark.

118. Apple is informed and believes, and on that basis alleges, that Samsung has gained profits by virtue of its infringement of Apple's Registered Icon Trademarks and the Purple iTunes Store Trademark.

119. Apple will suffer and is suffering irreparable harm from Samsung's infringement of Apple's Registered Icon Trademarks and the Purple iTunes Store Trademark insofar as Apple's invaluable good will is being eroded by Samsung's continuing infringement. Apple has no adequate remedy at law to compensate it for the loss of business reputation, customers, market position, confusion of potential customers and good will flowing from the Samsung's infringing activities. Apple is entitled to an injunction against Samsung's continuing infringement of Apple's Registered Icon Trademarks and the Purple iTunes Store Trademark. Unless enjoined, Samsung will continue its infringing conduct.

120. Because Samsung's actions have been committed with intent to damage Apple and to confuse and deceive the public, Apple is entitled to treble its actual damages or Samsung's profits, whichever is greater, and to an award of costs and, this being an exceptional case, reasonable attorneys' fees pursuant to 15 U.S.C. § 1117(a) and 1117(b).

FIFTH CLAIM FOR RELIEF

(Unfair Business Practices – California Business and Professions Code § 17200, *et seq.*)

121. Apple incorporates and realleges paragraphs 1 through 120 of this Complaint.

122. The acts of Samsung described above constitute fraudulent and unlawful business practices as defined by California Business & Professions Code § 17200, *et seq*.

123. Apple has valid and protectable prior rights in the Apple Product Trade Dress, the Registered Trade Dress, the iTunes Eighth Note and CD Design Trademark, the Purple iTunes Store Trademark, and the Registered Icon Trademarks. The Apple Product Trade Dress and the Registered Trade Dress do not serve any function other than to identify Apple as the source of its mobile products. The Apple Product Trade Dress and the Registered Trade Dress are inherently distinctive, and, through Apple's long use, have come to be associated solely with Apple as the source of the products on which it is used.

124. Samsung's use of its infringing trade dress is likely to cause confusion as to the source of Samsung's products and is likely to cause others to be confused or mistaken into believing that there is a relationship between Samsung and Apple or that Samsung's products are affiliated with or sponsored by Apple.

125. The above-described acts and practices by Samsung are likely to mislead or deceive the general public and therefore constitute fraudulent business practices in violation of California Business & Professions Code §§ 17200, *et seq*.

126. The above-described acts constitute unfair competition under Section 43(a) of the Lanham Act, 15 U.S.C. § 1125(a) and trademark and trade dress infringement under Section 32 of the Lanham Act, 15 U.S.C. § 1114, and are therefore unlawful acts in violation of California Business & Professions Code §§ 17200, *et seq*.

127. Samsung acted willfully and intentionally in designing its infringing trade dress, with full knowledge of Samsung's prior rights in the distinctive Apple Product Trade Dress, Registered Trade Dress, the iTunes Eighth Note and CD Design Trademark, the Purple iTunes Store Trademark, and the Registered Icon Trademarks and with an intent to cause confusion or mistake or to deceive customers into believing that there is an affiliation between Samsung and Apple or between Samsung's products and Apple's products.

128. The unlawful and fraudulent business practices of Samsung described above present a continuing threat to, and is meant to deceive members of, the public in that Samsung

continues to promote its products by wrongfully trading on the goodwill of the Apple Product Trade Dress, Registered Trade Dress, the iTunes Eighth Note and CD Design Trademark, Purple iTunes Store Trademark, and the Registered Icon Trademarks.

129. As a direct and proximate result of these acts, Samsung has received, and will continue to profit from, the strength of the Apple Product Trade Dress, Registered Trade Dress, the iTunes Eighth Note and CD Design Trademark, the Purple iTunes Store Trademark, and the Registered Icon Trademarks.

130. As a direct and proximate result of Samsung's wrongful conduct, Apple has been injured in fact and has lost money and profits, and such harm will continue unless Samsung's acts are enjoined by the Court. Apple has no adequate remedy at law for Samsung's continuing violation of Apple's rights.

131. Samsung should be required to restore to Apple any and all profits earned as a result of their unlawful and fraudulent actions, or provide apple with any other restitutionary relief as the Court deems appropriate.

SIXTH CLAIM FOR RELIEF

(Unjust Enrichment)

132. Apple incorporates and realleges paragraphs 1 through 131 of this Complaint.

133. As a result of the conduct alleged herein, Samsung has been unjustly enriched to Apple's detriment. Apple seeks a worldwide accounting and disgorgement of all ill gotten gains and profits resulting from Samsung's inequitable activities.

SEVENTH CLAIM FOR RELIEF

(Infringement of the '022 Patent)

134. Apple incorporates and realleges paragraphs 1 through 133 of this Complaint.

135. Samsung has infringed and continues to infringe one or more claims of the '022 Patent by using, selling and/or offering to sell, in the United States and/or importing into the United States, one or more of the Samsung mobile communication devices identified in this Complaint. Samsung's infringing activities violate 35 U.S.C. § 271.

EIGTH CLAIM FOR RELIEF

(Infringement of the ’381 Patent)

136. Apple incorporates and realleges paragraphs 1 through 135 of this Complaint.

137. Samsung has infringed and continues to infringe one or more claims of the ’381 Patent by using, selling and/or offering to sell, in the United States and/or importing into the United States, one or more of the Samsung mobile communication devices identified in this Complaint. Samsung’s infringing activities violate 35 U.S.C. § 271.

NINTH CLAIM FOR RELIEF

(Infringement of the ’134 Patent)

138. Apple incorporates and realleges paragraphs 1 through 137 of this Complaint.

139. Samsung has infringed and continues to infringe one or more claims of the ’134 Patent by using, selling and/or offering to sell, in the United States and/or importing into the United States, one or more of the Samsung mobile communication devices identified in this Complaint. Samsung’s infringing activities violate 35 U.S.C. § 271.

TENTH CLAIM FOR RELIEF

(Infringement of the ’828 Patent)

140. Apple incorporates and realleges paragraphs 1 through 139 of this Complaint.

141. Samsung has infringed and continues to infringe one or more claims of the ’828 Patent by using, selling and/or offering to sell, in the United States and/or importing into the United States, one or more of the Samsung mobile communication devices identified in this Complaint. Samsung’s infringing activities violate 35 U.S.C. § 271.

ELEVENTH CLAIM FOR RELIEF

(Infringement of the ’915 Patent)

142. Apple incorporates and realleges paragraphs 1 through 141 of this Complaint.

143. Samsung has infringed and continues to infringe one or more claims of the ’915 Patent by using, selling and/or offering to sell, in the United States and/or importing into the United States, one or more of the Samsung mobile communication devices identified in this Complaint. Samsung’s infringing activities violate 35 U.S.C. § 271.

TWELFTH CLAIM FOR RELIEF

(Infringement of the ’891 Patent)

144. Apple incorporates and realleges paragraphs 1 through 143 of this Complaint.

145. Samsung has infringed and continues to infringe one or more claims of the ’891 Patent by using, selling and/or offering to sell, in the United States and/or importing into the United States, one or more of the Samsung mobile communication devices identified in this Complaint. Samsung’s infringing activities violate 35 U.S.C. § 271.

THIRTEENTH CLAIM FOR RELIEF

(Infringement of the ’533 Patent)

146. Apple incorporates and realleges paragraphs 1 through 145 of this Complaint.

147. Samsung has infringed and continues to infringe one or more claims of the ’533 Patent by using, selling and/or offering to sell, in the United States and/or importing into the United States, one or more of the Samsung mobile communication devices identified in this Complaint. Samsung’s infringing activities violate 35 U.S.C. § 271.

FOURTEENTH CLAIM FOR RELIEF

(Infringement of the ’D790 Patent)

148. Apple incorporates and realleges paragraphs 1 through 147 of this Complaint.

149. Samsung has infringed and continues to infringe the ’D790 Patent by using, selling and/or offering to sell in the United States, and/or importing into the United States one or more of the Samsung mobile communication devices identified in this Complaint, which embody the design covered by the ’D790 design patent.

FIFTEENTH CLAIM FOR RELIEF

(Infringement of the ’D016 Patent)

150. Apple incorporates and realleges paragraphs 1 through 149 of this Complaint.

151. Samsung has infringed and continues to infringe the ’D016 Patent by using, selling and/or offering to sell in the United States, and/or importing into the United States one or more of the Samsung mobile communication devices identified in this Complaint, which embody the design covered by the ’D016 design patent.

SIXTEENTH CLAIM FOR RELIEF

(Infringement of the ’D677 Patent)

152. Apple incorporates and realleges paragraphs 1 through 151 of this Complaint.

153. Samsung has infringed and continues to infringe the ’D677 Patent by using, selling and/or offering to sell in the United States, and/or importing into the United States one or more of the Samsung mobile communication devices identified in this Complaint, which embody the design covered by the ’D677 design patent.

PRAYER FOR RELIEF

WHEREFORE, Apple prays for relief, as follows:

1. A judgment that Samsung has infringed one of more claims of each of Apple’s asserted patents;

2. An order and judgment preliminarily and permanently enjoining Samsung and its officers, directors, agents, servants, employees, affiliates, attorneys, and all others acting in privity or in concert with them, and their parents, subsidiaries, divisions, successors and assigns, from further acts of infringement of Apple’s asserted patents;

3. A judgment awarding Apple all damages adequate to compensate for Samsung’s infringement of Apple’s asserted patents, and in no event less than a reasonable royalty for Samsung’s acts of infringement, including all pre-judgment and post-judgment interest at the maximum rate permitted by law;

4. A judgment awarding Apple all damages, including treble damages, based on any infringement found to be willful, pursuant to 35 U.S.C. § 284, together with prejudgment interest

5. An order preliminarily and permanently enjoining Samsung and its officers, directors, agents, servants, employees, affiliates, attorneys, and all others acting in privity or in concert with them, and their parents, subsidiaries, divisions, successors and assigns, from directly or indirectly infringing the Apple Product Trade Dress, Registered Icon Trademarks, Purple iTunes Store Trademark, and iTunes Eighth Note and CD Design Trademark, or using any other product or packaging design or designations similar to or likely to cause confusion with the Apple Product Trade Dress, Registered Icon Trademarks, Purple iTunes Store Trademark, and iTunes

Eighth Note and CD Design Trademark; from passing off Samsung's products as being associated with and or sponsored or affiliated with Apple; from committing any other unfair business practices directed toward obtaining for themselves the business and customers of Apple; and from committing any other unfair business practices directed toward devaluing or diminishing the brand or business of Apple.

6. Actual damages suffered by Apple as a result of Samsung's unlawful conduct, in an amount to be proven at trial, as well as prejudgment interest as authorized by law;

7. Reasonable funds for future corrective advertising;

8. An accounting of Samsung's profits pursuant to 15 U.S.C. § 1117;

9. A judgment trebling any damages award pursuant to 15 U.S.C. § 1117;

10. Punitive damages pursuant to California Civil Code § 3294;

11. Restitutionary relief against Samsung and in favor of Apple, including disgorgement of wrongfully obtained profits and any other appropriate relief;

12. Costs of suit and reasonable attorneys' fees; and

13. Any other remedy to which Apple may be entitled, including all remedies provided for in 15 U.S.C. § 1117, Cal. Bus. & Prof. Code §§ 17200, *et seq.*, 17500, *et seq.*, and under any other California law.

Dated: April 15, 2011

HAROLD J. MCELHINNY
MICHAEL A. JACOBS
JENNIFER LEE TAYLOR
JASON R. BARTLETT
MORRISON & FOERSTER LLP

By: _____
MICHAEL A. JACOBS

Attorneys for Plaintiff
APPLE INC.

DEMAND FOR JURY TRIAL

Pursuant to Rule 38(b) of the Federal Rules of Civil Procedure, Apple hereby demands trial by jury on all issues raised by the Complaint.

Dated: April 15, 2011

HAROLD J. MCELHINNY
MICHAEL A. JACOBS
JENNIFER LEE TAYLOR
JASON R. BARTLETT
MORRISON & FOERSTER LLP

By: _____
MICHAEL A. JACOBS

Attorneys for Plaintiff
APPLE INC.

Exhibit 1

US007812828B2

(12) **United States Patent**
Westerman et al.

(10) **Patent No.: US 7,812,828 B2**
(45) **Date of Patent: Oct. 12, 2010**

(54) **ELLIPSE FITTING FOR MULTI-TOUCH SURFACES**

(75) Inventors: **Wayne Westerman**, San Francisco, CA (US); **John G. Elias**, Townsend, DE (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 707 days.

(21) Appl. No.: **11/677,958**

(22) Filed: **Feb. 22, 2007**

(65) **Prior Publication Data**

US 2007/0139395 A1 Jun. 21, 2007

Related U.S. Application Data

(63) Continuation of application No. 11/015,434, filed on Dec. 17, 2004, now Pat. No. 7,339,580, which is a continuation of application No. 09/236,513, filed on Jan. 25, 1999, now Pat. No. 6,323,846.

(60) Provisional application No. 60/072,509, filed on Jan. 26, 1998.

(51) **Int. Cl.**
G06F 3/041 (2006.01)

(52) **U.S. Cl.** **345/173**; 345/174; 345/175; 178/18.01; 178/18.03

(58) **Field of Classification Search** 345/173–178; 178/18.01, 18.03, 19.01, 20.01; 715/863
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

3,333,160	A	7/1967	Gorski
3,541,541	A	11/1970	Engelhart
3,662,105	A	5/1972	Hurst et al.
3,798,370	A	3/1974	Hurst
4,237,421	A	12/1980	Waldron
4,246,452	A	1/1981	Chandler
4,266,144	A	5/1981	Bristol
4,290,061	A	9/1981	Serrano
4,291,303	A	9/1981	Cutler et al.
4,550,221	A	10/1985	Mabusth
4,561,002	A	12/1985	Chiu
4,672,364	A	6/1987	Lucas
4,672,558	A	6/1987	Beckes et al.

(Continued)

FOREIGN PATENT DOCUMENTS

CA	1243096	10/1988

(Continued)

OTHER PUBLICATIONS

Chang, C-C. et al. (Aug. 1, 1993). "A Hashing-Oriented Nearest Neighbor Searching Scheme," *Pattern Recognition Letters*, 14(8):625-630.

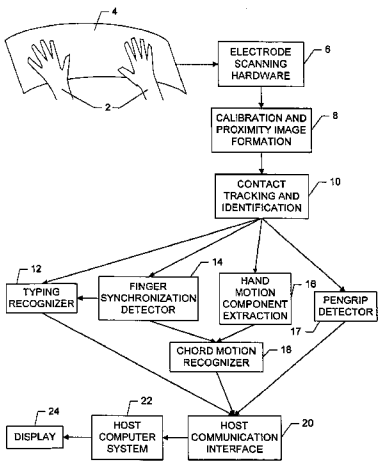
(Continued)

Primary Examiner—Amare Mengistu
Assistant Examiner—Koosha Sharifi
(74) *Attorney, Agent, or Firm*—Morrison & Foerster LLP

(57) **ABSTRACT**

Apparatus and methods are disclosed for simultaneously tracking multiple finger and palm contacts as hands approach, touch, and slide across a proximity-sensing, multi-touch surface. Identification and classification of intuitive hand configurations and motions enables unprecedented integration of typing, resting, pointing, scrolling, 3D manipulation, and handwriting into a versatile, ergonomic computer input device.

35 Claims, 45 Drawing Sheets



U.S. PATENT DOCUMENTS

4,692,809	A		9/1987	Beining et al.
4,695,827	A		9/1987	Beining et al.
4,733,222	A		3/1988	Evans
4,734,685	A		3/1988	Watanabe
4,746,770	A		5/1988	McAvinney
4,771,276	A		9/1988	Parks
4,788,384	A		11/1988	Bruere-Dawson et al.
4,806,846	A		2/1989	Kerber
4,898,555	A		2/1990	Sampson
4,968,877	A		11/1990	McAvinney et al.
5,003,519	A		3/1991	Noirjean
5,017,030	A		5/1991	Crews
5,178,477	A		1/1993	Gambaro
5,189,403	A		2/1993	Franz et al.
5,194,862	A		3/1993	Edwards
5,224,861	A		7/1993	Glass et al.
5,241,308	A		8/1993	Young
5,252,951	A		10/1993	Tannenbaum et al.
5,270,711	A		12/1993	Knapp et al.
5,281,966	A		1/1994	Walsh
5,305,017	A		4/1994	Gerpheide
5,327,161	A		7/1994	Logan et al.
5,345,543	A		9/1994	Capps et al.
5,369,228	A		11/1994	Faust
5,376,948	A		12/1994	Roberts
5,379,057	A		1/1995	Clough et al.
5,398,310	A		3/1995	Tchao et al.
5,442,742	A		8/1995	Greyson et al.
5,463,388	A		10/1995	Boie et al.
5,463,696	A		10/1995	Beernink et al.
5,471,578	A		11/1995	Moran et al.
5,479,528	A		12/1995	Speeter
5,483,261	A		1/1996	Yasutake
5,488,204	A		1/1996	Mead et al.
5,495,077	A		2/1996	Miller et al.
5,513,309	A		4/1996	Meier et al.
5,523,775	A		6/1996	Capps
5,530,455	A		6/1996	Gillick et al.
5,543,590	A		8/1996	Gillespie et al.
5,543,591	A		8/1996	Gillespie et al.
5,548,667	A		8/1996	Tu
5,563,632	A		10/1996	Roberts
5,563,996	A		10/1996	Tchao
5,565,658	A		10/1996	Gerpheide et al.
5,579,036	A		11/1996	Yates, IV
5,581,276	A		12/1996	Cipolla et al.
5,581,484	A		12/1996	Prince
5,581,681	A		12/1996	Tchao et al.
5,583,946	A		12/1996	Gourdol
5,589,856	A		12/1996	Stein et al.
5,590,219	A		12/1996	Gourdol
5,592,566	A		1/1997	Pagallo et al.
5,594,469	A		1/1997	Freeman et al.
5,594,810	A		1/1997	Gourdol
5,596,694	A		1/1997	Capps
5,612,719	A		3/1997	Beernink et al.
5,625,715	A		4/1997	Trew et al.
5,631,805	A		5/1997	Bonsall
5,633,955	A		5/1997	Bozinovic et al.
5,634,102	A		5/1997	Capps
5,636,101	A		6/1997	Bonsall et al.
5,642,108	A		6/1997	Gopher et al.
5,644,657	A		7/1997	Capps et al.
5,666,113	A		9/1997	Logan
5,666,502	A		9/1997	Capps
5,666,552	A		9/1997	Greyson et al.
5,675,361	A		10/1997	Santilli
5,675,362	A		10/1997	Clough et al.
5,677,710	A		10/1997	Thompson-Rohrlich
5,689,253	A		11/1997	Hargreaves et al.
5,710,844	A		1/1998	Capps et al.
5,729,249	A		3/1998	Yasutake
5,729,250	A		3/1998	Bishop et al.
5,730,165	A		3/1998	Philipp
5,736,976	A		4/1998	Cheung
5,741,990	A		4/1998	Davies
5,745,116	A		4/1998	Pisutha-Amond
5,745,716	A		4/1998	Tchao et al.
5,746,818	A		5/1998	Yatake
5,748,184	A	*	5/1998	Shieh 345/173
5,748,269	A		5/1998	Harris et al.
5,764,222	A		6/1998	Shieh
5,767,457	A		6/1998	Gerpheide et al.
5,767,842	A		6/1998	Korth
5,790,104	A		8/1998	Shieh
5,790,107	A		8/1998	Kasser et al.
5,802,516	A		9/1998	Shwarts et al.
5,808,567	A		9/1998	McCloud
5,809,267	A		9/1998	Moran et al.
5,821,690	A		10/1998	Martens et al.
5,821,930	A		10/1998	Hansen
5,823,782	A		10/1998	Marcus et al.
5,825,351	A		10/1998	Tam
5,825,352	A	*	10/1998	Bisset et al. 345/173
5,835,079	A		11/1998	Shieh
5,837,947	A		11/1998	Teterwak
5,854,625	A		12/1998	Frisch et al.
5,856,824	A		1/1999	Shieh
5,877,748	A		3/1999	Redlich
5,880,411	A		3/1999	Gillespie et al.
5,883,619	A		3/1999	Ho et al.
5,886,697	A		3/1999	Naughton et al.
5,896,126	A	*	4/1999	Shieh 345/173
5,898,434	A		4/1999	Small et al.
5,917,476	A		6/1999	Czerniecki
5,920,309	A		7/1999	Bisset et al.
5,923,319	A		7/1999	Bishop et al.
5,933,134	A		8/1999	Shieh
5,943,043	A		8/1999	Furuhata et al.
5,943,044	A		8/1999	Martinelli et al.
5,982,302	A		11/1999	Ure
5,995,106	A		11/1999	Naughton et al.
6,002,389	A		12/1999	Kasser
6,002,808	A		12/1999	Freeman
6,020,881	A		2/2000	Naughton et al.
6,031,524	A		2/2000	Kunert
6,037,882	A		3/2000	Levy
6,050,825	A		4/2000	Nichol et al.
6,052,339	A		4/2000	Frenkel et al.
6,061,177	A		5/2000	Fujimoto
6,072,494	A		6/2000	Nguyen
6,084,576	A		7/2000	Leu et al.
6,097,371	A		8/2000	Siddiqui et al.
6,107,997	A		8/2000	Ure
6,128,003	A		10/2000	Smith et al.
6,131,299	A		10/2000	Raab et al.
6,135,958	A		10/2000	Mikula-Curtis et al.
6,144,380	A		11/2000	Schwarts et al.
6,147,680	A	*	11/2000	Tareev 345/173
6,154,209	A		11/2000	Naughton et al.
6,160,551	A		12/2000	Naughton et al.
6,188,391	B1		2/2001	Seely et al.
6,198,515	B1		3/2001	Cole
6,208,329	B1		3/2001	Ballare
6,222,465	B1		4/2001	Kumar et al.
6,239,790	B1		5/2001	Martinelli et al.
6,243,071	B1		6/2001	Shwarts et al.
6,246,862	B1		6/2001	Grivas et al.
6,249,606	B1		6/2001	Kiraly et al.
6,255,604	B1		7/2001	Tokioka et al.
6,278,443	B1		8/2001	Amro et al.
6,288,707	B1		9/2001	Philipp
6,289,326	B1		9/2001	LaFleur
6,292,178	B1		9/2001	Bernstein et al.

6,323,846	B1	11/2001	Westerman et al.
6,323,849	B1	11/2001	He et al.
6,344,861	B1	2/2002	Naughton et al.
6,347,290	B1	2/2002	Bartlett
6,377,009	B1	4/2002	Philipp
6,380,931	B1	4/2002	Gillespie et al.
6,411,287	B1	6/2002	Scharff et al.
6,414,671	B1	7/2002	Gillespie et al.
6,421,234	B1	7/2002	Ricks et al.
6,452,514	B1	9/2002	Philipp
6,457,355	B1	10/2002	Philipp
6,466,036	B1	10/2002	Philipp
6,515,669	B1	2/2003	Mohri
6,525,749	B1	2/2003	Moran et al.
6,535,200	B2	3/2003	Philipp
6,543,684	B1	4/2003	White et al.
6,543,947	B2	4/2003	Lee
6,570,557	B1	5/2003	Westerman et al.
6,593,916	B1	7/2003	Aroyan
6,610,936	B2	8/2003	Gillespie et al.
6,624,833	B1	9/2003	Kumar et al.
6,639,577	B2	10/2003	Eberhard
6,650,319	B1	11/2003	Hurst et al.
6,658,994	B1	12/2003	McMillan
6,670,894	B2	12/2003	Mehring
6,677,932	B1	1/2004	Westerman
6,677,934	B1	1/2004	Blanchard
6,724,366	B2	4/2004	Crawford
6,757,002	B1	6/2004	Oross et al.
6,803,906	B1	10/2004	Morrison et al.
6,842,672	B1	1/2005	Straub et al.
6,856,259	B1	2/2005	Sharp
6,888,536	B2	5/2005	Westerman et al.
6,900,795	B1	5/2005	Knight, III et al.
6,927,761	B2	8/2005	Badaye et al.
6,942,571	B1	9/2005	McAllister et al.
6,965,375	B1	11/2005	Gettemy et al.
6,972,401	B2	12/2005	Akitt et al.
6,977,666	B1	12/2005	Hedrick
6,985,801	B1	1/2006	Straub et al.
6,992,659	B2	1/2006	Gettemy
7,031,228	B2	4/2006	Born et al.
7,084,859	B1	8/2006	Pryor
7,129,416	B1	10/2006	Steinfeld et al.
7,240,289	B2	7/2007	Naughton et al.
RE40,153	E	3/2008	Westerman et al.
7,339,580	B2	3/2008	Westerman et al.
2002/0118848	A1	8/2002	Karpenstein
2002/0140668	A1	10/2002	Crawford
2003/0006974	A1	1/2003	Clough et al.
2003/0076301	A1	4/2003	Tsuk et al.
2003/0076303	A1	4/2003	Huppi
2003/0076306	A1	4/2003	Zadesky et al.
2003/0095095	A1	5/2003	Pihlaja
2003/0095096	A1	5/2003	Robbin et al.
2003/0098858	A1	5/2003	Perski et al.
2003/0132950	A1	7/2003	Surucu et al.
2003/0164820	A1	9/2003	Kent
2003/0206202	A1	11/2003	Moriya
2003/0234768	A1	12/2003	Rekimoto et al.
2004/0146688	A1	7/2004	Treat
2004/0263484	A1	12/2004	Montysalo et al.
2005/0012723	A1	1/2005	Pallakoff
2005/0052425	A1	3/2005	Zadeski et al.
2005/0104867	A1	5/2005	Westerman et al.
2005/0110768	A1	5/2005	Marriott et al.
2006/0022955	A1	2/2006	Kennedy
2006/0022956	A1	2/2006	Lengeling et al.
2006/0026521	A1	2/2006	Hotelling et al.
2006/0026535	A1	2/2006	Hotelling et al.
2006/0026536	A1	2/2006	Hotelling et al.
2006/0032680	A1	2/2006	Elias et al.
2006/0033724	A1	2/2006	Chaudhri et al.
2006/0053387	A1	3/2006	Ording
2006/0066582	A1	3/2006	Lyon et al.
2006/0085757	A1	4/2006	Andre et al.
2006/0097991	A1	5/2006	Hotelling et al.
2006/0197753	A1	9/2006	Hotelling
2006/0232567	A1	10/2006	Westerman et al.
2006/0238517	A1	10/2006	King et al.
2006/0238518	A1	10/2006	Westerman et al.
2006/0238519	A1	10/2006	Westerman et al.
2006/0238520	A1	10/2006	Westerman et al.
2006/0238521	A1	10/2006	Westerman et al.
2006/0238522	A1	10/2006	Westerman et al.
2007/0070050	A1	3/2007	Westerman et al.
2007/0070051	A1	3/2007	Westerman et al.
2007/0070052	A1	3/2007	Westerman et al.
2007/0078919	A1	4/2007	Westerman et al.
2007/0081726	A1	4/2007	Westerman et al.
2007/0229464	A1	10/2007	Hotelling et al.
2007/0236466	A1	10/2007	Hotelling
2007/0247429	A1	10/2007	Westerman
2007/0257890	A1	11/2007	Hotelling et al.
2007/0268273	A1	11/2007	Westerman et al.
2007/0268274	A1	11/2007	Westerman et al.
2007/0268275	A1	11/2007	Westerman et al.
2008/0041639	A1	2/2008	Westerman et al.
2008/0042986	A1	2/2008	Westerman et al.
2008/0042987	A1	2/2008	Westerman et al.
2008/0042988	A1	2/2008	Westerman et al.
2008/0042989	A1	2/2008	Westerman et al.
2008/0128182	A1	6/2008	Westerman et al.
2009/0021489	A1	1/2009	Westerman et al.
2009/0160816	A1	6/2009	Westerman et al.
2009/0244031	A1	10/2009	Westerman et al.
2009/0244032	A1	10/2009	Westerman et al.
2009/0244033	A1	10/2009	Westerman et al.
2009/0249236	A1	10/2009	Westerman et al.
2009/0251435	A1	10/2009	Westerman et al.
2009/0251438	A1	10/2009	Westerman et al.
2009/0251439	A1	10/2009	Westerman et al.

FOREIGN PATENT DOCUMENTS

DE	10251296	A1	5/2004
EP	0 546 704	A2	6/1993
EP	0 546 704	A3	6/1993
EP	0288692		7/1993
EP	0 622 722	A2	11/1994
EP	0 622 722	A3	11/1994
EP	0 622 722	B1	11/1994
EP	0664504		1/1995
EP	0464908		9/1996
EP	0 817 000	A1	1/1998
EP	0 817 000	B1	1/1998
EP	1014295		1/2002
GB	2330670	A	4/1999
JP	02-056017	A	2/1990
JP	02-304613	A	12/1990
JP	04-233619	A	8/1992
JP	06-083523	A	3/1994
JP	06-332620	A	12/1994
JP	07-057103	A	3/1995
JP	07-129312	A	5/1995
JP	08-286830	A	11/1996
JP	2000-163031	A	6/2000
WO	WO-92/02000		2/1992
WO	WO-96/07981	A1	3/1996
WO	WO-97/18547		5/1997
WO	WO-97/23738		7/1997
WO	WO-97/36225		10/1997
WO	WO-97/40744		11/1997
WO	WO-98/14863		4/1998
WO	WO-99/38149	A1	7/1999
WO	WO-03/088176		10/2003

WO WO2006/023569 3/2006

OTHER PUBLICATIONS

Crowley, J.L. (Mar. 1, 1997). "Vision for Man-Machine Interaction," *Robotics and Autonomous Systems*, 19(3-4):347-358.
Davis, J. et al. (May 2, 1994). "Recognizing Hand Gestures," *European Conference on Computer Vision*, Stockholm, SE, 1:331-340.
European Search Report mailed Dec. 12, 2008, for EP Application No. 06016855.6 filed Jan. 25, 1999, six pages.
European Search Report mailed Dec. 23, 2008, for EP Application No. 06016831.7 filed Jan. 25, 1999, seven pages.
European Search Report mailed Jan. 9, 2009, for EP Application No. 06016832.5 filed Jan. 25, 1999, four pages.
Heap, T. et al. (Oct. 14, 1996). "Towards 3D Hand Tracking Using a Deformable Model," *Proceedings of the 2nd International Conference*, Killington, VT, USA, Oct. 14-16, 1996, *Automatic Face and Gesture Recognition, IEEE Comput. Soc.*, pp. 140-145.
International Search Report mailed Dec. 28, 2005, for PCT Application No. PCT/US05/03325, filed Feb. 3, 2005, two pages.
Nirei, K. et al. (Aug. 5, 1996). "Human Hand Tracking from Binocular Image Sequences," *Proceedings of the 1996 IEEE IECON 22nd International Conference, Taipei, Taiwan, Aug. 5-10, 1996, Industrial Electronics, Control, and Instrumentation* 1(5):297-302.
Non-Final Office Action mailed Sep. 21, 2007, for U.S. Appl. No. 11/428,515, filed Jul. 3, 2006, seven pages.
Non-Final Office Action mailed Sep. 24, 2007, for U.S. Appl. No. 11/428,506, filed Jul. 3, 2006, six pages.
Non-Final Office Action mailed Sep. 24, 2007, for U.S. Appl. No. 11/428,521, filed Jul. 3, 2006, six pages.
Non-Final Office Action mailed Jan. 28, 2008, for U.S. Appl. No. 11/428,522, filed Jul. 3, 2006, seven pages.
Non-Final Office Action mailed Feb. 4, 2008, for U.S. Appl. No. 11/428,515, filed Jul. 3, 2006, six pages.
Non-Final Office Action mailed Jul. 9, 2008, for U.S. Appl. No. 11/428,521, filed Jul. 3, 2006, 11 pages.
Non-Final Office Action mailed Sep. 2, 2008, for U.S. Appl. No. 11/428,522, filed Jul. 3, 2006, six pages.
Non-Final Office Action mailed Sep. 15, 2008, for U.S. Appl. No. 11/428,506, filed Jul. 3, 2006, eight pages.
Non-Final Office Action mailed Oct. 31, 2008, for U.S. Appl. No. 11/428,515, filed Jul. 3, 2006, seven pages.
Non-Final Office Action mailed Feb. 17, 2009, for U.S. Appl. No. 11/428,522, filed Jul. 3, 2006, six pages.
Notice of Allowability mailed Feb. 11, 2009, for U.S. Appl. No. 11/428,521, filed Jul. 3, 2006, five pages.
U.S. Appl. No. 11/332,861, filed Jan. 13, 2006 which is a Reissue of 6,677,932.
Westerman, W. (Jan. 1, 1999). "Hand Tracking, Finger Identification, and Chordic Manipulation on a Multi-Touch Surface," Dissertation, University of Delaware, pp. 1-333.
Final Office Action mailed Mar. 19, 2009, for U.S. Appl. No. 11/428,506, filed Jul. 3, 2006, seven pages.
Non-Final Office Action mailed Mar. 18, 2009, for U.S. Appl. No. 11/428,515, filed Jul. 3, 2006, 12 pages.
Non-Final Office Action mailed Apr. 2, 2009, for U.S. Appl. No. 11/428,501, filed Jul. 3, 2006, 11 pages.
Non-Final Office Action mailed Apr. 2, 2009, for U.S. Appl. No. 11/428,503, filed Jul. 3, 2006, 12 pages.
Notice of Allowability mailed Jul. 10, 2009, for U.S. Appl. No. 11/428,521, filed Jul. 3, 2006, four pages.
Rubine, D. (May 1992). "Combining Gestures and Direct Manipulation," CHI '92, pp. 659-660.
U.S. Appl. No. 12/342,027, filed Dec. 22, 2008, by Westerman et al.
U.S. Appl. No. 12/422,197, filed Apr. 10, 2009, by Westerman et al.
U.S. Appl. No. 12/422,205, filed Apr. 10, 2009, by Westerman et al.
U.S. Appl. No. 12/422,212, filed Apr. 10, 2009, by Westerman et al.
U.S. Appl. No. 12/422,222, filed Apr. 10, 2009, by Westerman et al.
U.S. Appl. No. 12/422,225, filed Apr. 10, 2009, by Westerman et al.
U.S. Appl. No. 12/434,439, filed May 1, 2009, by Westerman et al.
U.S. Appl. No. 12/479,573, filed Jun. 5, 2009, by Westerman et al.
U.S. Appl. No. 12/479,617, filed Jun. 5, 2009, by Westerman et al.
U.S. Appl. No. 12/479,678, filed Jun. 5, 2009, by Westerman et al.
Cui, Y. et al. (1996). "Hand Segmentation Using Learning-Based Prediction and Verification for Hand Sign Recognition," *Proceedings of the 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 88-93.
Final Office Action mailed Nov. 19, 2009, for U.S. Appl. No. 11/428,515, filed Jul. 3, 2006, 14 pages.
Non-Final Office Action mailed Aug. 25, 2009, for U.S. Appl. No. 11/428,522, filed Jul. 3, 2006, six pages.
Non-Final Office Action mailed Sep. 2, 2009, for U.S. Appl. No. 11/559,736, filed Nov. 14, 2006, 12 pages.
Non-Final Office Action mailed Oct. 5, 2009, for U.S. Appl. No. 11/559,799, filed Nov. 14, 2006, 14 pages.
Non-Final Office Action mailed Oct. 6, 2009, for U.S. Appl. No. 11/559,763, filed Nov. 14, 2006, 24 pages.
Non-Final Office Action mailed Oct. 14, 2009, for U.S. Appl. No. 11/428,501, filed Jul. 3, 2006, six pages.
Non-Final Office Action mailed Oct. 29, 2009, for U.S. Appl. No. 11/559,822, filed Nov. 14, 2006, 11 pages.
Non-Final Office Action mailed Oct. 30, 2009, for U.S. Appl. No. 11/428,503, filed Jul. 3, 2006, nine pages.
Non-Final Office Action mailed Dec. 22, 2009, for U.S. Appl. No. 11/559,833, filed Nov. 14, 2006, six pages.
Non-Final Office Action mailed Jan. 27, 2010, for U.S. Appl. No. 11/428,522, filed Jul. 3, 2006, five pages.
Notice of Allowability mailed Sep. 2, 2009, for U.S. Appl. No. 11/428,506, filed Jul. 3, 2006, five pages.
Rubine, D.H. (Dec. 1991). "The Automatic Recognition of Gestures," CMU-CS-91-202, Submitted in Partial Fulfillment of the Requirements of the Degree of Doctor of Philosophy in Computer Science at Carnegie Mellon University, 285 pages.
U.S. Appl. No. 10/654,108, filed Sep. 2, 2003 entitled "Ambidextrous Mouse".
U.S. Appl. No. 10/789,676, filed Feb. 27, 2004 entitled "Shape Detecting Input Device".
"4-Wire Resistive Touchscreens" obtained from http://www.touchscreens.com/intro-touchtypes-4resistive.html generated Aug. 5, 2005.
"5-Wire Resistive Touchscreens" obtained from http://www.touchscreens.com/intro-touchtypes-resistive.html generated Aug. 5, 2005.
"A Brief Overview of Gesture Recognition" obtained from http://www.dai.ed.ac.uk/Cvonline/LOCA_COPIES/COHEN/gesture_overview.html, generated Apr. 20, 2004.
"Capacitive Touchscreens" obtained from http://www.touchscreens.com/intro-touchtypes-capacitive.html generated Aug. 5, 2005.
"Capacitive Position Sensing" obtained from http://www.synaptics.com/technology/cps.cfm generated Aug. 5, 2005.
"Comparing Touch Technologies" obtained from http://www.touchscreens.com/intro-touchtypes.html generated Oct. 10, 2004.
"GlidePoint®" obtained from http://www.cirque.com/technology/technology_gp.html generated Aug. 5, 2005.
"How do touchscreen monitors know where you're touching?" obtained from http://www.electronics.howstuffworks.com/question716.html generated Aug. 5, 2005.
"How does a touchscreen work?" obtained from http://www.touchscreens.com/intro-anatomy.html generated Aug. 5, 2005.
"Infrared Touchscreens" obtained from http://www.touchscreens.com/intro-touchtypes-infrared.html generated Aug. 5, 2005.
"Mouse Emulation" FingerWorks obtained from http://www.fingerworks.com/gesture_guide_mouse.html generated Aug. 30, 2005.
"Mouse Gestures in Opera" obtained from http://www.opera.com/products/desktop/mouse/index.dml generated Aug. 30, 2005.
"Mouse Gestures," Optim oz, May 21, 2004.
"MultiTouch Overview" FingerWorks obtained from http://www.fingerworks.com/multoverview.html generated Aug. 30, 2005.
"Near Field Imaging Touchscreens" obtained from http://www.touchscreens.com/intro-touchtypes-nfi.html generated Aug. 5, 2005.
"PenTouch Capacitive Touchscreens" obtained from http://www.touchscreens.com/intro-touchtypes-pentouch.html generated Aug. 5, 2005.

"Surface Acoustic Wave Touchscreens" obtained from http://www.touchscreens.com/intro-touchtypes-saw.html generated Aug. 5, 2005.
"Symbol Commander" obtained from http://www.sensiva.com/symbolcomander/, generated Aug. 30, 2005.
"Tips for Typing" FingerWorks http://www.fingerworks.com/mini_typing.html generated Aug. 30, 2005.
"Touch Technologies Overview" 2001, 3M Touch Systems, Massachusetts.
"Wacom Components—Technology" obtained from http://www.wacom-components.com/english/tech.asp generated on Oct. 10, 2004.
"Watershed Algorithm" http://rsb.info.nih.gov/ij/plugins/watershed.html generated Aug. 5, 2005.
"FingerWorks—Gesture Guide—Application Switching," obtained from http://www.fingerworks.com/gesture_guide_apps.html, generated on Aug. 27, 2004, 1-pg.
"FingerWorks—Gesture Guide—Editing," obtained from http://www.fingerworks.com/gesure_guide_editing.html, generated on Aug. 27, 2004, 1-pg.
"FingerWorks—Gesture Guide—File Operations," obtained from http://www.fingerworks.com/gesture_guide_files.html, generated on Aug. 27, 2004, 1-pg.
"FingerWorks—Gesture Guide—Text Manipulation," obtained from http://www.fingerworks.com/gesture_guide_text_manip.html, generated on Aug. 27, 2004, 2-pg.
"FingerWorks—Gesture Guide—Tips and Tricks," obtained from http://www.fingerworks.com/gesture_guide_tips.html, generated Aug. 27, 2004, 2-pgs.
"FingerWorks—Gesture Guide—Web," obtained from http://www.fingerworks.com/gesture_guide_web.html, generated on Aug. 27, 2004, 1-pg.
"FingerWorks—Guide to Hand Gestures for USB Touchpads," obtained from http://www.fingerworks.com/igesture_userguide.html, generated Aug. 27, 2004, 1-pg.
"FingerWorks—iGesture—Technical Details," obtained from http://www.fingerworks.com/igesture_tech.html, generated Aug. 27, 2004, 1-pg.
"FingerWorks—The Only Touchpads with Ergonomic Full-Hand Resting and Relaxation!" obtained from http://www.fingerworks.com/resting.html, Copyright 2001, 1-pg.
"FingerWorks—Tips for Typing on the Mini," obtained from http://www.fingerworks.com/mini_typing.html, generated on Aug. 27, 2004, 2-pgs.
"iGesture Pad—the MultiFinger USB TouchPad with Whole-Hand Gestures," obtained from http://www.fingerworks.com/igesture.html, generated Aug. 27, 2004, 2-pgs.
Bier, et al., "Toolglass and Magic Lenses: The see-through interface" In James Kijiya, editor, Computer Graphics (SIGGRAPH '93 Proceedings), vol. 27, pp. 73-80, Aug. 1993.
Buxton, W. et al. (Jul. 22, 1985). "Issues and Techniques in Touch-Sensitive Tablet Input," *Proceedings ACM Siggraph*, pp. 215-224.
Cutler, L.D. et al. (Apr. 27, 1997). "Two-Handed Direct Manipulation on the Responsive Workbench," *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, Providence, RI, Apr. 27-30, 1997, pp. 107-114.
Davis, J. et al. (Oct. 31, 1994). "Determining 3-D Hand Motion," *Signals, Systems and Computers, 1994 Conference Record of the 28th Asilomar Conference on Pacific Grove*, CA, Oct. 31-Nov. 2, 1994, Los Alamitos, CA, pp. 1262-1266.
Douglas et al., The Ergonomics of Computer Pointing Devices (1997).
European Search Report received in EP 1 621 989 (@ *Beyer Weaver & Thomas, LLP*) dated Mar. 27, 2006.
EVB Elektronik "TSOP6238 IR Receiver Modules for Infrared Remote Control Systems" dated Jan. 2004 1-pg.
Fisher et al., "Repetitive Motion Disorders: The Design of Optimal Rate—Rest Profiles," Human Factors, 35(2):283-304 (Jun. 1993).
Fukumoto, et al., "ActiveClick: Tactile Feedback for Touch Panels," In CHI 2001 Summary, pp. 121-122, 2001.
Fukumoto and Yoshinobu Tonomura, "Body Coupled Fingering: Wireless Wearable Keyboard," *CHI 97*, pp. 147-154 (Mar. 1997).
Hardy, "Fingerworks" Mar. 7, 2002; BBC World On Line.
Hillier and Gerald J. Lieberman, *Introduction to Operations Research* (1986).
Jacob et al., "Integrality and Separability of Input Devices," *ACM Transactions on Computer-Human Interaction*, 1:3-26 (Mar. 1994).
Kinkley et al., "Touch-Sensing Input Devices," in CHI '99 Proceedings, pp. 223-230, 1999.
KIONX "KXP84 Series Summary Data Sheet" copyright 2005,dated Oct. 21, 2005, 4-pgs.
Lee et al., "A Multi-Touch Three Dimensional Touch-Sensitive Tablet," in CHI '85 Proceedings, pp. 121-128, 2000.
Lee, "A Fast Multiple-Touch-Sensitive Input Device," Master's Thesis, University of Toronto (1984).
Lee, S.K. et al. (Apr. 1985). "A Multi-Touch Three Dimensional Touch-Sensitive Tablet," *Proceedings of CHI: ACM Conference on Human Factors in Computing Systems*, pp. 21-25.
Matsushita et al., "HoloWall: Designing a Finger, Hand, Body and Object Sensitive Wall," In Proceedings of UIST '97, Oct. 1997.
Pavlovic, V.I. et al. (Jul. 1997). "Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19(7):677-695.
Quantum Research Group "QT510 / QWheel™ Touch Slider IC" copyright 2004-2005, 14-pgs.
Quek, "Unencumbered Gestural Interaction," *IEEE Multimedia*, 3:36-47 (Winter 1996).
Radwin, "Activation Force and Travel Effects on Overexertion in Repetitive Key Tapping," *Human Factors*, 39(1):130-140 (Mar. 1997).
Rekimoto "SmartSkin: An Infrastructure for Freehand Manipulation on Interactive Surfaces" CHI 2002, Apr. 20-25, 2002.
Rekimoto et al., "ToolStone: Effective Use of the Physical Manipulation Vocabularies of Input Devices," In Proc. of UIST 2000, 2000.
Rubine et al., "Programmable Finger-Tracking Instrument Controllers," *Computer Music Journal*, vol. 14, No. 1 (Spring 1990).
Rutledge et al., "Force-To-Motion Functions for Pointing," Human-Computer Interaction—INTERACT (1990).
Subatai Ahmad, "A Usable Real-Time 3D Hand Tracker," Proceedings of the 28th Asilomar Conference on Signals, Systems and Computers—Part 2 (of2), vol. 2 (Oct. 1994).
Texas Instruments "TSC2003 / I2C Touch Screen Controller" Data Sheet SBAS 162, dated Oct. 2001, 20-pgs.
Wellner, "The Digital Desk Calculators: Tangible Manipulation on a Desk Top Display" In ACM UIST '91 Proceedings, pp. 27-34, Nov. 1991.
Williams, "Applications for a Switched-Capacitor Instrumentation Building Block" Linear Technology Application Note 3, Jul. 1985, pp. 1-16.
Yamada et al., "A Switched-Capacitor Interface for Capacitive Pressure Sensors" IEEE Transactions on Instrumentation and Measurement, vol. 41, No. 1, Feb. 1992, pp. 81-86.
Yeh et al., "Switched Capacitor Interface Circuit for Capacitive Transducers" 1985 IEEE.
Zhai et al., "Dual Stream Input for Pointing and Scrolling," *Proceedings of CHI '97 Extended Abstracts* (1997).
Zimmerman et al., "Applying Electric Field Sensing to Human-Computer Interfaces," In CHI '85 Proceedings, pp. 280-287, 1995.
International Search Report received in corresponding PCT application No. PCT/US2006/008349 dated Oct. 6, 2006.
International search report for PCT/US99/01454, mailed May 14, 1999.
European supplementary search report for European Patent Application No. 99904228.6, mailed Feb. 16, 2005.
Supplementary European search report for U.S. Appl. No. 99904228.6, mailed Jul. 8, 2005.
European examination report for European patent Application No. 99904228.6, mailed Apr. 20, 2006.
European examination report for European Patent Application No. 99904228.6, mailed Mar. 23, 2007.
Extended European search report for European Patent Application No. 06016858.0, mailed Dec. 21, 2007.
Extended European search report for European Patent Application No. 06016856.4, mailed Mar. 14, 2008.
European examination report for European Patent Application No. 06016830.9, mailed Aug. 6, 2008.

European search report for European Patent Application No. 06016830.9 mailed Dec. 3, 2007.
European examination report for European Patent Application No. 06016856.4 mailed Sep. 16, 2008.
Notice of Allowance mailed Mar. 23, 2010, for U.S. Appl. No. 11/428,501, filed Jul. 3, 2006, eight pages.
European Search Report mailed Feb. 17, 2010, for EP Application No. 06016857.2, six pages.
European Search Report mailed Feb. 24, 2010, for EP Application No. 06016833.3, nine pages.
Sun Microsystems. (1992). "The Star7 PDA Prototype," located at <http://www.youtube.com/watch?v=Ahg8OBYixL0, last visited Jan. 15, 2010, seven pages.
Tomita, A. Jr. et al. (Nov. 9, 1997). "An Image Processing Method for Use in a GUI for the Visually Impaired," *Proceedings of the IECON '97: 23rd International Conference on Industrial Electronics, Control and Instrumentation*, New Orleans, LA, Nov. 9-14, 1997, pp. 264-268.

* cited by examiner

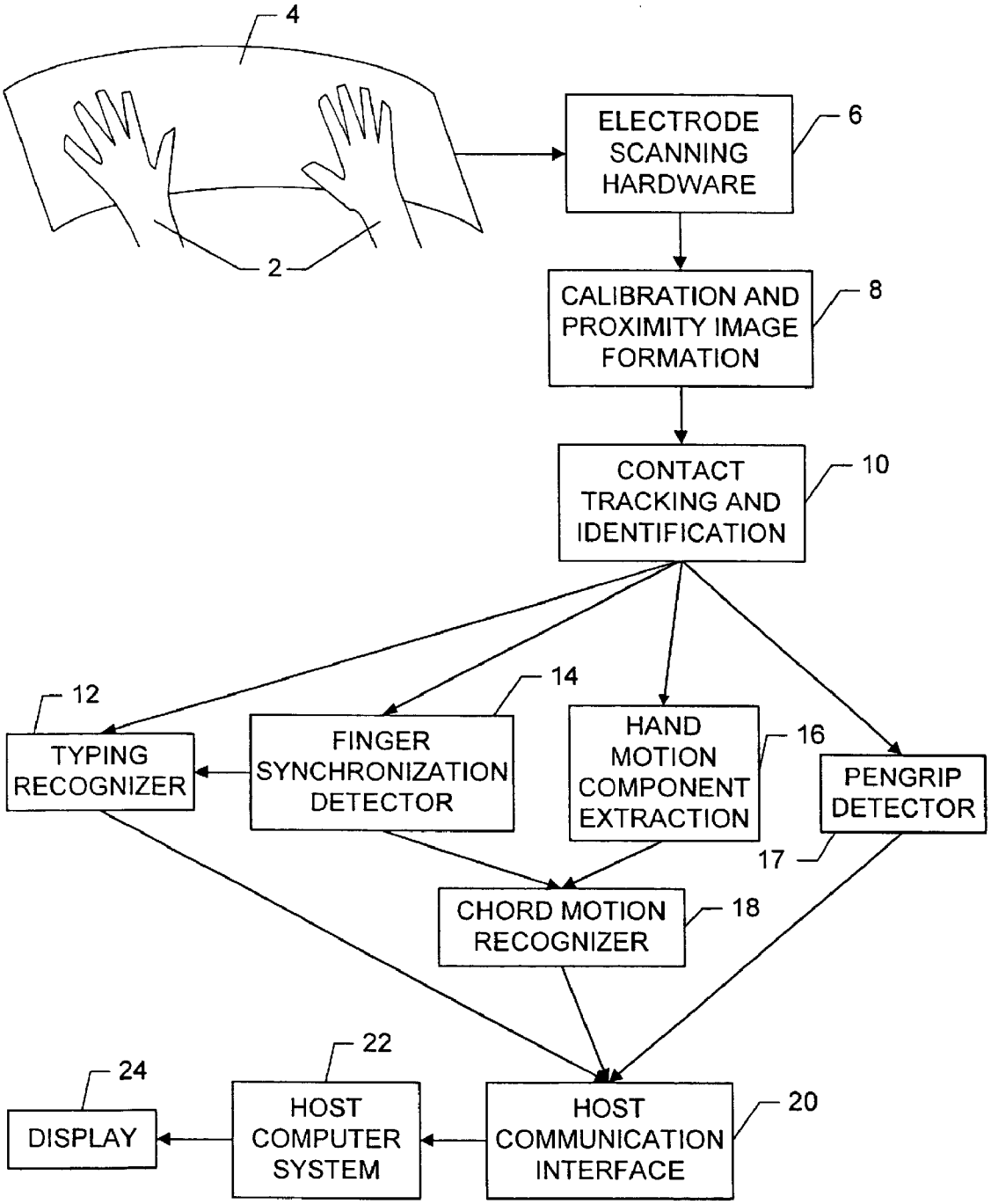

4
2
ELECTRODE SCANNING HARDWARE
6
CALIBRATION AND PROXIMITY IMAGE FORMATION
8
CONTACT TRACKING AND IDENTIFICATION
10
12
TYPING RECOGNIZER
FINGER SYNCHRONIZATION DETECTOR
14
HAND MOTION COMPONENT EXTRACTION
16
PENGRIP DETECTOR
17
CHORD MOTION RECOGNIZER
18
22
24
DISPLAY
HOST COMPUTER SYSTEM
HOST COMMUNICATION INTERFACE
20

FIG. 1

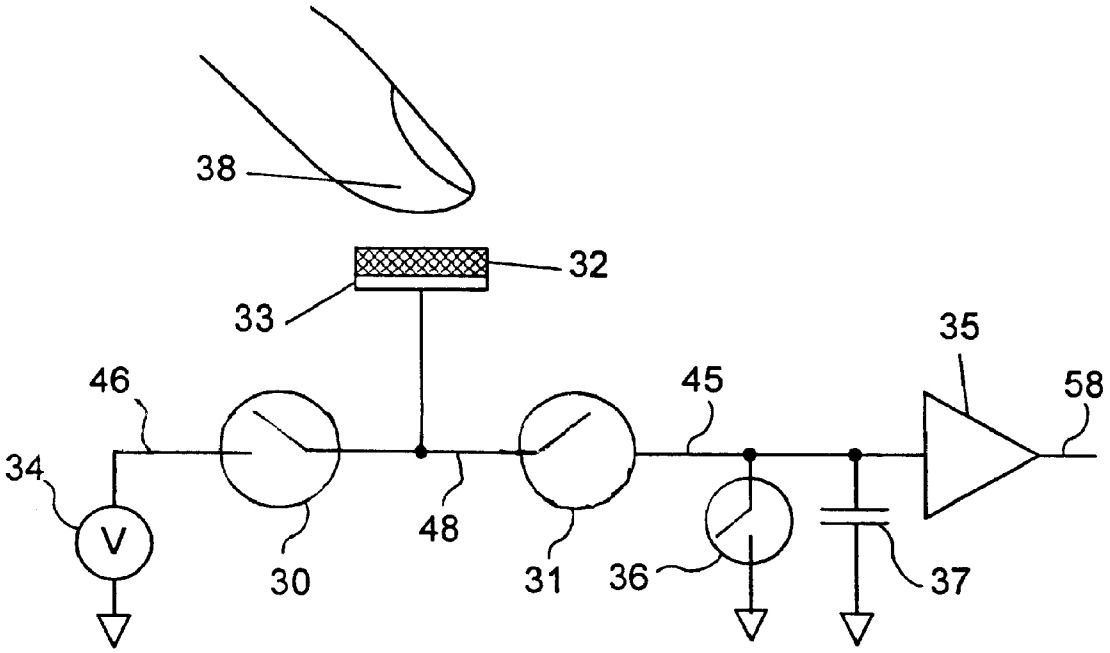


FIG. 2

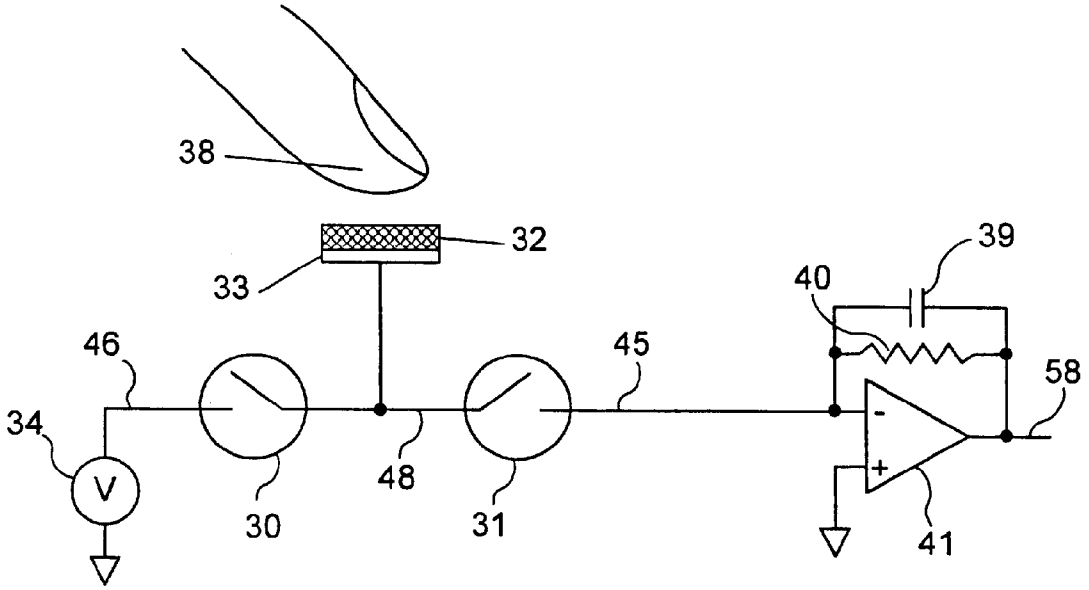

38
32
33
39
40
46
45
58
34
V
48
30
31
41

FIG. 3A

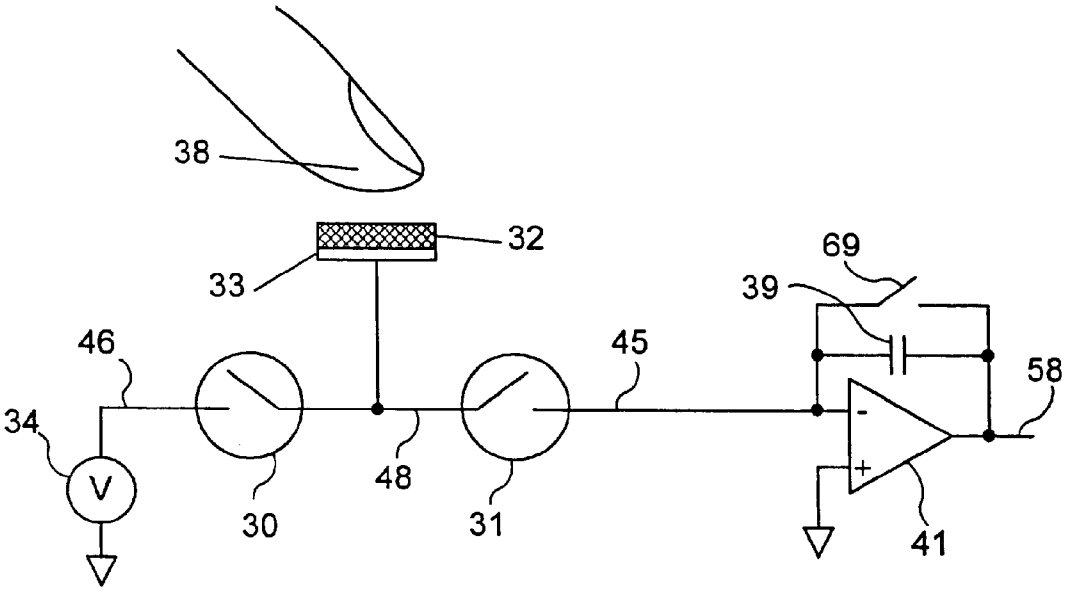

38
32
69
33
39
46
45
58
34
V
48
30
31
41

FIG. 3B

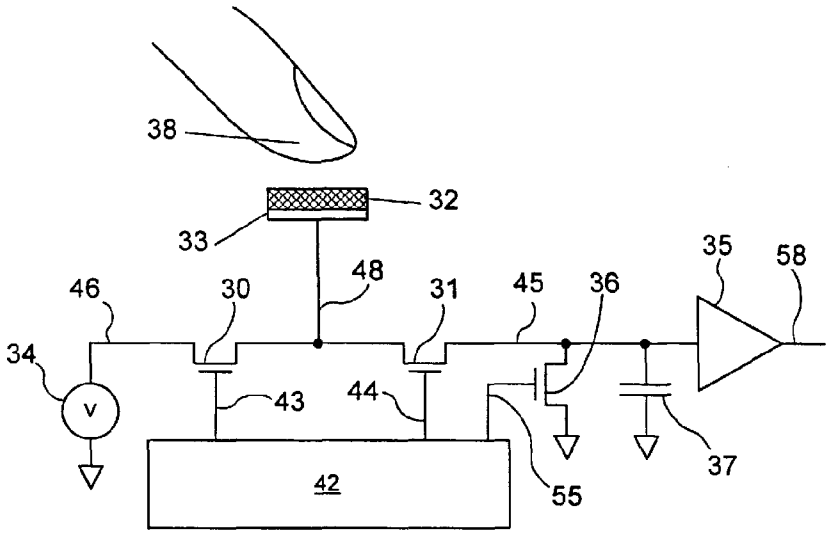


FIG. 4A

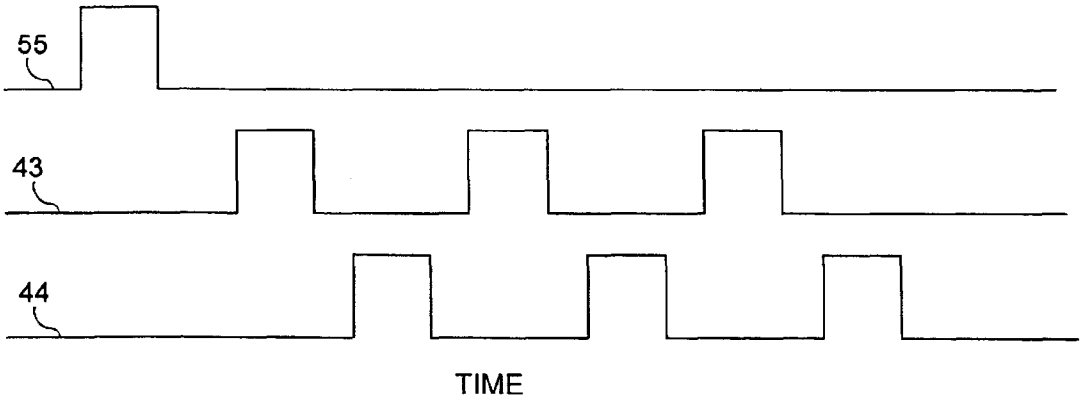


FIG. 4B

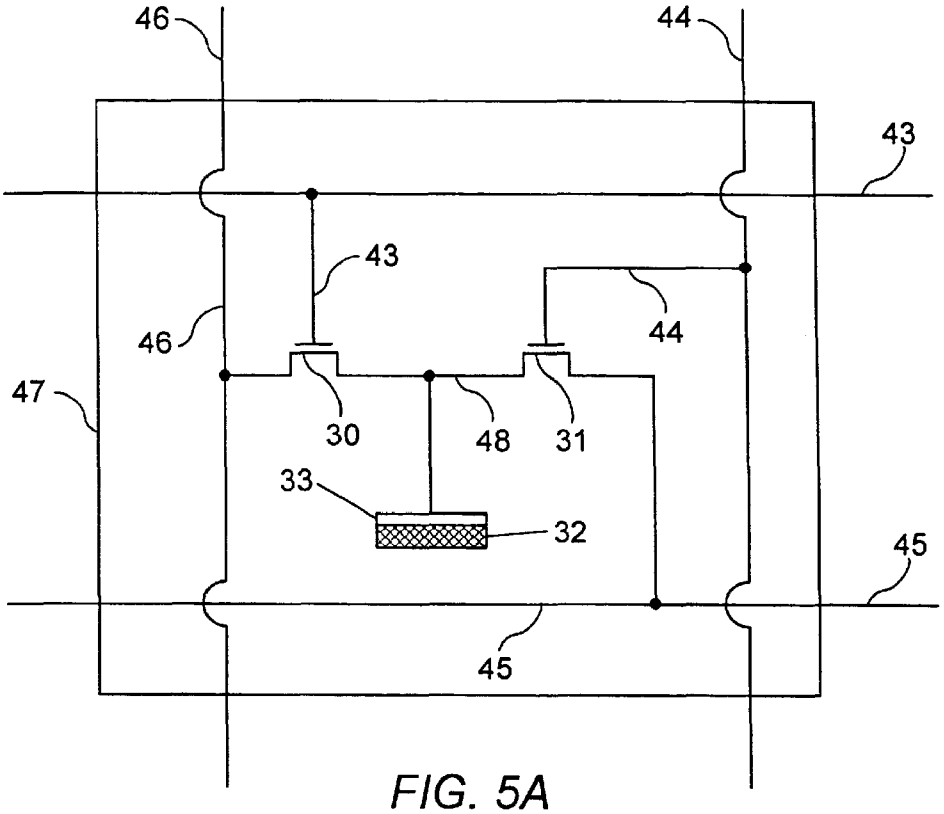


FIG. 5A

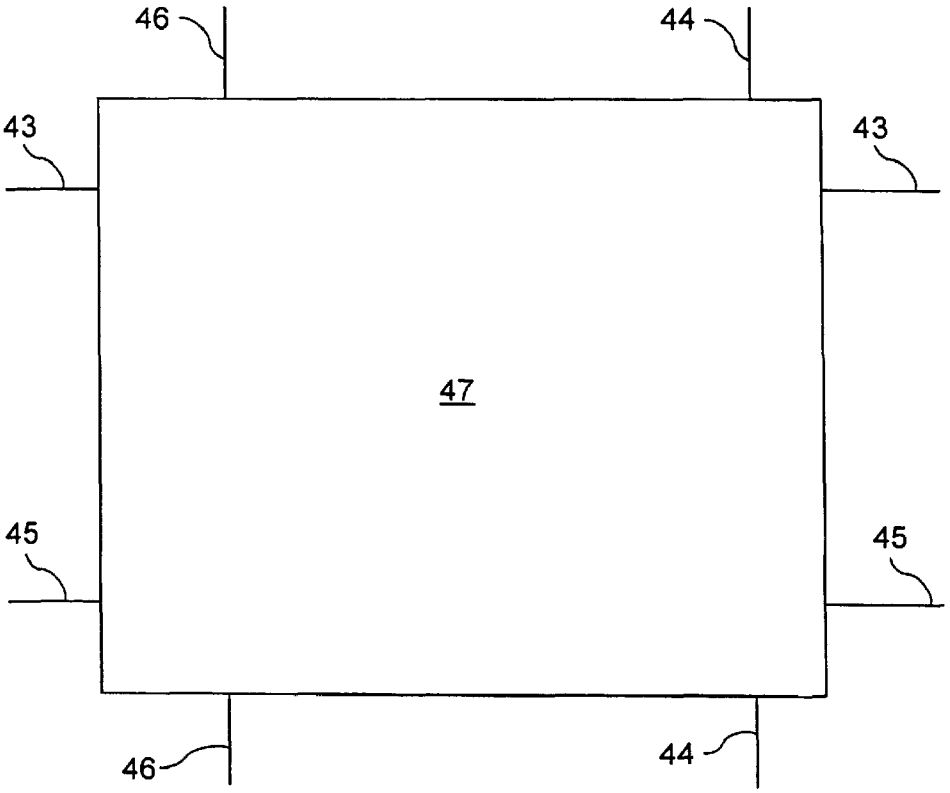


FIG. 5B

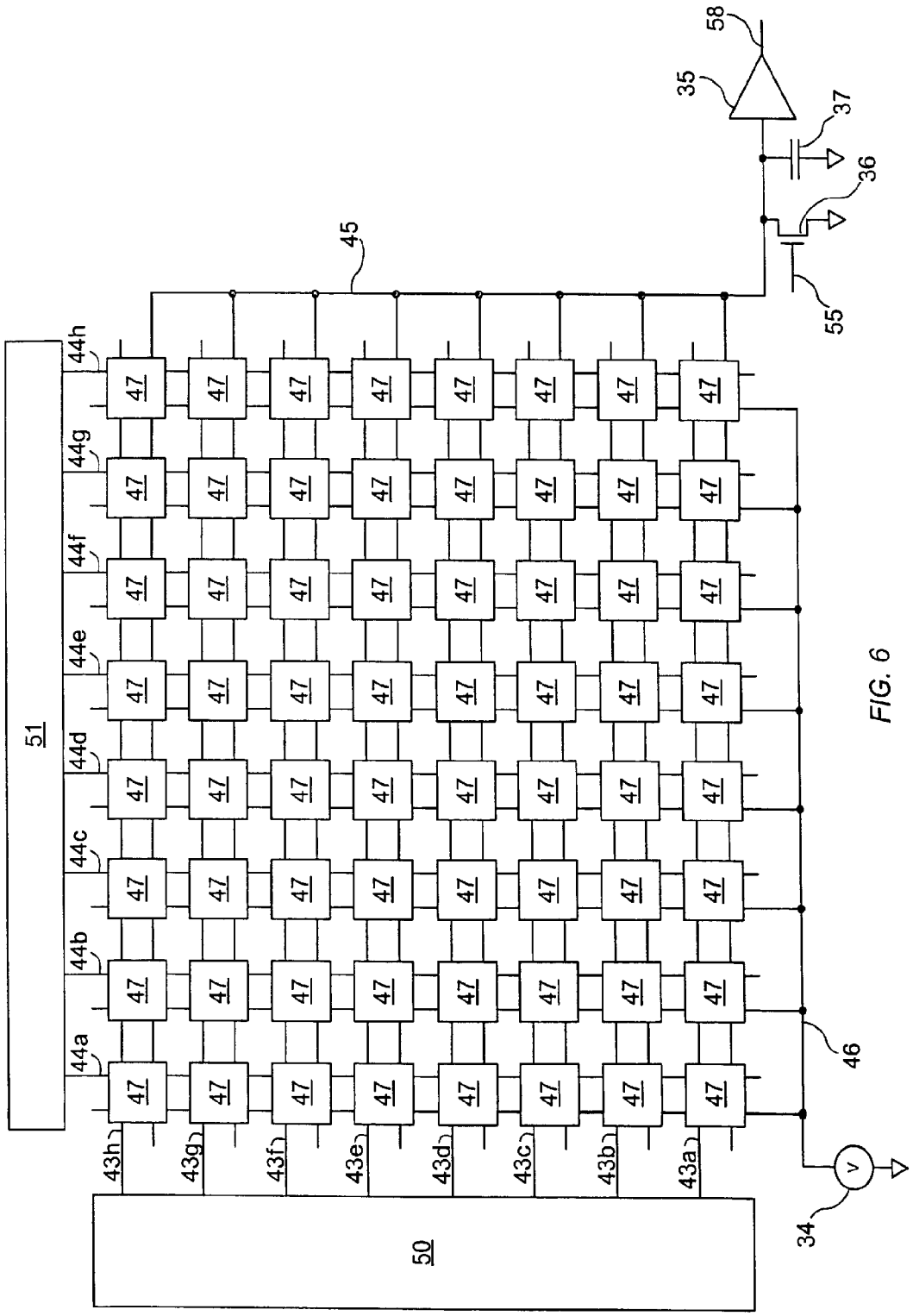

58
35
37
36
45
55
51
44h
44g
44f
44e
44d
44c
44b
44a
47
46
43h
43g
43f
43e
43d
43c
43b
43a
50
34
V

FIG. 6

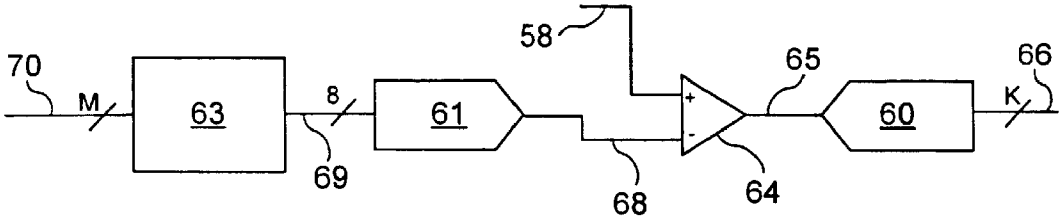

58
70
M
63
8
61
69
68
64
65
60
66
K

FIG. 7A

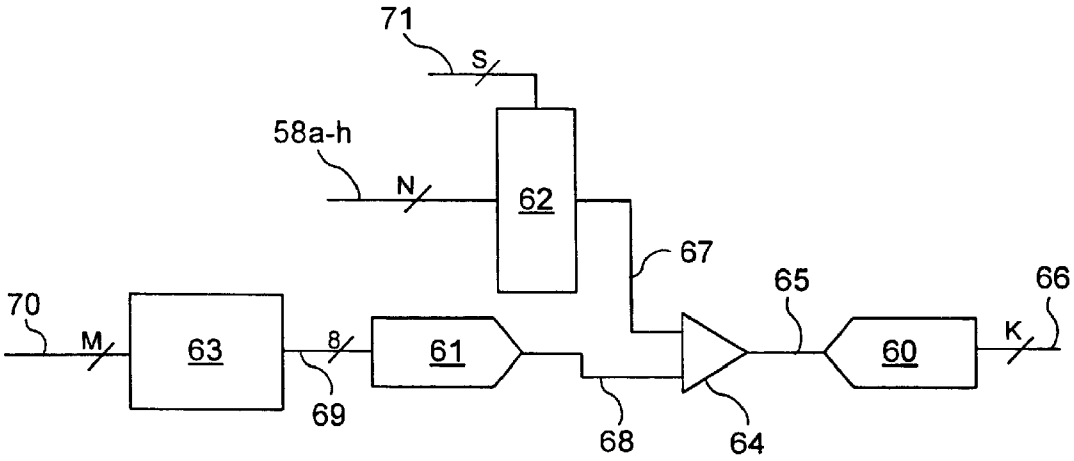

71
S
58a-h
N
62
67
70
M
63
8
61
69
68
64
65
60
66
K

FIG. 7B

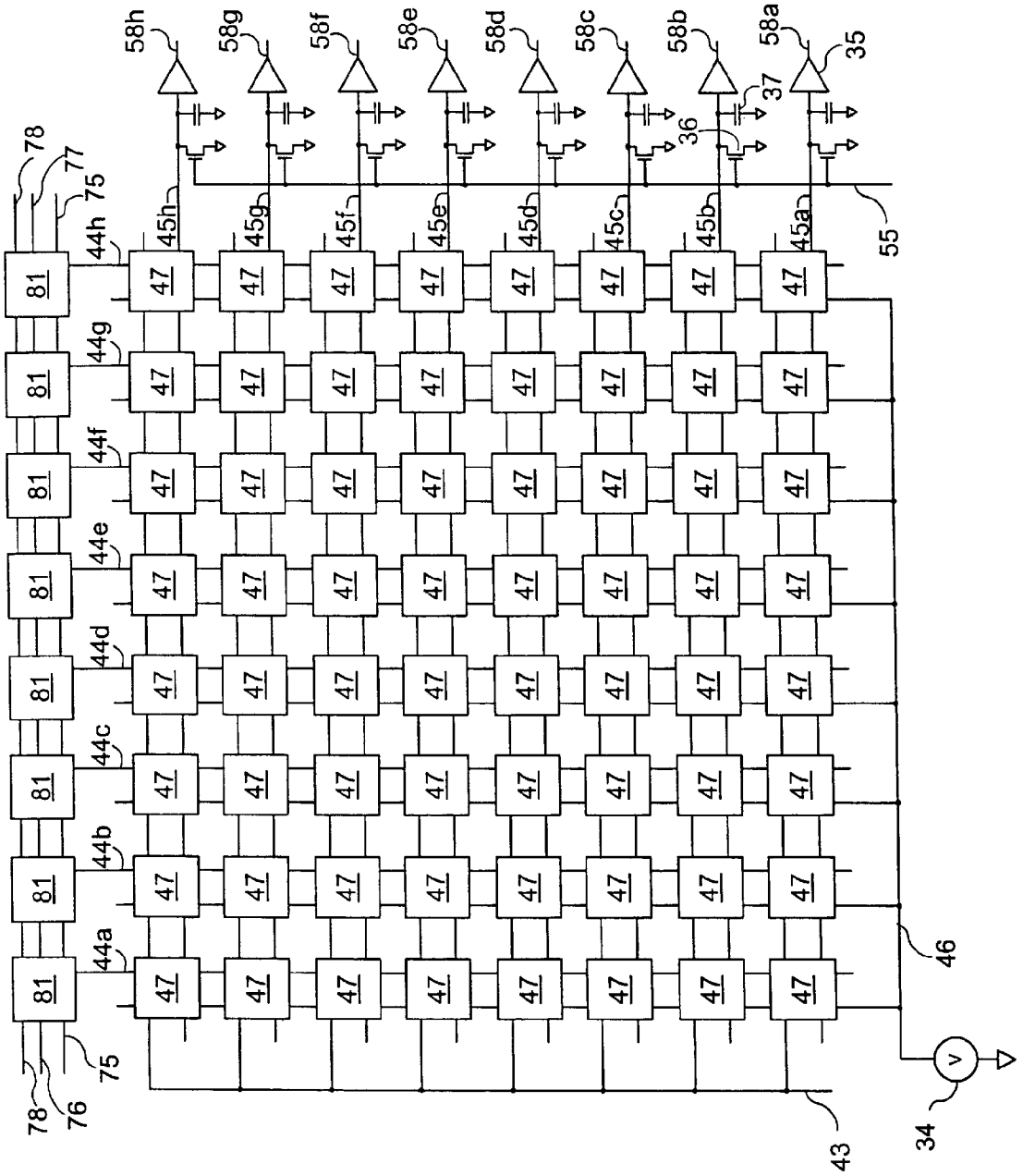

58h
58g
58f
58e
58d
58c
58b
58a
35
37
36
55
78
77
75
44h
44g
44f
44e
44d
44c
44b
44a
45h
45g
45f
45e
45d
45c
45b
45a
81
47
46
43
34
76

FIG. 8

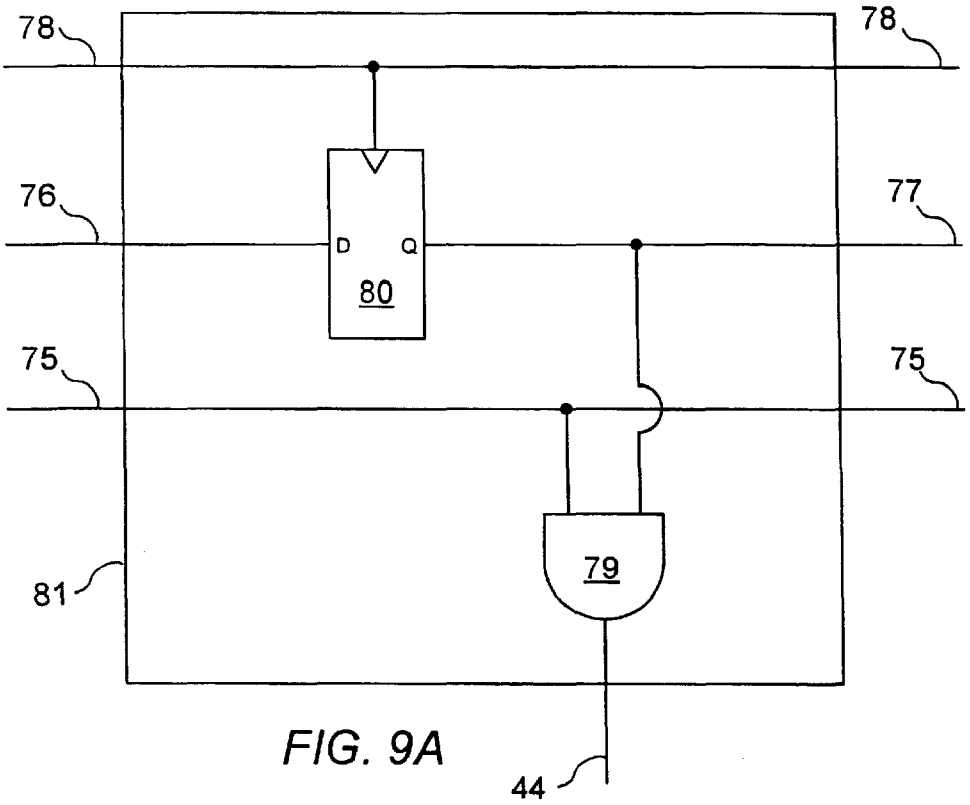


FIG. 9A

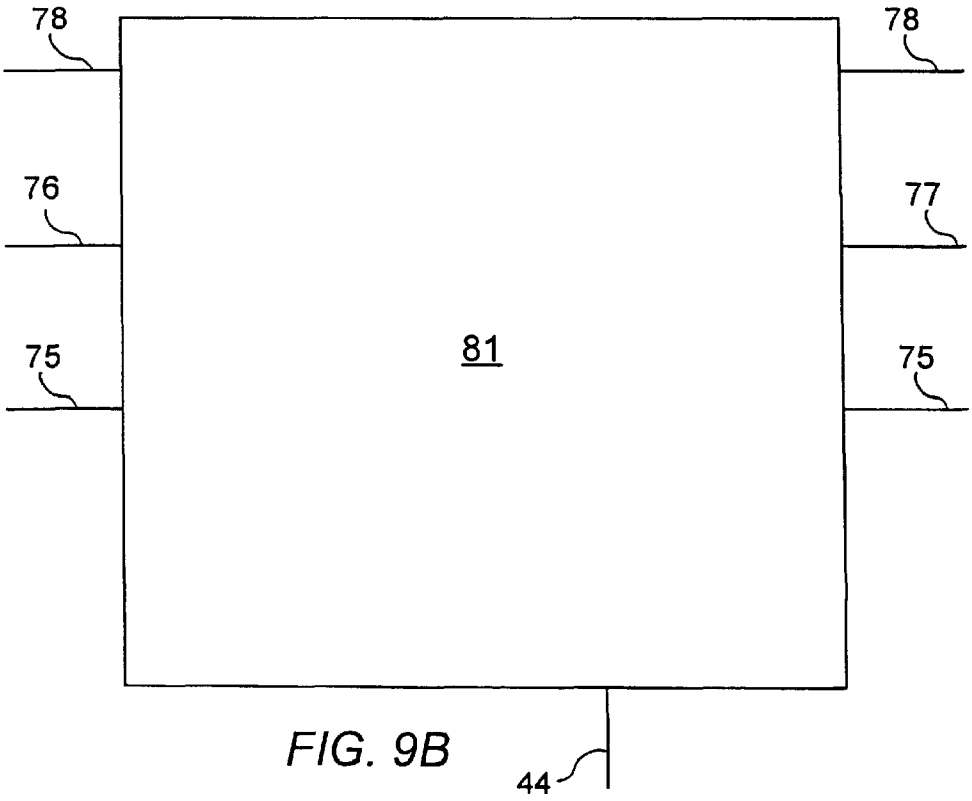


FIG. 9B

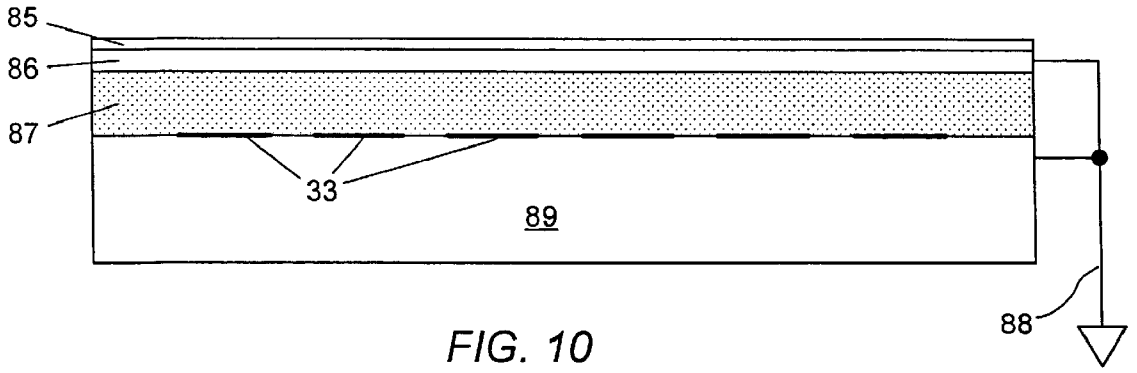


FIG. 10

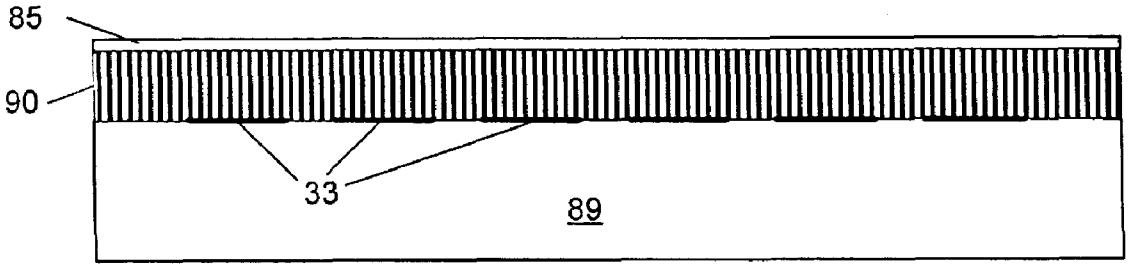


FIG. 11

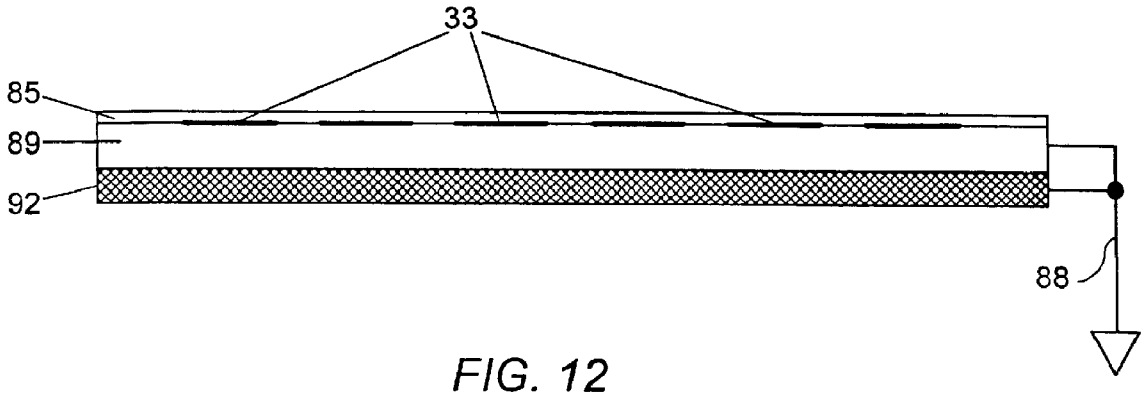


FIG. 12

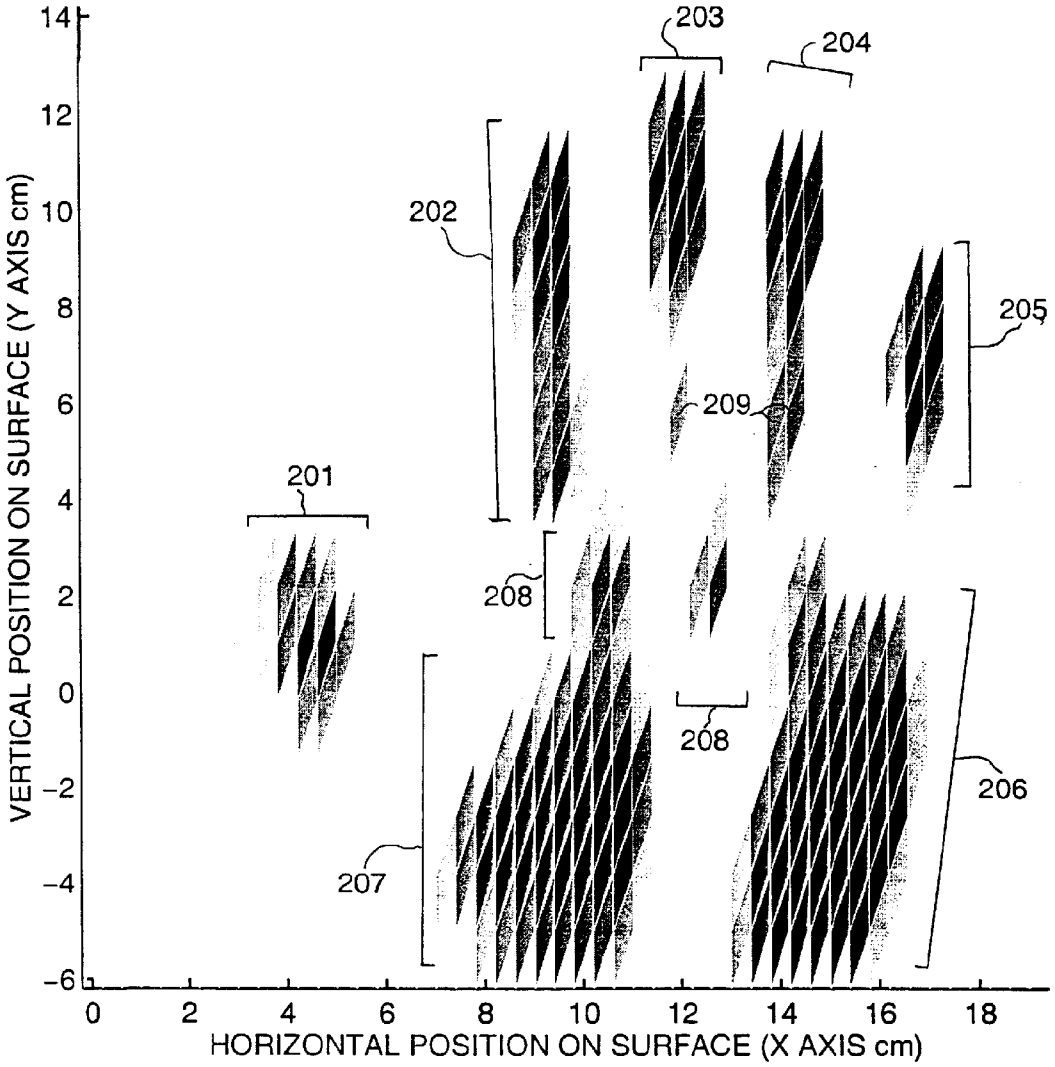

VERTICAL POSITION ON SURFACE (Y AXIS cm)
HORIZONTAL POSITION ON SURFACE (X AXIS cm)
14
12
10
8
6
4
2
0
−2
−4
−6
0
2
4
6
8
10
12
14
16
18
201
202
203
204
205
206
207
208
208
209

FIG. 13

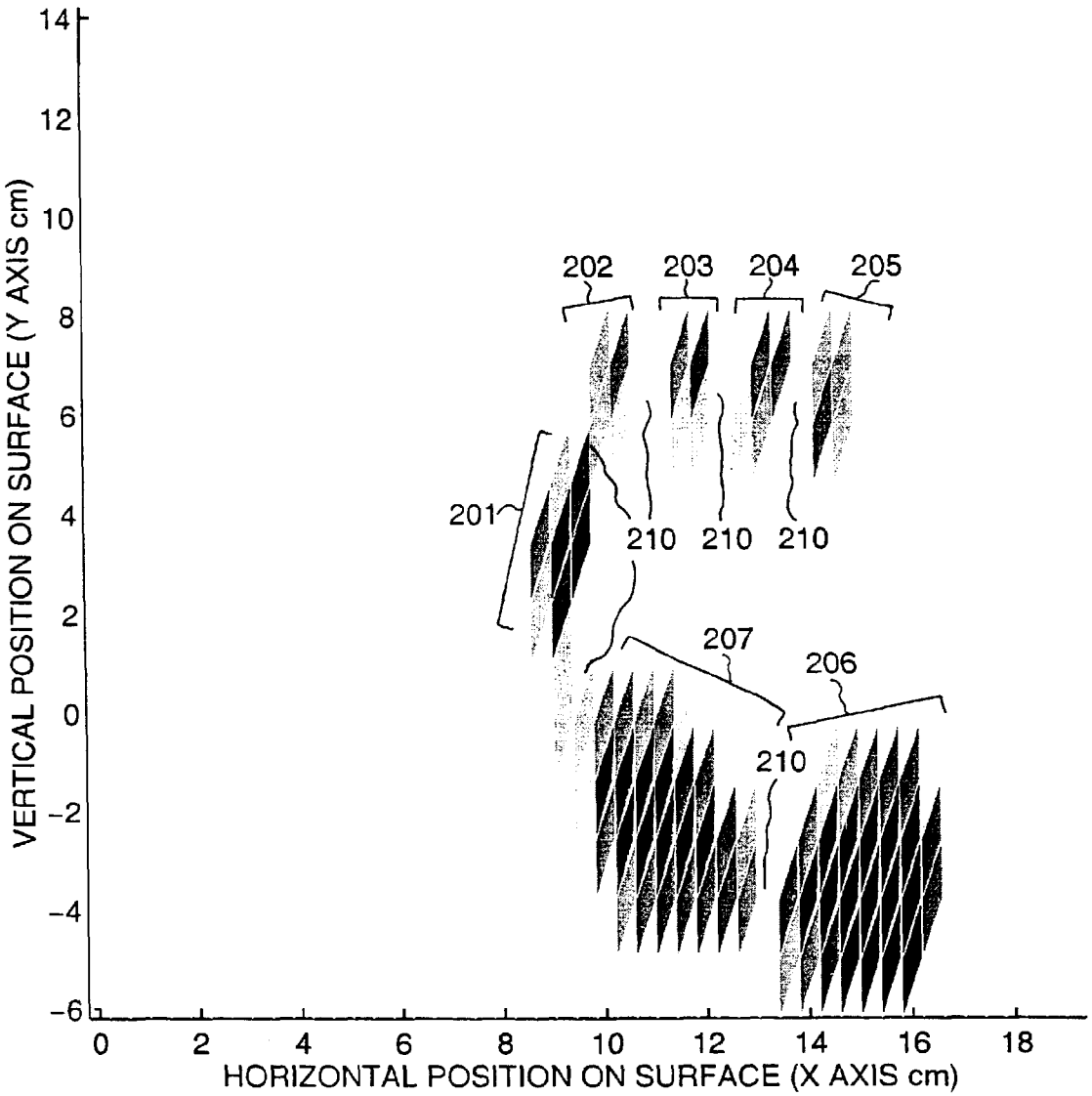

VERTICAL POSITION ON SURFACE (Y AXIS cm)
14
12
10
8
6
4
2
0
-2
-4
-6
0
2
4
6
8
10
12
14
16
18
HORIZONTAL POSITION ON SURFACE (X AXIS cm)
202
203
204
205
201
210
210
210
207
206
210

FIG. 14

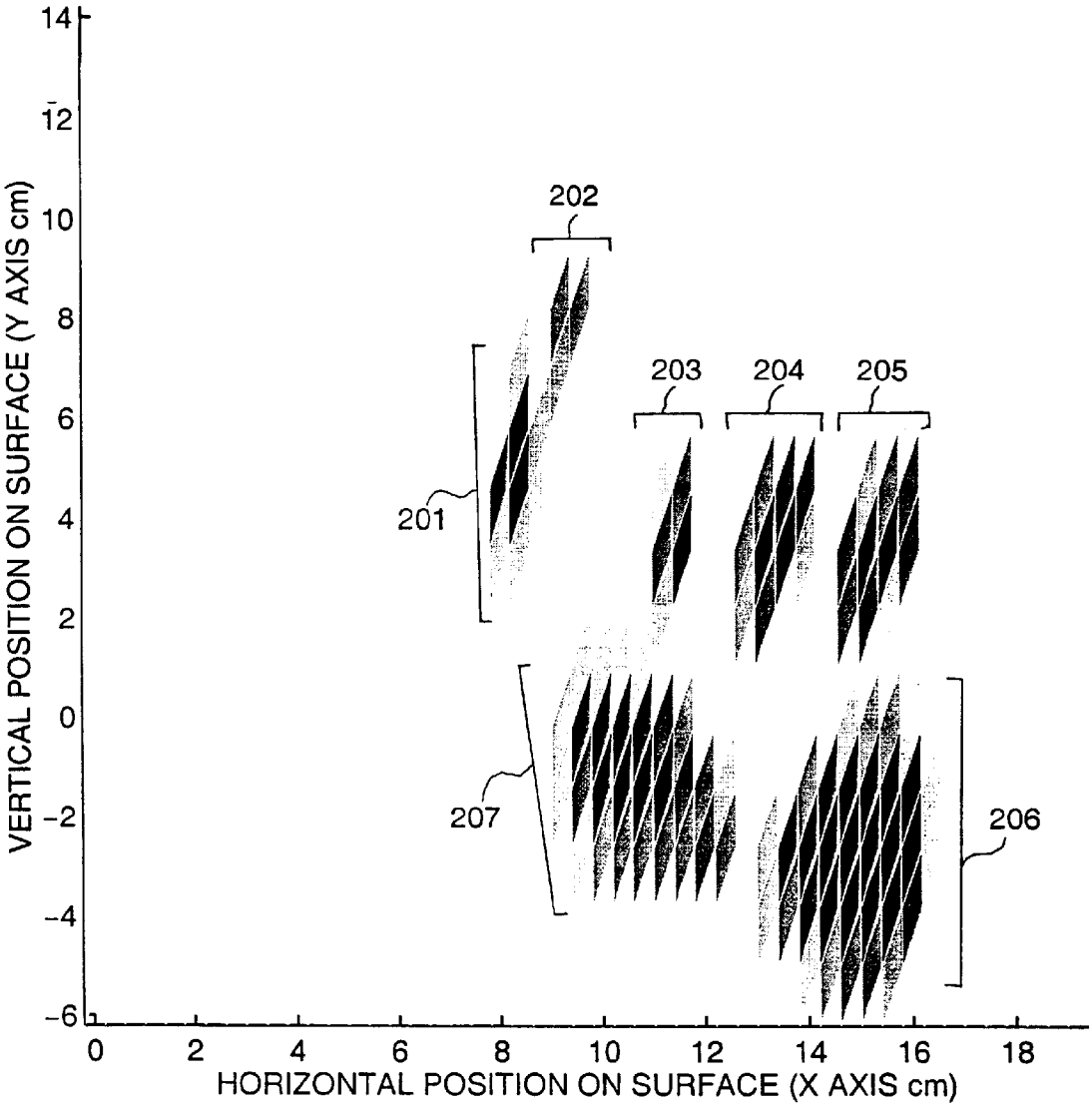

VERTICAL POSITION ON SURFACE (Y AXIS cm)
14
12
10
8
6
4
2
0
-2
-4
-6
0
2
4
6
8
10
12
14
16
18
HORIZONTAL POSITION ON SURFACE (X AXIS cm)
201
202
203
204
205
206
207

FIG. 15

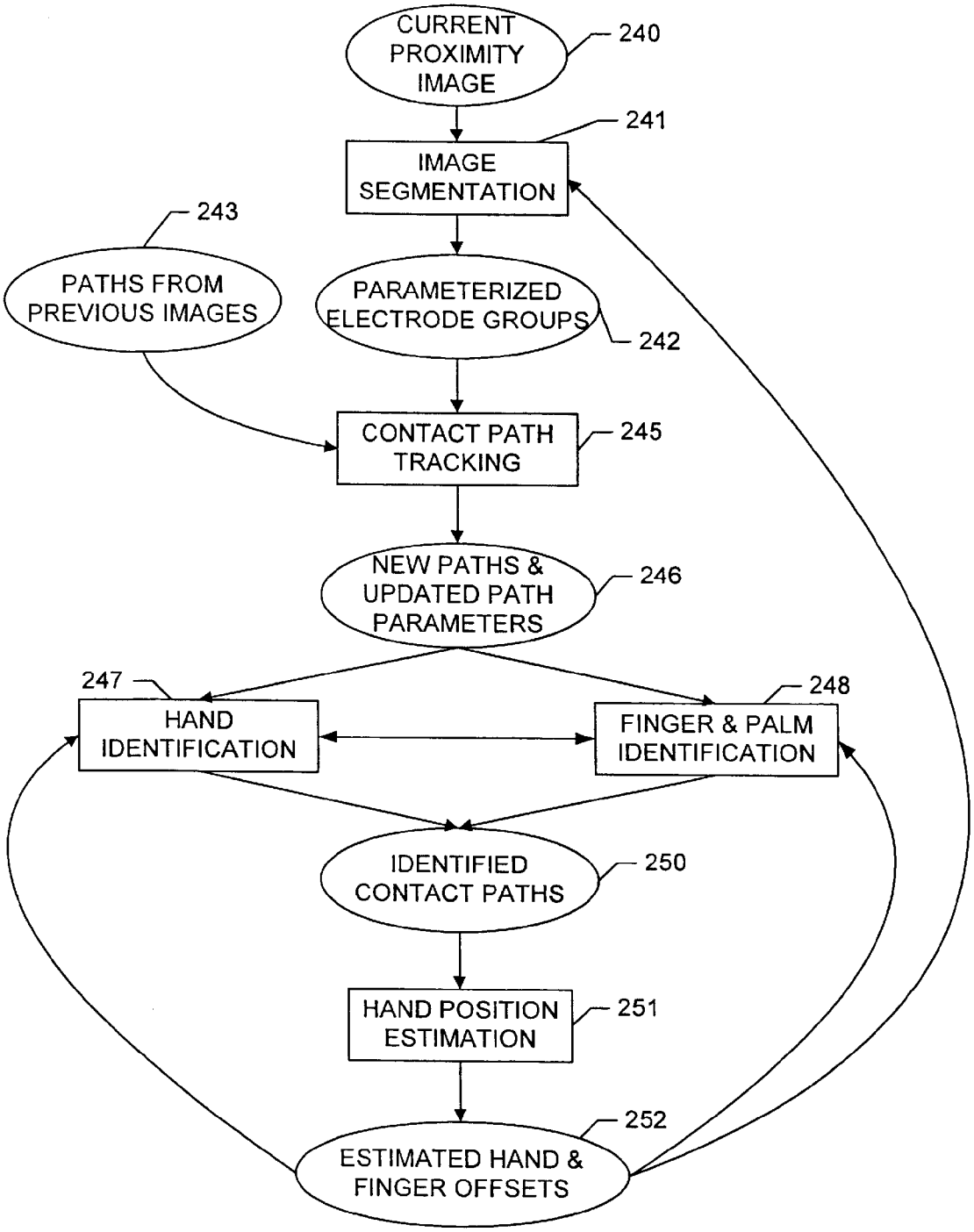

CURRENT PROXIMITY IMAGE
240
IMAGE SEGMENTATION
241
243
PATHS FROM PREVIOUS IMAGES
PARAMETERIZED ELECTRODE GROUPS
242
CONTACT PATH TRACKING
245
NEW PATHS & UPDATED PATH PARAMETERS
246
247
HAND IDENTIFICATION
248
FINGER & PALM IDENTIFICATION
IDENTIFIED CONTACT PATHS
250
HAND POSITION ESTIMATION
251
252
ESTIMATED HAND & FINGER OFFSETS

FIG. 16

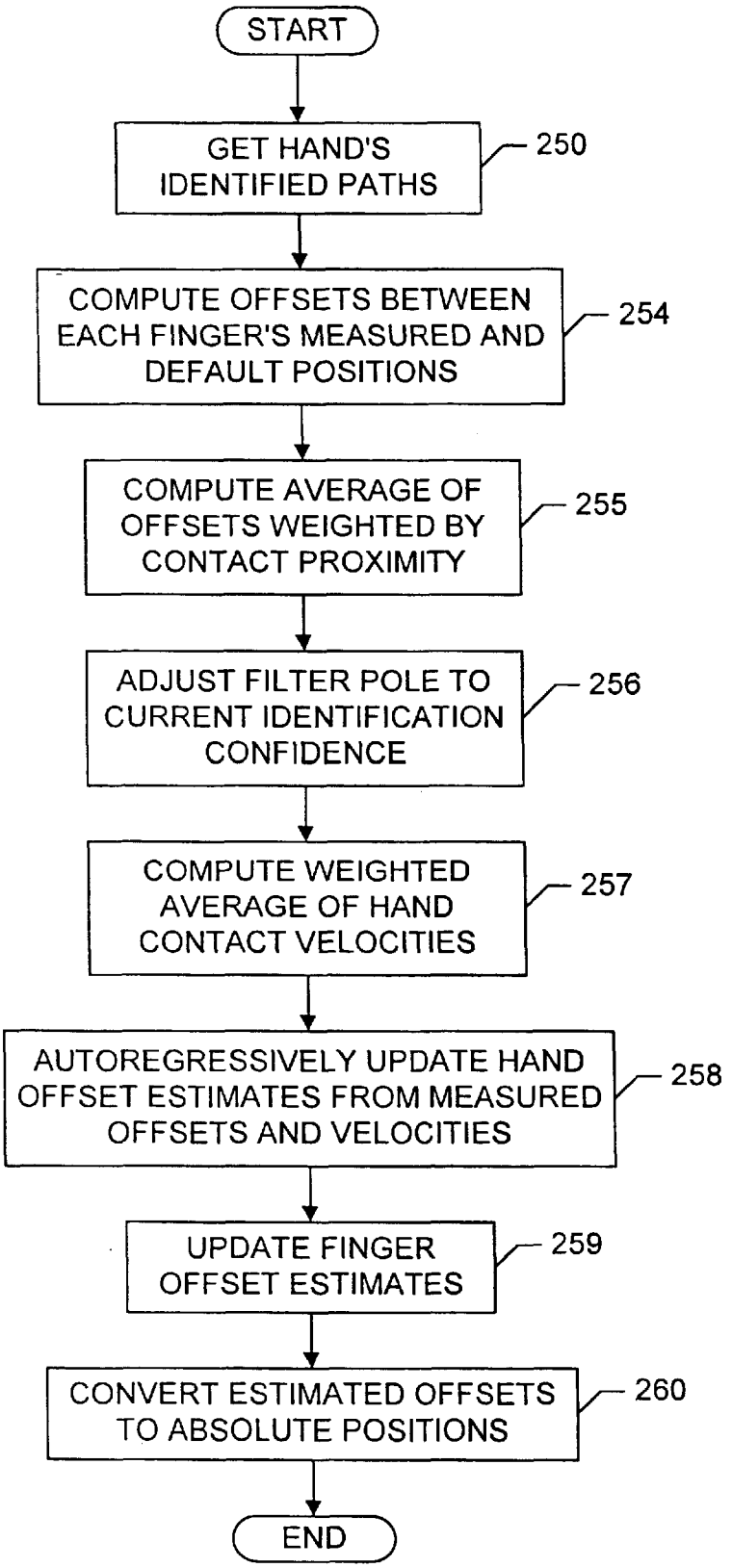

START
GET HAND'S IDENTIFIED PATHS
250
COMPUTE OFFSETS BETWEEN EACH FINGER'S MEASURED AND DEFAULT POSITIONS
254
COMPUTE AVERAGE OF OFFSETS WEIGHTED BY CONTACT PROXIMITY
255
ADJUST FILTER POLE TO CURRENT IDENTIFICATION CONFIDENCE
256
COMPUTE WEIGHTED AVERAGE OF HAND CONTACT VELOCITIES
257
AUTOREGRESSIVELY UPDATE HAND OFFSET ESTIMATES FROM MEASURED OFFSETS AND VELOCITIES
258
UPDATE FINGER OFFSET ESTIMATES
259
CONVERT ESTIMATED OFFSETS TO ABSOLUTE POSITIONS
260
END

FIG. 17

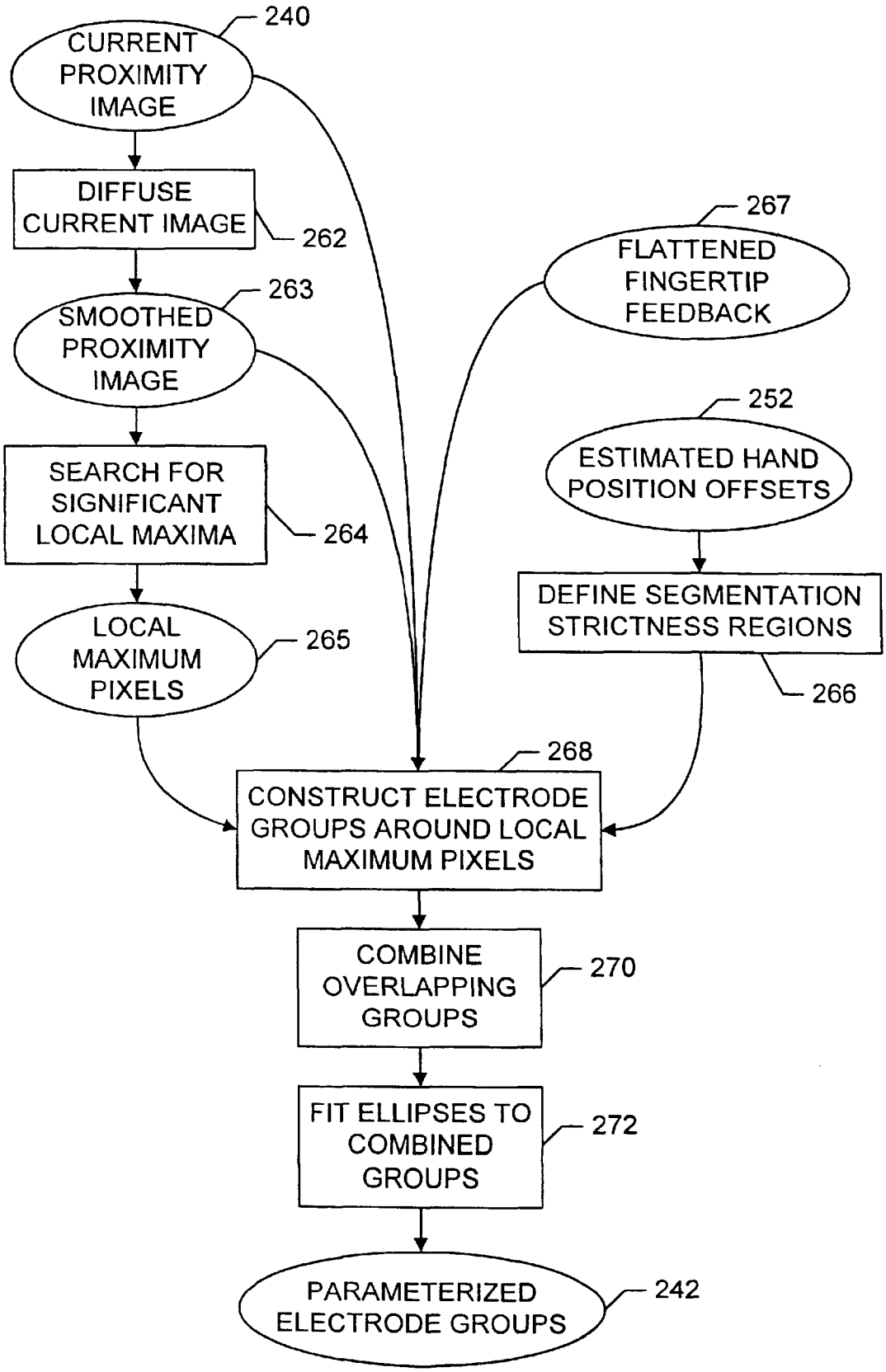

240
CURRENT PROXIMITY IMAGE
DIFFUSE CURRENT IMAGE
262
263
SMOOTHED PROXIMITY IMAGE
SEARCH FOR SIGNIFICANT LOCAL MAXIMA
264
LOCAL MAXIMUM PIXELS
265
267
FLATTENED FINGERTIP FEEDBACK
252
ESTIMATED HAND POSITION OFFSETS
DEFINE SEGMENTATION STRICTNESS REGIONS
266
268
CONSTRUCT ELECTRODE GROUPS AROUND LOCAL MAXIMUM PIXELS
COMBINE OVERLAPPING GROUPS
270
FIT ELLIPSES TO COMBINED GROUPS
272
PARAMETERIZED ELECTRODE GROUPS
242

FIG. 18

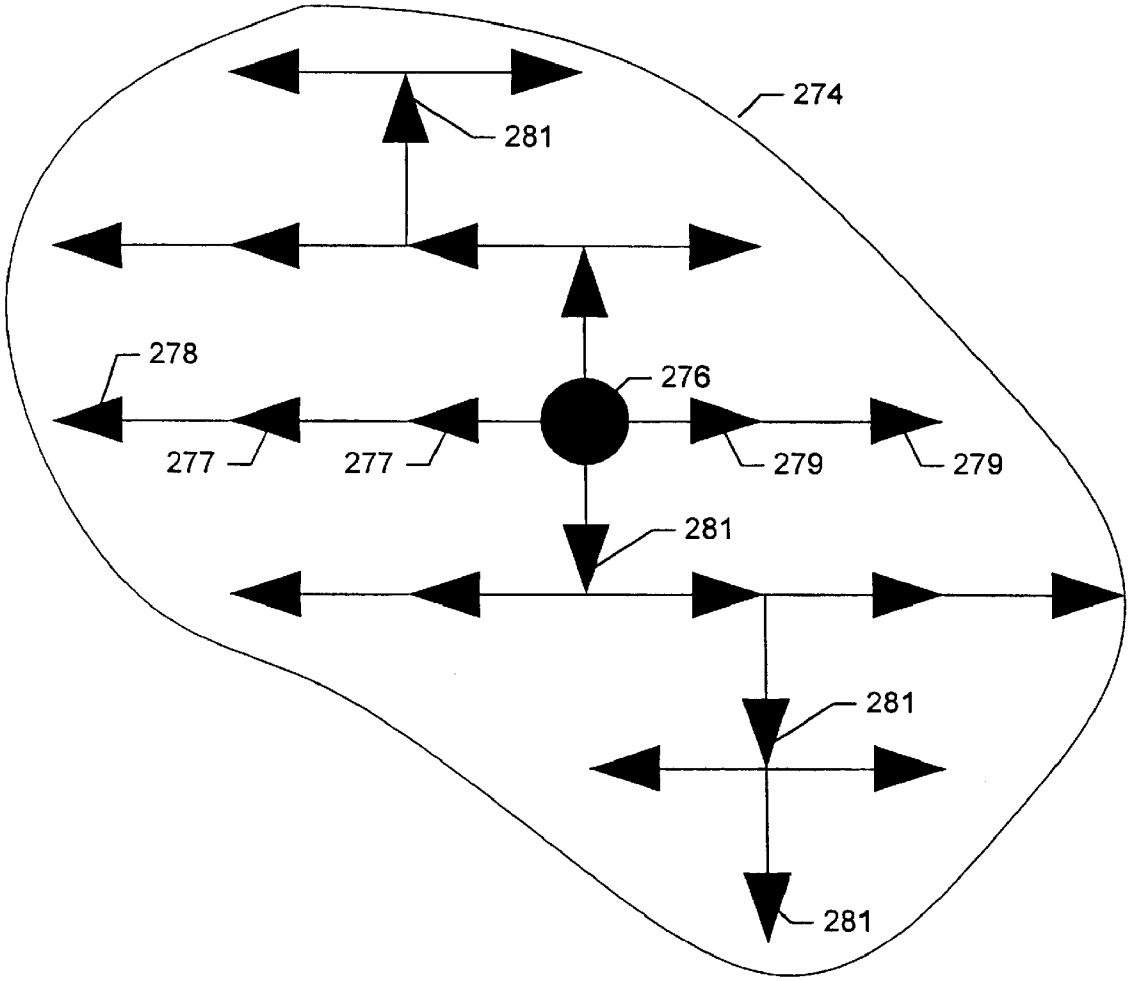


FIG. 19

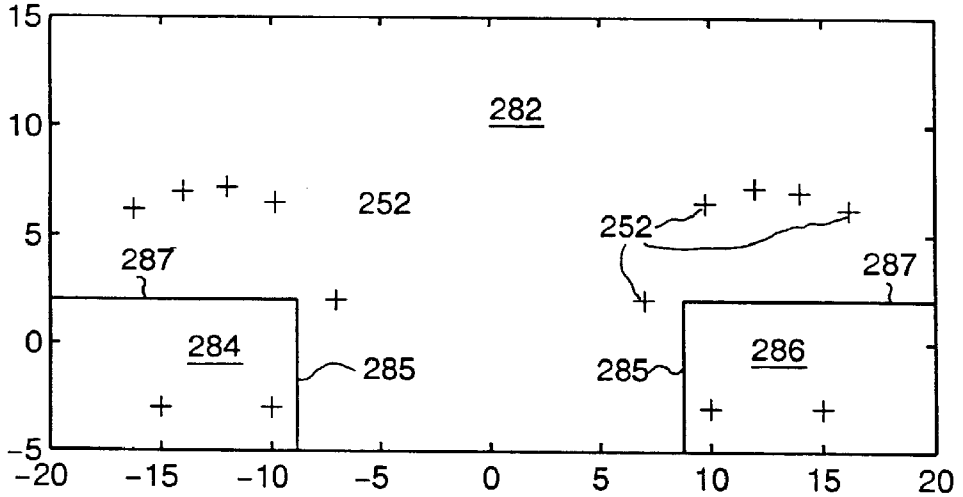


FIG. 20A

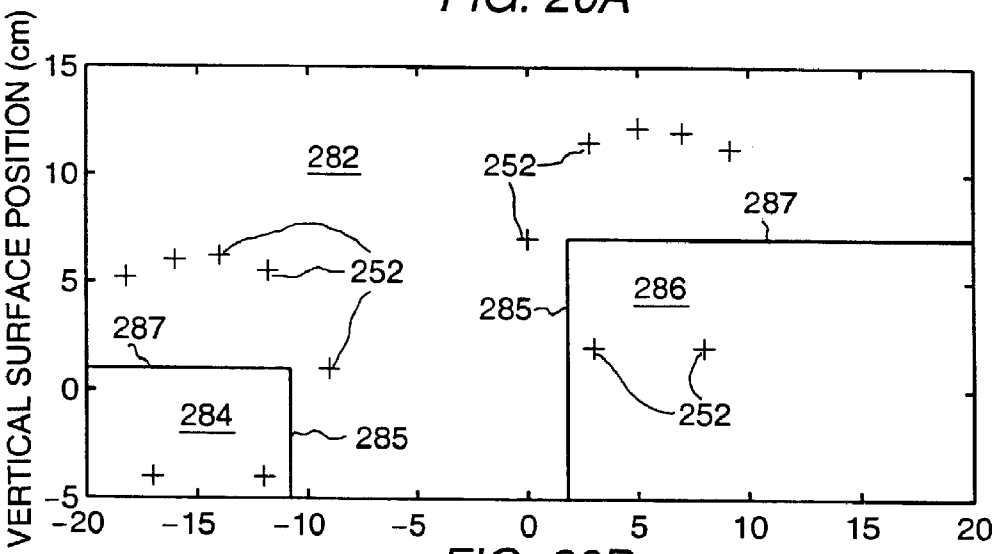


FIG. 20B

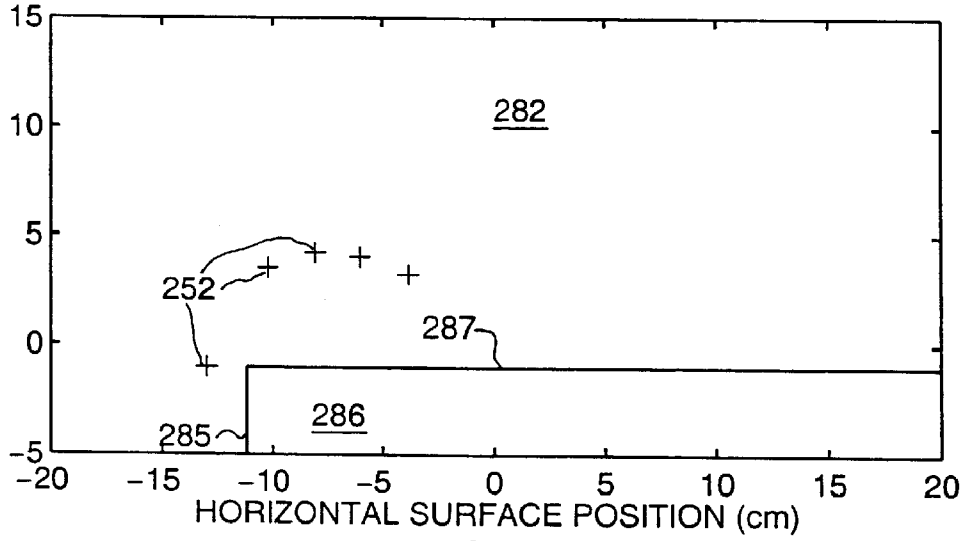


FIG. 20C

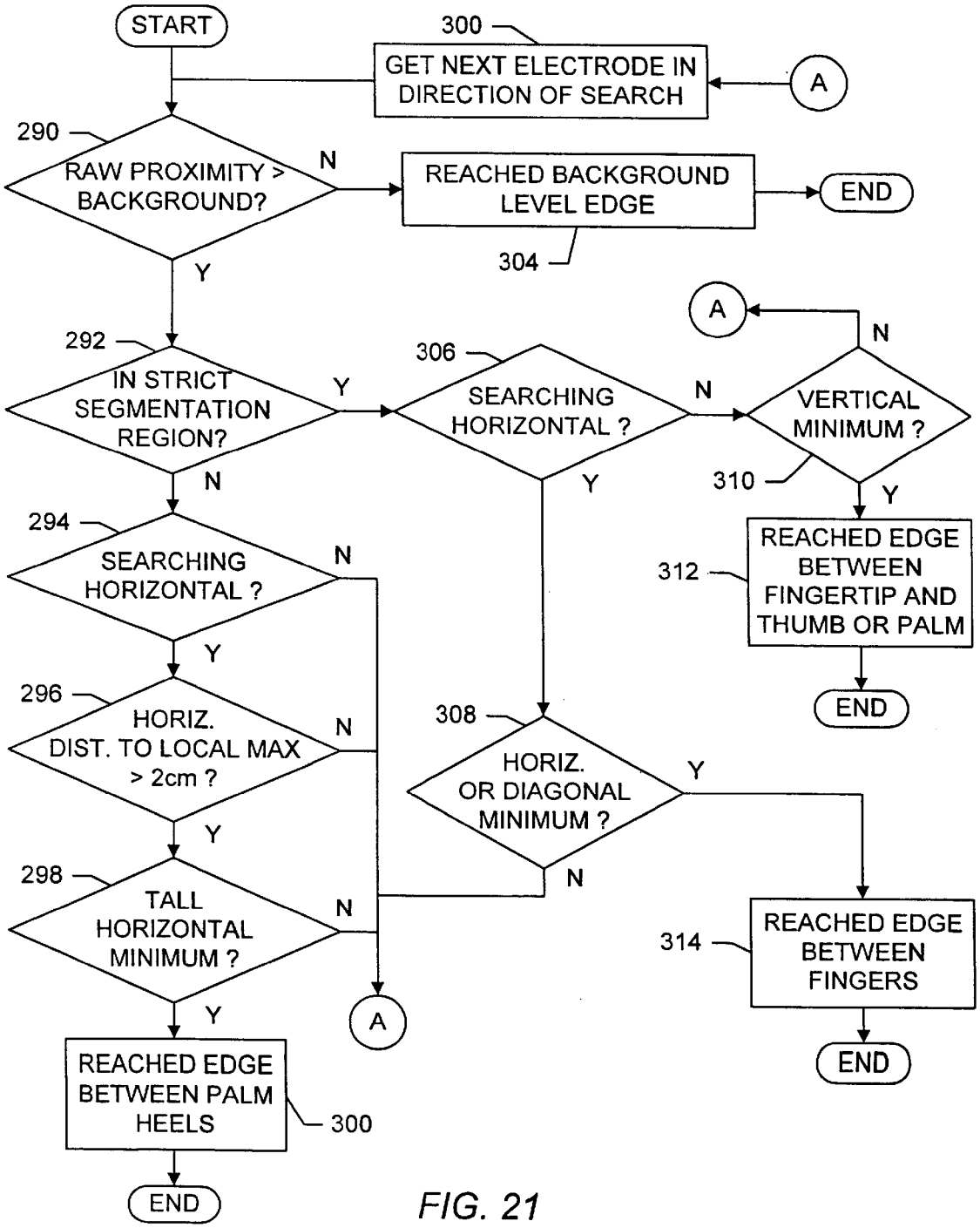

START
300
GET NEXT ELECTRODE IN DIRECTION OF SEARCH
A
290
RAW PROXIMITY > BACKGROUND?
N
REACHED BACKGROUND LEVEL EDGE
END
304
Y
A
N
292
IN STRICT SEGMENTATION REGION?
Y
306
SEARCHING HORIZONTAL ?
N
VERTICAL MINIMUM ?
N
Y
310
Y
294
SEARCHING HORIZONTAL ?
N
312
REACHED EDGE BETWEEN FINGERTIP AND THUMB OR PALM
Y
END
296
HORIZ. DIST. TO LOCAL MAX > 2cm ?
N
308
HORIZ. OR DIAGONAL MINIMUM ?
Y
Y
N
298
TALL HORIZONTAL MINIMUM ?
N
314
REACHED EDGE BETWEEN FINGERS
Y
A
END
REACHED EDGE BETWEEN PALM HEELS
300
END

FIG. 21

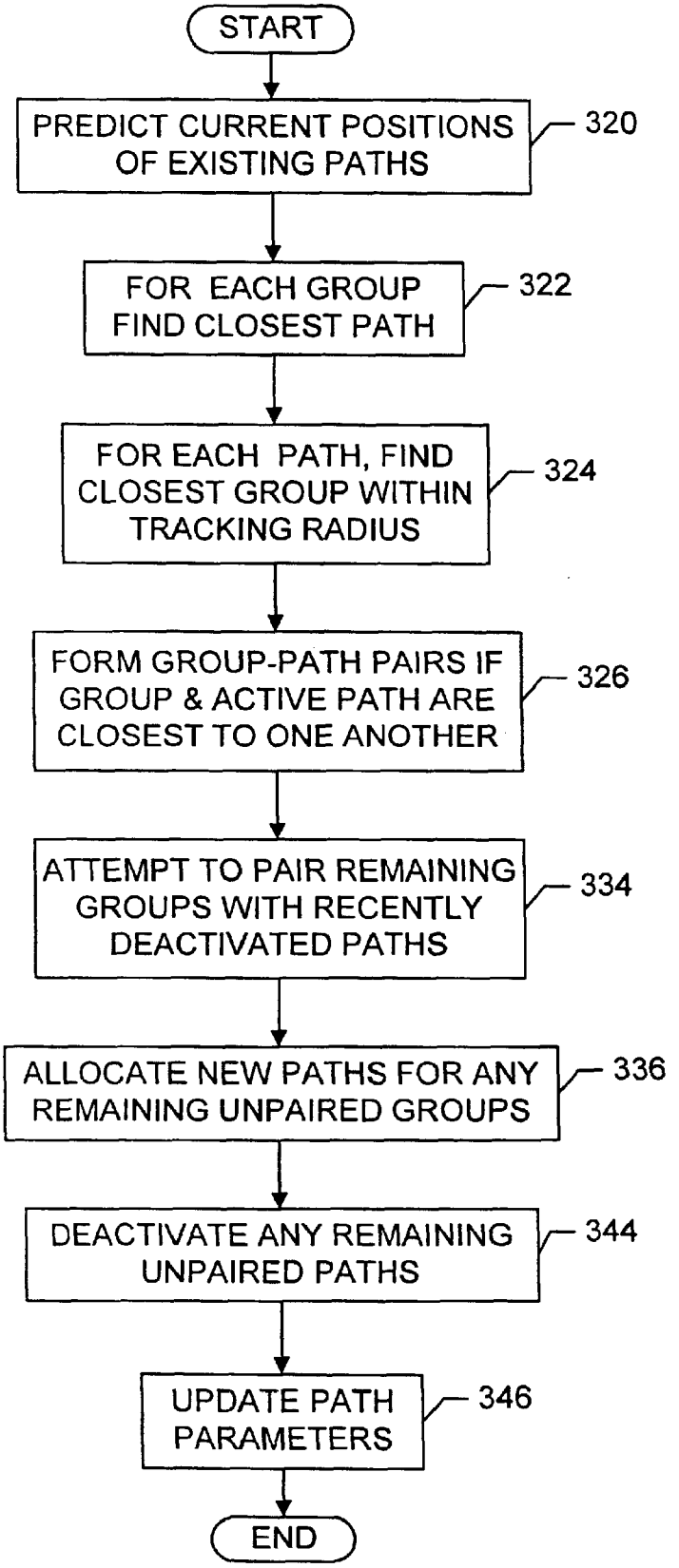

START
PREDICT CURRENT POSITIONS OF EXISTING PATHS
320
FOR EACH GROUP FIND CLOSEST PATH
322
FOR EACH PATH, FIND CLOSEST GROUP WITHIN TRACKING RADIUS
324
FORM GROUP-PATH PAIRS IF GROUP & ACTIVE PATH ARE CLOSEST TO ONE ANOTHER
326
ATTEMPT TO PAIR REMAINING GROUPS WITH RECENTLY DEACTIVATED PATHS
334
ALLOCATE NEW PATHS FOR ANY REMAINING UNPAIRED GROUPS
336
DEACTIVATE ANY REMAINING UNPAIRED PATHS
344
UPDATE PATH PARAMETERS
346
END

FIG. 22

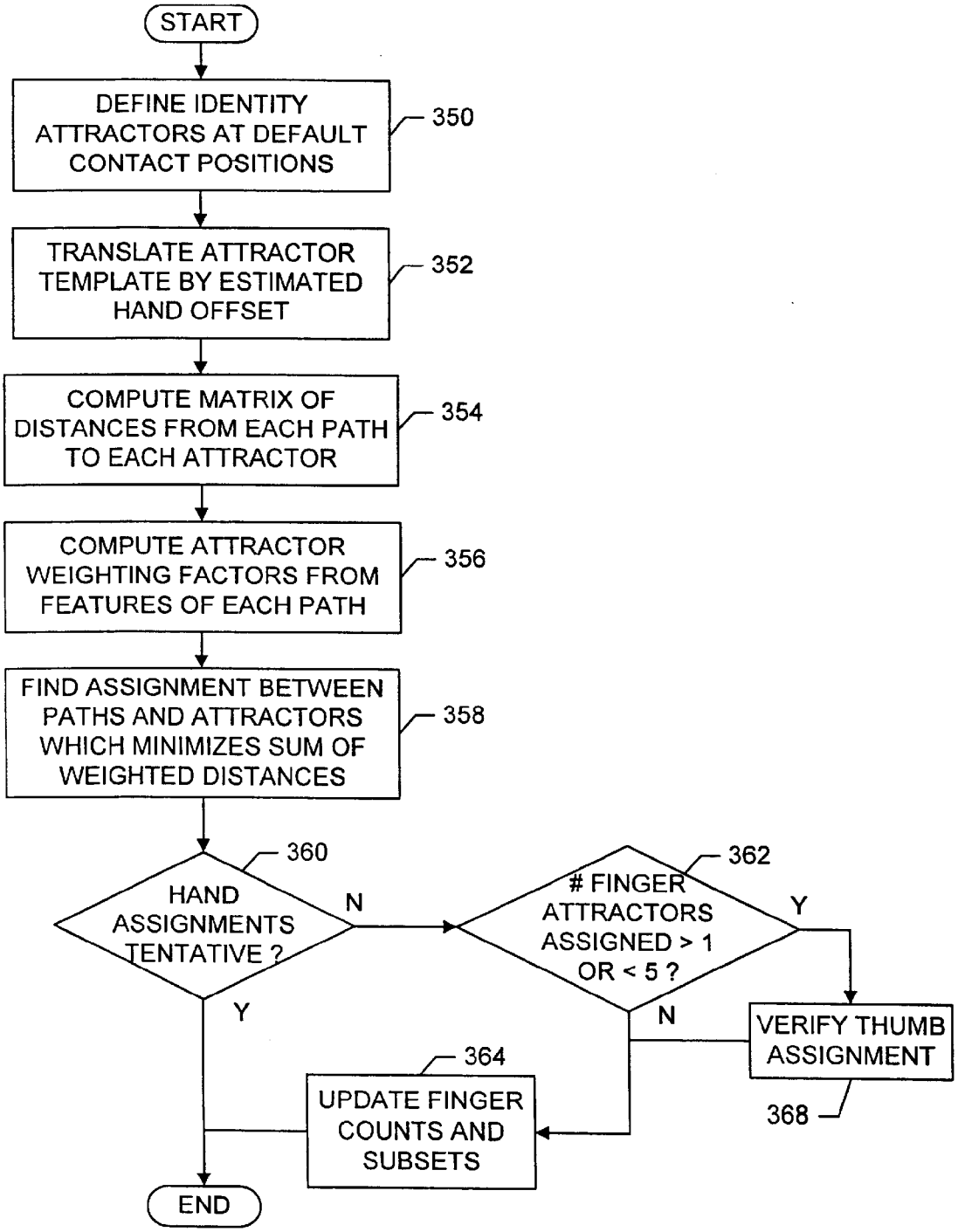

START
DEFINE IDENTITY ATTRACTORS AT DEFAULT CONTACT POSITIONS
350
TRANSLATE ATTRACTOR TEMPLATE BY ESTIMATED HAND OFFSET
352
COMPUTE MATRIX OF DISTANCES FROM EACH PATH TO EACH ATTRACTOR
354
COMPUTE ATTRACTOR WEIGHTING FACTORS FROM FEATURES OF EACH PATH
356
FIND ASSIGNMENT BETWEEN PATHS AND ATTRACTORS WHICH MINIMIZES SUM OF WEIGHTED DISTANCES
358
360
HAND ASSIGNMENTS TENTATIVE ?
N
Y
362
FINGER ATTRACTORS ASSIGNED > 1 OR < 5 ?
Y
N
VERIFY THUMB ASSIGNMENT
368
364
UPDATE FINGER COUNTS AND SUBSETS
END

FIG. 23

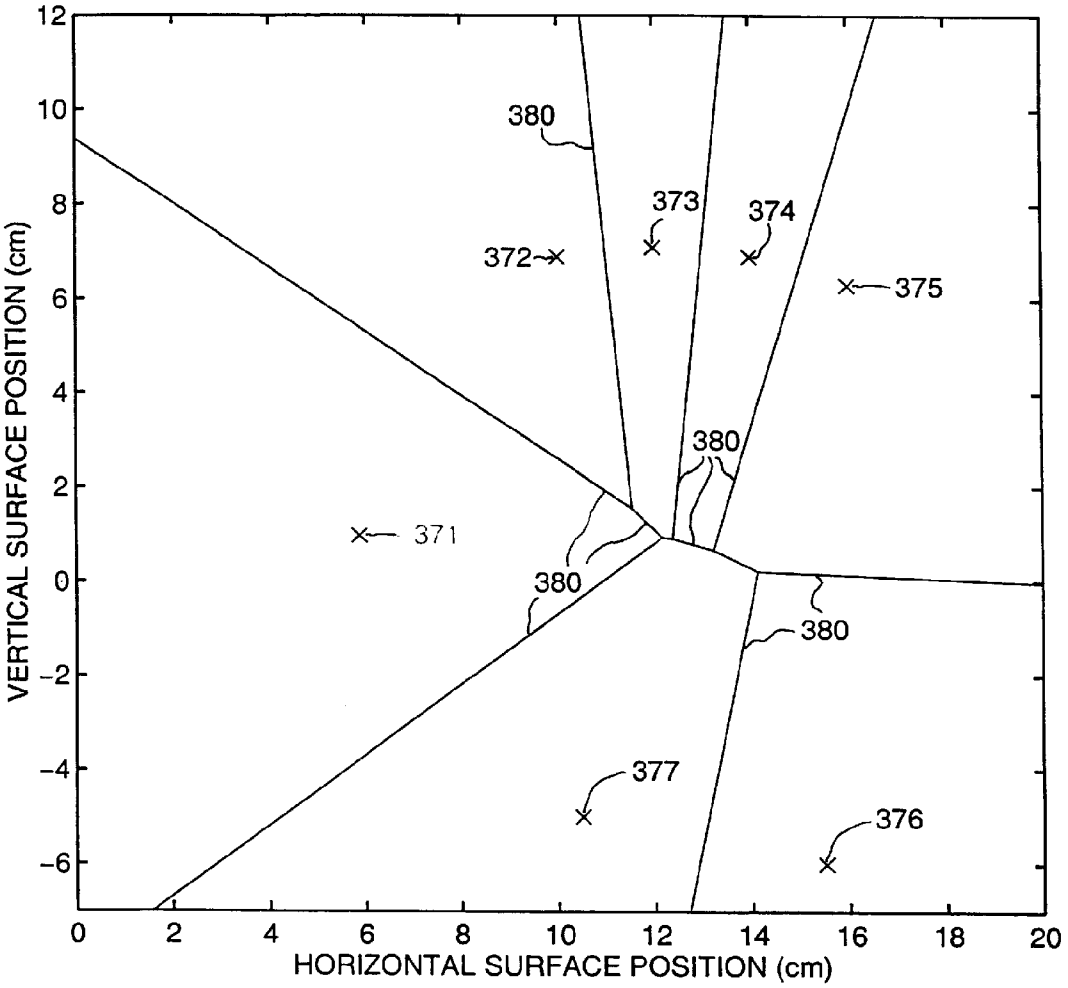

VERTICAL SURFACE POSITION (cm)
HORIZONTAL SURFACE POSITION (cm)
12
10
8
6
4
2
0
-2
-4
-6
0
2
4
6
8
10
12
14
16
18
20
380
373
374
372
375
380
371
380
380
377
376

FIG. 24

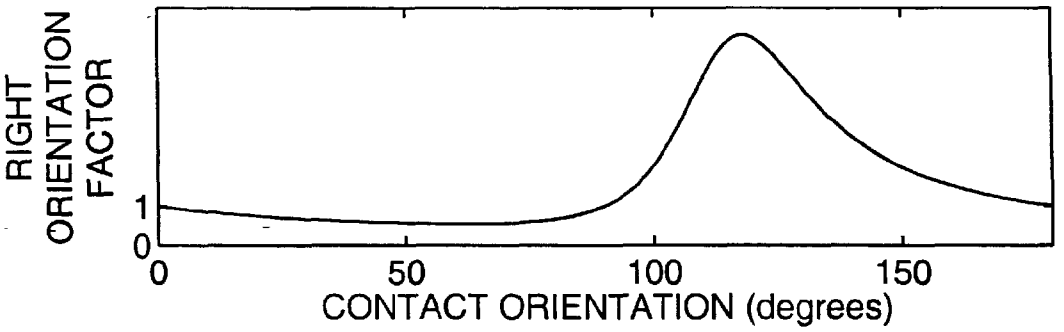


FIG. 25A

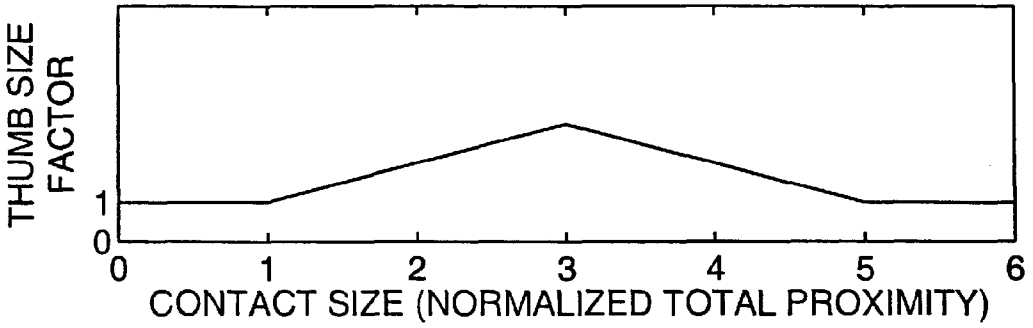


FIG. 25B

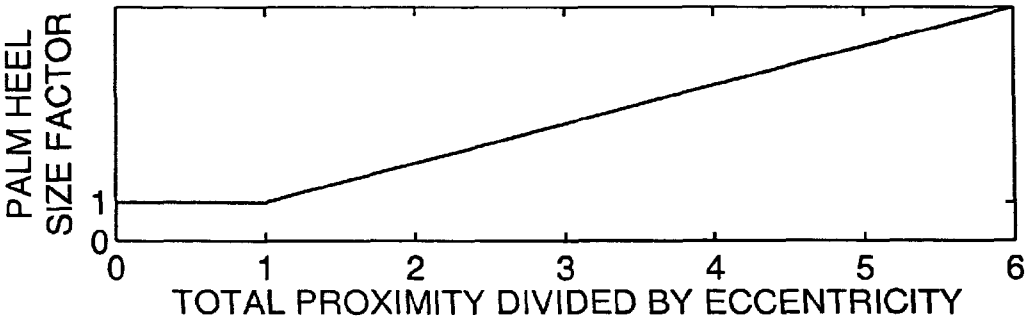


FIG. 25C

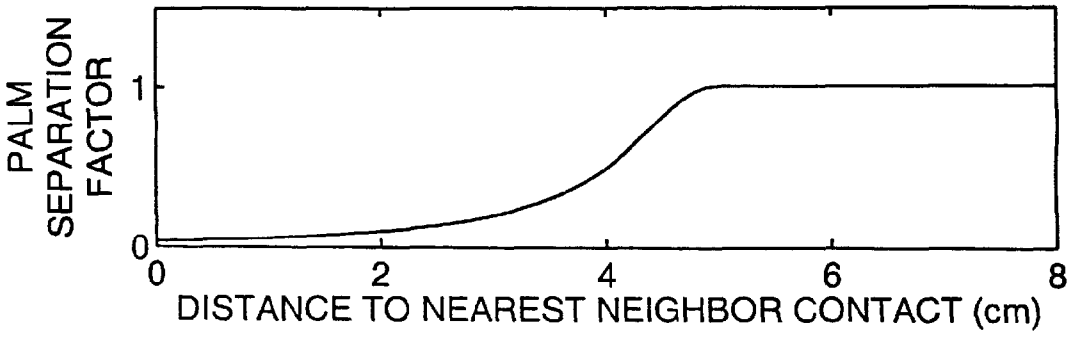


FIG. 25D

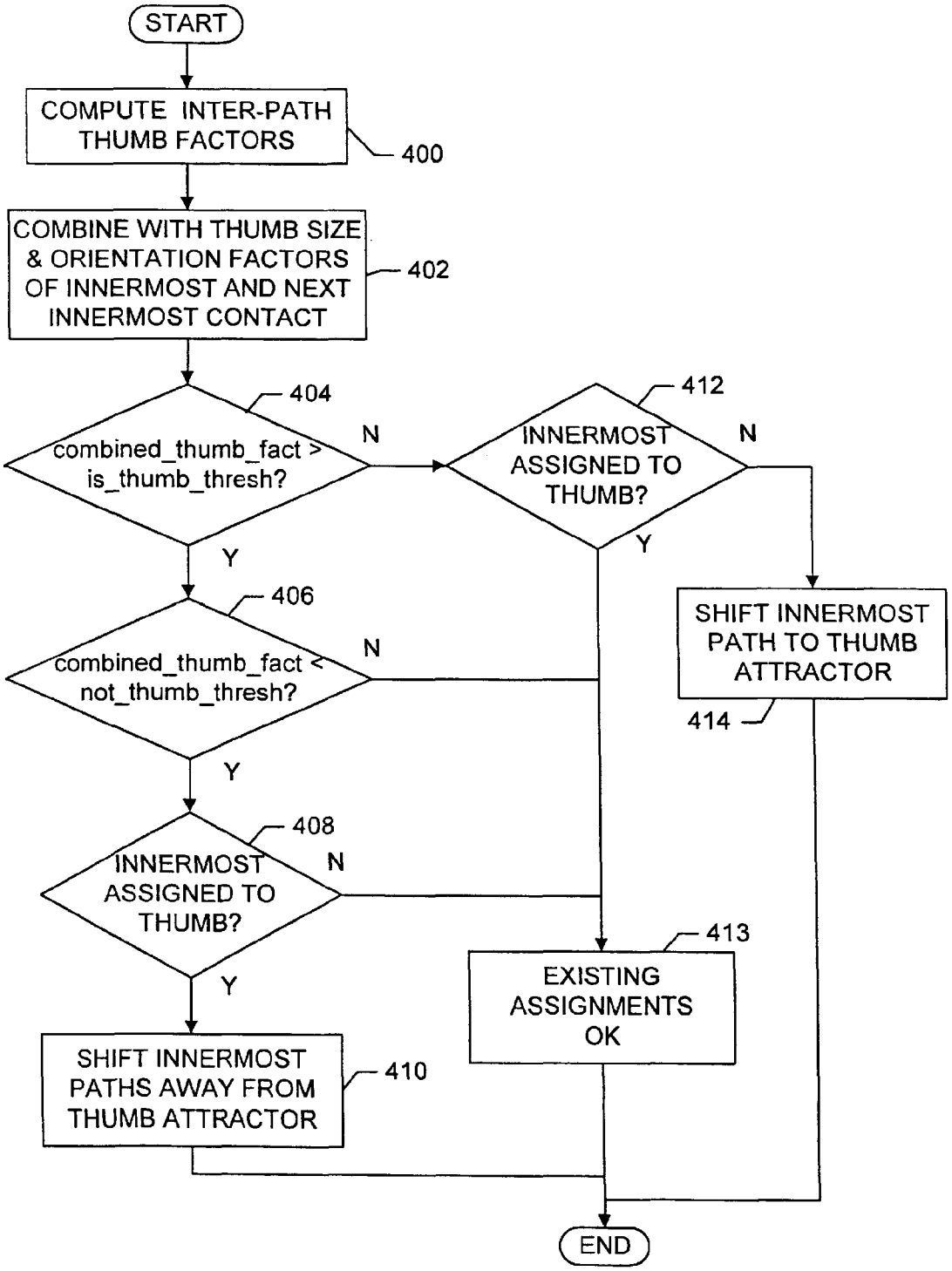

START
COMPUTE INTER-PATH THUMB FACTORS
400
COMBINE WITH THUMB SIZE & ORIENTATION FACTORS OF INNERMOST AND NEXT INNERMOST CONTACT
402
404
combined_thumb_fact > is_thumb_thresh?
N
Y
412
INNERMOST ASSIGNED TO THUMB?
N
Y
SHIFT INNERMOST PATH TO THUMB ATTRACTOR
414
406
combined_thumb_fact < not_thumb_thresh?
N
Y
408
INNERMOST ASSIGNED TO THUMB?
N
Y
413
EXISTING ASSIGNMENTS OK
SHIFT INNERMOST PATHS AWAY FROM THUMB ATTRACTOR
410
END

FIG. 26

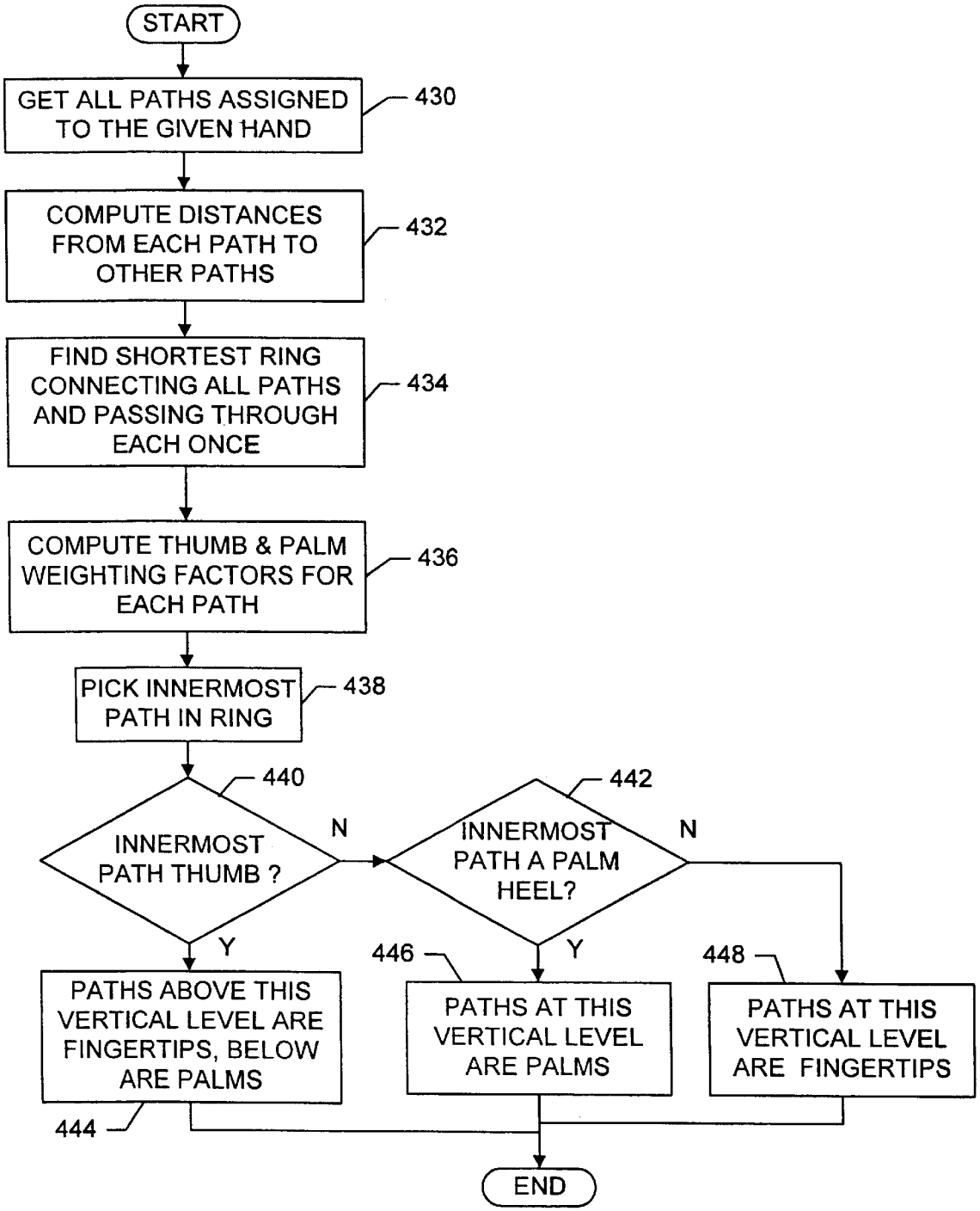

START
GET ALL PATHS ASSIGNED TO THE GIVEN HAND
430
COMPUTE DISTANCES FROM EACH PATH TO OTHER PATHS
432
FIND SHORTEST RING CONNECTING ALL PATHS AND PASSING THROUGH EACH ONCE
434
COMPUTE THUMB & PALM WEIGHTING FACTORS FOR EACH PATH
436
PICK INNERMOST PATH IN RING
438
440
INNERMOST PATH THUMB ?
N
Y
442
INNERMOST PATH A PALM HEEL?
N
Y
446
448
PATHS ABOVE THIS VERTICAL LEVEL ARE FINGERTIPS, BELOW ARE PALMS
PATHS AT THIS VERTICAL LEVEL ARE PALMS
PATHS AT THIS VERTICAL LEVEL ARE FINGERTIPS
444
END

FIG. 27

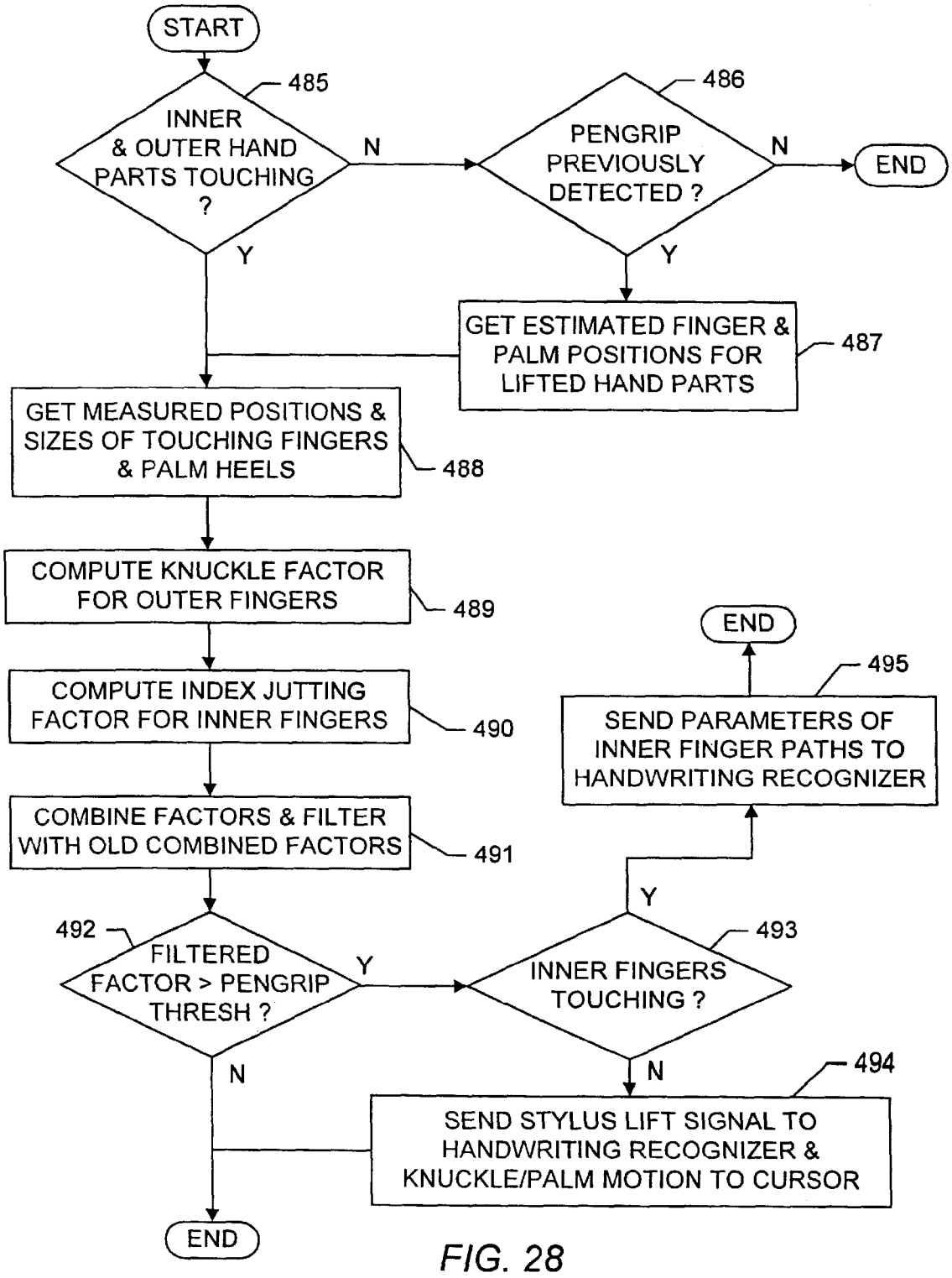

START
485
INNER & OUTER HAND PARTS TOUCHING ?
N
Y
486
PENGRIP PREVIOUSLY DETECTED ?
N
END
Y
GET ESTIMATED FINGER & PALM POSITIONS FOR LIFTED HAND PARTS
487
GET MEASURED POSITIONS & SIZES OF TOUCHING FINGERS & PALM HEELS
488
COMPUTE KNUCKLE FACTOR FOR OUTER FINGERS
489
COMPUTE INDEX JUTTING FACTOR FOR INNER FINGERS
490
COMBINE FACTORS & FILTER WITH OLD COMBINED FACTORS
491
492
FILTERED FACTOR > PENGRIP THRESH ?
Y
N
END
493
INNER FINGERS TOUCHING ?
Y
N
494
SEND STYLUS LIFT SIGNAL TO HANDWRITING RECOGNIZER & KNUCKLE/PALM MOTION TO CURSOR
END
495
SEND PARAMETERS OF INNER FINGER PATHS TO HANDWRITING RECOGNIZER

FIG. 28

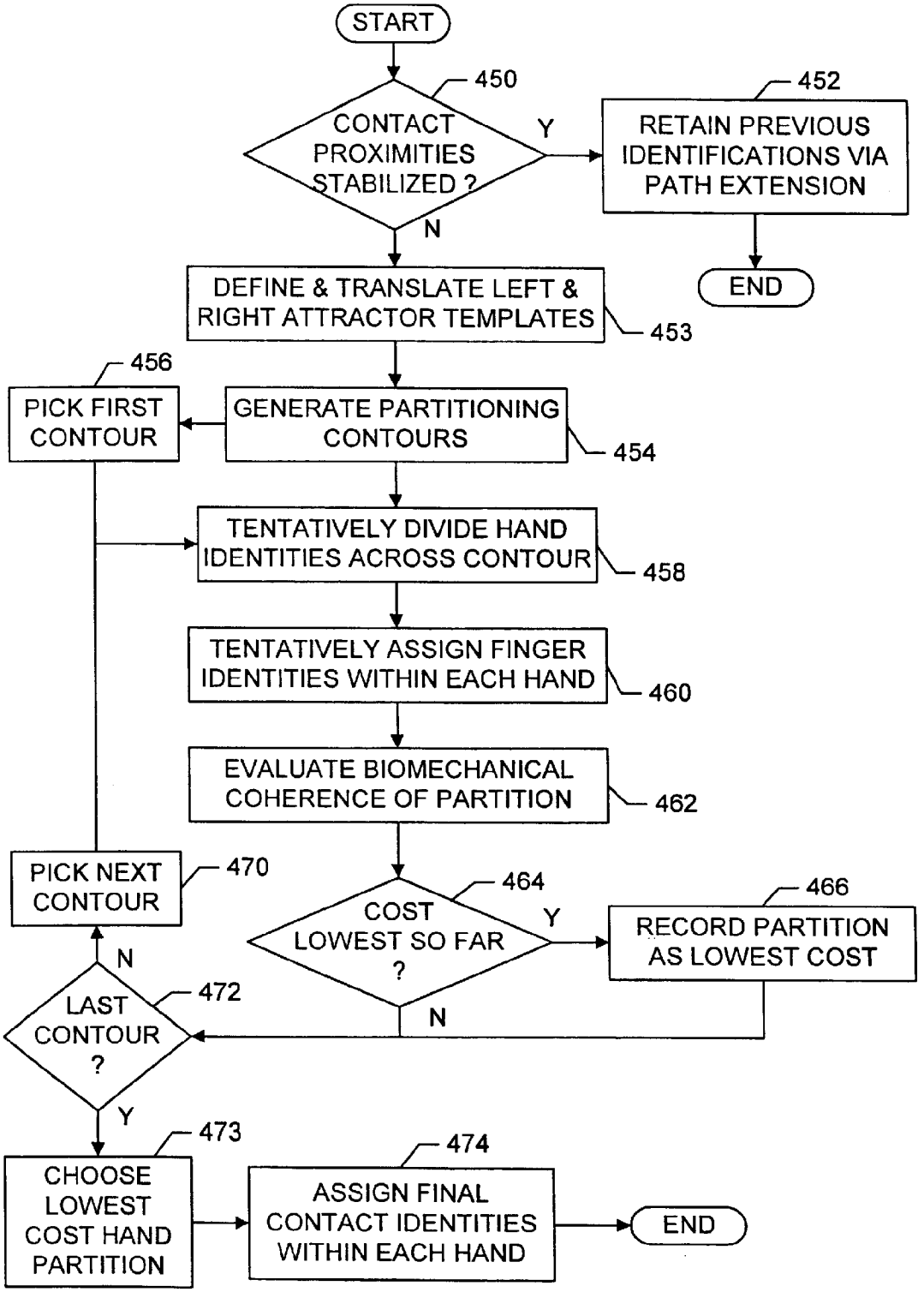

START
450
CONTACT PROXIMITIES STABILIZED ?
Y
N
452
RETAIN PREVIOUS IDENTIFICATIONS VIA PATH EXTENSION
END
DEFINE & TRANSLATE LEFT & RIGHT ATTRACTOR TEMPLATES
453
456
PICK FIRST CONTOUR
GENERATE PARTITIONING CONTOURS
454
TENTATIVELY DIVIDE HAND IDENTITIES ACROSS CONTOUR
458
TENTATIVELY ASSIGN FINGER IDENTITIES WITHIN EACH HAND
460
EVALUATE BIOMECHANICAL COHERENCE OF PARTITION
462
PICK NEXT CONTOUR
470
464
COST LOWEST SO FAR ?
Y
N
466
RECORD PARTITION AS LOWEST COST
N
LAST CONTOUR ?
472
Y
473
CHOOSE LOWEST COST HAND PARTITION
474
ASSIGN FINAL CONTACT IDENTITIES WITHIN EACH HAND
END

FIG. 29

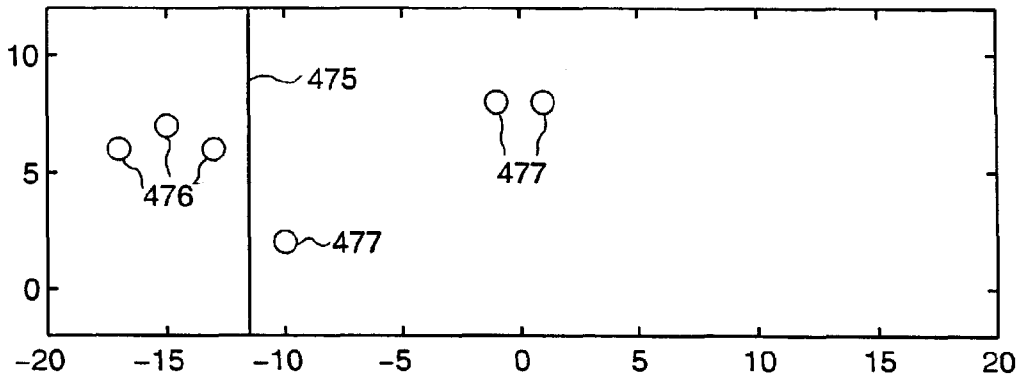


FIG. 30A

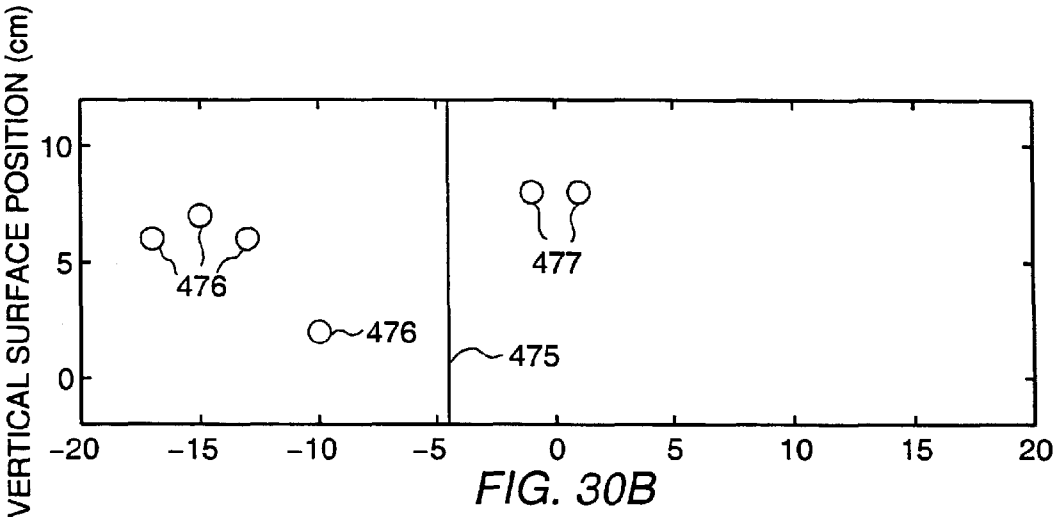


FIG. 30B

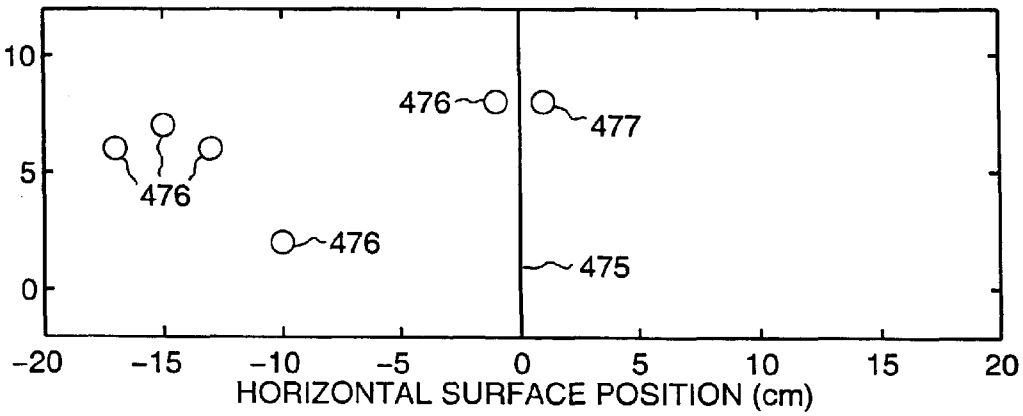


FIG. 30C

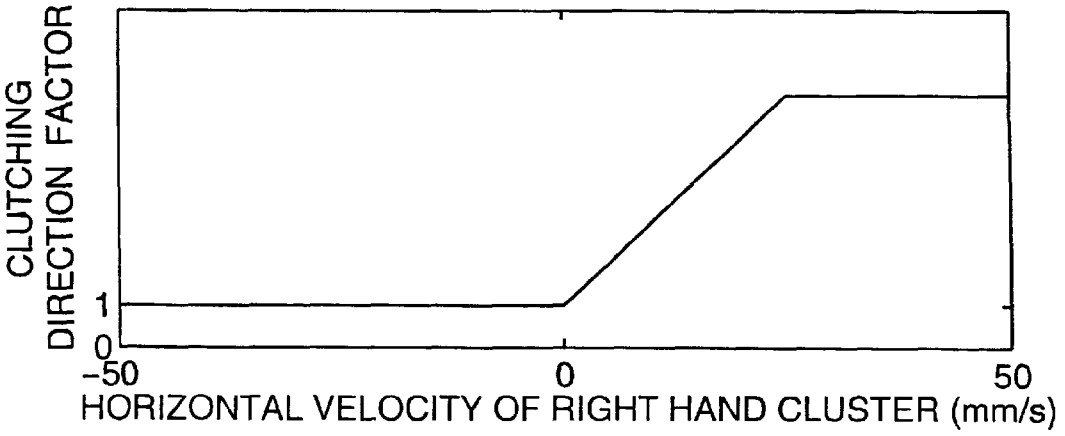


FIG. 31A

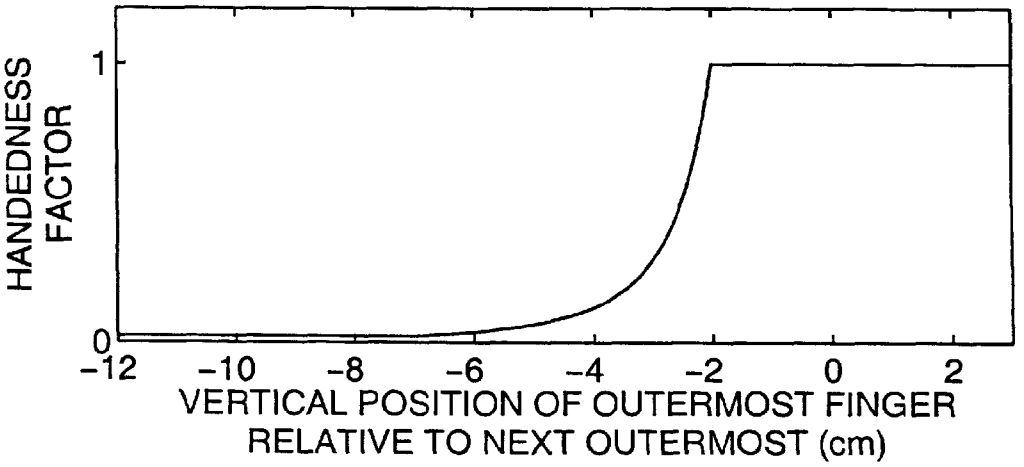


FIG. 31B

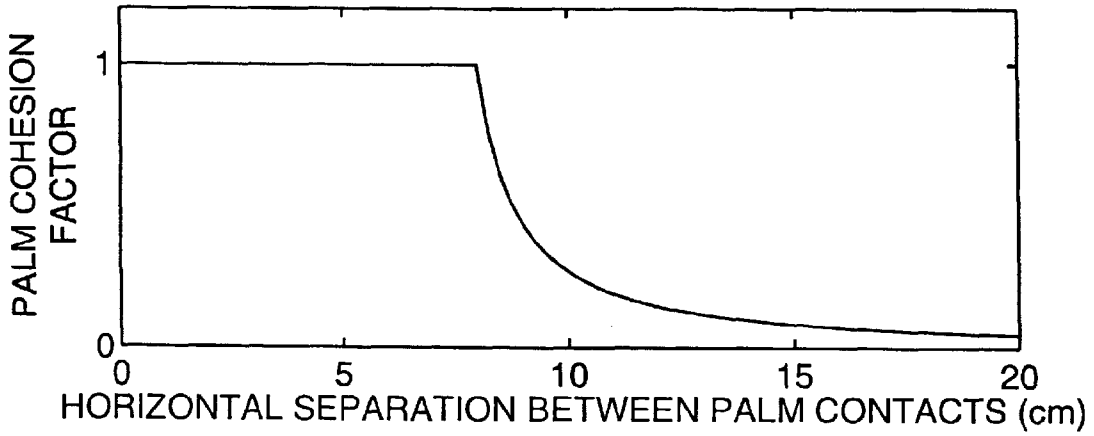


FIG. 31C

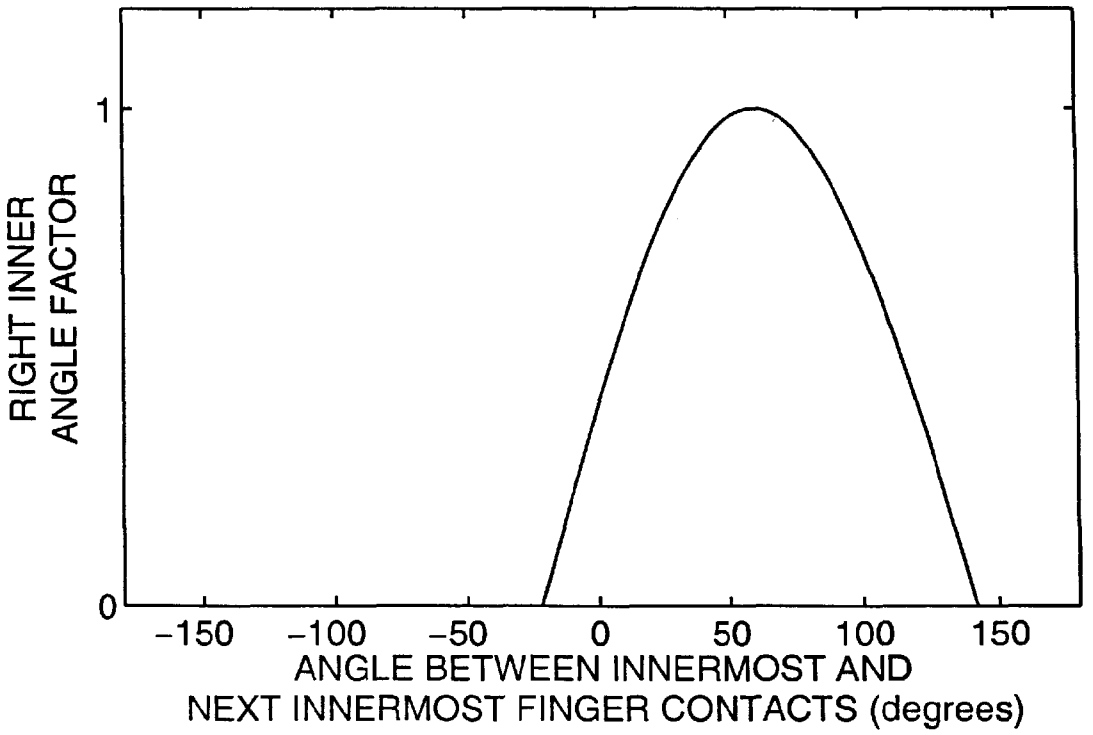

RIGHT INNER
ANGLE FACTOR
1
0
−150
−100
−50
0
50
100
150
ANGLE BETWEEN INNERMOST AND
NEXT INNERMOST FINGER CONTACTS (degrees)

FIG. 32

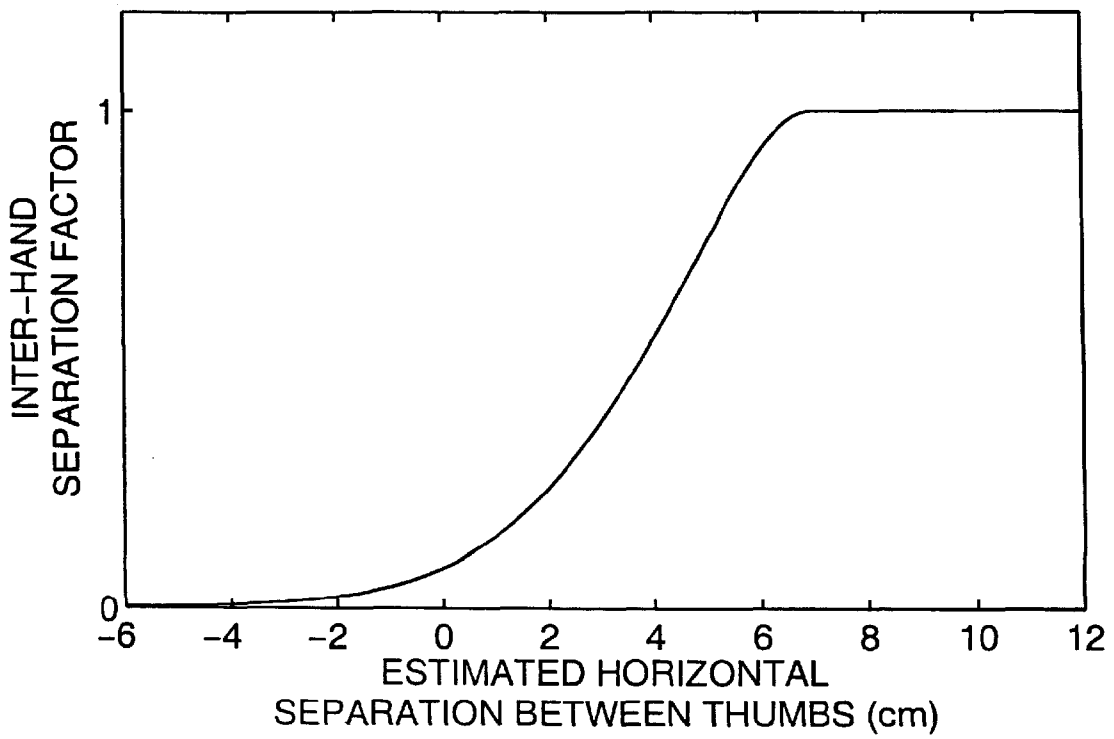

INTER-HAND
SEPARATION FACTOR
1
0
−6
−4
−2
0
2
4
6
8
10
12
ESTIMATED HORIZONTAL
SEPARATION BETWEEN THUMBS (cm)

FIG. 33

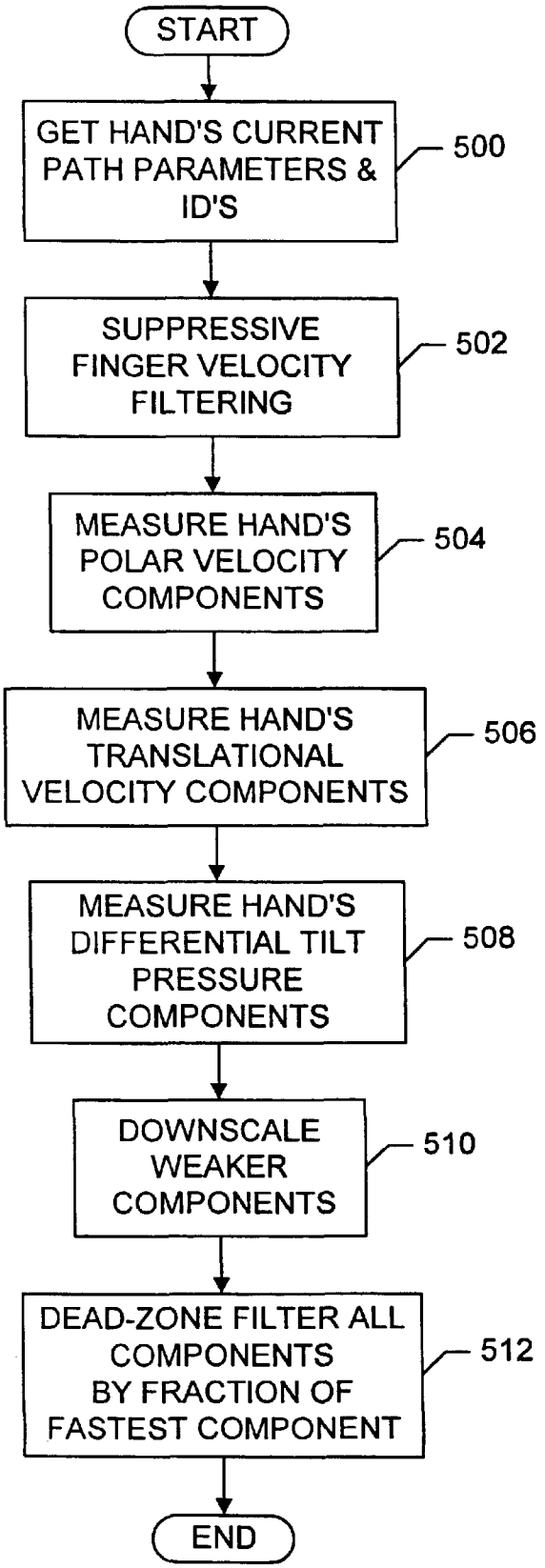

START
GET HAND'S CURRENT PATH PARAMETERS & ID'S
500
SUPPRESSIVE FINGER VELOCITY FILTERING
502
MEASURE HAND'S POLAR VELOCITY COMPONENTS
504
MEASURE HAND'S TRANSLATIONAL VELOCITY COMPONENTS
506
MEASURE HAND'S DIFFERENTIAL TILT PRESSURE COMPONENTS
508
DOWNSCALE WEAKER COMPONENTS
510
DEAD-ZONE FILTER ALL COMPONENTS BY FRACTION OF FASTEST COMPONENT
512
END

FIG. 34

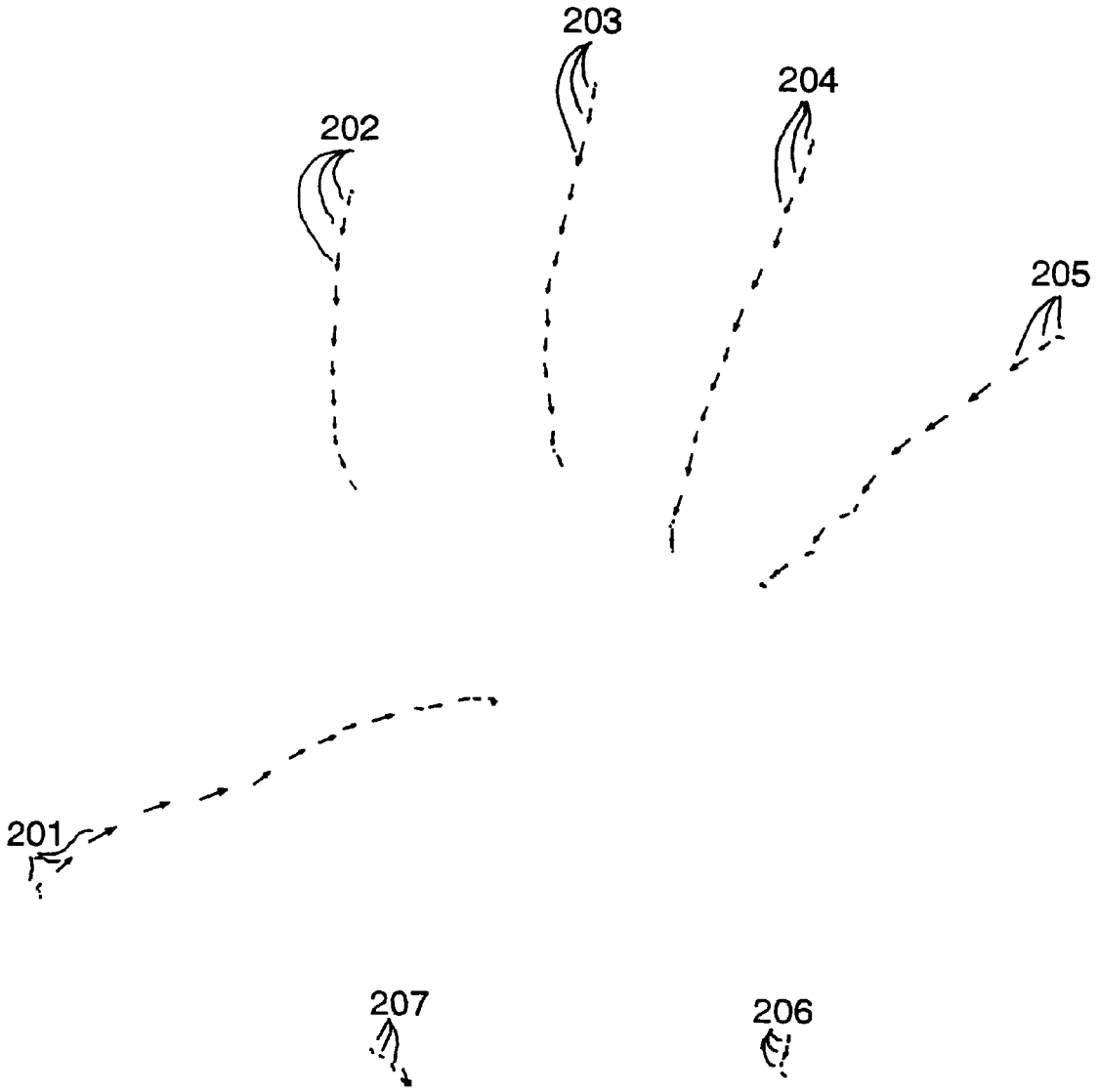


FIG. 35

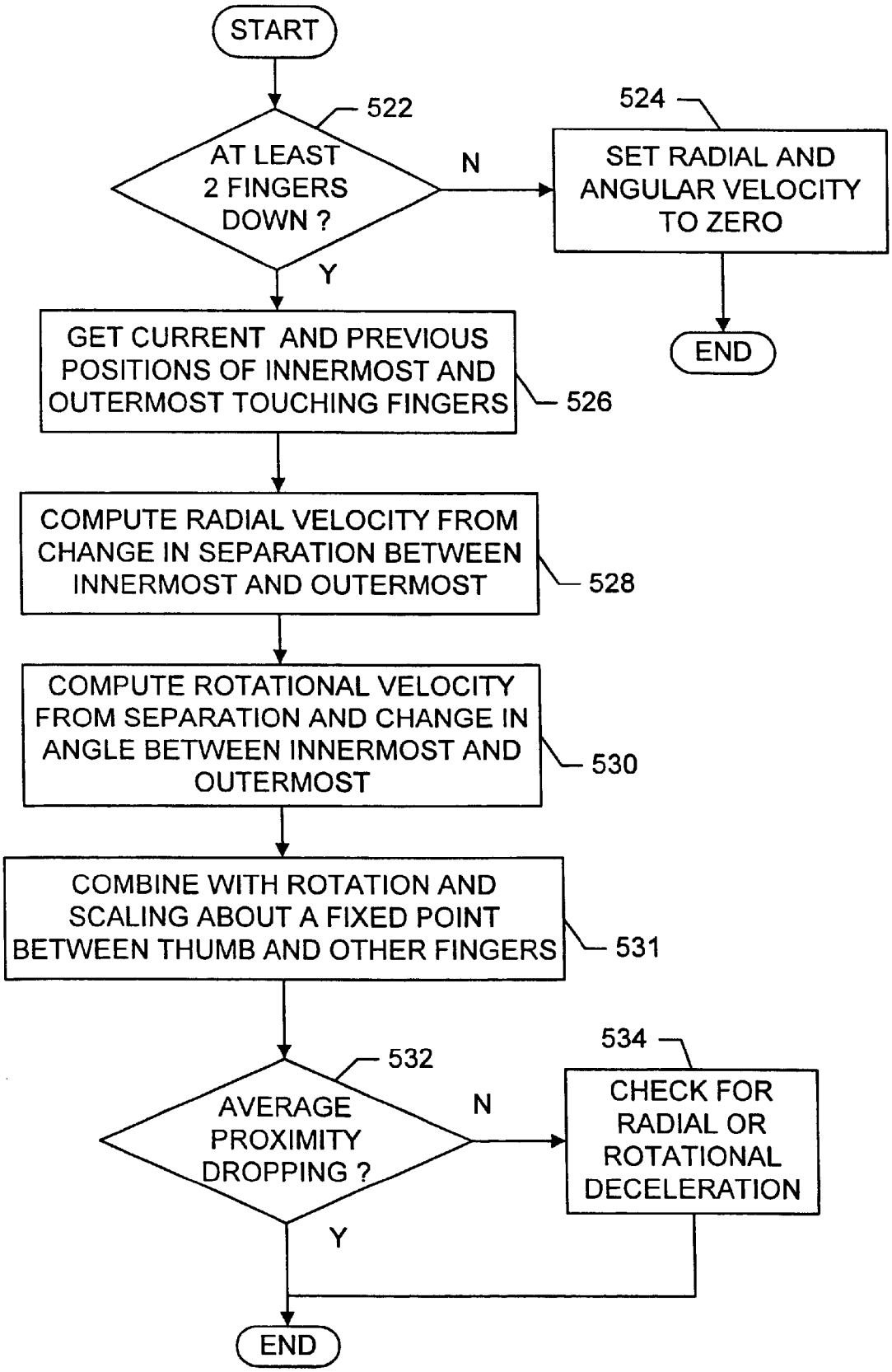

START
522
AT LEAST 2 FINGERS DOWN ?
N
524
SET RADIAL AND ANGULAR VELOCITY TO ZERO
END
Y
GET CURRENT AND PREVIOUS POSITIONS OF INNERMOST AND OUTERMOST TOUCHING FINGERS
526
COMPUTE RADIAL VELOCITY FROM CHANGE IN SEPARATION BETWEEN INNERMOST AND OUTERMOST
528
COMPUTE ROTATIONAL VELOCITY FROM SEPARATION AND CHANGE IN ANGLE BETWEEN INNERMOST AND OUTERMOST
530
COMBINE WITH ROTATION AND SCALING ABOUT A FIXED POINT BETWEEN THUMB AND OTHER FINGERS
531
532
AVERAGE PROXIMITY DROPPING ?
N
534
CHECK FOR RADIAL OR ROTATIONAL DECELERATION
Y
END

FIG. 36

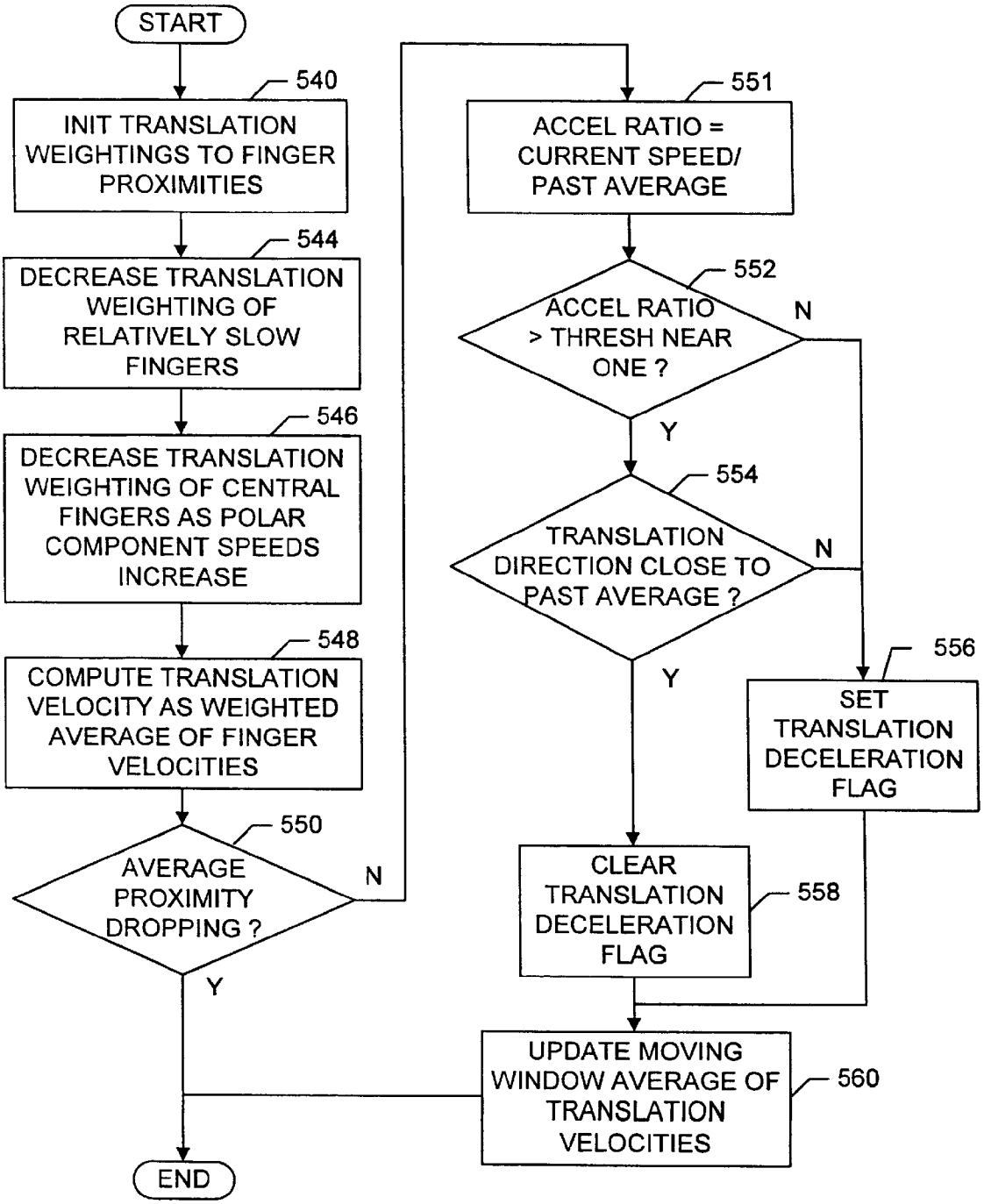

START
540
INIT TRANSLATION WEIGHTINGS TO FINGER PROXIMITIES
544
DECREASE TRANSLATION WEIGHTING OF RELATIVELY SLOW FINGERS
546
DECREASE TRANSLATION WEIGHTING OF CENTRAL FINGERS AS POLAR COMPONENT SPEEDS INCREASE
548
COMPUTE TRANSLATION VELOCITY AS WEIGHTED AVERAGE OF FINGER VELOCITIES
550
AVERAGE PROXIMITY DROPPING ?
N
Y
END
551
ACCEL RATIO = CURRENT SPEED/ PAST AVERAGE
552
ACCEL RATIO > THRESH NEAR ONE ?
N
Y
554
TRANSLATION DIRECTION CLOSE TO PAST AVERAGE ?
N
Y
556
SET TRANSLATION DECELERATION FLAG
CLEAR TRANSLATION DECELERATION FLAG
558
UPDATE MOVING WINDOW AVERAGE OF TRANSLATION VELOCITIES
560

FIG. 37

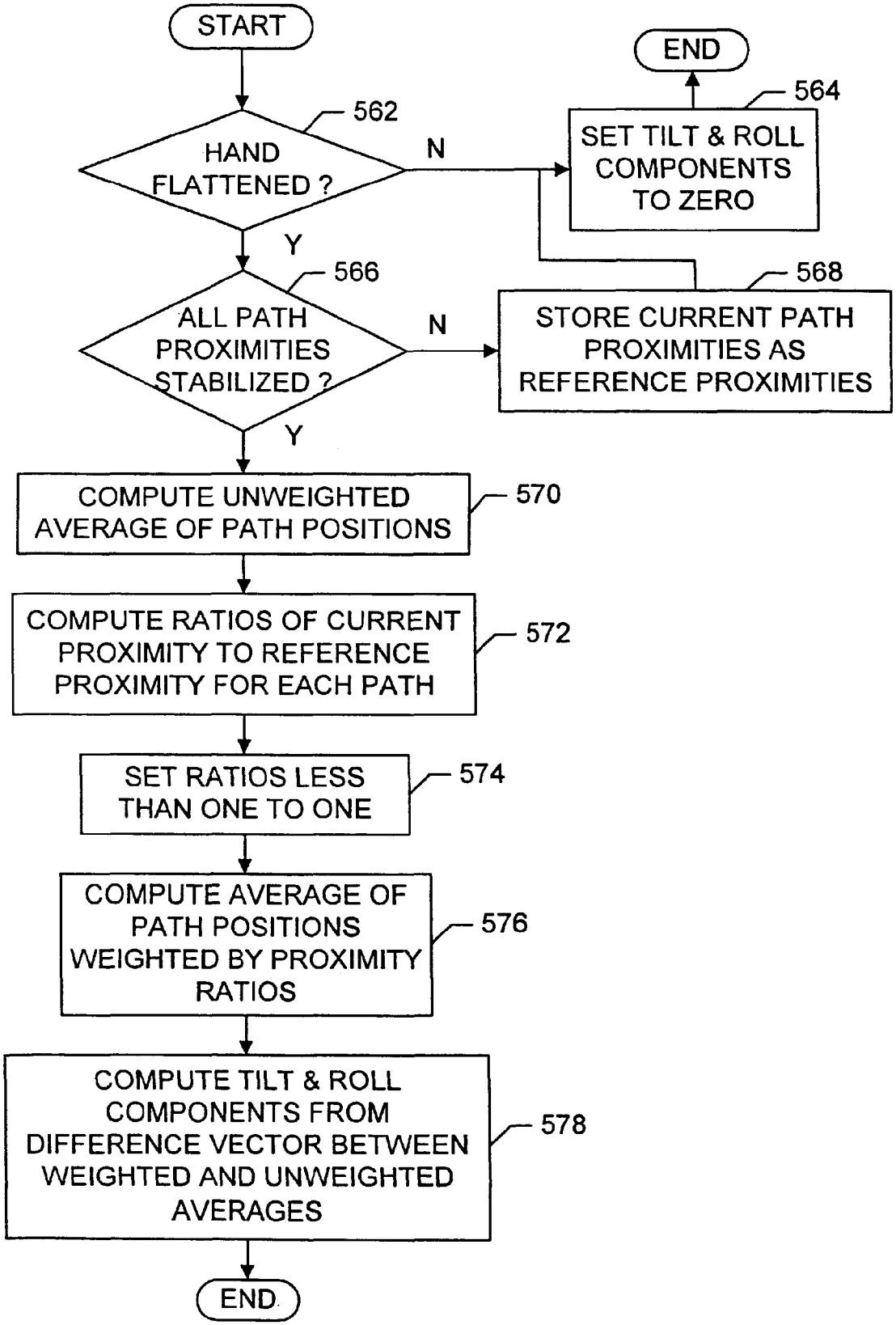

START
HAND FLATTENED ?
562
N
Y
SET TILT & ROLL COMPONENTS TO ZERO
564
END
ALL PATH PROXIMITIES STABILIZED ?
566
N
Y
STORE CURRENT PATH PROXIMITIES AS REFERENCE PROXIMITIES
568
COMPUTE UNWEIGHTED AVERAGE OF PATH POSITIONS
570
COMPUTE RATIOS OF CURRENT PROXIMITY TO REFERENCE PROXIMITY FOR EACH PATH
572
SET RATIOS LESS THAN ONE TO ONE
574
COMPUTE AVERAGE OF PATH POSITIONS WEIGHTED BY PROXIMITY RATIOS
576
COMPUTE TILT & ROLL COMPONENTS FROM DIFFERENCE VECTOR BETWEEN WEIGHTED AND UNWEIGHTED AVERAGES
578
END

FIG. 38

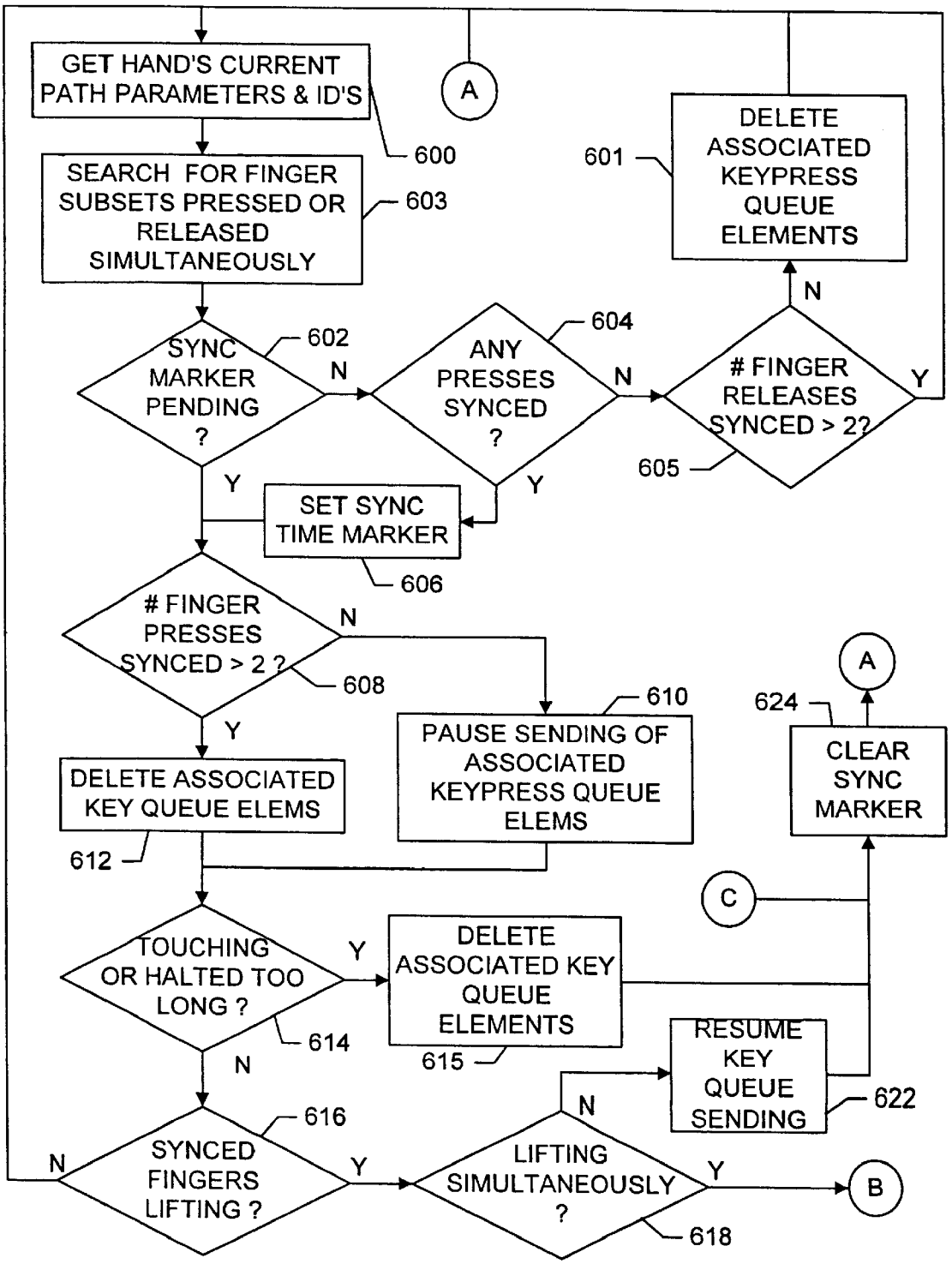

GET HAND'S CURRENT PATH PARAMETERS & ID'S
600
A
SEARCH FOR FINGER SUBSETS PRESSED OR RELEASED SIMULTANEOUSLY
603
DELETE ASSOCIATED KEYPRESS QUEUE ELEMENTS
601
N
SYNC MARKER PENDING ?
602
N
ANY PRESSES SYNCED ?
604
N
FINGER RELEASES SYNCED > 2?
Y
605
Y
Y
SET SYNC TIME MARKER
606
FINGER PRESSES SYNCED > 2 ?
N
608
Y
PAUSE SENDING OF ASSOCIATED KEYPRESS QUEUE ELEMS
610
A
624
CLEAR SYNC MARKER
DELETE ASSOCIATED KEY QUEUE ELEMS
612
C
TOUCHING OR HALTED TOO LONG ?
Y
614
DELETE ASSOCIATED KEY QUEUE ELEMENTS
615
N
RESUME KEY QUEUE SENDING
622
N
SYNCED FINGERS LIFTING ?
616
N
Y
LIFTING SIMULTANEOUSLY ?
Y
618
B

FIG. 39A

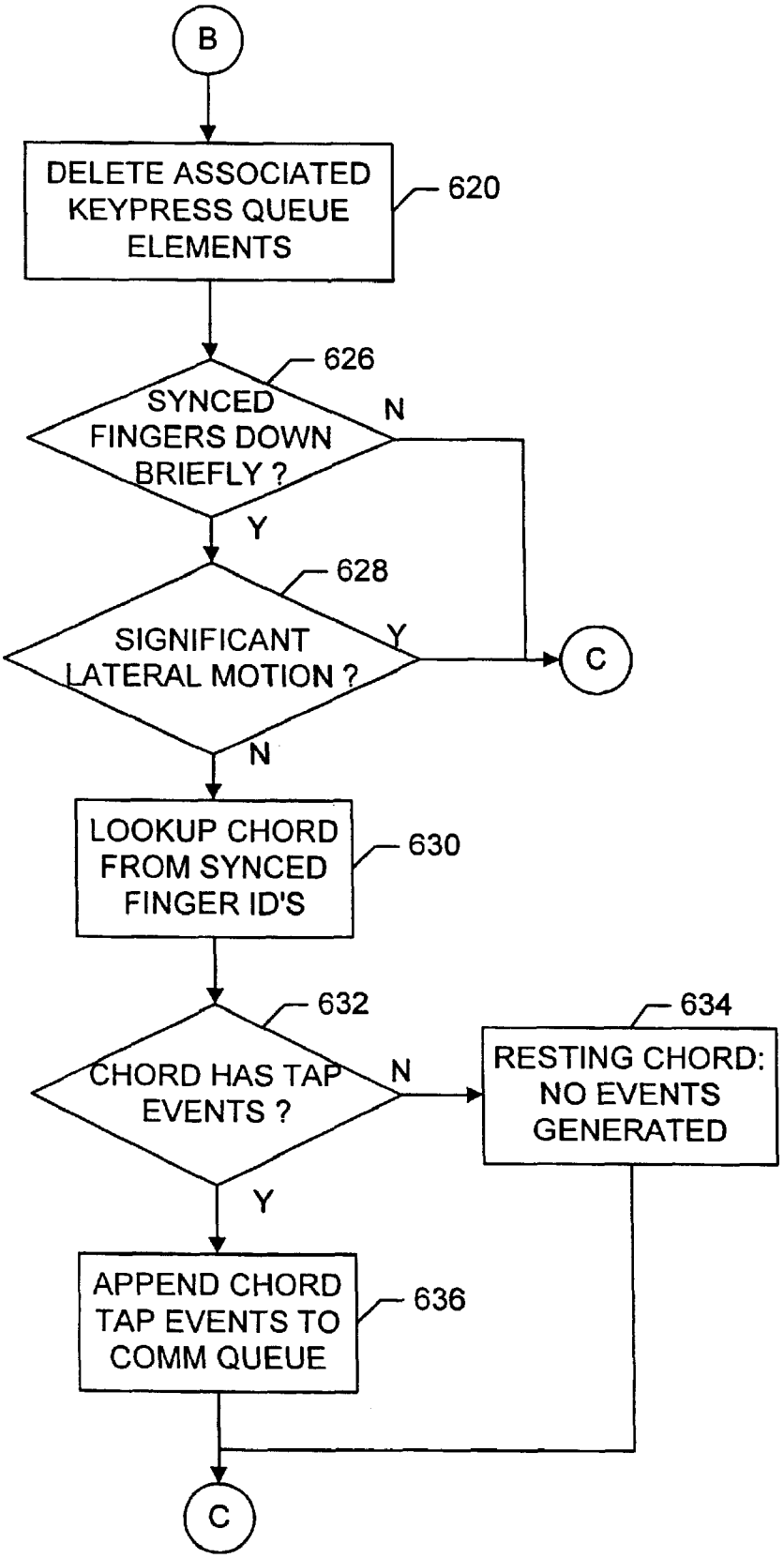

B
DELETE ASSOCIATED KEYPRESS QUEUE ELEMENTS
620
626
SYNCED FINGERS DOWN BRIEFLY ?
N
Y
628
SIGNIFICANT LATERAL MOTION ?
Y
C
N
LOOKUP CHORD FROM SYNCED FINGER ID'S
630
632
CHORD HAS TAP EVENTS ?
N
634
RESTING CHORD: NO EVENTS GENERATED
Y
APPEND CHORD TAP EVENTS TO COMM QUEUE
636
C

FIG. 39B

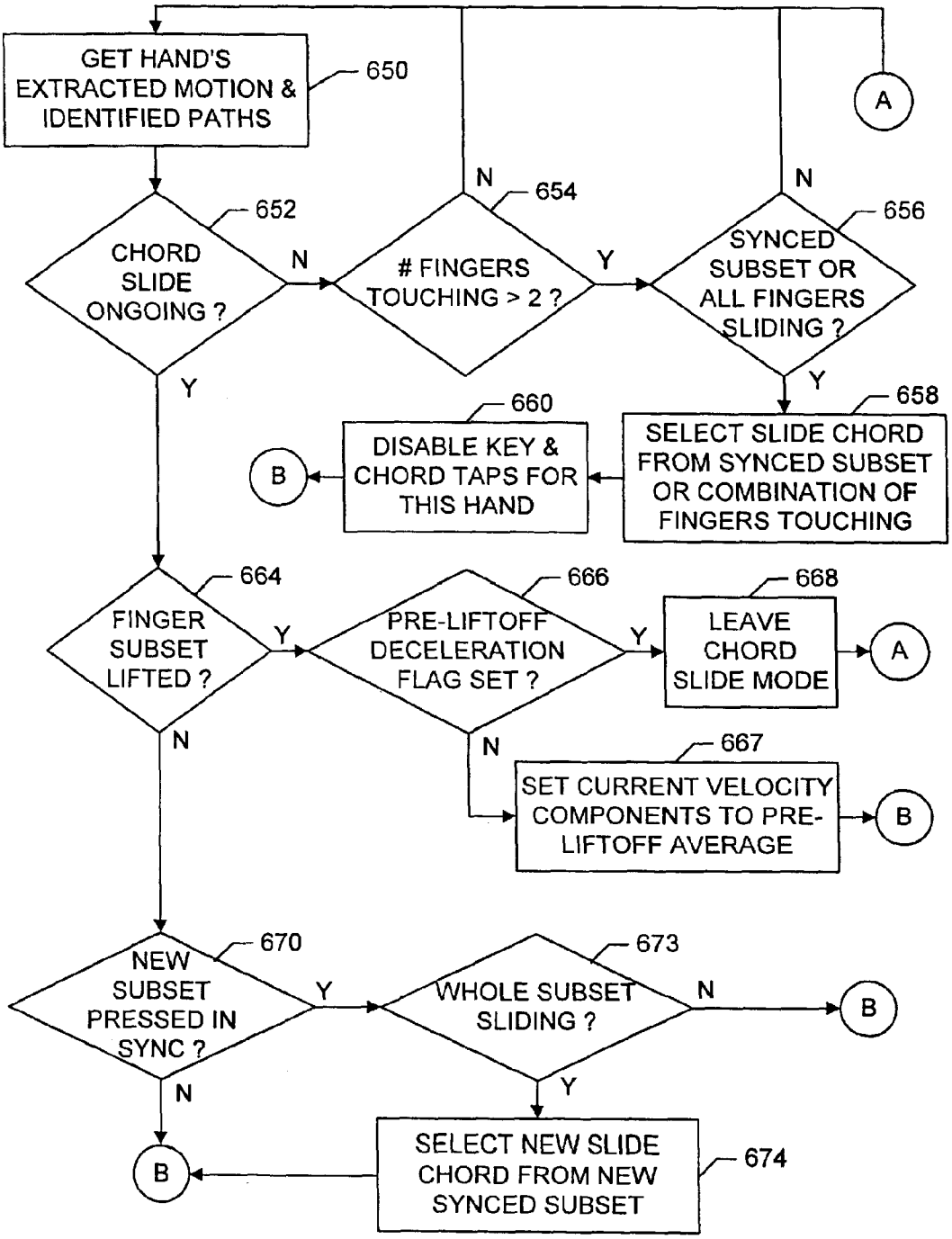

GET HAND'S EXTRACTED MOTION & IDENTIFIED PATHS
650
A
CHORD SLIDE ONGOING ?
652
N
FINGERS TOUCHING > 2 ?
654
Y
SYNCED SUBSET OR ALL FINGERS SLIDING ?
656
Y
DISABLE KEY & CHORD TAPS FOR THIS HAND
660
B
SELECT SLIDE CHORD FROM SYNCED SUBSET OR COMBINATION OF FINGERS TOUCHING
658
FINGER SUBSET LIFTED ?
664
PRE-LIFTOFF DECELERATION FLAG SET ?
666
LEAVE CHORD SLIDE MODE
668
SET CURRENT VELOCITY COMPONENTS TO PRE-LIFTOFF AVERAGE
667
NEW SUBSET PRESSED IN SYNC ?
670
WHOLE SUBSET SLIDING ?
673
SELECT NEW SLIDE CHORD FROM NEW SYNCED SUBSET
674

FIG. 40A

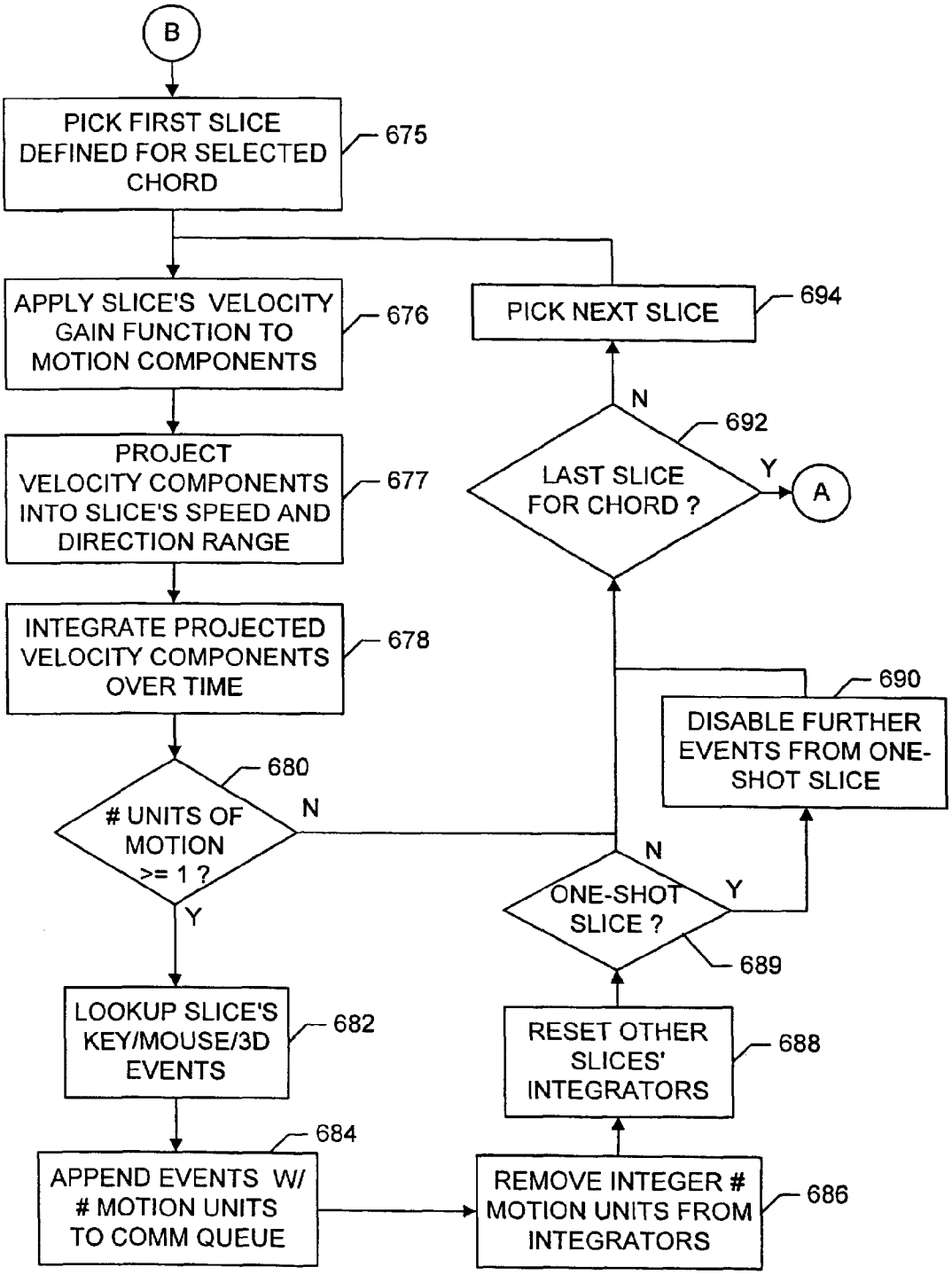

B
PICK FIRST SLICE DEFINED FOR SELECTED CHORD
675
APPLY SLICE'S VELOCITY GAIN FUNCTION TO MOTION COMPONENTS
676
PROJECT VELOCITY COMPONENTS INTO SLICE'S SPEED AND DIRECTION RANGE
677
INTEGRATE PROJECTED VELOCITY COMPONENTS OVER TIME
678
UNITS OF MOTION >= 1 ?
680
N
Y
LOOKUP SLICE'S KEY/MOUSE/3D EVENTS
682
APPEND EVENTS W/ # MOTION UNITS TO COMM QUEUE
684
REMOVE INTEGER # MOTION UNITS FROM INTEGRATORS
686
RESET OTHER SLICES' INTEGRATORS
688
ONE-SHOT SLICE ?
689
N
Y
DISABLE FURTHER EVENTS FROM ONE-SHOT SLICE
690
LAST SLICE FOR CHORD ?
692
N
Y
A
PICK NEXT SLICE
694

FIG. 40B

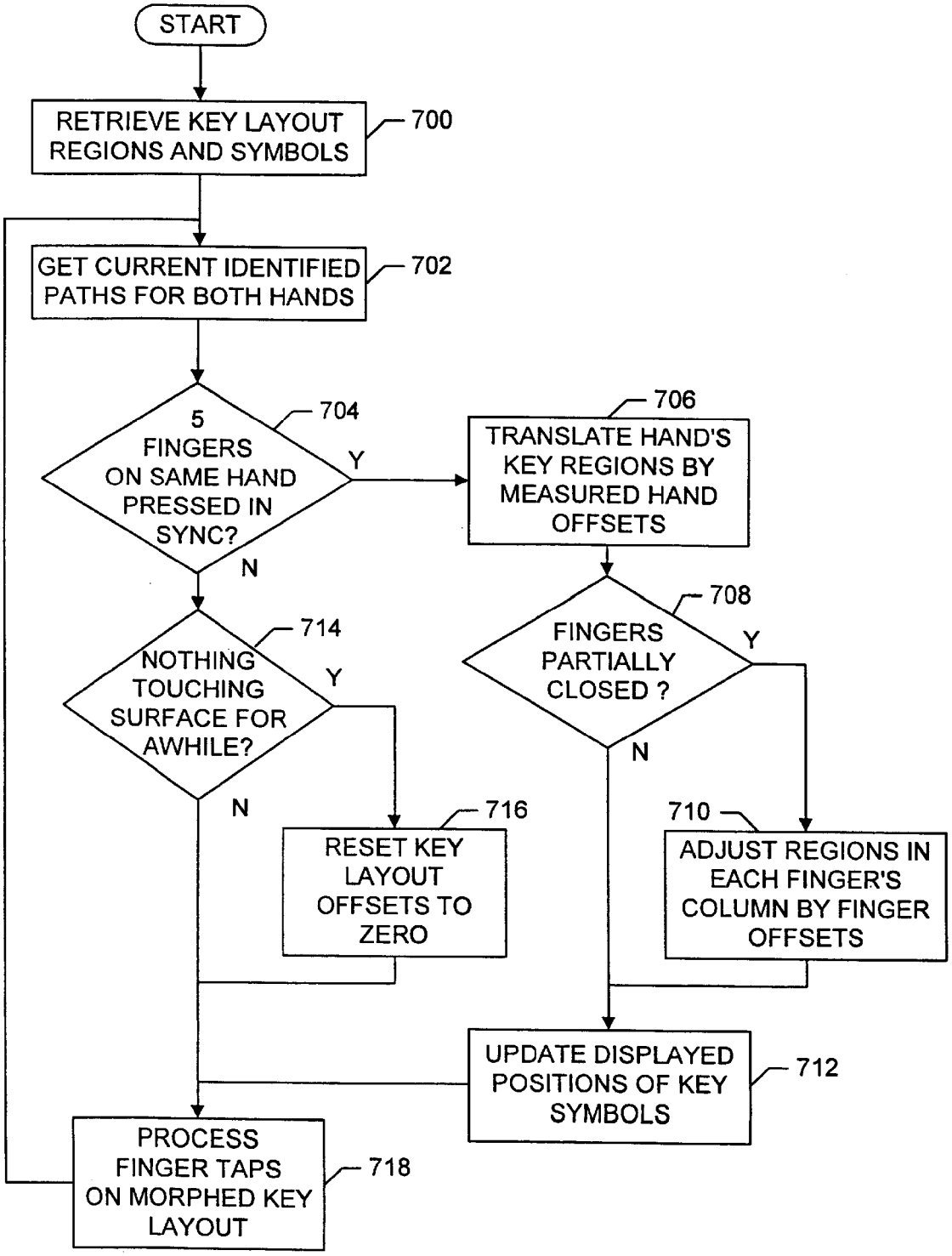

START
RETRIEVE KEY LAYOUT REGIONS AND SYMBOLS
700
GET CURRENT IDENTIFIED PATHS FOR BOTH HANDS
702
5 FINGERS ON SAME HAND PRESSED IN SYNC?
704
Y
N
TRANSLATE HAND'S KEY REGIONS BY MEASURED HAND OFFSETS
706
FINGERS PARTIALLY CLOSED ?
708
Y
N
NOTHING TOUCHING SURFACE FOR AWHILE?
714
Y
N
RESET KEY LAYOUT OFFSETS TO ZERO
716
ADJUST REGIONS IN EACH FINGER'S COLUMN BY FINGER OFFSETS
710
UPDATE DISPLAYED POSITIONS OF KEY SYMBOLS
712
PROCESS FINGER TAPS ON MORPHED KEY LAYOUT
718

FIG. 41

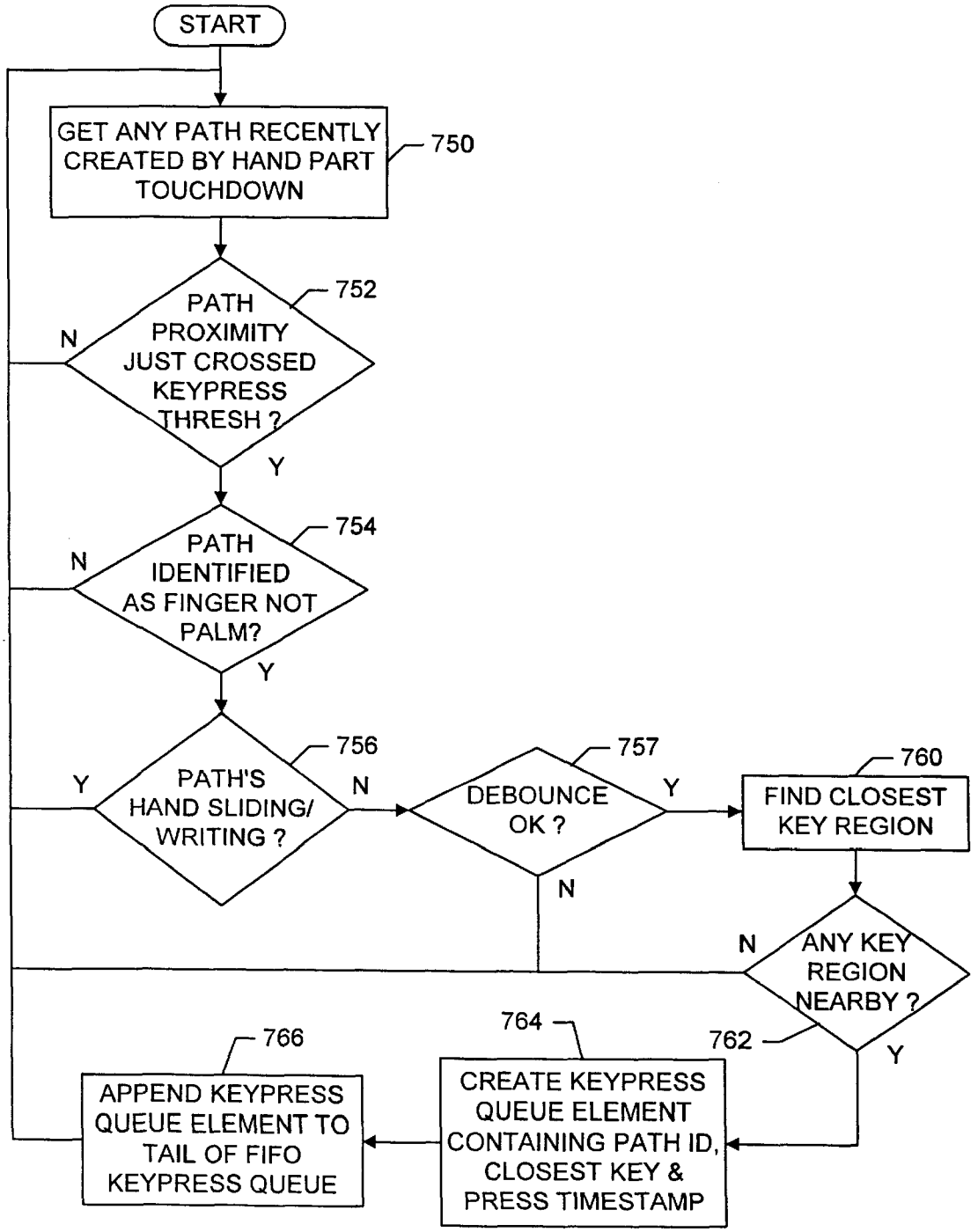

START
GET ANY PATH RECENTLY CREATED BY HAND PART TOUCHDOWN
750
PATH PROXIMITY JUST CROSSED KEYPRESS THRESH ?
752
N
Y
PATH IDENTIFIED AS FINGER NOT PALM?
754
N
Y
PATH'S HAND SLIDING/ WRITING ?
756
Y
N
DEBOUNCE OK ?
757
Y
N
FIND CLOSEST KEY REGION
760
ANY KEY REGION NEARBY ?
762
N
Y
CREATE KEYPRESS QUEUE ELEMENT CONTAINING PATH ID, CLOSEST KEY & PRESS TIMESTAMP
764
APPEND KEYPRESS QUEUE ELEMENT TO TAIL OF FIFO KEYPRESS QUEUE
766

FIG. 42

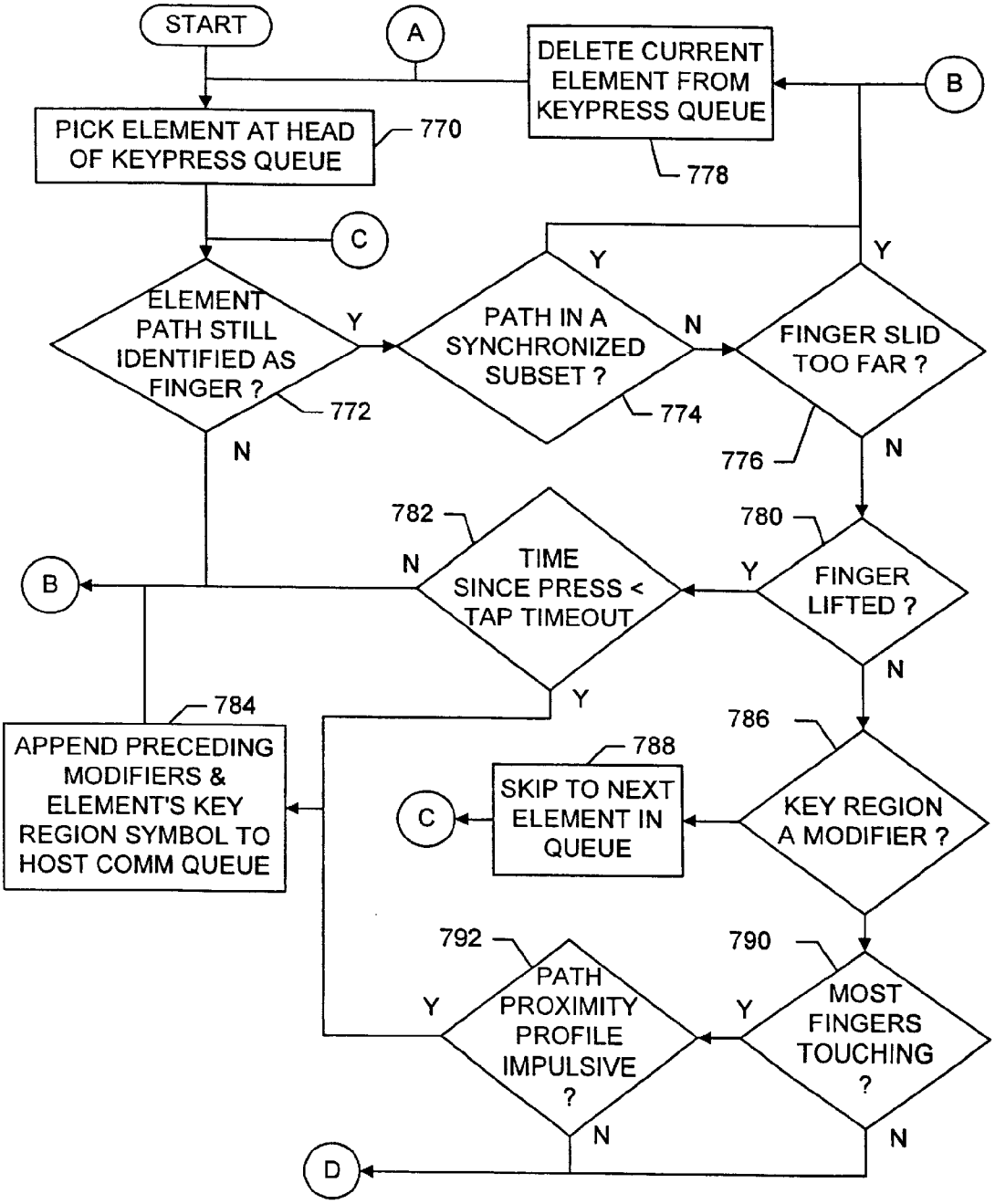

START
A
DELETE CURRENT ELEMENT FROM KEYPRESS QUEUE
B
PICK ELEMENT AT HEAD OF KEYPRESS QUEUE
770
778
C
ELEMENT PATH STILL IDENTIFIED AS FINGER ?
Y
PATH IN A SYNCHRONIZED SUBSET ?
N
FINGER SLID TOO FAR ?
772
774
776
N
782
780
B
N
TIME SINCE PRESS < TAP TIMEOUT
Y
FINGER LIFTED ?
N
Y
784
786
APPEND PRECEDING MODIFIERS & ELEMENT'S KEY REGION SYMBOL TO HOST COMM QUEUE
788
C
SKIP TO NEXT ELEMENT IN QUEUE
KEY REGION A MODIFIER ?
792
790
Y
PATH PROXIMITY PROFILE IMPULSIVE ?
Y
MOST FINGERS TOUCHING ?
N
N
D

FIG. 43A

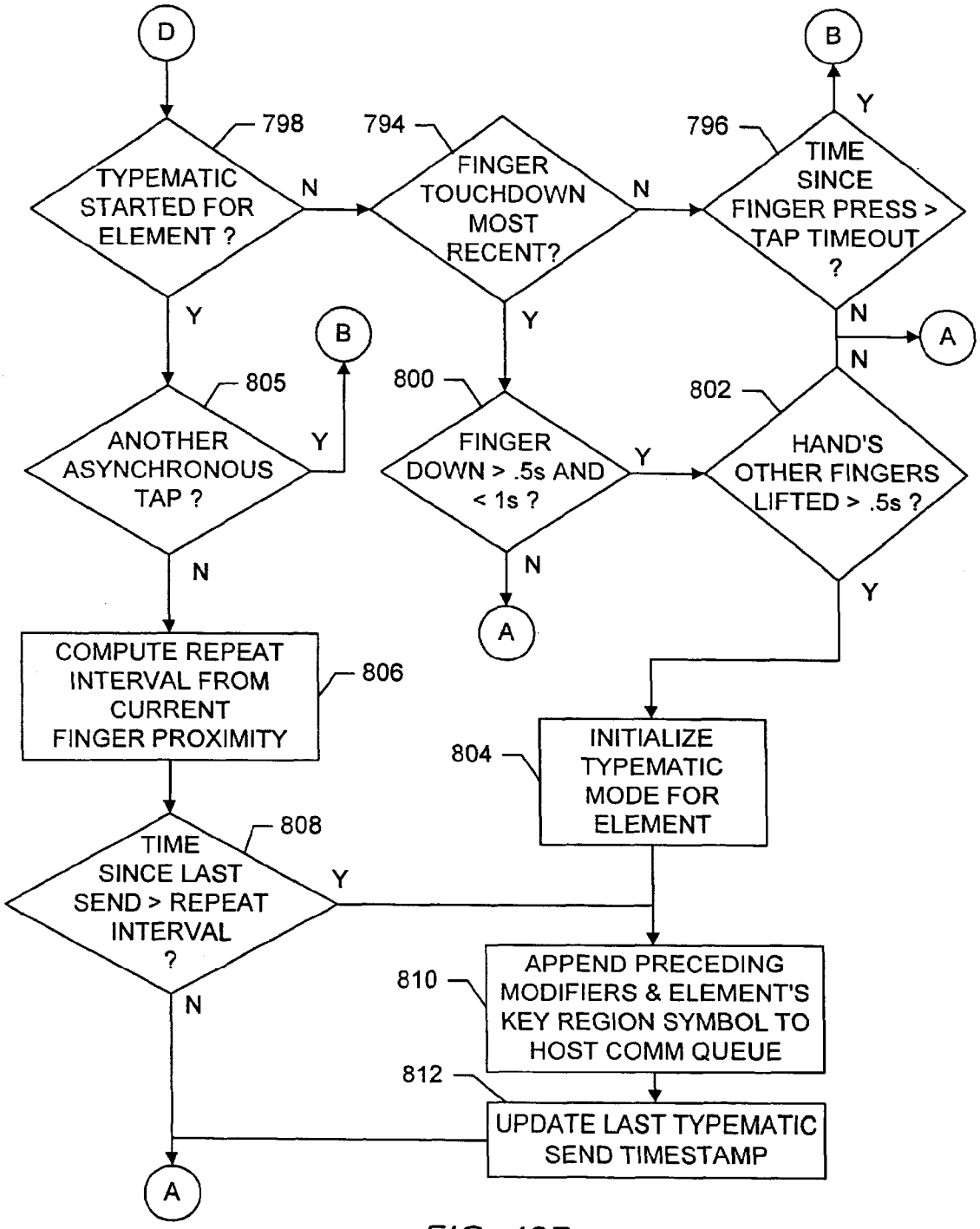

D
798
TYPEMATIC STARTED FOR ELEMENT ?
N
Y
794
FINGER TOUCHDOWN MOST RECENT?
N
Y
796
TIME SINCE FINGER PRESS > TAP TIMEOUT ?
Y
B
N
A
N
B
805
ANOTHER ASYNCHRONOUS TAP ?
Y
N
800
FINGER DOWN > .5s AND < 1s ?
Y
N
A
802
HAND'S OTHER FINGERS LIFTED > .5s ?
Y
COMPUTE REPEAT INTERVAL FROM CURRENT FINGER PROXIMITY
806
804
INITIALIZE TYPEMATIC MODE FOR ELEMENT
808
TIME SINCE LAST SEND > REPEAT INTERVAL ?
Y
N
810
APPEND PRECEDING MODIFIERS & ELEMENT'S KEY REGION SYMBOL TO HOST COMM QUEUE
812
UPDATE LAST TYPEMATIC SEND TIMESTAMP
A

FIG. 43B

ELLIPSE FITTING FOR MULTI-TOUCH SURFACES

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of 11/015,434, entitled "Method and Apparatus for Integrating Manual Input," filed Dec. 17, 2004 now U.S. Pat. No. 7,339,580, which is a continuation of 09/236,513 (now Pat. No. 6,323,846) filed Jan. 25, 1999 which claims the benefit of provisional application 60/072,509, filed Jan. 26, 1998, each of which is hereby incorporated by reference in its entirety. This application is also related to Application Ser. No. 11/428,501, entitled "Capacitive Sensing Arrangement," 11/428,503, entitled "Touch Surface," 11/428,506, entitled "User Interface Gestures," 11/428,515, entitled "User Interface Gestures", 11/428,522, entitled "Identifying Contacts on a Touch Surface," 11/428,521, entitled "Identifying Contacts on a Touch Surface", 11/559,736, entitled "Multi-Touch Contact Tracking Algorithm", 11/559,763, "Multi-Touch Contact Motion Extraction," 11/559,799, entitled "Multi-Touch Contact Motion Extraction," 11/559,822, entitled "Multi-Touch Contact Motion Extraction," 11/559,833, entitled Multi-Touch Hand Position Offset Computation, each of which is hereby incorporated by reference in its entirety.

BACKGROUND OF THE INVENTION

A. Field of the Invention

The present invention relates generally to methods and apparatus for data input, and, more particularly, to a method and apparatus for integrating manual input.

B. Description of the Related Art

Many methods for manual input of data and commands to computers are in use today, but each is most efficient and easy to use for particular types of data input. For example, drawing tablets with pens or pucks excel at drafting, sketching, and quick command gestures. Handwriting with a stylus is convenient for filling out forms which require signatures, special symbols, or small amounts of text, but handwriting is slow compared to typing and voice input for long documents. Mice, finger-sticks and touchpads excel at cursor pointing and graphical object manipulations such as drag and drop. Rollers, thumbwheels and trackballs excel at panning and scrolling. The diversity of tasks that many computer users encounter in a single day call for all of these techniques, but few users will pay for a multitude of input devices, and the separate devices are often incompatible in a usability and an ergonomic sense. For instance, drawing tablets are a must for graphics professionals, but switching between drawing and typing is inconvenient because the pen must be put down or held awkwardly between the fingers while typing. Thus, there is a long-felt need in the art for a manual input device which is cheap yet offers convenient integration of common manual input techniques.

Speech recognition is an exciting new technology which promises to relieve some of the input burden on user hands. However, voice is not appropriate for inputting all types of data either. Currently, voice input is best-suited for dictation of long text documents. Until natural language recognition matures sufficiently that very high level voice commands can be understood by the computer, voice will have little advantage over keyboard hot-keys and mouse menus for command and control. Furthermore, precise pointing, drawing, and manipulation of graphical objects is difficult with voice commands, no matter how well speech is understood. Thus, there

will always be a need in the art for multi-function manual input devices which supplement voice input.

A generic manual input device which combines the typing, pointing, scrolling, and handwriting capabilities of the standard input device collection must have ergonomic, economic, and productivity advantages which outweigh the unavoidable sacrifices of abandoning device specialization. The generic device must tightly integrate yet clearly distinguish the different types of input. It should therefore appear modeless to the user in the sense that the user should not need to provide explicit mode switch signals such as buttonpresses, arm relocations, or stylus pickups before switching from one input activity to another. Epidemiological studies suggest that repetition and force multiply in causing repetitive strain injuries. Awkward postures, device activation force, wasted motion, and repetition should be minimized to improve ergonomics. Furthermore, the workload should be spread evenly over all available muscle groups to avoid repetitive strain.

Repetition can be minimized by allocating to several graphical manipulation channels those tasks which require complex mouse pointer motion sequences. Common graphical user interface operations such as finding and manipulating a scroll bar or slider control are much less efficient than specialized finger motions which cause scrolling directly, without the step of repositioning the cursor over an on-screen control. Preferably the graphical manipulation channels should be distributed amongst many finger and hand motion combinations to spread the workload. Touchpads and mice with auxilliary scrolling controls such as the Cirque®™ Smartcat touchpad with edge scrolling, the IBM®™ ScrollPoint™ mouse with embedded pointing stick, and the Roller Mouse described in U.S. Pat. No. 5,530,455 to Gillick et al. represent small improvements in this area, but still do not provide enough direct manipulation channels to eliminate many often-used cursor motion sequences. Furthermore, as S. Zhai et al. found in "Dual Stream Input for Pointing and Scrolling," Proceedings of CHI '97 Extended Abstracts (1997), manipulation of more than two degrees of freedom at a time is very difficult with these devices, preventing simultaneous panning, zooming and rotating.

Another common method for reducing excess motion and repetition is to automatically continue pointing or scrolling movement signals once the user has stopped moving or lifts the finger. Related art methods can be distinguished by the conditions under which such motion continuation is enabled. In U.S. Pat. No. 4,734,685, Watanabe continues image panning when the distance and velocity of pointing device movement exceed thresholds. Automatic panning is, stopped by moving the pointing device back in the opposite direction, so stopping requires additional precise movements. In U.S. Pat. No. 5,543,591 to Gillespie et al., motion continuation occurs when the finger enters an edge border region around a small touchpad. Continued motion speed is fixed and the direction corresponds to the direction from the center of the touchpad to the finger at the edge. Continuation mode ends when the finger leaves the border region or lifts off the pad. Disadvantageously, users sometimes pause at the edge of the pad without intending for cursor motion to continue, and the unexpected motion continuation becomes annoying. U.S. Pat. No. 5,327,161 to Logan et al. describes motion continuation when the finger enters a border area as well, but in an alternative trackball emulation mode, motion continuation can be a function solely of lateral finger velocity and direction at liftoff. Motion continuation decays due to a friction factor or can be stopped by a subsequent touchdown on the surface. Disadvantageously, touch velocity at liftoff is not a reliable indicator of the user's desire for motion continuation since

when approaching a large target on a display at high speeds the user may not stop the pointer completely before liftoff. Thus it would be an advance in the art to provide a motion continuation method which does not become activated unexpectedly when the user really intended to stop pointer movement at a target but happens to be on a border or happens to be moving at significant speed during liftoff.

Many attempts have been made to embed pointing devices in a keyboard so the hands do not have to leave typing position to access the pointing device. These include the integrated pointing key described in U.S. Pat. No. 5,189,403 to Franz et al., the integrated pointing stick disclosed by J. Rutledge and T. Selker in "Force-to-Motion Functions for Pointing," Human-Computer Interaction—INTERACT '90, pp. 701-06 (1990), and the position sensing keys described in U.S. Pat. No. 5,675,361 to Santilli. Nevertheless, the limited movement range and resolution of these devices, leads to poorer pointing speed and accuracy than a mouse, and they add mechanical complexity to keyboard construction. Thus there exists a need in the art for pointing methods with higher resolution, larger movement range, and more degrees of freedom yet which are easily accessible from typing hand positions.

Touch screens and touchpads often distinguish pointing motions from emulated button clicks or keypresses by assuming very little lateral fingertip motion will occur during taps on the touch surface which are intended as clicks. Inherent in these methods is the assumption that tapping will usually be straight down from the suspended finger position, minimizing those components of finger motion tangential to the surface. This is a valid assumption if the surface is not finely divided into distinct key areas or if the user does a slow, "hunt and peck" visual search for each key before striking. For example, in U.S. Pat. No. 5,543,591 to Gillespie et al., a touchpad sends all lateral motions to the host computer as cursor movements. However, if the finger is lifted soon enough after touchdown to count as a tap and if the accumulated lateral motions are not excessive, any sent motions are undone and a mouse button click is sent instead. This method only works for mouse commands such as pointing which can safely be undone, not for dragging or other manipulations. In U.S. Pat. No. 5,666,113 to Logan, taps with less than about 1/16" lateral motion activate keys on a small keypad while lateral motion in excess of 1/16" activates cursor control mode. In both patents cursor mode is invoked by default when a finger stays on the surface a long time.

However, fast touch typing on a surface divided into a large array of key regions tends to produce more tangential motions along the surface than related art filtering techniques can tolerate. Such an array contains keys in multiple rows and columns which may not be directly under the fingers, so the user must reach with the hand or flex or extend fingers to touch many of the key regions. Quick reaching and extending imparts significant lateral finger motion while the finger is in the air which may still be present when the finger contacts the surface. Glancing taps with as much as 1/4" lateral motion measured at the surface can easily result. Attempting to filter or suppress this much motion would make the cursor seem sluggish and unresponsive. Furthermore, it may be desirable to enter a typematic or automatic key repeat mode instead of pointing mode when the finger is held in one place on the surface. Any lateral shifting by the fingertip during a prolonged finger press would also be picked up as cursor jitter without heavy filtering. Thus, there is a need in the art for a method to distinguish keying from pointing on the same surface via more robust hand configuration cues than lateral motion of a single finger.

An ergonomic typing system should require minimal key tapping force, easily distinguish finger taps from resting hands, and cushion the fingers from the jarring force of surface impact. Mechanical and membrane keyboards rely on the spring force in the keyswitches to prevent activation when the hands are resting on the keys. This causes an irreconcilable tradeoff between the ergonomic desires to reduce the fatigue from key activating force and to relax the full weight of the hands onto the keys during rest periods. Force minimization on touch surfaces is possible with capacitive or active optical sensing, which do not rely on finger pressure, rather than resistive-membrane or surface-acoustic-wave sensing techniques. The related art touch devices discussed below will become confused if a whole hand including its four fingertips a thumb and possibly palm heels, rests on the surface. Thus, there exists a long felt need in the art for a multi-touch surface typing system based on zero-force capacitive sensing which can tolerate resting hands and a surface cushion.

An ergonomic typing system should also adapt to individual hand sizes tolerate variations in typing style, and support a range of healthy hand postures. Though many ergonomic keyboards have been proposed, mechanical keyswitches can only be repositioned at great cost. For example, the keyboard with concave keywells described by Hargreaves et al. in U.S. Pat. No. 5,689,253 fits most hands well but also tends to lock the arms in a single position. A touch surface key layout could easily be morphed, translated, or arbitrarily reconfigured as long as the changes did not confuse the user. However, touch surfaces may not provide as much laterally orienting tactile feedback as the edges of mechanical keyswitches. Thus, there exists a need in the art for a surface typing recognizer which can adapt a key layout to fit individual hand postures and which can sustain typing accuracy if the hands drift due to limited tactile feedback.

Handwriting on smooth touch surfaces using a stylus is well-known in the art, but it typically does not integrate well with typing and pointing because the stylus must be put down somewhere or held awkwardly during other input activities. Also, it may be difficult to distinguish the handwriting activity of the stylus from pointing motions of a fingertip. Thus there exists a need in the art for a method to capture coarse handwriting gestures without a stylus and without confusing them with pointing motions.

Many of the input differentiation needs cited above could be met with a touch sensing technology which distinguishes a variety of hand configurations and motions such as sliding finger chords and grips. Many mechanical chord keyboards have been designed to detect simultaneous downward activity from multiple fingers, but they do not detect lateral finger motion over a large range. Related art shows several examples of capacitive touchpads which emulate a mouse or keyboard by tracking a single finger. These typically measure the capacitance of or between elongated wires which are laid out in rows and columns. A thin dielectric is interposed between the row and column layers. Presence of a finger perturbs the self or mutual capacitance for nearby electrodes. Since most of these technologies use projective row and column sensors which integrate on one electrode the proximity of all objects in a particular row or column, they cannot uniquely determine the positions of two or more objects as discussed in S. Lee, "A Fast Multiple-Touch-Sensitive Input Device," University of Toronto Masters Thesis (1984). The best they can do is count fingertips which happen to lie in a straight row, and even that will fail if a thumb or palm is introduced in the same column as a fingertip.

In U.S. Pat. Nos. 5,565,658 and 5,305,017, Gerpheide et al. measure the mutual capacitance between row and column electrodes by driving one set of electrodes at some clock frequency and sensing how much of that frequency is coupled onto a second electrode set. Such synchronous measurements are very prone to noise at the driving frequency, so to increase signal-to-noise ratio they form virtual electrodes comprised of multiple rows or multiple columns, instead of a single row and column, and scan through electrode combinations until the various mutual capacitances are nulled or balanced. The coupled signal increases with the product of the rows and columns in each virtual electrodes, but the noise only increases with the sum, giving a net gain in signal-to-noise ratio for virtual electrodes consisting of more than two rows and two columns. However, to uniquely distinguish multiple objects, virtual electrode sizes would have to be reduced so the intersection of the row and column virtual electrodes would be no larger than a finger tip, i.e., about two rows and two columns, which will degrade the signal-to-noise ratio. Also, the signal-to-noise ratio drops as row and column lengths increase to cover a large area.

In U.S. Pat. Nos. 5,543,591, 5,543,590, and 5,495,077, Gillespie et al measure the electrode-finger self-capacitance for row and column electrodes independently. Total electrode capacitance is estimated by measuring the electrode voltage change caused by injecting or removing a known amount of charge in a known time. All electrodes can be measured simultaneously if each electrode has its own drive/sense circuit. The centroid calculated from all row and column electrode signals establishes an interpolated vertical and horizontal position for a single object. This method may in general have higher signal-to-noise ratio than synchronous methods, but the signal-to-noise ratio is still degraded as row and column lengths increase. Signal-to-noise ratio is especially important for accurately locating objects which are floating a few millimeters above the pad. Though this method can detect such objects, it tends to report their position as being near the middle of the pad, or simply does not detect floating objects near the edges.

Thus there exists a need in the art for a capacitance-sensing apparatus which does not suffer from poor signal-to-noise ratio and the multiple finger indistinguishability problems of touchpads with long row and column electrodes.

U.S. Pat. No. 5,463,388 to Boie et al. has a capacitive sensing system applicable to either keyboard or mouse input, but does not consider the problem of integrating both types of input simultaneously. Though they mention independent detection of arrayed unit-cell electrodes, their capacitance transduction circuitry appears too complex to be economically reproduced at each electrode. Thus the long lead wires connecting electrodes to remote signal conditioning circuitry can pickup noise and will have significant capacitance compared to the finger-electrode self-capacitance, again limiting signal-to-noise ratio. Also, they do not recognize the importance of independent electrodes for multiple finger tracking, or mention how to track multiple fingers on an independent electrode array.

Lee built an early multi-touch electrode array, with 7 mm by 4 mm metal electrodes arranged in 32 rows and 64 columns. The "Fast Multiple-Touch-Sensitive Input Device (FMTSID)" total active area measured 12" by 16", with a 0.075 mm Mylar dielectric to insulate fingers from electrodes. Each electrode had one diode connected to a row charging line and a second diode connected to a column discharging line. Electrode capacitance changes were measured singly or in rectangular groups by raising the voltage on one or more row lines, selectively charging the electrodes in those rows, and then timing the discharge of selected columns to ground through a discharge resistor. Lee's design required only two diodes per electrode, but the principal disadvantage of Lee's design is that the column diode reverse bias capacitances allowed interference between electrodes in the same column.

All of the related capacitance sensing art cited above utilize interpolation between electrodes to achieve high pointing resolution with economical electrode density. Both Boie et al. and Gillespie et al. discuss compultattion of a centroid from all row and column electrode readings. However, for multiple finger detection, centroid calculation must be carefully limited around local maxima to include only one finger at a time. Lee utilizes a bisective search technique to find local maxima and then interpolates only on the eight nearest neighbor electrodes of each local maximum electrode. This may work fine for small fingertips, but thumb and palm contacts may cover more than nine electrodes. Thus there exists a need in the art for improved means to group exactly those electrodes which are covered by each distinguishable hand contact and to compute a centroid from such potentially irregular groups.

To take maximum advantage of multi-touch surface sensing, complex proximity image processing is necessary to track and identify the parts of the hand contacting the surface at any one time. Compared to passive optical, images, proximity images provide clear indications of where the body contacts the surface, uncluttered by luminosity variation and extraneous objects in the background. Thus proximity image filtering and segmentation stages can be simpler and more reliable than in computer vision approaches to free-space hand tracking such as S. Alimad, "A Usable Real-Time 3D Hand Tracker," Proceedings of the 28th Asilomar Conference on Signals, Systems, and Computers—Part 2, vol. 2, IEEE (1994) or Y. Cui and J. Wang, "Hand Segmentation Using Learning-Based Prediction and Verification for Hand Sign Recognition," Proceedings of the 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 88-93 (1996). However, parts of the hand such as intermediate finger joints and the center of the palms do not show up in capacitive proximity images at all if the hand is not flattened on the surface. Without these intermediate linkages between fingertips and palms the overall hand structure can only be guessed at, making hand contact identification very difficult. Hence the optical flow and contour tracking techniques which have been applied to free-space hand sign language recognition as in F. Quek, "Unencumbered Gestural Interaction," IEEE Multimedia, vol. 3, pp. 36-47 (1996), do not address the special challenges of proximity image tracking.

Synaptics Corp. has successfully fabricated their electrode array on flexible mylar film rather than stiff circuit board. This is suitable for conforming to the contours of special products, but does not provide significant finger cushioning for large surfaces. Even if a cushion was placed under the film, the lack of stretchability in the film, leads, and electrodes would limit the compliance afforded by the compressible material. Boie et al suggests that placing cornressihle insulators on top of the electrode array cushions finger impact. However, an insulator more than about one millimeter thick would seriously attenuate the measured finger-electrode capacitances. Thus there exists a need in the art for a method to transfer finger capacitance influences through an arbitrarily thick cushion.

SUMMARY OF THE INVENTION

It is a primary object of the present invention to provide a system and method for integrating different types of manual

input such as typing, multiple degree-of-freedom manipulation, and handwriting on a multi-touch surface.

It is also an object of the present invention to provide a system and method for distinguishing different types of manual input such as typing, multiple degree-of-freedom manipulation, and handwriting on a multi-touch surface, via different hand configurations which are easy for the user to learn and easy for the system to recognize.

It is a further object of the present invention to provide an improved capacitance-transducing apparatus that is cheaply implemented near each electrode so that two-dimensional sensor arrays of arbitrary size and resolution can be built without degradation in signal to noise.

It is a further object of the present invention to provide an electronic system which minimizes the number of sensing electrodes necessary to obtain proximity images with such resolution that a variety of hand configurations can be distinguished.

Yet another object of the present invention is to provide a multi-touch surface apparatus which is compliant and contoured to be comfortable and ergonomic under extended use.

Yet another object of the present invention is to provide tactile key or hand position feedback without impeding hand resting on the surface or smooth, accurate sliding across the surface.

It is a further object of the present invention to provide an electronic system which can provide images of flesh proximity to an array of sensors with such resolution that a variety of hand configurations can be distinguished.

It is another object of the present invention to provide an improved method for invoking cursor motion continuation only when the user wants it by not invoking it when significant deceleration is detected.

Another object of the present invention is to identify different hand parts as they contact the surface so that a variety of hand configurations can be recognized and used to distinguish different kinds of input activity.

Yet another object of the present invention is to reliably extract rotation and scaling as well as translation degrees of freedom from the motion of two or more hand contacts to aid in navigation and manipulation of two-dimensional electronic documents.

It is a further object of the present invention to reliably extract tilt and roll degrees of freedom from hand pressure differences to aid in navigation and manipulation of three-dimensional environments.

Additional objects and advantages of the invention will be set forth in part in the description which follows, and in part will be obvious from the description, or may be learned by practice of the invention. The objects and advantages of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims.

To achieve the objects and in accordance with the purpose of the invention, as embodied and broadly described herein, the invention comprises a sensing device that is sensitive to changes in self-capacitance brought about by changes in proximity of a touch device to the sensing device, the sensing device comprising: two electrical switching means connected together in series having a common node, an input node, and an output node; a dielectric-covered sensing electrode connected to the common node between the two switching means; a power supply providing an approximately constant voltage connected to the input node of the series-connected switching means; an integrating capacitor to accumulate charge transferred during multiple consecutive switchings of the series connected switching means; another switching means connected in parallel across the integrating capacitor to deplete its residual charge; and a voltage-to-voltage translation device connected to the output node of the series-connected switching means which produces a voltage representing the magnitude of the self-capacitance of the sensing device. Alternatively, the sensing device comprises: two electrical switching means connected together in series having a common node, an input node, and an output node; a dielectric-covered sensing electrode connected to the common node between the two switching means; a power supply providing an approximately constant voltage connected to the input node of the series-connected switching means; and an integrating current-to-voltage translation device connected to the output node of the series connected switching means, the current-to-voltage translation device producing a voltage representing the magnitude of the self-capacitance of the sensing device.

To further achieve the objects, the present invention comprises a multi-touch surface apparatus for detecting a spatial arrangement of multiple touch devices on or near the surface of the multi-touch apparatus, comprising: one of a rigid or flexible surface; a plurality of two-dimensional arrays of one of the sensing devices (recited in the previous paragraph) arranged on the surface in groups wherein the sensing devices within a group have their output nodes connected together and share the same integrating capacitor, charge depletion switch, and voltage-to-voltage translation circuitry; control circuitry for enabling a single sensor device from each two-dimensional array; means for selecting the sensor voltage data from each two-dimensional array; voltage measurement circuitry to convert sensor voltage data to a digital code; and circuitry for communicating the digital code to another electronic device. The sensor voltage data selecting means comprises one of a multiplexing circuitry and a plurality of voltage measurement circuits.

To still further achieve the objects, the present invention comprises a multi-touch surface apparatus for sensing diverse configurations and activities of touch devices and generating integrated manual input to one of an electronic or electromechanical device, the apparatus comprising: an array of one of the proximity sensing devices described above; a dielectric cover having symbols printed thereon that represent action-to-be-taken when engaged by the touch devices; scanning means for forming digital proximity images from the array of sensing devices; calibrating means for removing background offsets from the proximity images; recognition means for interpreting the configurations and activities of the touch devices that make up the proximity images; processing means for generating input signals in response to particular touch device configurations and motions; and communication means for sending the input signals to the electronic or electromechanical device.

To even further achieve the objects, the present invention comprises a multi-touch surface apparatus for sensing diverse configurations and activities of fingers and palms of one or more hands near the surface and generating integrated manual input to one of an electronic or electromechanical device, the apparatus comprising: an array of proximity sensing means embedded in the surface; scanning means for forming digital proximity images from the proximities measured by the sensing means; image segmentation means for collecting into groups those proximity image pixels intensified by contact of the same distinguishable part of a hand; contact tracking means for parameterizing hand contact features and trajectories as the contacts move across successive proximity images, contact identification means for determining which hand and which part of the hand is causing each surface contact; syn-

chronization detection means for identifying subsets of identified contacts which touchdown or liftoff the surface at approximately the same time, and for generating command signals in response to synchronous taps of multiple fingers on the surface; typing recognition means for generating intended key symbols from asynchronous finger taps; motion component extraction means for compressing multiple degrees of freedom of multiple fingers into degrees of freedom common in two and three dimensional graphical manipulation; chord motion recognition means for generating one of command and cursor manipulation signals in response to motion in one or more extracted degrees of freedom by a selected combination of fingers; pen grip detection means for recognizing contact arrangements which resemble the configuration of the hand when gripping a pen, generating inking signals from motions of the inner fingers, and generating cursor manipulation signals from motions of the palms while the inner fingers are lifted; and communication means for sending the sensed configurations and activities of finger and palms to one of the electronic and electromechanical device.

To further achieve the objects, the present invention comprises a method for tracking and identifying hand contacts in a sequence of proximity images in order to support interpretation of hand configurations and activities related to typing, multiple degree-of-freedom manipulation via chords, and handwriting, the method comprising the steps of: segmenting each proximity image into groups of electrodes which indicate significant proximity, each group representing proximity of a distinguishable hand part or other touch device; extracting total proximity, position, shape, size, and orientation parameters from each group of electrodes; tracking group paths through successive proximity images including detection of path endpoints at contact touchdown and liftoff; computing velocity and filtered position vectors along each path; assigning a hand and finger identity to each contact path by incorporating relative path positions and velocities, individual contact features, and previous estimates of hand and finger positions; and maintaining estimates of hand and finger positions from trajectories of paths currently assigned to the fingers, wherein the estimates provide high level feedback to bias segmentations and identifications in future images.

To still further achieve the objects, the present invention comprises a method for integrally extracting multiple degrees of freedom of hand motion from sliding motions of two or more fingers of a hand across a multi-touch surface, one of the fingers preferably being the opposable thumb, the method comprising the steps of: tracking across successive scans of the proximity sensor array the trajectories of individual hand parts on the surface; finding an innermost and an outermost finger contact from contacts identified as fingers on the given hand; computing a scaling velocity component from a change in a distance between the innermost and outermost finger contacts; computing a rotational velocity component from a change in a vector angle between the innermost and outermost finger contacts; computing a translation weighting for each contacting finger; computing translational velocity components in two dimensions from a translation weighted average of the finger velocities tangential to surface; suppressively filtering components whose speeds are consistently lower than the fastest components; transmitting the filtered velocity components as control signals to an electronic or electromechanical device.

To even further achieve the objects, the present invention comprises a manual input integration method for supporting diverse hand input activities such as resting the hands, typing, multiple degree-of-freedom manipulation, command gesturing and handwriting on a multi-touch surface, the method enabling users to instantaneously switch between the input activities by placing their hands in different configurations comprising distinguishable combinations of relative hand contact timing, proximity, shape, size, position, motion and/or identity across a succession of surface proximity images, the method comprising the steps of: tracking each touching hand part across successive proximity images; measuring the times when each hand part touches down and lifts off the surface; detecting when hand parts touch down or lift off simultaneously; producing discrete key symbols when the user asynchronously taps, holds, or slides a finger on key regions defined on the surface; producing discrete mouse button click commands, key commands, or no signals when the user synchronously taps two or more fingers from the same hand on the surface; producing gesture commands or multiple degree-of-freedom manipulation signals when the user slides two or more fingers across the surface; and sending the produced symbols, commands and manipulation signals as input to an electronic or an electro-mechanical device.

To still even further achieve the objects, the present invention comprises a method for choosing what kinds of input signals will be generated and sent to an electronic or electromechanical device in response to tapping or sliding of fingers on a multi-touch surface, the method comprising the following steps: identifying each contact on the surface as either a thumb, fingertip or palm; measuring the times when each hand part touches down and lifts off the surface; forming a set of those fingers which touch down from the all finger floating state before any one of the fingers lifts back off the surface; choosing the kinds of input signals to be generated by further distinctive motion of the fingers from the combination of finger identities in the set; generating input signals of this kind when further distinctive motions of the fingers occur; forming a subset any two or more fingers which touch down synchronously after at least one finger has lifted back off the surface; choosing a new kinds of input signals to be generated by further distinctive motion of the fingers from the combination of finger identities in the subset; generating input signals of this new kind when further distinctive motions of the fingers occur; and continuing to form new subsets, choose and generate new kinds of input signals in response to liftoff and synchronous touchdowns until all fingers lift off the surface.

To further achieve the objects, the present invention comprises a method for continuing generation of cursor movement or scrolling signals from a tangential motion of a touch device over a touch-sensitive input device surface after touch device liftoff from the surface if the touch device operator indicates that cursor movement continuation is desired by accelerating or failing to decelerate the tangential motion of the touch device before the touch device is lifted, the method comprising the following steps: measuring, storing and transmitting to a computing device two or more representative tangential velocities during touch device manipulation; computing and storing a liftoff velocity from touch device positions immediately prior to the touch device liftoff; comparing the liftoff velocity with the representative tangential velocities, and entering a mode for continuously moving the cursor if a tangential liftoff direction approximately equals the representative tangential directions and a tangential liftoff speed is greater than a predetermined fractional multiple of representative tangential speeds; continuously transmitting cursor movement signals after liftoff to a computing device such that the cursor movement velocity corresponds to one of the representative tangential velocities; and ceasing transmission of the cursor movement signals when the touch device engages the surface again, if comparing means detects significant

deceleration before liftoff, or if the computing device replies that the cursor can move no farther or a window can scroll no farther.

It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention as claimed.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate several embodiments of the invention and together with the description, serve to explain the principles of the invention. In the drawings:

FIG. **1** is a block diagram of the integrated manual input apparatus;

FIG. **2** is a schematic drawing of the proximity sensor with voltage amplifier;

FIG. **3** is a schematic drawing of the proximity sensor with integrating current amplifier;

FIG. **4** is a schematic drawing of the proximity sensor implemented with field effect transistors;

FIG. **5** is a schematic drawing of the proximity sensor as used to implement 2D arrays of proximity sensors;

FIG. **6** is a block diagram showing a typical architecture for a 2D array of proximity sensors where all sensors share the same amplifier;

FIG. **7** is a block diagram of circuitry used to convert proximity sensor output to a digital code;

FIG. **8** is a block diagram showing a typical architecture for a 2D array of proximity sensors where sensors within a row share the same amplifier;

FIG. **9** is a schematic of a circuit useful for enabling the output gates of all proximity sensors within a group (arranged in columns);

FIG. **10** is a side view of a 2D proximity sensor array that is sensitive to the pressure exerted by non-conducting touch objects;

FIG. **11** is a, side view of a 2D proximity sensor array that provides a compliant surface without loss of spatial sensitivity;

FIG. **12** is a side view of a 2D proximity sensor array that is sensitive to both the proximity of conducting touch objects and to the pressure exerted by non-conducting touch objects;

FIG. **13** is an example proximity image of a hand flattened onto the surface with fingers outstretched;

FIG. **14** is an example proximity image of a hand partially closed with fingertips normal to surface;

FIG. **15** is an example proximity image of a hand in the pen grip configuration with thumb and index fingers pinched;

FIG. **16** is a data flow diagram of the hand tracking and contact identification system;

FIG. **17** is a flow chart of hand position estimation:

FIG. **18** is a data flow diagram of proximity image segmentation;

FIG. **19** is a diagram of the boundary search pattern during construction of an electrode group;

FIG. **20**A is a diagram of the segmentation strictness regions with both hands in their neutral, default position on surface;

FIG. **20**B is a diagram of the segmentation strictness regions when the hands are in asymmetric positions on surface;

FIG. **20**C is a diagram of the segmentation strictness regions when the right hand crosses to the left half of the surface and the left hand is off the surface;

FIG. **21** is a flow chart of segmentation edge testing;

FIG. **22** is a flow chart of persistent path tracking;

FIG. **23** is a flow chart of the hand part identification algorithm;

FIG. **24** is a Voronoi cell diagram constructed around hand part attractor points;

FIG. **25**A is a plot of orientation weighting factor for right thumb, right inner palm, and left outer palm versus contact orientation;

FIG. **25**B is a plot of thumb size factor versus contact size;

FIG. **25**C is a plot of palm size factor versus ratio of total contact proximity to contact eccentricity;

FIG. **25**D is a plot of palm separation factor versus distance between a contact and it nearest neighbor contact;

FIG. **26** is a flow chart of the thumb presence verification algorithm;

FIG. **27** is a flow chart of an alternative hand part identification algorithm;

FIG. **28** is a flow chart of the pen grip detection process:

FIG. **29** is a flow chart of the hand identification algorithm:

FIGS. **30**A-C show three different hand partition hypotheses for a fixed arrangement of surface contacts;

FIG. **31**A is a plot of the hand clutching direction factor versus horizontal hand velocity;

FIG. **31**B is a plot of the handedness factor versus vertical position of outermost finger relative to next outermost;

FIG. **31**C is a plot of the palm cohesion factor versus maximum horizontal separation between palm contacts within a hand;

FIG. **32** is a plot of the inner finger angle factor versus the angle between the innermost and next innermost finger contacts;

FIG. **33** is a plot of the inter-hand separation factor versus the estimated distance between the right thumb and left thumb;

FIG. **34** is a flow chart of hand motion component extraction;

FIG. **35** is a diagram of typical finger trajectories when hand is contracting;

FIG. **36** is a flow chart of radial and angular hand velocity extraction;

FIG. **37** is a flow chart showing extraction of translational hand velocity components;

FIG. **38** is a flow chart of differential hand pressure extraction;

FIG. **39**A is a flow chart of the finger synchronization detection loop;

FIG. **39**B is a flow chart of chord tap detection;

FIG. **40**A is a flow chart of the chord motion recognition loop;

FIG. **40**B is a flow chart of chord motion event generation;

FIG. **41** is a flow chart of key layout morphing;

FIG. **42** is a flow chart of the keypress detection loop;

FIG. **43**A is a flow chart of the keypress acceptance and transmission loop; and

FIG. **43**B is a flow chart of typematic emulation.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Reference will now be made in detail to the present preferred embodiments of the invention, examples of which are illustrated in the accompanying drawings. Wherever possible the same reference numbers will be used throughout the drawings to refer to the same or like parts.

FIG. **1** is a system block diagram of the entire, integrated manual input apparatus. Sensor embedded in the multi-touch

surface **2** detect proximity of entire flattened hands **4**, fingertips thumbs, palms, and other conductive touch devices to the surface **2**. In a preferred embodiment, the surface is large enough to comfortably accommodate both hands **4** and is arched to reduce forearm pronation.

In alternative embodiments the multi-touch surface **2** may be large enough to accommodate motion of one hand, but may be flexible so it can be fitted to an armrest or clothing.

Electronic scanning hardware **6** controls and reads from each proximity sensor of a sensor array. A calibration module **8** constructs a raw proximity image from a complete scan of the sensor array and subtracts off any background sensor offsets. The background sensor offsets can simply be a proximity image taken when nothing is touching the surface.

The offset-corrected proximity image is then passed on to the contact tracking and identification module **10**, which segments the image into distinguishable hand-surface contacts, tracks and identifies them as they move through successive images.

The paths of identified contacts are passed on to a typing recognizer module **12**, finger synchronization detection module **14**, motion component extraction module **16**, and pen grip detection module **17**, which contain software algorithms to distinguish hand configurations and respond to detected hand motions.

The typing recognizer module **12** responds to quick presses and releases of fingers which are largely asynchronous with respect to the activity of other fingers on the same hand. It attempts to find the key region nearest to the location of each finger tap and forwards the key symbols or commands associated with the nearest key region to the communication interface module **20**.

The finger synchronization detector **14** checks the finger activity within a hand for simultaneous presses or releases of a subset of fingers. When such simultaneous activity is detected it signals the typing recognizer to ignore or cancel keystroke processing for fingers contained in the synchronous subset. It also passes on the combination of finger identities in the synchronous subset to the chord motion recognizer **18**.

The motion component extraction module **16** computes multiple degrees of freedom of control from individual finger motions during easily performable hand manipulations on the surface **2**, such as hand translations, hand rotation about the wrist, hand scaling by grasping with the fingers, and differential hand tilting.

The chord motion recognizer produces chord tap or motion events dependent upon both the synchronized finger subset identified by the synchronization detector **14** and on the direction and speed of motion extracted in **16**. These events are then posted to the host communication interface **20**.

The pen grip detection module **17** checks for specific arrangements of identified hand contacts which indicate the hand is configured as if gripping a pen. If such an arrangement is, detected, it forwards the movements of the gripping fingers as inking events to the host communication interface **20**. These inking events can either lay digital ink on the host computer display for drawing or signature capture purposes, or they can be further interpreted by handwriting recognition software which is well known in the art. The detailed steps within each of the above modules will be further described later.

The host communication interface keeps events from both the typing recognizer **12** and chord motion recognizer **18** in a single temporally ordered queue and dispatches them to the host computer system **22**. The method of communication between the interface **20** and host computer system **22** can vary widely depending on the function and processing power of the host computer. In a preferred embodiment, the communication would take place over computer cables via industry standard protocols such as Apple Desktop Bus, PS/2 keyboard and mouse protocol for PCs, or Universal Serial Bus (USB). In alternative embodiments the software processing of modules **10-18** would be performed within the host computer **22**. The multi-touch surface apparatus would only contain enough hardware to scan the proximity sensor array **6**, form proximity images **8**, and compress and send them to the host computer over a wireless network. The host communication interface **20** would then play the role of device driver on the host computer, conveying results of the proximity image recognition process as input to other applications residing on the host computer system **22**.

In a preferred embodiment the host computer system outputs to a visual display device **24** so that the hands and fingers **4** can manipulate graphical objects on the display screen. However, in alternative embodiments the host computer might output to an audio display or control a machine such as a robot.

The term "proximity" will only be used in reference to the distance or pressure between a touch device such as a finger and the surface **2**, not in reference to the distance between adjacent fingers. "Horizontal" and "vertical" refer to x and y directional axes within the surface plane. Proximity measurements are then interpreted as pressure in a z axis normal to the surface. The direction "inner" means toward the thumb of a given hand, and the direction "outer" means towards the pinky finger of a given hand. For the purposes of this description, the thumb is considered a finger unless otherwise noted, but it does not count as a fingertip. "Contact" is used as a general term for a hand part when it touches the surface and appears in the current proximity image, and for the group and path data structures which represent it.

FIG. **2** is a schematic diagram of a device that outputs a voltage **58** dependent on the proximity of a touch device **38** to a conductive sense electrode **33**. The proximity sensing device includes two electrical switching means **30** and **31** connected together in series having a common node **48**, an input node **46**, and an output node **45**. A thin dielectric material **32** covers the sensing electrode **33** that is electrically connected to the common node **48**. A power supply **34** providing an approximately constant voltage is connected between reference ground and the input node **46**. The two electrical switches **30** and **31** gate the flow of charge from the power supply **34** to an integrating capacitor **37**. The voltage across the integrating capacitor **37** is translated to another voltage **58** by a high-impedance voltage amplifier **35**. The plates of the integrating capacitor **37** can be discharged by closing electrical switch **36** until the voltage across the integrating capacitor **37** is near zero. The electrical switches **30** and **31** are opened and closed in sequence but are never closed at the same time, although they may be opened at the same time as shown in FIG. **2**. Electrical switch **30** is referred to as the input switch; electrical switch **31** is referred to as the output switch; and, electrical switch **36** is referred to as the shorting switch.

The proximity sensing device shown in FIG. **2** is operated by closing and opening the electrical switches **30**, **31**, and **36** in a particular sequence after which the voltage output from the amplifier **58**, which is dependent on the proximity of a touch device **38**, is recorded. Sensor operation begins with all switches in the open state as shown in FIG. **2**. The shorting switch **36** is then closed for a sufficiently long time to reduce the charge residing on the integrating capacitor **37** to a low level. The shorting switch **37** is then opened. The input switch

30 is then closed thus allowing charge to flow between the power supply and the common node 48 until the voltage across the input switch 30 becomes zero. Charge Q will accumulate on the sensing electrode 33 according to

$$Q=V(e*A)/D \quad (1)$$

where V is the voltage of the power supply 34, e is the permittivity of the dielectric sensing electrode cover 32 and the air gap between the cover and the touch device 38, D is the thickness of this dielectric region, and A is the overlap area of the touch device 38 and the sensing electrode 33. Therefore the amount of charge accumulating on the sensing electrode 33 will depend, among other things, on the area of overlap of the touch device 38 and the sensing electrode 33 and the distance between the touch device 38 and the sensing electrode 33. The input switch 30 is opened after the voltage across it has become zero, or nearly so. Soon after input switch 30 is opened the output switch 31 is closed until the voltage across it is nearly zero. Closing the output switch 31 allows charge to flow between the sensing electrode 33 and the integrating capacitor 37 resulting in a voltage change across the integrating capacitor 37 according to:

$$\text{delta}V=(V-Vc)/(1+C*D/e*A) \quad (2)$$

where Vc is the voltage across the integrating capacitor 37 before the output switch 31 was closed, C is the capacitance of the integrating capacitor 37, and A and D are equal to their values when input switch 30 was closed as shown in Equation 1. Multiple switchings of the input 30 and output 31 switches as described above produce a voltage on the integrating capacitor 37 that reflects the proximity of a touch device 38 to the sensing electrode 33.

FIG. 3A is a schematic diagram of the proximity sensor in which the shorting transistor 36 and the voltage-to-voltage translation device 35 are replaced by a resistor 40 and a current-to-voltage translation device 41, respectively. The integrating function of capacitor 37 shown in FIG. 2 is, in this variation of the proximity sensor, carried out by the capacitor 39 shown in FIG. 3A. Those skilled in the art will see that this variation of the proximity sensor produces a more linear output 58 from multiple switchings of the input and output switches, depending on the relative value of the resistor 40. Alternatively, the resistor 40 can be replaced by a shorting switch 69 (cf. FIG. 3B) to improve linearity. Although, the circuits shown in FIG. 3 provide a more linear output than the circuit shown in FIG. 2 the circuits of FIG. 3 generally require dual power supplies while the circuit of FIG. 2 requires only one.

The electrical switches shown in, FIG. 2 can be implemented with various transistor technologies: discrete, integrated, thin film, thick film, polymer, optical, etc. One such implementation is shown in FIG. 4A where field effect transistors (FETs) are used as the input 30, output 31, and shorting 36 switches. The FETs are switched on and off by voltages applied to their gate terminals (43, 44, and 55). For the purpose of this description we will assume the FET is switched on when its gate voltage is logic 1 and switched off when its gate voltage is logic 0. A controller 42 is used to apply gate voltages as a function of time as shown in FIG. 4B. In this example, a sequence of three pairs of pulses (43 and 44) are applied to the input and output transistor gates. Each pair of pulses 43 and 44 produces a voltage change across the integrating capacitor 37 as shown in Equation 2. The number of pulse pairs applied to input 43 and output 44 gates depends on the desired voltage across integrating capacitor 37. In typical applications the number is between one and several hundred pulse-pairs.

FIG. 5 shows the proximity sensor circuitry appropriate for use in a system comprising an array of proximity sensors 47 as in a multi-touch surface system. The proximity sensor 47 consists of the input transistor 30, the output transistor 31, the sensing electrode 33, the dielectric cover 32 for the sensing electrode 33, and conductive traces 43, 44, 45, and 46. The conductive traces are arranged so as to allow the proximity sensors 47 comprising a 2D array to be closely packed and to share the same conductive traces, thus reducing the number of wires needed in a system. FIG. 6 shows an example of such a system where the input nodes 46 of all proximity sensors are connected together and connected to a power supply 34. The output nodes 45 of all proximity sensors are connected together and connected to a single integrating capacitor 37, a single shorting transistor 36, and a single voltage-to-voltage amplifier 35. In this implementation, a single proximity sensor 47 is enabled at a time by applying a logic 1 signal first to its input gate 43 and then to its output gate 44. This gating of a single proximity sensor 47 one at a time is done by input gate controller 50 and output gate controller 51. For example, to enable the proximity sensor 47 in the lower right corner the input gate controller 50 would output a logic one pulse on conductive trace 43*a*. This is followed by a logic one pulse on conductive trace 44*h* produced by output gate controller 51. Repetition of this pulse as shown in FIG. 4B would cause charge to build up on integrating capacitor 37 and a corresponding voltage to appear at the output of the amplifier 58. The entire array of proximity sensors 47 is thus scanned by enabling a single sensor at a time and recording its output.

FIG. 7A is a schematic of typical circuitry useful for converting the proximity sensor output 58 to a digital code appropriate for processing by computer. The proximity sensor output 58 is typically non-zero even when there is no touch device (e.g., ref. no. 38 in FIG. 2) nearby. This non-zero signal is due to parasitic or stray capacitance present at the common node 48 of the proximity sensor and is of relatively constant value. It is desirable to remove this non-zero background signal before converting the sensor output 58 to a digital code. This is done by using a differential amplifier 64 to subtract a stored record of the background signal 68 from the sensor output 58. The resulting difference signal 65 is then converted to a digital code by an ADC (analog to digital converter) 60 producing a K-bit code 66. The stored background signal is first recorded by sampling the array of proximity sensors 47 (FIG. 6) with no touch devices nearby and storing a digital code specific for each proximity sensor 47 in a memory device 63. The particular code corresponding to the background signal of each proximity sensor is selected by an M-bit address input 70 to the memory device 63 and applied 69 to a DAC (digital to analog converter) 61.

The 2D array of proximity sensors 47 shown in FIG. 6 can be connected in groups so as to improve the rate at which the entire array is scanned. This is illustrated in FIG. 8 where the groups are arranged as columns of proximity sensors. In this approach, the input nodes of the proximity sensors are connected together and connected to a power supply 34, as in FIG. 6. The output gates 44 are also connected in the same way. However, the input gates 43 are now all connected together and the output nodes 45 are connected to only those proximity sensors 47 within a row and to a dedicated voltage amplifier 35. With this connection method, all of the proximity sensors in a column are enabled at a time, thus reducing the time to scan the array by a factor N, where N is the number of proximity sensors in a group. The outputs 58*a-h* could con-

nect to dedicated converter circuitry as shown in FIG. **7**A or alternatively each output **58***a-h* could be converted one at a time using the circuitry shown in FIG. **7**B. In this figure, the output signals from each group **58***a-h* are selected one at a time by multiplexer **62** and applied to the positive input of the differential amplifier **64**. With this later approach, it is assumed that the ADC **60** conversion time is much faster than the sensor enable time, thus providing the suggested speed up in sensor array scanning.

FIG. **9** shows a typical circuit useful for the control of the proximity sensor's output gate **44**. It consists of three input signals **75**, **76**, **78** and two output signals **44**, **77**. The output gate signal **44** is logic 1 when both inputs to AND gate **79** are logic 1. The AND input signal **77** becomes logic 1 if input signal **76** is logic 1 when input signal **78** transitions from logic 0 to logic 1, otherwise it remains logic 0. A linear array of these circuits **81** can be connected end-to-end to enable the output gates of a single group of proximity sensors at a time as shown in FIG. **8**.

FIG. **10** shows a cover for the multi-touch surface **89** that permits the system to be sensitive to pressure exerted by non-conducting touch objects (e.g., gloved fingers) contacting the multi-touch surface. This cover comprises a deformable dielectric touch layer **85**, a deformable conducting layer **86**, and a compliant dielectric layer **87**. The touch surface **85** would have a symbol set printed on it appropriate for a specific application, and this surface could be removed and replaced with another one having a different symbol set. The conducting layer **86** is electrically connected **88** to the reference ground of the proximity sensor's power supply **34**. When a touch object presses on the top surface **85** it causes the conducting surface **86** under the touch device to move closer to the sensing electrode **33** of the proximity sensor. This results in a change in the amount of charge stored on the sensing electrode **33** and thus the presence of the touch object can be detected. The amount of charge stored will depend on the pressure exerted by the touch object. More pressure results in more charge stored as indicated in Equation 1.

To obtain a softer touch surface on the multi-touch device a thicker and more, compliant dielectric cover could be used. However, as the dielectric thickness increases the effect of the touch device on the sensing electrodes **33** spreads out thus lowering spatial resolution. A compliant anisotropically-conducting material can be used to counter this negative effect while also providing a soft touch surface. FIG. **11** shows a cover in which a compliant anisotropically-conducting material **90** is set between a thin dielectric cover **85** and the sensing electrodes **33**. If the conductivity of the compliant material **90** is oriented mostly in the vertical direction, the image formed by a touch device on the surface **85** will be translated without significant spreading to the sensing electrodes **33**, thus preserving spatial resolution while providing a compliant touch surface.

FIG. **12** shows a cross section of a multi-touch surface that senses both the proximity and pressure of a touch device. The touch layer **85** is a thin dielectric that separates touch devices from the sensing electrodes **33**. Proximity sensing is relative to this surface. The electrodes **33** and associated switches and conductors are fabricated on a compliant material **89** which is attached to a rigid metal base **92**. The metal base **92** is electrically connected **88** to the reference ground of the proximity sensor's power supply **34**. When a touch device presses on the touch surface **85** it causes the sensing electrodes **33** directly below to move closer to the rigid metal base **92**. The distance moved depends on the pressure applied and thus the pressure exerted by a touch device can be detected as described before.

To illustrate typical properties of hand contacts as they appear in proximity images, FIGS. **13**-**15** contain sample images captured by a prototype array of parallelogram-shaped electrodes. Shading of each electrode darkens to indicate heightened proximity signals as flesh gets closer to the surface, compresses against the surface due to hand pressure, and overlaps the parallelogram more completely. Note that the resolution of these images is in no way intended to limit the scope of the invention, since certain applications such as handwriting recognition will clearly require finer electrode arrays than indicated by the electrode size in these sample images. In the discussion that follows, the proximity data measured at one electrode during a particular scan cycle constitutes one "pixel" of the proximity image captured in that scan cycle.

FIG. **13** shows a right hand flattened against the surface with fingers outstretched. At the far left is the oblong thumb **201** which tends to point off at about 120-degrees. The columnar blobs arranged in an arc across the top of the image are the index finger **202**, middle finger **203**, ring finger **204** and pinky finger **205**. Flesh from the proximal finger joint, or proximal phalanges **209**, will appear below each fingertip if the fingers are fully extended. The inner **207** and outer **206** palm heels cause the pair of very large contacts across the bottom of the image. Forepalm calluses **213** are visible at the center of the hand if the palm is fully flattened. This image shows that all the hand contacts are roughly oval-shaped, but they differ in pressure, size, orientation, eccentricity and spacing relative to one another. This image includes all of the hand parts which can touch the surface from the bottom of one hand but in many instances only a few of these parts will be touching the surface, and the fingertips may roam widely in relation to the palms as fingers are flexed and extended.

FIG. **14** shows another extreme in which the hand is partially closed. The thumb **201** is adducted toward the fingertips **202**-**208** and the fingers are flexed so the fingertips come down normal instead of tangential to the surface. The height and intensity of fingertip contacts is lessened somewhat because the boney tip rather than fleshy pulp pad is actually touching the surface, but fingertip width remains the same. Adjacent fingertips **202**-**205** and thumb **201** are so close together as to be distinguishable only by slight proximity valleys **210** between them. The proximal phalange finger joints are suspended well above the surface and do not appear in the image, nor do the forepalm calluses. The palm heels **206**, **207** are somewhat shorter since only the rear of the palm can touch the surface when fingers are flexed, but the separation between them is unchanged. Notice that the proximity images are uncluttered by background objects. Unlike optical images, only conductive objects within a few millimeters of the surface show up at all.

FIG. **15** is a proximity image of a right hand in a pen grip configuration. The thumb **201** and index fingertip **202** are pinched together as if they were holding a pen but in this case they are touching the surface instead. Actually the thumb and index finger appear the same here as in FIG. **14**. However, the middle **203**, ring **204**, and pinky **205** fingers are curled under as if making a fist, so the knuckles from the top of the fingers actually touch the surface instead of the finger tips. The curling under of the knuckles actually places them behind the pinched thumb **201** and index fingertip **202** very close to the palm heels **206**, **207**. The knuckles also appear larger than the curled fingertips of FIG. **14** but the same size as the flattened fingertips in FIG. **13**. These differences in size and arrangement will be measured by the pen grip detector **17** to distinguish this pen grip configuration from the closed and flattened hand configurations.

FIG. **16** represents the data flow within the contact tracking and identification module **10**. The image segmentation process **241** takes the most recently scanned proximity image data **240** and segments it into groups of electrodes **242** corresponding to the distinguishable hand parts of FIG. **13**. The filtering and segmentation rules applied in particular regions of the image are partially determined by feedback of the estimated hand offset data **252**. The image segmentation process **241** outputs a set of electrode group data structures **242** which are parameterized by fitting an ellipse to the positions and proximity measurements of the electrodes within each group.

The path tracking process **245** matches up the parameterized electrode groups **242** with the predicted continuations of contact path data structures **243** extracted from previous images. Such path tracking ensures continuity of contact representation across proximity images. This makes it possible to measure the velocity of individual hand contacts and determine when a hand part lifts off the surface, disappearing from future images. The path tracking process **245** updates the path positions, velocities, and contact geometry features from the parameters of the current groups **242** and passes them on to the contact identification processes **247** and **248**. For notational purposes, groups and unidentified paths will be referred to by data structure names of the form Gi and Pi respectively, where the indices i are arbitrary except for the null group G**0** and null path P**0**. Particular group and path parameters will be denoted by subscripts to these structure names and image scan cycles will be denoted by bracketed indices, so that, for example, $P2_x$, [n] represents the horizontal position of path **2** in the current proximity image, and $P2_x$, [n−1] represents the position in the previous proximity image. The contact identification system is hierarchically split into a hand identification process **247** and within-hand finger and palm identification process **248**. Given a hand identification for each contact, the finger and palm identification process **248** utilizes combinatorial optimization and fuzzy pattern recognition techniques to identify the part of the hand causing each surface contact. Feedback of the estimated hand offset helps identify hand contacts when so few contacts appear in the image that the overall hand structure is not apparent.

The hand identification process **247** utilizes a separate combinatorial optimization algorithm to find the assignment of left or right hand identity to surface contacts which results in the most biomechanically consistent within-hand identifications. It also receives feedback of the estimated hand and finger offsets **252**, primarily for the purpose of temporarily storing the last measured hand position after fingers in a hand lift off the surface. Then if the fingers soon touch back down in the same region they will more likely receive their previous hand identifications.

The output of the identification processes **247** and **248** is the set of contact paths with non-zero hand and finger indices attached. For notational purposes identified paths will be referred to as F**0** for the unidentified or null finger, F**1** for the thumb **201**, F**2** for the index finger **202**, F**3** for the middle finger **203**, F**4** for the ring finger **204**, F**5** for the pinky finger **205**, F**6** the outer palm heel **206**. F**7** for the inner palm heel **207**, and F**8** for the forepalm calluses **208**. To denote a particular hand identity this notation can be prefixed with an L for left hand or R for right hand, so that, for example, RF**2** denotes the right index finger path. When referring to a particular hand as a whole. LH denotes the left hand and RH denotes the right hand. In the actual algorithms left hand identity is represented by a −1 and right hand by +1, so it is easy to reverse the handedness of measurements taken across the vertical axis of symmetry.

It is also convenient to maintain for each hand a set of bitfield data registers for which each bit represents touchdown, continued contact or liftoff of a particular finger. Bit positions within each bit field correspond to the hand part indices above. Such registers can quickly be tested with a bit mask to determine whether a particular subset of fingers has touched down. Alternatively, they can be fed into a lookup table to find the input events associated with a particular finger chord (combination of fingers). Such finger identity bitfields are needed primarily by the synchronization detector **14** and chord motion recognizer **18**.

The last process within the tracking and identification subsystem is the hand position estimator **251**, which as described above provides biasing feedback to the identification and segmentation processes. The hand position estimator is intended to provide a conservative guess **252** of lateral hand position under all conditions including when the hand is floating above the surface without touching. In this case the estimate represents a best guess of where the hand will touch down again. When parts of a hand are touching the surface, the estimate combines the current position measurements of currently identified hand parts with past estimates which may have been made from more or less reliable identifications.

The simplest but inferior method of obtaining a hand position measurement would be to average the positions of all the hand's contacts regardless of identity. If hand parts **201**-**207** were all touching the surface as in FIG. **13** the resulting centroid would be a decent estimate, lying somewhere under the center of the palm since the fingers and palm heels typically form a ring around the center of the palm. However, consider when only one hand contact is available for the average. The estimate would assume the hand center is at the position of this lone contact, but if the contact is from the right thumb the hand center would actually be 4-8 cm to the right, or if the contact is from a palm heel the hand center is actually 4-6 cm higher, or if the lone contact is from the middle finger the hand center should actually be actually 4-6 cm lower.

FIG. **17** shows the detailed steps within the hand position estimator **251**. The steps must be repeated for each hand separately. In a preferred embodiment, the process utilizes the within-hand contact identifications (**250**) to compute (step **254**) for each contact an offset between the measured contact position ($Fi_x[n]$,$Fi_y[n]$) and the default position of the particular finger or palm heel (Fi_{defx},Fi_{defy}) with hand part identity i. The default positions preferably correspond to finger and palm positions when the hand is in a neutral posture with fingers partially closed, as when resting on home row of a keyboard. Step **255** averages the individual contact offsets to obtain a measured hand offset, ($H_{mox}[n]$,$H_{moy}[n]$):

$$H_{mox}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n](Fi_x[n] - Fi_{defx})}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \tag{3}$$

$$H_{moy}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n](Fi_y[n] - Fi_{defy})}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \tag{4}$$

Preferably the weighting $Fi_{mow}[n]$ of each finger and palm heel is approximately its measured total proximity, i.e., $Fi_{mow}[n]=Fi_z[n]$. This ensures that lifted fingers, whose proximity is zero, have no influence on the average, and that contacts with

lower than normal proximity, whose measured positions and identities are less accurate, have low influence. Furthermore, if palm heels are touching, their large total proximities will dominate the average. This is beneficial because the palm heels, being immobile relative to the hand center compared to the highly flexible fingers, supply a more reliable indication of overall hand position. When a hand is not touching the surface, i.e., when all proximities are zero, the measured offsets are set to zero. This will cause the filtered hand position estimate below to decay toward the default hand position.

As long as the contact identifications are correct, this hand position measurement method eliminates the large errors caused by assuming lone contacts originate from the center of the hand. Flexing of fingers from their default positions will not perturb the measured centroid more than a couple centimeters. However, this scheme is susceptible to contact misidentification, which can cause centroid measurement errors of up to 8 cm if only one hand part is touching. Therefore, the current measured offsets are not used directly, but are averaged with previous offset estimates ($H_{eox}[n-1],H_{eoy}[n-1]$) using a simple first-order autoregressive filter, forming current offset estimates ($H_{eox}[n],H_{eoy}[n]$).

Step **256** adjusts the filter pole $H_{oa}[n]$ according to confidence in the current contact identifications. Since finger identifications accumulate reliability as more parts of the hand contact the surface one simple measure of identification confidence: is the number of fingers which have touched down from the hand since the hand last left the surface. Contacts with large total proximities also improve identification reliability because they have strong disambiguating features such as size and orientation. Therefore $H_{oa}[n]$ is set roughly proportional to the maximum finger count plus the sum of contact proximities for the hand. $H_{oa}[n]$ must of course be normalized to be between zero and one or the filter will be unstable. Thus when confidence in contact identifications is high, i.e., when many parts of the hand firmly touch the surface, the autoregressive filter favors the current offset measurements. However, when only one or two contacts have reappeared since hand liftoff, the filter emphasizes previous offset estimates in the hope that they were based upon more reliable identifications.

The filtered offsets must also maintain a conservative estimate of hand position while the hand is floating above the surface for optimal segmentation and identification as the hand touches back down. If a hand lifts off the surface in the middle of a complex sequence of operations and must, quickly touch down again, it will probably touch down close to where it lifted off. However, if the operation sequence has ended, the hand is likely to eventually return to the neutral posture, or default position, to rest. Therefore, while a hand is not touching the surface, $H_{oa}[n]$ is made small enough that the estimated offsets gradually decay to zero at about the same rate as a hand lazily returns to default position.

When $H_{oa}[n]$ is made small due to low identification confidence, the filter tracking delay becomes large enough to lag behind a pair of quickly moving fingers by several centimeters. The purpose of the filter is to react slowly to questionable changes in contact identity, not to smooth contact motion. This motion tracking delay can be safely eliminated by adding the contact motion measured between images to the old offset estimate. Step **257** obtains motion from the average, ($H_{mvx}[n],H_{mvy}[n]$) of the current contact velocities:

$$H_{mvx}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n]Fi_{vx}[n]}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \tag{5}$$

$$H_{mvy}[n] = \frac{\sum_{i=1}^{i=7} FI_{mow}[n]Fi_{vy}[n]}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \tag{6}$$

The current contact velocities. ($Fi_{vx}[n],F_{vy}[n]$), are retrieved from the path tracking process **245**, which measures them independent of finger identity. Step **258** updates the estimated hand offsets ($H_{eox}[n],H_{eoy}[n]$) using the complete filter equations:

$$H_{eox}[n]=H_{oa}[n]H_{mox}[n]+(1-H_{oa}[n])(H_{eox}[n-1]+H_{mox}[n]\Delta t) \tag{7}$$

$$H_{eoy}[n]=H_{oa}[n]H_{moy}[n]+(1-H_{oa}[n])(H_{eoy}[n-1]+H_{moy}[n]\Delta t) \tag{8}$$

Finally, to provide a similarly conservative estimate of the positions of particular fingers step **259** computes individual finger offsets ($Fi_{eox}[n],Fi_{eoy}[n]$) from the distance between identified contacts and their corresponding default finger positions less the estimated hand offsets. For each identifiable contact i, the offsets are computed as:

$$Fi_{eox}[n]=H_{oa}[n](H_{mox}[n]+Fi_x[n]-Fi_{defx})+(1-H_{oa}[n])(Fi_{eox}[n-1]+Fi_{vx}[n]\Delta t) \tag{9}$$

$$FI_{eoy}[n]=H_{oa}[n](H_{moy}[n]+Fi_y[n]-Fi_{defy})+(1-H_{oa}[n])(Fi_{eoy}[n-1]+Fi_{vy}[n]\Delta t) \tag{10}$$

These finger offsets reflect deviations of finger flexion and extension from the neutral posture. If the user places the fingers in an extreme configuration such as the flattened hand configuration, the collective magnitudes of these finger offsets can be used as an indication of user hand size and finger length compared to the average adult.

The parameters ($H_{eox}[n],H_{eoy}[n]$) and ($Fi_{eox}[n],Fi_{eoy}[n]$) for each hand and finger constitute the estimated hand and finger offset data **252**, which is fed back to the segmentation and identification processes during analysis of the next proximity image. If the other processes need the estimate in absolute coordinates, they can simply add (step **260**) the supplied offsets to the default finger positions, but in many cases the relative offset representation is actually more convenient.

It should be clear to those skilled in the art that many improvements can be made to the above hand position estimation procedure which remain well within the scope of this invention, especially in the manner of guessing the position of lifted hands. One improvement is to make the estimated hand offsets decay toward zero at a constant speed when a hand is lifted rather than decay exponentially. Also, the offset computations for each hand have been independent as described so far. It is actually advantageous to impose a minimum horizontal separation between the estimated left hand position and estimated right hand position such that when a hand such as the right hand slides to the opposite side of the board while the other hand is lifted, the estimated position of the other hand is displaced. In this case the estimated position of the lifted left hand would be forced from default to the far left of the surface, possibly off the surface completely. If the right hand is lifted and the left is not, an equation like the following can be applied to force the estimated right hand position out of the way:

$$Rh_{eox}[n]:=\min(RH_{eox}[n],(LF1_{defx}-RF1_{defx})+Lh_{eox}[n]+\text{min\_hard\_sep}) \quad (11)$$

where $(LF1_{defx}-RF1_{defx})$ is the default separation between left and right thumbs, is the minimum horizontal separation to be imposed, and $LH_{eox}[n]$ is the current estimated offset of the left hand.

FIG. **18** represents the data flow within the proximity image segmentation process **241**. Step **262** makes a spatially smoothed copy **263** of the current proximity image **240** by passing a two-dimensional diffusion operator or Gaussian kernel over it. Step **264** searches the smoothed image **263** for local maximum pixels **265** whose filtered proximity exceeds a significance threshold and exceeds the filtered proximities of nearest neighbor pixels. The smoothing reduces the chance that an isolated noise spike on a single electrode will result in a local maximum-which exceeds the significance threshold, and consolidates local maxima to about one per distinguishable fleshy contact.

Process **268** then constructs a group of electrodes or pixels which register significant proximity around each local maximum pixel by searching outward from each local maximum for contact edges. Each electrode encountered before reaching a contact boundary is added to the local maximum's group. FIG. **19** shows the basic boundary electrode search pattern for an example contact boundary **274**. In this diagram, an electrode or image pixel lies at the tip of each arrow. The search starts at the local maximum pixel **276**, proceeds to the left pixels **277** until the boundary **274** is detected. The last pixel before the boundary **278** is marked as an edge pixel, and the search resumes to the right **279** of the local maximum pixel **276**. Once the left and right edges of the local maximum's row have been found, the search recurses to the rows above and below, always starting **281** in the column of the pixel in the previous row which had the greatest proximity. As the example illustrates, the resulting set of pixels or electrodes is connected in the mathematical sense but need not be rectangular. This allows groups to closely fit the typical oval-shape of flesh contacts without leaving electrodes out or including those from adjacent contacts.

If contacts were small and always well separated, edges could simply be established wherever proximity readings fell to the background level. But sometimes fingertips are only separated by a slight valley or shallow saddle point **210**. To segment adjacent fingertips the partial minima of these valleys must be detected and used as group boundaries. Large palm heel contacts, on the other hand, may exhibit partial minima due to minor nonuniformities in flesh proximity across the contact. If all electrodes under the contact are to be collected in a single group, such partial minima must be ignored. Given a hand position estimate the segmentation system can apply strict edge detection rules in regions of the image where fingertips and thumb are expected to appear but apply sloppy edge detection rules in regions of the image where palms are expected to, appear. This ensures that adjacent fingertips are not joined into a single group and that each palm heel is not broken into multiple groups.

Step **266** of FIG. **18** defines the positions of these segmentation regions using the hand position estimates **252** derived from analyses of previous images. FIG. **20**A shows the extent of the strict and sloppy segmentation regions while the hands are in their default positions, making estimated offsets for both hands zero. Plus signs in the diagram **252** indicate the estimated position of each finger and palm heel in each hand. Rectangular outlines in the lower corners represent the left **284** and right **286** sloppy segmentation regions where partial minima are largely ignored. The T-shaped region remaining is the strict segmentation region **282**, where proximity saddle points must serve as contact boundaries. As a preferred embodiment the sloppy regions are rectangular, their inner boundaries **285** are placed just inside of the columns where the index fingers **202** are expected to lie, and the upper boundaries **287** are placed at the estimated vertical levels of their respective thumbs **201**. The outer and lower boundaries of the sloppy regions are determined by the outside edges of the surface. Due to the decay in estimated hand offsets after hands leave the surface, the sloppy segmentation regions return to the positions shown after the hands have stayed off the surface a few seconds, regardless of hand position at liftoff. FIG. **20**B shows how the sloppy regions follow the estimated hand positions **252** as the right hand moves toward the upper left and the left hand moves toward the lower left. This ensures that the palms and only the palms fall in the sloppy regions as long as the hand position estimates are correct.

FIG. **20**C shows that the left sloppy region **284** is moved left off the surface entirely when the left hand is lifted off the surface and the right hand slides to the left side of the surface. This prevents the fingers of one hand from entering the sloppy segmentation region of the opposite hand. This effect is implemented by imposing a minimum horizontal separation between the sloppy regions and, should the regions get too close to one another, letting the hand with the most surface contacts override the estimated position of the hand with fewer contacts. FIG. **21** is a detailed flow chart of the edge tests which are applied at each searched electrode depending on whether the electrode is in a strict or sloppy segmentation region. Decision diamond **290** checks whether the unsmoothed proximity of the electrode is greater than the background proximity levels. If not, the electrode is labeled an edge electrode in step **304** regardless of the segmentation region or search direction, and in step **305** the search returns to the row maximum to recurse in another direction. If the unsmoothed proximity is significant farther tests are applied to the smoothed proximity of neighboring electrodes depending on whether decision diamond **292** decides the search electrode is in a sloppy or strict region.

If a strict region search is advancing horizontally within a row, decision diamond **306** passes to decision diamond **308** which tests whether the electrode lies in a horizontal or diagonal partial minimum with respect to its nearest neighbor electrodes. If so, a proximity valley between adjacent fingers has probably been detected, the electrode is labeled as an edge **314** and search resumes in other directions **305**. If not, the search continues on the next electrode in the row **302**. If a strict region search is advancing vertically to the next row, decision diamond **306** passes to decision diamond **310** which tests whether the electrode lies in a vertical partial minimum with respect to the smoothed proximity of its nearest neighbor electrodes. If so, a proximity valley between a finger and the thumb has probably been detected, the electrode is labeled as an edge **312** and search resumes in other directions **305**. If not, the search continues into the next row **302**. If decision diamond **294** determines that a sloppy region search is advancing horizontally within a row, stringent horizontal minimum tests are performed to check for the crease or proximity valley between the inner and outer palm heels. To qualify, the electrode must be more than about 2 cm horizontal distance from the originating local maximum, as checked by decision diamond **296**. Also the electrode must be part of a tall valley or partial horizontal minimum which extends to the rows above and below and the next-nearest neighbors within the row, as checked by decision diamond **298**. If so, the electrode is labeled as an edge **300** and search recurses in other directions **305**. All other partial minima within the sloppy regions are

ignored, so the search continues **302** until a background level edge is reached on an upcoming electrode.

In sloppy segmentation regions it is possible for groups to overlap significantly because partial minima between local maxima do not act as boundaries. Typically when this happens the overlapping groups are part of a large fleshy contact such as a palm which, even after smoothing, has multiple local maxima. Two groups are defined to be overlapping if the search originating local maximum electrode of one group is also an element of the other group. In the interest of presenting only one group per distinguishable fleshy contact to the rest of the system, step **270** of FIG. **18** combines overlapping groups into single supergroups before parameter extraction. Those skilled in the art will realize that feedback from high-level analysis of previous images can be applied in various alternative ways to improve the segmentation process and still lie well within the scope of this invention. For example, additional image smoothing in sloppy segmentation regions could consolidate each palm heel contact into a single local maximum which would pass strict segmentation region boundary tests. Care must be taken with this approach however, because too much smoothing can cause finger pairs which unexpectedly enter sloppy palm regions to be joined into one group. Once a finger pair is joined the finger identification process **248** has no way to tell that the fingertips are actually not a single palm heel, so the finger identification process will be unable to correct the hand position estimate or adjust the sloppy regions for proper segmentation of future images.

More detailed forms of feedback than the hand position estimate can be utilized as well. For example, the proximal phalanges(**209** in FIG. **13**) are actually part of the finger but tend to be segmented into separate groups than the fingertips by the vertical minimum test **310**. The vertical minimum test is necessary to separate the thumb group from index fingertip group in the partially closed FIG. **14** and pen grip FIG. **15** hand configurations. However, the proximal phalanges of flattened fingers can be distinguished from a thumb behind a curled fingertip by the fact that it is very difficult to flatten one long finger without flattening the other long fingers. To take, advantage of this constraint, a flattened finger flag **267** is set whenever two or more of the contacts identified as index through pinky in previous images are larger than normal, reliably indicating that fingertips are flattening. Then decision diamond **310** is modified during processing of the current image to ignore the first vertical minimum encountered during search of rows below the originating local minimum **276**. This allows the proximal phalanges to be included in the fingertip group but prevents fingertip groups from merging with thumbs or forepalms. The last step **272** of the segmentation process is to extract shape, size, and position parameters from each electrode group. Group position reflects hand contact position and is necessary to determine finger velocity. The total group proximity, eccentricity, and orientation are used by higher level modules to help distinguish finger, palm, and thumb contacts.

Provided G_E is the set of electrodes in group G, e_z is the unsmoothed proximity of an electrode or pixel e, and e_x and e_y are the coordinates on the surface of the electrode center in centimeters, to give a basic indicator of group position, the proximity-weighted center, or centroid, is computed from positions and proximities of the group's electrodes:

$$G_z = \sum_{e \in G_E} e_z \tag{12}$$

$$G_x = \sum_{e \in G_E} \frac{e_z e_x}{G_z} \tag{13}$$

$$G_y = \sum_{e \in G_E} \frac{e_z e_y}{G_z} \tag{14}$$

Note that since the total group proximity G_z integrates proximity over each pixel in the group, it depends upon both of the size of a hand part, since large hand parts tend to cause groups with more pixels, and of the proximity to or pressure on the surface of a hand part.

Since most groups are convex, their shape is well approximated by ellipse parameters. The ellipse fitting procedure requires a unitary transformation of the group covariance matrix G_{eov} of second moments Q_{xx}, Q_{xy}, G_{yy}:

$$G_{cov} = \begin{bmatrix} G_{xx} & G_{xy} \\ G_{yx} & G_{yy} \end{bmatrix} \tag{15}$$

$$G_{xx} = \sum_{e \in G_E} e_z (G_x - e_x)^2 \tag{16}$$

$$G_{yx} = G_{xy} = \sum_{e \in G_E} e_z (G_x - e_x)(G_y - e_y) \tag{17}$$

$$G_{yy} = \sum_{e \in G_E} e_x (G_y - e_y)^2 \tag{18}$$

The eigenvalues λ_0 and λ_1 of the covariance matrix G_{eov} determine the ellipse axis lengths and orientation G_θ:

$$G_{major} = \sqrt{\lambda_0} \tag{19}$$

$$G_{minor} = \sqrt{\lambda_1} \tag{20}$$

$$G_\theta = \arctan\left(\frac{\lambda_0 - G_{xx}}{G_{xy}}\right) \tag{21}$$

where G_θ is uniquely wrapped into the range (0,180°).

For convenience while distinguishing fingertips from palms at higher system levels, the major and minor axis lengths are converted via their ratio into an eccentricity $G_\in$:

$$G_\varepsilon = \frac{G_{major}}{G_{minor}} \tag{22}$$

Note that since the major axis length is always greater than or equal to the minor axis length, the eccentricity will always be greater than or equal to one. Finally, the total group proximity is empirically renormalized so that the typical curled fingertip will have a total proximity around one:

$$G_z' = \frac{G_z}{Z_{averageFingertip}} \tag{23}$$

On low resolution electrode arrays, the total group proximity G_z is a more reliable indicator of contact size as well as finger pressure than the fitted ellipse parameters. Therefore, if proximity images have low resolution, the orientation and eccentricity of small contacts are set to default values rather than their measured values, and total group proximity G_z is used as the primary measure of contact size instead of major and minor axis lengths.

FIG. **22** shows the steps of the path tracking process, which chains together those groups from successive proximity images which correspond to the same physical hand contact. To determine where each hand part has moved since the last proximity image, the tracking process must decide which current groups should be matched with which existing contact paths. As a general rule, a group and path arising from the same contact will be closer to one another than to other groups and paths. Also, biomechanical constraints on lateral finger velocity and acceleration limit how far a finger can travel between images. Therefore a group and path should not be matched unless they are within a distance known as the tracking radius of one another. Since the typical lateral separation between fingers is greater than the tracking radius for reasonable image scan rates touchdown and liftoff are easily detected by the fact that touchdown usually causes a new group to appear outside the tracking radii of existing paths, and liftoff will leave an active path without a group within its tracking radius. To prevent improper breaking of paths at high finger speeds each path's tracking radius P_{rtruck} can be made dependent on its existing speed and proximity.

The first step **320** predicts the current locations of surface contacts along existing trajectories using path positions and velocities measured from previous images. Applying previous velocity to the location prediction improves the prediction except when a finger suddenly starts or stops or changes direction. Since such high acceleration events occur less often than zero acceleration events, the benefits of velocity-based prediction outweigh the potentially bad predictions during finger acceleration. Letting $P_x[n-1],P_y[n-1]$ be the position of path P from time step n−1 and $P_{vx}[n-1]$. $P_{vy}[n-1]$ the last known velocity, the velocity-predicted path continuation is then:

$$P_{predx}[n]=P_x[n-1]+\Delta t P_{vx}[n-1] \tag{24}$$

$$P_{predy}[n]=P_y[n-1]+\Delta t P_{vy}[n-1] \tag{25}$$

Letting the set of paths active in the previous image be PA, and let the set electrode groups constructed in the current image be G, step **322** finds for each group Gk the closest active path and records the distance to it:

$$??????? \tag{26}$$

$$\text{Gk closestPdist 2=min Pl PA d 2(Gk,Pl) Gk G} \tag{27}$$

where the squared Euclidean distance is an easily computed distance metric:

$$a^2(Gk,Pl)=(Gk_x-Pl_{predx})^2+(Gk_y-Pl_{predy})^2 \tag{28}$$

Step **324** then finds for each active path Pl, the closest active group and records the distance to it:

$$\text{9 Pl closestG=arg min Gk Gd 2(Gk,Pl) Pl PA} \tag{29}$$

$$\text{Pl closestGdist 2=min Gk Gd 2(Gk,Pl) Pl PA} \tag{30}$$

In step **326**, an active group Gk and path Pl are only paired with one another if they are closest to one another, i.e., $Gk_{closestP}$ and $Pl_{closestG}$ refer to one another, and the distance between them is less than the tracking radius. All of the following conditions must hold:

$$Gk_{closestP}\equiv Pl \tag{31}$$

$$Pl_{closestG}\equiv Gk \tag{32}$$

$$Pl_{closestGdist2}<Pl_{mack2} \tag{33}$$

To aid in detection of repetitive taps of the same finger, it may be useful to preserve continuity of path assignment between taps over the same location. This is accomplished in step Via USPTO EFS **334** by repeating steps **322-326** using only groups which were left unpaired above and paths which were deactivated within the last second or so due to finger liftoff.

In step **336**, any group which has still not be paired with an active or recently deactivated path is allocated a new path, representing touchdown of a new finger onto the surface. In step **344**, any active path which cannot be so paired with a group is deactivated, representing hand part liftoff from the surface.

Step **346** incorporates the extracted parameters of each group into its assigned path via standard filtering techniques. The equations shown below apply simple autoregressive filters to update the path position ($P_x[n],P_y[n],P_z[n]$), velocity ($P_x[n],P_y[n]$), and shape ($P_\theta[n]$, $P_\in[n]$) parameters from corresponding group parameters, but Kalman or finite impulse response filters would also be appropriate.

If a path P has just been started by group G at time step n, i.e., a hand part has just touched down, its parameters are initialized as follows:

$$P_x[n]=G_x \tag{34}$$

$$P_y[n]=G_y \tag{35}$$

$$P_z[n]=G_z \tag{36}$$

$$P_\theta[n]=G_\theta \tag{37}$$

$$P_\in[n]=G_\in \tag{38}$$

$$P_{vx}[n]=0 \tag{39}$$

$$P_{vy}[n]=0 \tag{40}$$

$$P_{vz}[n]=G_z/\Delta t \tag{41}$$

else if group G is a continuation of active path P[n−1] to time step n:

$$P_x[n]=G_\alpha G_x+(1-G_\alpha)(P_{predx}[n-1]) \tag{42}$$

$$P_y[n]=G_\alpha G_y+(1-G_\alpha)(P_{predy}[n-1]) \tag{43}$$

$$P_z[n]=G_\alpha G_z+(1-G_\alpha)(P_{predz}[n-1]) \tag{44}$$

$$P_\theta[n]=G_\alpha G_\theta+(1-G_\alpha)(P_\theta[n-1]) \tag{45}$$

$$P_\in[n]=G_\alpha G_\in+(1-G_\alpha)(P_\in[n-1]) \tag{46}$$

$$P_{vx}[n]=(P_x[n]-P_x[n-1])/\Delta t \tag{47}$$

$$P_{vy}[n]=(P_y[n]-P_y[n-1])/\Delta t \tag{48}$$

$$P_{vz}[n]=(P_z[n]-P_z[n-1])/\Delta t \tag{49}$$

It is also useful to compute the magnitude P_{speed} and angle P_{dir} from the velocity vector (P_{vx}, P_{vx}). Since the reliability of position measurements increases considerably with total proximity P_z, the low-pass filter pole $G_\alpha$is decreased for groups with total proximities lower than normal. Thus when

signals are weak, the system relies heavily on the previously established path velocity, but when the finger firmly touches the surface causing a strong, reliable signal, the system relies entirely on the current group centroid measurement.

The next process within the tracking module is contact identification. On surfaces large enough for multiple hands, the contacts of each hand tend to form a circular cluster, and the clusters tend to remain separate because users like to avoid entangling the fingers of opposite hands. Because the arrangement of fingers within a hand cluster is independent of the location of and arrangement within the other hand's cluster, the contact identification system is hierarchically split. The hand identification process **247** first decides to which cluster each contact belongs. Then a within-cluster identification process **248** analyzes for each hand the arrangement of contacts within the hand's cluster, independent of the other hand's cluster. Because within-cluster or finger identification works the same for each hand regardless of how many hands can fit on the surface, it will be described first. The description below is for identification within the right hand. Mirror symmetry must be applied to some parameters before identifying left hand contacts.

FIG. **23** shows the preferred embodiment of the finger identification process **248**. For the contacts assigned to each hand this embodiment attempts to match contacts to a template of hand part attractor points, each attractor point having an identity which corresponds to a particular finger or palm heel. This matching between contact paths and attractors should be basically one to one but in the case that some hand parts are not touching the surface, some attractors will be left unfilled, i.e., assigned to the null path or dummy paths.

Step **350** initializes the locations of the attractor points to the approximate positions of the corresponding fingers and palms when the hand is in a neutral posture with fingers partially curled. Preferably these are the same default finger locations (Fi_{defx},Fi_{defy}) employed in hand offset estimation. Setting the distances and angles between attractor points from a half-closed hand posture allows the matching algorithm to perform well for a wide variety of finger flexions and extensions.

The resulting attractor points tend to lie in a ring as displayed by the crosses in FIG. **24**. The identities of attractor points **371-377** correspond to the identities of hand parts **201-207**. If the given hand is a left hand, the attractor ring must be mirrored about the vertical axis from that shown. FIG. **24** also includes line segments **380** forming the Voronoi cell around each attractor point. Every point within an attractor's Voronoi cell is closer to that attractor than any other attractor. When there is only one contact in the cluster and its features are not distinguishing, the assignment algorithm effectively assigns the contact to the attractor point of the Voronoi cell which the contact lies within. When there are multiple surface contacts in a hand cluster, they could all lie in the same Voronoi cell, so the assignment algorithm must perform a global optimization which takes into account all of the contact positions at once.

Alternative embodiments can include additional attractors for other hand part or alternative attractor arrangements for atypical hand configurations. For example, attractors for forepalm contacts can be placed at the center of the ring, but since the forepalms typically do not touch the surface unless the rest of the hand is flattened onto the surface as well, forepalm attractors should be weighted such that contacts are assigned to them only when no regular attractors are left unassigned.

For optimal matching accuracy the ring should be kept roughly centered on the hand cluster. Therefore step **352** translates all of the attractor points for a given hand by the hand's estimated position offset. The final attractor positions ($Aj_x[n]$,$Aj_y[n]$) are therefore given by:

$$Aj_x[n]=H_{eox}[n]+Fj_{defx} \tag{50}$$

$$Aj_y[n]=H_{eoy}[n]+Fj_{defy} \tag{51}$$

In alternative embodiments the attractor ring can also be rotated or scaled by estimates of hand rotation and size such as the estimated finger offsets, but care must be taken that wrong finger offset estimates and identification errors do not reinforce one another by severely warping the attractor ring.

Once the attractor template is in place, step **354** constructs a square matrix $[d_{ij}]$ of the distances in the surface plane from each active contact path Pi to each attractor point Aj. If there are fewer surface contacts than attractors, the null path P**0**, which has zero distance to each attractor, takes place of the missing contacts. Though any distance metric can be used, the squared Euclidean distance,

$$d_{ij}=(Aj_x[n]-Pi_x[n])^2+(Aj_y[n]-Pi_y[n]) \tag{52}$$

is preferred because it specially favors assignments wherein the angle between any pair of contacts is close to the angle between the pair of attractors assigned to those contacts. This corresponds to the biomechanical constraint that fingertips avoid crossing over one another, especially while touching a surface.

In step **356**, the distances from each contact to selected attractors are weighted according to whether the geometrical features of the given contact match those expected from the hand part that the attractor represents. Since the thumb and palm heels exhibit the most distinguishing geometrical features, weighting functions are computed for the thumb and palm heel attractors, and distances to fingertip attractors are unchanged. In a preferred embodiment, each weighting function is composed of several factor versus feature relationships such as those plotted approximately in FIG. **25**. Each factor is designed to take on a default value of 1 when its feature measurement provides no distinguishing information, take on larger values if the measured contact feature uniquely resembles the given thumb or palm hand part, and take on smaller values if the measured feature is inconsistent with the given attractor's hand part. The factor relationships can be variously stored and computed as lookup tables, piecewise linear functions, polynomials, trigonometric functions, rational functions, or any combination of these. Since assignment between a contact and an attractor whose features match is favored as the weighted distance between becomes smaller, the distances are actually weighted (multiplied) with the reciprocals of the factor relationships shown.

FIG. **25**A shows the right thumb and right inner palm heel orientation factor versus orientation of a contact's fitted ellipse. Orientation of these hand parts tends to be about 120°, whereas fingertip and outer palm heel contacts are usually very close to vertical (90°), and orientation of the left thumb and left inner palm heel averages 60°. The right orientation factor therefore approaches a maximum at 120°. It approaches the default value of 1 at 0°, 90°, and 180° where orientation is inconclusive of identity, and reaches a minimum at 60°, the favored orientation of the opposite thumb or palm heel. The corresponding relationship for the left thumb and inner palm heel orientation factor is flipped about 90°.

FIG. **25**B approximately plots the thumb size factor. Since thumb size as indicated by total proximity tends to peak at two or three times the size of the typical curled fingertip, the thumb size factor peaks at these sizes. Unlike palm heels, thumb contacts can not be much larger than two or three times the default fingertip size, so the thumb factor drops back down

for larger sizes. Since any hand part can appear small when touching the surface very lightly or just starting to touch-down, small size is not distinguishing, so the size factor defaults to **1** for very small contacts.

FIG. **25**C approximately plots the palm heel size factor. As more pressure is applied to the palms, the palm heel contacts can grow quite large, remaining fairly round as they do so. Thus the palm heel size factor is much like the thumb size factor except the palm factor is free to increase indefinitely. However, fingertip contacts can grow by becoming taller as the fingers are flattened. But since finger width is constant, the eccentricity of an ellipse fitted to a growing fingertip contact increases in proportion to the height. To prevent flattened fingers from having a large palm factor, has little effect for palms, whose eccentricity remains near **1**, but cancels the high proximities of flattened fingertips. Though directly using fitted ellipse width would be less accurate for low resolution electrode arrays, the above ratio basically captures contact width.

Another important distinguishing feature of the palm heels is that wrist anatomy keeps the centroids of their contacts separated from one other and from the fingers by several centimeters. This is not true of the thumb and fingertips, which can be moved within a centimeter of one another via flexible joints. The inter-palm separation feature is measured by searching for the nearest neighbor contact of a given contact and measuring the distance to the neighbor. As plotted approximately in FIG. **25**D, the palm separation factor quickly decreases as the separation between the contact and its nearest neighbor falls below a few centimeters, indicating that the given contact (and its nearest neighbor) are not palm heels. Unlike the size and orientation factors which only become reliable as the weight of the hands fully compresses the palms, the palm separation factor is especially helpful in distinguishing the palm heels from pairs of adjacent fingertips because it applies equally well to light, small contacts.

Once the thumb and palm weightings have been applied to the distance matrix, step **358** finds the one-to-one assignment between attractors and contacts which minimizes the sum of weighted; distances between each attractor and it's assigned contact. For notational purposes, let a new matrix $[c_{ij}]$ hold the weighted distances:

$$c_{ij} = \begin{cases} d_{ij} / (Pi_{thumb\_size\_fact} Pi_{orient\_fact}) & \text{if } j = 1 \\ d_{ij} & \text{if } 2 \le j \le 5 \\ d_{ij} / (Pi_{palm\_size\_fact} Pi_{palm\_sep\_fact}) & \text{if } j = 6 \\ d_{ij} / (Pi_{palm\_size\_fact} Pi_{palm\_sep\_fact}) & \text{if } j = 7 \end{cases} \tag{53}$$

Mathematically the optimization can then be stated as finding the permutation $\{\pi_1, \ldots, \pi_7\}$ of integer hand part identities $\{1, \ldots, 7\}$ which minimizes:

$$\sum_{i=1}^{7} c_{i\pi_i} \tag{54}$$

where c_{ij} is the weighted distance from contact i to attractor j, and contact i and attractor j are considered assigned to one another when $\pi_i \equiv j$. This combinatorial optimization problem, known more specifically in mathematics as an assignment problem, can be efficiently solved by a variety of well-known mathematical techniques, such as branch and bound, localized combinatorial search, the Hungarian method, or network flow solvers. Those skilled in the art will recognize that this type of combinatorial optimization problem has a mathematically equivalent dual representation in which the optimization is reformulated as a maximization of a sum of dual parameters. Such reformulation of the above hand part identification method as the dual of attractor-contact distance minimization remains well within the scope of this invention.

To avoid unnecessary computation, decision diamond **360** ends the finger identification process at this stage if the hand assignment of the given contact cluster is only a tentative hypothesis being evaluated by the hand identification module **247**. However, if the given hand assignments are the final preferred hypothesis, further processes verify finger identities and compile identity statistics such as finger counts.

The identifications produced by this attractor assignment method are highly reliable when all five fingers are touching the surface or when thumb and palm features are unambiguous. Checking that the horizontal coordinates for identified fingertip contacts are in increasing order easily verifies that fingertip identities are not erroneously swapped. However, when-only two to four fingers are touching, yet no finger strongly exhibits thumb size or orientation features, the assignment of the innermost finger contact may wrongly indicate whether the contact is the thumb. In this case, decision diamond **362** employs a thumb verification process **368** to take further measurements between the innermost finger contact and the other fingers. If these further measurements strongly suggest the innermost finger contact identity is wrong, the thumb verification process changes the assignment of the innermost finger contact. Once the finger assignments are verified, step **364** compiles statistics about the assignments within each hand such as the number of touching fingertips and bitfields of touching finger identities. These statistics provide convenient summaries of identification results for other modules.

FIG. **26** shows the steps within the thumb verification module. The first **400** is to compute several velocity, separation, and angle factors for the innermost contact identified as a finger relative to the other contacts identified as fingers. Since these inter-path measurements presuppose a contact identity ordering, they could not have easily been included as attractor distance weightings because contact identities are not known until the attractor distance minimization is complete. For the factor descriptions below, let FI be the innermost finger contact, FN be the next innermost finger contact, FO be the outermost finger contact.

The separation between thumb and index finger is often larger than the separations between fingertips, but all separations tend to grow as the fingers are outstretched. Therefore an inner separation factor inner_separation_fact is defined as the ratio of the distance between the innermost and next innermost finger contacts to the average of the distances between other adjacent fingertip contacts, avg_separation: 12 innerseparationfact min

$$innerseparationfact \approx \min\left(1, \frac{\sqrt{(FI_x - FN_x)^2 + (FI_y - FN_y)^2}}{avgseparation}\right) \tag{55}$$

The factor is clipped to be greater than one since an innermost separation less than the average can occur regardless of whether thumb or index finger is the innermost touching finger. In case there are only two finger contacts, a default average separation of 2-3 cm is used. The factor tends to

become larger than one if the innermost contact is actually the thumb but remains near one if the innermost contact is a fingertip.

Since the thumb rarely moves further forward than the fingertips except when the fingers are curled into a fist, the angle between the innermost and next innermost finger contact can help indicate whether the innermost finger contact is the thumb. For the right hand the angle of the vector from the thumb to the index finger is most often 60°, though it ranges to 0 as the thumb moves forward and to 120° as the thumb adducts under the palm. This is reflected in the approximate plot of the inner angle factor in FIG. **32**, which peaks at 60° and approaches 0 toward 0° and 120°. If the innermost finger contact is actually an index fingertip, the measured angle between innermost and next innermost contact will likely be between 30° and −60°, producing a very small angle factor.

The inner separation and angle factors are highly discriminating of neutral thumb postures, but users often exceed the above cited separation and angle ranges when performing hand scaling or rotation gestures. For instance, during an anti-pinch gesture, the thumb may start pinched against the index or middle fingertip, but then the thumb and fingertip slide away from one another. This causes the inner separation factor to be relatively small at the start of the gesture. Similarly, the thumb-index angle can also exceed the range expected by the inner angle factor at the beginning or end of hand rotation gestures, wherein the fingers rotate as if turning a screw. To compensate, the inner separation and angle factors are fuzzy OR'ed with expansion and rotation factors which are selective for symmetric finger scalings or rotations centered on a point between the thumb and fingertips.

When defined by the following approximate equation, the expansion factor peaks as the innermost and outermost finger contacts slide at approximately the same speed and in opposite directions, parallel to the vector between them:

$$\text{expansionfact} \approx -\sqrt{FI_{\text{speed}}[n] \times FO_{\text{speed}}[n]} \times \cos(FI_{dir}[n] - \angle(FI[n],(FO[n])) \times \cos(FO_{dir}[n] - \angle(FI[n],FO[n])) \quad (56)$$

$$\text{expansnsion\_fact} := \max(0, \text{expansion\_fact}) \quad (57)$$

where ∠(FI[n], FO[n]) is the angle between the fingers:

$$\angle(FI[n], FO[n]) = \arctan\left(\frac{FI_y[n] - FO_y[n]}{FI_x[n] - FO_x[n]}\right) \quad (58)$$

Translational motions of both fingers in the same direction produce negative factor values which are clipped to zero by the max operation. Computing the geometric rather than arithmetic mean of the innermost and outermost speeds aids selectivity by producing a large expansion factor only when speeds of both contacts are high.

The rotation factor must also be very selective. If the rotation factor was simply proportional to changes in the angle between innermost and outermost finger, it would erroneously grow in response to asymmetries in finger motion such as when the innermost finger starts translating downward while the outermost contact is stationary. To be more selective, the rotation factor must favor symmetric rotation about an imaginary pivot between the thumb and fingertips. The approximate rotation factor equation below peaks as the innermost and outermost finger move in opposite directions, but in this case the contacts should move perpendicular to the vector between them:

$$\text{rotationfact} = -\sqrt{FI_{\text{speed}}[n] FO_{\text{speed}}[n]} \times \sin(FI_{dir}[n] - \angle(FI[n],FO[n])) \times \sin(FO_{dir}[n] - \angle(FI[n],FO[n])) \quad (59)$$

$$\text{rotation\_fact} := \max(0, \text{rotation\_fact}) \quad (60)$$

Since motions which maximize this rotation factor are easy to perform between the opposable thumb and another finger but difficult to perform between two fingertips the rotation factor is a robust indicator of thumb presence.

Finally, a fuzzy logic expression (step **402**) combines these inter-contact factors with the thumb feature factors for the innermost and next innermost finger contacts. In a preferred embodiment, this fuzzy logic expression for the combined_thumb_fact takes the form:

$$\text{combined\_thumb\_fact} \approx (\text{inner\_separation\_fact} \times \text{angle\_fact} + \text{expansion\_fact} + \text{rotation\_fact}) \times (FI_{orient\_fact}/FN_{orient\_fact}) \times (FI_{thumb\_size\_fact}/FN_{thumb\_size\_fact}) \quad (61)$$

The feature factor ratios of this expression attempt to compare the features of the innermost contact to current features of the next innermost contact, which is already known to be a fingertip. If the innermost contact is also a fingertip its features should be similar to the next innermost, causing the ratios to remain near one. However, thumb-like features on the innermost contact will cause the ratios to be large. Therefore if the combined thumb factor exceeds a high threshold, diamond **404** decides the innermost finger contact is definitely a thumb. If decision diamond **412** determines the contact is not already assigned to the thumb attractor **412**, step **414** shifts the contact assignment inward on the attractor ring to the thumb attractor. Otherwise, if decision diamond **406** determines that the combined thumb factor is less than a low threshold, the innermost contact is most definitely not the thumb. Therefore if decision diamond **408** finds the contact assigned to the thumb attractor, step **410** shifts the innermost contact assignment and any adjacent finger contacts outward on the attractor ring to unfill the thumb attractor. If the combined_thumb_fact is between the high and low threshold or if the existing assignments agree with the threshold decisions, step **413** makes no assignment changes.

The hand contact features and interrelationships introduced here to aid identification can be measured and combined in various alternative ways yet remain well within the scope of the invention. In alternative embodiments of the multi-touch surface apparatus which include raised, touch-insensitive palm rests, palm identification and its requisite attractors and factors may be eliminated. Geometrical parameters can be optimally adapted to measurements of individual user hand size taken while the hand is flattened. However, the attractor-based identification method already tolerates variations in a single person's finger positions due to finger flexion and extension which are as great or greater than the variations in hand size across adult persons. Therefore adaptation of the thumb and palm size factors to a person's average finger and palm heel proximities is more important than adaptation of attractor positions to individual finger lengths, which will only add marginal performance improvements.

As another example of an alternative method for incorporating these features and relationships into a hand contact identifier, FIG. **27** diagrams an alternative finger identification embodiment which does not include an attractor template. To order the paths from finger and palm contacts within a given hand **430**, step **432** constructs a two-dimensional matrix of the distances from each contact to the other contacts. In step **434**, a shortest path algorithm well known from the theory of network flow optimization then finds the shortest graph cycle connecting all the contact paths and passing

through each once **434**. Since hand contacts tend to lie in a ring this shortest graph cycle will tend to connect adjacent contacts, thus establishing a sensible ordering for them.

The next step **438** is to pick a contact at an extreme position in the ring such as the innermost or outermost and test whether it is a thumb (decision diamond **440**) or palm (decision diamond **442**). This can be done using contact features and fuzzy logic expressions analogous to those utilized in the thumb verification process and the, attractor weightings. If the innermost path is a thumb, step **444** concludes that contacts above are most likely fingertips, and contacts in the ring below the thumb are most likely palms. If (**442**) the innermost path is a palm heel, step **446** concludes the paths significantly above the innermost must be fingers while paths at the same vertical level should be palms. The thumb and palm tests are then repeated for the contacts adjacent in the ring to the innermost until any other thumb or palm contacts are found. Once any thumb and palm contacts are identified, step **448** identifies remaining fingertip contacts by their respective ordering in the ring and their relatively high vertical position.

Since this alternative algorithm does not include an attractor template to impose constraints on relative positions, the fuzzy verification functions for each contact may need to include measurements of the vertical position of the contact relative to other contacts in the ring and relative to the estimated hand offset. The attractor template embodiment is preferred over this alternative embodiment because the attractor embodiment more elegantly incorporates expected angles between contacts and the estimated hand offset into the finger identification process.

Hand identification is needed for multi-touch surfaces which are large enough to accomodate both hands simultaneously and which have the left and right halves of the surface joined such that a hand can roam freely across the middle to either half of the surface. The simplest method of hand identification would be to assign hand identity to each contact according to whether the contact initially touched down in the left or right half of the surface. However, if a hand touched down in the middle, straddling the left and right halves, some of the hand's contacts would end up assigned to the left hand and others to the right hand. Therefore more sophisticated methods which take into account the clustering properties of hand contacts must be applied to ensure all contacts from the same hand get the same identity. Once all surface contacts are initially identified, the path tracking module can reliably retain existing identifications as a hand slides from one side of the surface to the other.

The thumb and inner palm contact orientations and the relative thumb placement are the only contact features independent of cluster position which distinguish a lone cluster of right hand contacts from a cluster of left hand contacts. If the thumb is lifted off the surface, a right hand contact cluster appears nearly indistinguishable from a left hand cluster. In this case cluster identification must still depend heavily on which side of the board the cluster starts on, but the identity of contacts which recently lifted off nearby also proves helpful. For example, if the right hand moves from the right side to the middle of the surface and lifts off, the next contacts which appear in the middle will most likely be from the right hand touching back down, not from the left hand moving to the middle and displacing the right hand. The division between left and right halves of the surface should therefore be dynamic, shifting toward the right or left according to which hand was most recently near the middle. Since the hand offset estimates temporarily retain the last known hand positions after liftoff, such a dynamic division is implemented by tying the positions of left hand and right hand attractor templates to the estimated hard positions.

Though cases remain in which the user can fool the hand identification system with sudden placements of a hand in unexpected locations, the user may actually wish to fool the system in these cases. For example, users with only one hand free to use the surface may intentionally place the hand far onto the opposite half of the surface to access the chord input operations of the opposite hand. Therefore, when a hand cluster suddenly touches down well into the opposite half of the surface, it can safely be given the opposite halfs identity, regardless of its true identity. Arching the surface across the middle can also discourage users from sliding a hand to the opposite side by causing awkward forearm pronation should users do so.

FIG. **29** shows process details within the hand identification module **247**. Decision diamond **450** first determines whether the hand identification algorithm actually needs to be executed by checking whether all path proximities have stabilized. To maximize stability of the identifications, hand and finger identities need only be reevaluated when a new hand part touches down or disambiguating features of existing contacts become stronger. The contact size and orientation features are unreliable until the flesh fully compresses against the surface a few dozen milliseconds after initial surface contact. Therefore decision diamond **450** executes the hand identification algorithm for each proximity image in which a new contact appears and for subsequent proximity images in which the total proximity of any new contacts continues to increase. For images in which proximities of existing contacts have stabilized and no new contacts appear, path continuation as performed by the path tracking process **245** is sufficient to retain and extend (step **452**) the contact identifications computed from previous images.

Should the hand identification algorithm be invoked for the current image, the first step **453** is to define and position left and right hand attractor templates. These should be basically the same as the attractor templates (FIG. **24**, step **352**) used in within-hand identification, except that both left and right rings must now be utilized at once. The default placement of the rings relative to one another should correspond to the default left and right hand contact positions shown in FIG. **20**A. Each ring translates to follow the estimated position of its hand, just like the sloppy segmentation regions follow the hands in FIG. **20**B. Individual attractor points can safely be translated by their corresponding estimated finger offsets. Therefore the final attractor positions ($Aj_x[n]$,$Aj_y[n]$) for the left hand L and right hand H attractor rings are:

$$Laj_x[n]=Lh_{eox}[n]+LFj_{eox}[n]+Lfj_{defx} \quad (62)$$

$$Laj_y[n]=Lh_{eoy}[n]+LFj_{eoy}[n]+Lfj_{defy} \quad (63)$$

$$Raj_x[n]=Rh_{eox}[n]+RFj_{eox}[n]+Rfj_{defx} \quad (64)$$

$$Raj_y[n]=Rh_{eoy}[n]+RFj_{eoy}[n]+Rfj_{defy} \quad (65)$$

Basically the hand identification algorithm will compare the cost of assigning contacts to attractors in one ring versus the other, the cost depending on the sum of weighted distances between each contact and its assigned attractor. Adjusting the attractor ring with the estimated hand and finger offsets lowers the relative costs for assignment hypotheses which resemble recent hand assignments, helping to stabilize identifications across successive proximity images even when hands temporarily lift off.

Next a set of assignment hypotheses must be generated and compared. The most efficient way to generate sensible

hypotheses is to define a set of roughly vertical contour lines, one between each horizontally adjacent contact. Step **454** does this by ordering all surface contacts by their horizontal coordinates and establishing a vertical contour halfway between each pair of adjacent horizontal coordinates. FIGS. **30**A-C show examples of three different contours **475** and their associated assignment hypotheses for a fixed set of contacts. Each contour corresponds to a separate hypothesis, known also as a partition, in which all contacts to the left **476** of the contour are from the left hand, and all contacts to the right **477** of the contour are from the right hand. Contours are also necessary at the left and right ends of the surface to handle the hypotheses that all contacts on the surface are from the same hand. Contours which hypothesize more contacts on a given hand than can be caused by a single hand are immediately eliminated.

Generating partitions via vertical contours avoids all hypotheses in which contacts of one hand horizontally overlap or cross over contacts of the opposite hand. Considering that each hand can cause seven or more distinct contacts, this reduces the number of hand identity permutations to examine from thousands to at most a dozen. With fewer hypotheses to examine, the evaluation of each partition can be much more sophisticated, and if necessary, computationally costly.

The optimization search loop follows. Its goal is to determine which of the contours divides the contacts into a partition of two contact clusters such that the cluster positions and arrangement of contacts within each cluster best satisfy known anatomical and biomechanical constraints. The optimization begins by picking (step **456**) a first contour divider such as the leftmost and tentatively assigning (step **458**) any contacts to the left of the contour to the left hand and the rest to the right hand. Step **460** invokes the finger identification algorithm of FIG. **23**, which attempts to assign finger and palm identities to contacts within each hand. Decision diamond **360** avoids the computational expense of thumb verification **368** and statistics gathering **364** for this tentative assignment hypothesis.

Returning to FIG. **29**, step **462** computes a cost for the partition. This cost is meant to evaluate how well the tentatively identified contacts fit their assigned attractor ring and how well the partition meets between-hand separation constraints. This is done by computing for each hand the sum of weighted distances from each tentatively identified contact to its assigned attractor point as in Equation 54 of finger identification, including size and orientation feature factors for thumb and palm attractors. This sum represents the basic template fitting cost for a hand. Each hand cost is then weighted as a whole with the reciprocals of its clutching velocity, handedness, and palm cohesion factors. These factors, to be described below, represent additional constraints which are underemphasized by the weighted attractor distances. Finally, the weighted left and right hand costs are added together and scaled by the reciprocal of a hand separation factor to obtain a total cost for the partition.

If decision diamond **464** determines this total cost is lower than the total costs of the partitions evaluated so far **464**, step **466** records the partition cost as the lowest and records the dividing contour. Decision diamond **472** repeats this process for each contour **470** until the costs of all partitions have been evaluated. Step **473** chooses the partition which has the lowest cost overall as the actual hand partitioning **473**, and the hand identities of all contact paths are updated accordingly. Then step **474** reinvokes the within-hand finger contact identification process so that the thumb verification and statistics gathering processes are performed using the actual hand assignments.

Users often perform clutching motions in which the right hand, for example, lifts off from a slide at the right side of the surface, touches backdown in the middle of the surface, and resumes sliding toward the right. Therefore when a hand is detected touching down in the middle of the surface and sliding toward one side, it probably came from the at side. A hand velocity factor, plotted approximately in FIG. **31**A, captures this phenomenon by slightly increasing in value when a hand cluster's contacts are moving toward the cluster's assigned side of the board, thus decreasing the basic cost of the hand. The factor is a function of the average of the contacts' horizontal velocities the side of the surface the given cluster is assigned. Since high speeds do not necessarily give a stronger indication of user intent the factor saturates at moderate speeds.

Though the thumb orientation factors help identify which hand a thumb is from when the thumb lies in the ambiguous middle region of the surface, the vertical position of the thumb relative to other fingers in the same hand also gives a strong indication of handedness. The thumb tends to be positioned much lower than the fingertips, but the pinky tends to be only slightly lower than the other fingertips. The handedness factor plotted approximately in FIG. **31**B, takes advantage of this constraint by boosting the hand cost when the contact identified as the outermost fingertip is more than a couple centimeters lower than the next outermost fingertip contact. In such cases the tentative hand assignment for all contacts in the cluster is probably wrong. Since this causes the within-hand identification algorithm to fit the contacts to the wrong attractor ring, finger identities become reversed such that the supposedly lowered pinky is truly a lowered thumb of the opposite hand. Unfortunately, limited confidence can be placed in the handedness factor. Though the pinky should not appear lowered as much as the thumb the outer palm heel can, creating an ambiguity in which the thumb and fingertips of one hand have the same contact arrangement as the fingertips and outer palm heel of the opposite hand. This ambiguity can cause the handedness factor to be erroneously low for an accurately identified hand cluster, so the handedness factor is only used on clusters in the middle of the surface where hand position is ambiguous.

Distinguishing contact clusters is challenging because a cluster can become quite sparse and large when the fingers outstretched, with the pinky and thumb of the same hand spanning up to 20 cm. However, the palm can stretch very little in comparison, placing useful constraints on how far apart palm heel contacts and forepalms from the same hand can be. The entire palm region of an outstretched adult hand is about 10 cm square, so palm contact centroids should not be scattered over a region larger than about 8 cm. When a partition wrongly includes fingers from the opposite hand in a cluster, the within-cluster identification algorithm tends to assign the extra fingers from the opposite hand to palm heel and forepalm attractors. This usually causes the contacts assigned to the cluster's palm attractors to be scattered across the surface wider than is plausible for true palm contacts from a single hand. To punish such partitions, the palm cohesion factor quickly drops below one for a tentative hand cluster in which the supposed palm contacts are scattered over a region larger than 8 cm. Therefore its reciprocal will greatly increase the hand's basic cost. FIG. **31**C shows the value of the palm cohesion factor versus horizontal separation between palm contacts. The horizontal spread can be efficiently measured by finding the maximum and minimum horizontal coordinates of all contacts identified as palm heels or forepalms and taking the difference between the maximum and minimum. The measurement and factor value lookup are repeated for the

vertical separation, and the horizontal and vertical factors are multiplicatively combined to obtain the final palm cohesion factor.

FIG. **33** is an approximate plot of the inter-hand separation factor. This factor increases the total costs of partitions in which the estimated or actual horizontal positions of the thumbs from each hand approach or overlap. It is measured by finding the minimum of the horizontal offsets of right hand contacts with respect to their corresponding default finger positions. Similarly the maximum of the horizontal offsets of the left hand contacts with respect to their corresponding default finger positions is found. If the difference between these hand offset extremes is small enough to suggest the thumbs are overlapping the same columnar region of the surface while either touching the surface or floating above it, the separation factor becomes very small. Such overlap corresponds to a negative thumb separation in the plot. To encourage assignment of contacts which are within a couple centimeters of one another to the same cluster, the separation factor gradually begins to drop starting with positive separations of a few centimeters or less. The inter-hand separation factor is not applicable to partitions in which all surface contacts are assigned to the same hand, and takes on the default value of one in this case.

Alternative embodiments of this hand identification process can include additional constraint factors and remain well within the scope of this invention. For example, a velocity coherence factor could be computed to favor partitions in which all fingers within a cluster slide at approximately the same speed and direction, though each cluster as a whole has a different average speed and direction.

Sometimes irreversible decisions made by the chord motion recognizer or typing recognized on the basis of existing hand identifications prevent late changes in the identifications of hand contacts even when new proximity image information suggests existing identifications are wrong. This might be the case for a chord slide which generates input events that can not be undone, yet well into the slide new image information indicates some fingers in the chord should have been attributed to the opposite hand. In this case the user can be warned to stop the slide and check for possible input errors but in the meantime it is best to retain the existing identifications even if wrong, rather than switch to correct assignments which could have further unpredictable effects when added to the erroneous input events. Therefore once a chord slide has generated input events, the identifications of their existing paths may be locked so the hand identification algorithm can only swap identifications of subsequent new contacts.

This hand identification process can be modified for differently configured multi-touch surfaces and remain well within the scope of this invention. For surfaces which are so narrow that thumbs invade one another's space or so tall that one hand can lie above another, the contours need not be straight vertical lines. Additional contours could weave around candidate overlapping thumbs, or they could be perpendicular to the vector between the estimated hand positions. If the surface was large enough for more than one user, additional attractor rings would have to be provided for each additional hand, and multiple partitioning contours would be necessary per hypothesis to partition the surface into more than two portions. On a surface large enough for only one hand it might still be necessary to determine which hand was touching the surface. Then instead of hypothesizing different contours, the hand identification module would evaluate the hypotheses that either the left hand attractor ring or the right hand attractor ring was centered on the surface. If the surface was mounted on a pedestal to allow access from all sides, the hand identification module would also hypothesize various rotations of each attractor ring.

The attractor-based finger identification system **248** will successfully identify the individual hand contacts which comprise the pen grip hand configuration (FIG. **15**). However, additional steps are needed to distinguish the unique finger arrangement within the pen grip from the normal arrangement within the closed hand configuration (FIG. **14**). In this pen grip arrangement the outer fingers curl under toward the palms so their knuckles touch the surface and the index, finger juts out ahead of them. The pen grip detection module **17** employs a fuzzy pattern recognition process similar to the thumb verification process to detect this unique arrangement

An additional problem with handwriting recognition via the pen grip hand configuration is that the inner gripping fingers and sometimes the whole hand will be picked up between strokes, causing the distinguishing finger arrangement to temporarily disappear. Therefore the pen grip recognition process must have hysteresis to stay in handwriting mode between gripping finger lifts. In the preferred embodiment, hysteresis is obtained by temporal filtering of the combined fuzzy decision factors and by using the estimated finger positions in measurements of finger arrangement while the actual fingers are lifted off the surface. The estimated finger positions provide effective hysteresis because they temporarily retain the unique jutting arrangement before decaying back toward the normal arched fingertip positions a few seconds after liftoff.

FIG. **28** shows the steps within the pen grip detection module **17**. Decision diamond **485** determines whether all pen grip hand parts are touching the surface. If not decision diamond **486** causes the estimated finger and palm positions to be retrieved for any lifted parts in step **487** only if pen grip or handwriting mode is already active. Otherwise the process exits for lack of enough surface contacts. Thus the estimated finger positions cannot be used to start handwriting mode, but they can continue it. Step **488** retrieves the measured positions and sizes of fingers and palm heels which are touching the surface.

Step **489** computes a knuckle factor from the outer finger sizes and their vertical distance from the palm heels which peaks as the outer finger contacts become larger than normal fingertips and close to the palm heels. Step **490** computes a jutting factor from the difference between the vertical coordinates of the inner and outer fingers which peaks as the index fingertip juts further out in front of the knuckles. Step **491** combines the knuckle and jutting factors in a fuzzy logic expression and averages the result with previous results via an autoregressive or moving average filter. Decision diamond **492** continues or starts pen grip mode if the filtered expression result is above a threshold which may itself be variable to provide additional hysteresis. While in pen grip mode, typing **12** and chord motion recognition **18** are disabled for the pen gripping hand.

In pen grip mode, decision diamond **493** determines whether the inner gripping fingers are actually touching the surface. If so, step **495** generates inking events from the path parameters of the inner fingers and appends them to the outgoing event queue of the host communication interface. These inking events can either cause "digital ink" to be laved on the display **24** for drawing or signature capture purposes, or they can be intercepted by a handwriting recognition system and interpreted as gestures or language symbols. Handwriting recognition systems are well known in the art.

If the inner fingers are lifted, step **494** sends stylus raised events to the host communication interface to instruct the handwriting recognition system of a break between symbols. In some applications the user may need to indicate where the "digital ink" or interpreted symbols are to be inserted on the display by positioning a cursor. Though on a multi-touch surface a user could move the cursor by leaving the pen grip configuration and sliding a finger chord, it is preferable to allow cursor positioning without leaving the pen grip configuration. This can be supported by generating cursor positioning events from slides of the palm heels and outer knuckles. Since normal writing motions will also include slides of the palm heels and outer knuckles, palm motions should be ignored until the inner fingers have been lifted for a few hundred milliseconds.

Should the user actually pick up a conductive stylus and attempt to write with it, the hand configuration will change slightly because the inner gripping fingers will be directing the stylus from above the surface rather than touching the surface during strokes. Since the forearm tends to supinate more when actually holding a stylus, the inner palm heel may also stay off the surface while the hand rests on the sides of the pinky, ring finger and the outer palm heel. Though the outer palm heel may lie further outward than normal with respect to the pinky, the ring and pinky fingers will still appear as large knuckle contacts curled close to the outer palm. The tip of the stylus essentially takes the place of the index fingertip for identification purposes, remaining at or above the vertical level of the knuckles. Thus the pen grip detector can function in essentially the same way when the user writes with a stylus, except that the index fingertip path sent to the host communication interface will in actuality be caused by the stylus.

Technically, each hand has 24 degrees of freedom of movement in all finger joints combined, but as a practical matter, tendon linkage limitations make it difficult to move all of the joints independently. Measurements of finger contacts on a surface yield ten degrees of freedom in motion lateral to the surface, five degrees of freedom in individual fingertip pressure or proximity to the surface, and one degree of freedom of thumb orientation. However, many of these degrees of freedom have limited ranges and would require unreasonable twisting and dexterity from the average user to access independently.

The purpose of the motion component extraction module **16** is to extract from the 16 observable degrees of freedom enough degrees of freedom for common graphical manipulation tasks in two and three dimensions. In two dimensions the most common tasks are horizontal and vertical panning, rotating, and zooming or resizing. In three dimensions, two additional rotational degrees of freedom are available around the horizontal and vertical axes. The motion component extractor attempts to extract these 4-6 degrees of freedom from those basic hand motions which can be performed easily and at the same time without interfering with one another. When multiple degrees of freedom can be accessed at the same time they are said to be integral rather than separable, and integral input devices are usually faster because they allow diagonal motions rather than restricting motions to be along a single axis or degree of freedom at one time.

When only four degrees of freedom are needed, the basic motions can be whole hand translation, hand scaling by uniformly flexing or extending the fingers, and hand rotation either about the wrist as when unscrewing ajar lid or between the fingers as when unscrewing a nut. Not only are these hand motions easy to perform because they utilize motions which intuitively include the opposable thumb, they correspond cognitively to the graphical manipulation tasks of object rotation and sizing. Their only drawback is that the translational motions of all the fingers during these hand rotations and scalings do not cancel perfectly and can instead add up to a net translation in some direction in addition to the desired rotation or scaling. To allow all motions to be performed simultaneously so that the degrees of freedom are integral yet to prevent unintended translations from imperfectly performed scalings and rotations, the motion extractor preferentially weights the fingers whose translations cancel best and nonlinearly scales velocity components depending on their speeds relative to one another.

The processes within the motion component extractor **16** are shown in FIG. **34**. Step **500** first fetches the identified contact paths **250** for the given hand. These paths contain the lateral velocities and proximities to be used in the motion calculations, and the identifications are needed so that motion of certain fingers or palm heels which would degrade particular motion component calculations can be deemphasized.

The next step **502** applies additional filtering to the lateral contact velocities when finger proximity is changing rapidly. This is necessary because during finger liftoff and touch down on the surface, the front part of the fingertip often touches down before and lifts off after the back of the fingertip, causing a net downward or upward lateral translation in the finger centroid. Such proximity-dependent translations can be put to good use when slowly rolling the fingertip for fine positioning control, but they can also annoy the user if they cause the cursor to jump away from a selected position during finger liftoff. This is prevented by temporarily downscaling a finger's lateral velocity in proportion to large changes in the finger's proximity. Since other fingers within a hand tend to shift slightly as one finger lifts off, additional downscaling of each finger velocity is done in response to the maximum percent change in proximity among contacting fingers. Alternatively, more precise suppression can be obtained by subtracting from the lateral finger speed an amount proportional to the instantaneous change in finger contact height. This assumes that the perturbation in lateral finger velocity caused by finger liftoff is proportional to the change in contact height due to the back of the fingertip lifting off first or touching down last.

Process **504**, whose detailed steps are shown in FIG. **36**, measures the polar velocity components from radial (scaling) and rotational motion. Unless rotation is extracted from thumb orientation changes, at least two contacting fingers are necessary to compute a radial or angular velocity of the hand. Since thumb motion is much more independent of the other fingers than they are of one another, scalings and rotations are easier for the user to perform if one of these fingers is the opposable thumb, but the measurement method will work without the thumb. If decision diamond **522** determines that less than two fingers are touching the surface, step **524** sets the radial and rotational velocities of the hand to zero. FIG. **35** shows trajectories of each finger during a contractive hand scaling. The thumb **201** and pinky **205** travel in nearly opposite directions at roughly the same speed, so that the sum of their motions cancels for zero net translation, but the difference in their motions is maximized for a large net scaling. The central fingers **202-204** also move toward a central point but the palm heels remain stationary, failing to complement the flexing of the central fingers. Therefore the difference between motion of a central finger and any other finger is usually less than the difference between the pinky and thumb motions, and the sum of central finger velocities during a hand scaling adds up to a net vertical translation. Similar phenomena occur during hand rotations, except that if the rotation is centered at the wrist with forearm fixed rather than

centered at the forepalms, a net horizontal translation will appear in the sum of motions from any combination of fingers.

Since the differences in finger motion are usually greatest between thumb and pinky, step **526** only retrieves the current and previous positions of the innermost and outermost touching fingers for the hand scaling and rotation measurements.

Step **528** then computes the hand scaling velocity H_{xx} from the change in distance between the innermost finger FI and outermost finger FO with approximately the following equation:

$$Hvs[n] = \frac{d(FI[n], FO[n]) - d(fi[n-1], FO[n-1])}{\Delta t} \tag{66}$$

where d(FI[n],FO[n]) is the squared Euclidean distance between the fingers:

$$d(FI[n],FO[n]=\sqrt{(FI_x[n]-FO_x[n])^2+(FI_y[n]-FO_y[n])^2} \tag{67}$$

If one of the innermost or outermost fingers was not touching during the previous proximity image, the change in separation is assumed to be zero. Similarly, step **530** computes the hand rotational velocity H_{xx} from the change in angle between the innermost and outermost finger with approximately the following equation:

$$H_{vr}[n]\left(\frac{\angle(FI[n], FO[n]) - \angle(FI[n-1], FO[n-1]}{\Delta t}\right)\times\left(\frac{d(FI[n], FO[n]}{\pi}\right) \tag{68}$$

The change in angle is multiplied by the current separation to convert it to the same units as the translation and scaling components. These equations capture any rotation and scaling components of hand motion even if the hand is also translating as a whole, thus making the rotation and scaling degrees of freedom integral with translation.

Another reason the computations above are restricted to the thumb and pinky or innermost and outermost fingers is that users may want to make fine translating manipulations with the central fingers, i.e., index, middle, and ring, while the thumb and pinky remain stationary. If changes in distances or angles between the central fingers and the thumb were averaged with Equations 66-68, this would not be possible because central finger translations would cause the appearance of rotation or scaling with respect to the stationary thumb or pinky. However, Equations 56-60 applied in the thumb verification process are only sensitive to symmetric rotation and scaling about a fixed point between the fingers. They approach zero if any significant whole hand translation is occurring or the finger motions are not complementary. In case the user fails to properly move the outermost finger during a rotation or scaling gesture, step **531** uses equations of the approximate form of Equations 56-60 to compute rotation and scaling velocities between the thumb and any touching fingers other than the outermost. The resulting velocities are preferably combined with the results of Equations 66-68 via a maximum operation rather than an average in case translational motion causes the fixed point rotations or scalings to be zero. Finally, decision diamond **532** orders a check for radial or rotational deceleration **534** during motions prior to finger liftoff. The method for detecting radial or rotational deceleration is the same as that detailed in the description of translation extraction.

FIG. **37** shows the details of hand translational velocity measurements referred to in process **506** of FIG. **34**. The simplest way to compute a hand translation velocity would be to simply average the lateral velocities of each finger. However, the user expects the motion or control to display gain to be constant regardless of how many fingers are being moved, even if some are resting stationary. Furthermore, if the user is simultaneously scaling or rotating the hand, a simple average is sensitive to spurious net translations caused: by uncanceled central finger motions.

Therefore, in a preferred embodiment the translational component extractor carefully assigns weightings for each finger before computing the average translation. Step **540** initializes the translation weighting Fi_{vw} of each finger to its total contact proximity, i.e., $Fi_{vw}[n]=Fi_z[n]$. This ensures that fingers not touching the surface do not dilute the average with their zero velocities and that fingers which only touch lightly have less influence since their position and velocity measurements may be less reliable. The next step **544** decreases the weightings of fingers which are relatively stationary so that the control to display gain of intentionally moving fingers is not diluted. This can be done by finding the fastest moving finger, recording its speed as a maximum finger speed and scaling each finger's translation weighting in proportion to its speed divided by the maximum of the finger speeds, as shown approximately in the formula below:

$$Fi_{vw}[n]: = Fi_{vw}[n]\times\left(\frac{Fi_{speed}[n]}{\max_j Fi_{speed}[n]}\right)^{ptw} \tag{69}$$

where the power ptw adjusts the strength of the speed dependence. Note that step **544** can be skipped for applications such as computer-aided-design in which users desire both a normal cursor motion gain mode and a low gain mode. Lower cursor motion gain is useful for fine, short range positioning, and would be accessed by moving only one or two fingers while keeping the rest stationary.

Step **546** decreases the translation weightings for the central fingers during hand scalings and rotations, though it does not prevent the central fingers from making fine translational manipulations while the thumb and pinky are stationary. The formulas below accomplish this seamlessly by downscaling the central translation weightings as the magnitudes of the rotation and scaling velocities become significant compared to $K_{polarthresh}$:

$$Fi_{vwx}[n] \approx \frac{Fi_{vw}[n]\times K_{polarthresh}}{K_{polarthresh}+|H_{vr}[n]|} \tag{70}$$

$$Fi_{vwy}[n] \approx \frac{Fi_{vw}[n]\times K_{polarthresh}}{K_{polarthresh}+|H_{vr}[n]|+|H_{vs}[n]|} \tag{71}$$

where these equations are applied only to the central fingers whose identities i are between the innermost and outermost. Note that since hand scaling does not cause much horizontal translation bias, the horizontal translation weighting $Fi_{vwx}[n]$ need not be affected by hand scaling velocity $H_{vs}[n]$, as indicated by the lack of a hand scaling term in Equation 70. The translation weightings of the innermost and outermost fingers are unchanged by the polar component speeds, i.e., $FI_{vwx}[n]=FI_{vwx}[n]=FI_{vw}[n]$ and $FO_{vwx}[n]=FO_{vwx}[n]=FO_{vw}[n]$. Step **548** finally computes the hand translation velocity vector $(H_{vx}[n],H_{vy}[n])$ from the weighted average of the finger velocities:

$$H_{vx}[n] = \frac{\sum_{i-1}^{5} Fi_{vwx} Fi_{vx}}{\sum_{i-1}^{5} Fi_{vwx}} \tag{72}$$

$$H_{vy}[n] = \frac{\sum_{i-1}^{5} Fi_{vwy} Fi_{vy}}{\sum_{i-1}^{5} F_{vwy}} \tag{73}$$

The last part of the translation calculations is to test for the lateral deceleration of the fingers before liftoff, which reliably indicates whether the user wishes cursor motion to stop at liftoff. If deceleration is not detected prior to liftoff, the user may intend cursor motion to continue after liftoff, or the user may intend a special "one-shot" command to be invoked. Decision diamond **550** only invokes the deceleration tests while finger proximities are not dropping too quickly, to prevent the perturbations in finger centroids which can accompany finger liftoff from interfering with the deceleration measurements. Step **551** computes the percentage acceleration or ratio of current translation speed $|H_{vx}[n],H_{vy}[n])|$ to a past average translation speed preferably computed by a moving window average or autoregressive filter. Decision diamond **552** causes the translation deceleration flag to be set **556** if the acceleration ratio is less than a threshold. If this threshold is set greater than one, the user will have to be accelerating the fingers just prior to liftoff for cursor motion to continue. If the threshold is set just below one, cursor motion will reliably be continued as long as the user maintains a constant lateral speed prior to liftoff, but if the user begins to slow the cursor on approach to a target area of the display the deceleration flag will be set. Decision diamond **554** can also cause the deceleration flag to be set if the current translation direction is substantially different from an average of past directions. Such change in direction indicates the hand motion trajectory is curving, in which case cursor motion should not be continued after liftoff because accurately determining the direction to the user's intended target becomes very difficult. If neither deceleration nor curved trajectories are detected, step **558** clears the translation deceleration flag. This will enable cursor motion continuation should the fingers subsequently begin liftoff. Note that decision diamond **550** prevents the state of the translation deceleration flags from changing during liftoff so that the decision after liftoff to continue cursor motion depends on the state of the deceleration flag before liftoff began. The final step **560** updates the autoregressive or moving window average of the hand translation velocity vector, which can become the velocity of continued cursor motion after liftoff. Actual generation of the continued cursor motion signals occurs in the chord motion recognizer **18** as will be discussed with FIG. **40**.

Note that this cursor motion continuation method has several advantages over motion continuation methods in related art. Since the decision to continue motion depends on a percentage acceleration which inherently normalizes to any speed range, the user can intentionally invoke motion continuation from a wide range of speeds including very low speeds. Thus the user can directly invoke slow motion continuation to auto scroll a document at readable speeds. This is not true of Watanabe's method in U.S. Pat. No. 4,734,685, which only continues motion when the user's motion exceeds a high speed threshold, nor of Logan et al.'s method in U.S. Pat. No. 5,327,161, which if enabled for low finger speeds will undesirably continue motion when a user decelerates on approach to a large target but fails to stop completely before lifting off. Percentage acceleration also captures user intent more clearly than position of a finger in a border area. Position of a finger in a border area as used in U.S. Pat. No. 5,543,591 to Gillespie et al. is ambiguous because the cursor can reach its desired target on the display just as the finger enters the border, yet the touchpad device will continue cursor motion past the target because it thinks the finger has run out of space to move. In the present invention, on the other hand, the acceleration ratio will remain near one if the fingers can slide off the edge of the sensing array without hitting a physical barrier, sensibly invoking motion continuation. But if the fingers decelerate before crossing or stop on the edge of the sensing array, the cursor will stop as desired.

The details of the differential hand pressure extraction process **508** are shown in FIG. **38**. Fingertip proximity, quickly saturates when pressure is applied through the bony tip normal to a hard surface. Unless the surface itself is highly compliant, the best dynamic range of fingertip pressure is obtained with the fingers outstretched and hand nearly flattened so that the compressible soft pulp underneath the fingertips rests on the surface. Decision diamond **562** therefore causes the tilt and roll hand pressure components to be set to zero in step **564** and pressure extraction to abort unless the hand is nearly flattened. Inherent in the test for hand flattening **562** is a finger count to ensure that most of the five fingers and both palm heels are touching the surface to maximize the precision of the hand pressure measurements, though technically only three non-collinear hand contacts arranged like a tripod are necessary to establish tilt and roll pressures. Decision diamond **562** can also require the user to explicitly enable three-dimensional manipulation with an intuitive gesture such as placing all five fingers on the surface briefly tapping the palm heels on the surface, and finally resting the palm heels on the surface. Decision diamond **566** causes step **568** to capture and store reference proximities for each contact path when the proximity of all contacts have stabilized at the end of this initiation sequence. The tilt and roll pressure components are again zeroed **564** for the sensor array scan cycle during which this calibration is performed.

However, during subsequent scan cycles the user can tilt the hand forward applying more pressure to the fingertips or backward applying more pressure to the palm heels or the user can roll the hand outward onto the pinky and outer palm heel or inward applying more pressure to the thumb, index finger and inner palm heel. Step **5170** will proceed to calculate an unweighted average of the current contact positions. Step **572** computes for each hand part still touching the surface the ratio of current proximity to the reference proximity previously stored. To make these ratios less sensitive to accidental lifting of hand parts, step **574** clips them to be greater or equal to one so only increases in proximity and pressure register in, the tilt and roll measurements. Another average contact path position is computed in step **576**, but this one is weighted by the above computed proximity ratios for each path. The difference between these weighted and unweighted contact position averages taken in step **578** produces a vector whose direction can indicate the direction of roll or tilt and whose magnitude can control the rate of roll or tilt about x and y axes.

Since the weighted and unweighted position averages are only influenced by positions of currently contacting fingers and increases in contact pressure or proximity, the method is insensitive to finger liftoffs. Computation of reference-normalized proximity ratios in step **572** rather than absolute

changes in proximity prevents the large palm heel contacts from having undue influence on the weighted average position.

Since only the current contact positions are used in the average position computations, the roll and tilt vector is independent of lateral motions such as hand translation or rotation as long as the lateral motions do not disturb finger pressure, thus once again achieving integrality. However, hand scaling and differential hand pressure are difficult to use at the same time because flexing the fingers generally causes significant decreases in fingertip contact area and thus interferes with inference of fingertip pressure changes. When this becomes a serious problem, a total hand pressure component can be used as a sixth degree of freedom in place of the hand scaling component. This total pressure component causes cursor velocity along a z-axis in proportion to deviations of the average of the contact proximity ratios from one. Alternative embodiments may include further enhancements such as adapting the reference proximities to slow variations in resting hand pressure and applying a dead zone filter to ignore pressure difference vectors with small magnitudes

Despite the care taken to measure the polar velocity, translation velocity, and hand pressure components in such a way that the resultant vectors are independent of one another, uneven finger motion during hand scaling, rotation, or translation can still cause minor perturbations in measurements of one degree of freedom while primarily attempting to move in another. Non-linear filtering applied in steps **510** and **512** of FIG. **34** removes the remaining motion leakage between dominant components and nearly stationary components. In steps **510** each component velocity is downscaled by the ratio of its average speed to the maximum of all the component speeds, the dominant component speed:

$$H_{vx}[n] := H_{vx}[n] \times \left(\frac{H_{xyspeed}[n]}{dominantspeed}\right)^{pds} \tag{74}$$

$$H_{vy}[n] := H_{vy}[n] \times \left(\frac{H_{xyspeed}[n]}{dominantspeed}\right)^{pds} \tag{75}$$

$$H_{vs}[n] := H_{vs}[n] \times \left(\frac{H_{sspeed}[n]}{dominantspeed}\right)^{pds} \tag{76}$$

$$H_{vr}[n] := H_{vr}[n] \times \left(\frac{H_{rspeed}[n]}{dominantspeed}\right)^{pds} \tag{77}$$

where $H_{xyspeed}[n]$, $H_{sspeed}[n]$, and $H_{rspeed}[n]$ are autoregressive averages over time of the translation speed, scaling speed, and rotational speed, where:

$$\text{dominant\_speed} = \max(H_{xyspeed}[n], H_{sspeed}[n], H_{rspeed}[n]) \tag{78}$$

where pds controls the strength of the filter. As pdy is adjusted towards infinity the dominant component is picked out and all components less than the dominant tend toward zero producing the orthogonal cursor effect well-known in drawing applications. As pds is adjusted towards zero the filters have no effect. Preferably, pds is set in between so that components significantly slower than the dominant are slowed further, but components close to the dominant in speed are barely affected, preserving the possibility of diagonal motion in multiple degrees of freedom at once. The autoregressive averaging helps to pick out the component or components which are dominant over the long term and suppress the others even while the dominant components are slowing to a stop.

Step **512** takes a second pass with a related filter known as a dead-zone filter. A dead-zone filter produces zero output velocity for input velocities less than a speed threshold but produces output speeds in proportion to the difference between the input speed and the threshold for input velocities that exceed the threshold. Preferably the speed threshold or width of the dead zone is set to a fraction of the maximum of current component speeds. All velocity components are filtered using this same dead zone width. The final extracted component velocities are forwarded to the chord motion recognizer module **18** which will determine what if any input events should be generated from the motions.

FIG. **39**A shows the details of the finger synchronization detector module **14**. The synchronization detection process described below is repeated for each hand independently. Step **600** fetches proximity markers and identifications for the hand's current paths. The identifications will be necessary to ignore palm paths and identify combinations of synchronized fingers, while the proximity markers record the time at which each contact path first exceeds a press proximity threshold and the time at which each contact path drops below a release proximity threshold prior to total liftoff. Setting these proximity thresholds somewhat higher than the minimum proximity considered significant by the segmentation search process **264**, produces more precise finger press and release times.

Step **603** searches for subsets of fingers which touch down at about the same time and for subsets of fingers which lift off at about the same time. This can be done by recording each finger path along with its press time in a temporally ordered list as it crosses the press proximity threshold. Since the primary function of the palms is to support the forearms while the hands are resting, palm activity is ignored by the typing **12** and chord motion recognizers **18** except during differential hand pressure extraction and palm heel presses can be excluded from this list and most other synchronization tests. To check for synchronization between the two most recent finger presses, the press times of the two most recent entries in the list are compared. If the difference between their press times is less than a temporal threshold, the two finger presses are considered synchronized. If not, the most recent finger press is considered asynchronous. Synchronization among three or more fingers up to five is found by comparing press times of the three, four, or five most recent list entries. If the press time of the most recent entry is within a temporal threshold of the nth most recent entry, synchronization among the n most recent finger presses is indicated. To accommodate imprecision in touchdown across the hand, the magnitude of the temporal threshold should increase slightly in proportion to the number of fingers being tested for synchronization. The largest set of recent finger presses found to be synchronized is recorded as the synchronized subset, and the combination of finger identities comprising this subset is stored conveniently as a finger identity bitfield. The term subset is used because the synchronized press subset may not include all fingers currently touching the surface, as happens when a finger touches down much earlier than the other fingers yet remains touching as they simultaneously touch down. An ordered list of finger release times is similarly maintained and searched separately. Alternative embodiments may require that a finger still be touching the surface to be included in the synchronized press subset.

Decision diamond **602** checks whether a synchronization marker is pending from a previous image scan cycle. If not, decision diamond **604** checks whether the search **603** found a newly synchronized press subset in the current proximity image. If so, step **606** sets the temporal synchronization

marker to the oldest press within the new synchronized subset. Additional finger presses may be added to the subset during future scan cycles without affecting the value of this temporal synchronization marker. If there is currently no finger press synchronization, decision diamond **605** determines whether three or more fingers have just been released simultaneously. Simultaneous release of three or more fingers should not occur while typing with a set of fingers but does occur when lifting fingers off the surface from rest. Therefore simultaneous release of three or more fingers reliably indicates that the released fingers are not intended as keypresses and should be deleted from the keypress queue **605**, regardless of whether these same fingers touched down synchronously. Release synchronization of two fingers is not by itself a reliable indicator of typing intent and has no effect on the keypress queue. The keypress queue is described later with FIGS. **42**-**43**B.

Once a press synchronization marker for the hand is pending, further processing checks the number of finger presses which are synchronized and waits for release of the synchronized fingers. If decision diamond **608** finds three or more fingers in the synchronized press subset the user cannot possibly be typing with these fingers. Therefore step **612** immediately deletes the three or more synchronized presses from the keypress queue. This way they cannot cause key symbol transmission to the host, and transmission of key symbols from subsequent asynchronous presses is not blocked waiting for the synchronized fingers to be released.

However, when the synchronization only involves two finger presses **608**, it is difficult to know whether the user intended to tap a finger pair chord or intended to type two adjacent keys and accidentally let the key presses occur simultaneously. Since such accidental simultaneous presses are usually followed by asynchronous releases of the two fingers, but finger pair chords are usually released synchronously, the decision whether the presses are asynchronous key taps or chord taps must be delayed until finger release can be checked for synchronization. In the meantime, step **610** places a hold on the keypress queue to prevent transmission of key symbols from the possible finger chord or any subsequent finger presses. To prevent long backups in key transmission, decision diamond **614** will eventually release the queue hold by having step **615** delete the synchronized presses from the keypress queue if both fingers remain touching a long time. Though this aborts the hypothesis that the presses were intended as key taps, the presses are also less likely to be key taps if the fingers are not lifted soon after touchdown.

If the synchronized fingers are not lifting, decision diamond **616** leaves the synchronization marker pending so synchronization checks can be continued with updated path parameters **600** after the next scan cycle. If the synchronized fingers are lifting, but decision diamond **618** finds with the help of the synchronization release search **603** that they are doing so asynchronously **618**, step **622** releases any holds on the keypress queue assuming any synchronized finger pair was intended to be two keypresses. Though the synchronized finger presses are not deleted from the keypress queue at this point, they may have already been deleted in step **612** if the pressed subset contained more than two. Also, step **624** clears the temporal synchronization marker, indicating that no further synchronization tests need be done for this subset.

Continuing to FIG. **39**B, if the fingers synchronized during touchdown also lift simultaneously, step **618** removes them and any holds from the keypress queue in case they were a pair awaiting a positive release synchronization test. Further tests ensue to determine whether the synchronized fingers meet additional chord tap conditions. As with single finger taps, the synchronized fingers cannot be held on the surface more than about half a second if they are to qualify, as a chord tap. Decision diamond **626** tests this by thresholding the time between the release of the last remaining synchronized finger and the temporal press synchronization marker. A chord tap should also exhibit a limited amount of lateral finger motion, measured either as an average of peak finger speeds or distance traveled since touchdown in decision diamond **628**. If the quick release and limited lateral motion conditions are not met, step **624** clears the synchronization marker with the conclusion that the synchronized fingers were either just resting fingers or part of a chord slide.

If the chord tap conditions are met, step **630** looks up, using the synchronized subset bitfield, any input events such as mouse clicks or keyboard commands assigned to the combination of fingers in the chord tap. Some chords such as those including all four fingertips may be reserved as resting chords **634**, in which case decision diamond **632** will find they have no associated input events. If the chord does have tap input events, step **636** appends these to the main outgoing event queue of the host communication interface **20**. Finally step **624** clears the synchronization marker in readiness for future finger synchronizations on the given hand.

As a further precaution against accidental generation of chord taps while typing, it is also useful for decision diamond **632** to ignore through step **634** the first chord tap which comes soon after a valid keypress without a chord slide in between. Usually after typing the user will need to reposition the mouse cursor before clicking, requiring an intervening chord slide. If the mouse cursor happens to already be in place after typing, the user may have to tap the finger chord a second time for the click to be sent, but this is less risky than having an accidental chord tap cause an unintended mouse button click in the middle of a typing session.

FIG. **40**A shows the detailed steps of the chord motion recognizer module **18**. The chord motion recognition process described below is repeated for each hand independently. Step **650** retrieves the parameters of the hand's identified paths **250** and the hand's extracted motion components from the motion extraction module **16**. If a slide of a finger chord has not already started, decision diamond **652** orders slide initiation tests **654** and **656**. To distinguish slides from glancing finger taps during typing, decision diamond **654** requires at least two fingers from a hand to be touching the surface for slide mode to start. There may be some exceptions to this rule, such as allowing a single finger to resume a previous slide within a second or so after the previous slide chord lifts off the surface.

In a preferred embodiment, the user can start a slide and specify its chord in either of two ways. In the first way, the user starts with the hand floating above the surface, places some fingers on the surface possibly asynchronously, and begins moving all of these fingers laterally. Decision diamond **656** initiates the slide mode only when significant motion is detected in all the touching fingers. Step **658** selects the chord from the combination of fingers touching when significant motion is detected, regardless of touchdown synchronization. In this case coherent initiation of motion in all the touching fingers is sufficient to distinguish the slide from resting fingers, so synchronization of touchdown is not necessary. Also, novice users may erroneously try to start a slide by placing and sliding only one finger on the surface, forgetting that multiple fingers are necessary. Tolerance of asynchronous touchdown allows them to seamlessly correct this by subsequently placing and sliding the rest of the fingers desired for

the chord. The slide chord will then initiate without forcing the user to pick up all fingers and start over with synchronized finger touchdowns.

In the second way, the user starts with multiple fingers resting on the surface, lifts a subset of these fingers, touches a subset back down on the surface synchronously to select the chord, and begins moving the subset laterally to initiate the slide. Decision diamond **656** actually initiates the slide mode when it detects significant motion in all the fingers of the synchronized subset. Whether the fingers which remained resting on the surface during this sequence begin to move does not matter since in this case the selected chord is determined in step **658** by the combination of fingers in the synchronized press subset, not from the set of all touching fingers. This second way has the advantage that the user does not have to lift the whole hand from the surface before starting the slide, but can instead leave most of the weight of the hands resting on the surface and only lift and press the two or three fingers necessary to identify the most common finger chords.

To provide greater tolerance for accidental shifts in resting finger positions, decision diamond **656** requires both that all relevant fingers are moving at significant speed and that they are moving about the same speed. This is checked either by thresholding the geometric mean of the finger speeds or by thresholding the fastest finger's speed and verifying that the slowest finger's speed is at least a minimum fraction of the fastest finger's speed. Once a chord slide is initiated, step **660** disables recognition of key or chord taps by the hand at least until either the touching fingers or the synced subset lifts off.

Once the slide initiates, the chord motion recognizer could simply begin sending raw component velocities paired with the selected combination of finger identities to the host. However, in the interest of backward compatibility with the mouse and key event formats of conventional input devices, the motion event generation steps in FIG. **40**B convert motion in any of the extracted degrees of freedom into standard mouse and key command events which depend on the identity of the selected chord. To support such motion conversion, step **658** finds a chord activity structure in a lookup table using a bitfield of the identities of either the touching fingers or the fingers in the synchronized, subset. Different finger identity combinations can refer to the same chord activity structure. In the preferred embodiment, all finger combinations with the same number of non-thumb fingertips refer to the same chord activity structure, so slide chord activities are distinguished by whether the thumb is touching and how many non-thumb fingers are touching. Basing chord action on the number of fingertips rather than their combination still provides up to seven chords per hand yet makes chords easier for the user to memorize and perform. The user has the freedom to choose and vary which fingertips are used in chords requiring only one; two or three fingertips. Given this freedom, users naturally tend to pick combinations in which all touching fingertips are adjacent rather than combinations in which a finger such as the ring finger is lifted but the surrounding fingers such as the middle and pinky must touch. One chord typing study found that users can tap these finger chords in which all pressed fingertips are adjacent twice as fast as other chords.

The events in each chord activity structure are organized into slices. Each slice contains events to be generated in response to motion in a particular range of speeds and directions within the extracted degrees of freedom. For example, a mouse cursor slice could be allocated any translational speed and direction. However, text cursor manipulation requires four slices, one for each arrow key, and each arrow's slice integrates motion in a narrow direction range of translation. Each slice can also include motion sensitivity and so-called cursor acceleration parameters for each degree of freedom. These will be used to discretize motion into the units such as arrow key clicks or mouse clicks expected by existing host computer systems.

Step **675** of chord motion conversion simply picks the first slice in the given chord activity structure for processing. Step **676** scales the current values of the extracted velocity components by the slice's motion sensitivity and acceleration parameters. Step **677** geometrically projects or clips the scaled velocity components into the slice's defined speed and direction range. For the example mouse cursor slice, this might only involve clipping the rotation and scaling components to zero. But for an arrow key slice, the translation velocity vector is projected onto the unit vector pointing in the same direction as the arrow. Step **678** integrates each scaled and projected component velocity over time in the slice's accumulators until decision diamond **680** determines at least one unit of motion has been accumulated. Step **682** looks up the slice's preferred mouse, key, or three-dimensional input event format, attaches the number of accumulated motion units to the event; and step **684** dispatches the event to the outgoing queue of the host communication interface **20**. Step **686** subtracts the sent motion events from the accumulators, and step **688** optionally clears the accumulators of other slices. If the slice is intended to generate a single key command per hand motion, decision diamond **689** will determine that it is a one-shot slice so that step **690** can disable further event generation from it until a slice with a different direction intervenes. If the given slice is the last slice, decision diamond **692** returns to step **650** to await the next scan of the sensor array. Otherwise step **694** continues to integrate and convert the current motion for other slices.

Returning to FIG. **40**A, for some applications it may be desirable to change the selected chord whenever an additional finger touches down or one of the fingers in the chord lifts off. However, in the preferred embodiment, the selected chord cannot be changed after slide initiation by asynchronous finger touch activity. This gives the user freedom to rest or lift addition fingers as may be necessary to get the best precision in a desired degree of freedom. For example, even though the finger pair chord does not include the thumb, the thumb can be set down shortly after slide initiation to access the full dynamic range of the rotation and scaling degrees of freedom. In fact, all remaining lifted fingers can always be set down after initiation of any chord to allow manipulation by the whole hand. Likewise, all fingers but one can be lifted, yet translation will continue.

Though asynchronous finger touch activity is ignored, synchronized lifting and pressing of multiple fingers subsequent to slide initiation can create a new synchronized subset and change the selected chord. Preferably this is only allowed while the hand has paused but its fingers are still resting on the surface. Decision diamond **670** will detect the new subset and commence motion testing in decision diamond **673** which is analogous to decision diamond **656**. If significant motion is found in all fingers of the newly synchronized subset, step **674** will select the new subset as the slide chord and lookup a new chord activity structure in analogy to step **658**. Thus finger synchronization again allows the user to switch to a different activity without forcing the user to lift the whole hand from the surface. Integration of velocity components resumes but the events generated from the new chord activity structure will presumably be different.

It is advantageous to provide visual or auditory feedback to the user about which chord activity structure has been selected. This can be accomplished visually by placing a row of five light emitting diodes across the top of the multi-touch

surface, with one row per hand to be used on the surface. When entering slide mode, step **658** would turn on a combination of these lights corresponding to the combination of fingers in the selected chord. Step **674** would change the combination of active lights to match the new chord activity structure should the user select a new, chord, and step **668** would turn them off. Similar lights could be emulated on the host computer display **24**. The lights could also be flashed to indicate the finger combination detected during chord taps in step **636**. The implementation for auditory feedback would be similar, except light combinations would be replaced with tone or tone burst combinations.

The accumulation and event generation process repeats for all array scan cycles until decision diamond **664** detects liftoff by all the fingers from the initiating combination. Decision diamond **666** then checks the pre-liftoff deceleration flag of the dominant motion, component. The state of this flag is determined by step **556** or **558** of translation extraction (FIG. **37**) if translation is dominant, or by corresponding flags in step **534** of polar extraction. If there has been significant deceleration, step **668** simply exits the chord slide mode, setting the selected chord to null. If the flag indicates no significant finger deceleration prior to liftoff, decision diamond **666** enables motion continuation mode for the selected chord. While in this mode, step **667** applies the pre-liftoff weighted average (**560**) of dominant component velocity to the motion accumulators (**678**) in place of the current velocities, which are presumably zero since no fingers touch the surface. Motion continuation mode does not stop until any of the remaining fingers not in the synchronized subset are lifted or more fingers newly touch down. This causes decision diamond **664** to become false and normal slide activity with the currently selected chord to resume. Though the cursor or scrolling velocity does not decay during motion continuation mode, the host computer can send a signal instructing motion continuation mode to be canceled if the cursor reaches the edge of the screen or end of a document. Similarly, if any fingers remain on the surface during motion continuation, their translations can adjust the cursor or scrolling velocity.

In the preferred embodiment, the chord motion recognizers for each hand function independently and the input events for each chord can be configured independently. This allows the system to allocate tasks between hands in many different ways and to support a variety of bimanual manipulations. For example, mouse cursor motion can be allocated to the fingertip pair chord on both hands and mouse button drag to a triple fingertip chord on both hands. This way the mouse pointer can be moved and drug with either hand on either half of the surface. Primary mouse clicks would be generated by a tap of a fingertip pair on either half of the surface, and double-clicks could be ergonomically generated by a single tap of three fingertips on the surface. Window scrolling could be allocated to slides of four fingers on either hand.

Alternatively, mouse cursor manipulations could be allocated as discussed above to the right hand and right half of the surface, while corresponding text cursor manipulations are allocated to chords on the left hand. For instance, left fingertip pair movement would generate arrow key commands corresponding to the direction of motion, and three fingertips would generate shift arrow combinations for selection of text.

For host computer systems supporting manipulations in three or more degrees of freedom, a left hand chord could be selected to pan, zoom, and rotate the display background while a corresponding chord in the right hand could translate, resize and rotate a foreground object. These chords would not have to include the thumb since the thumb can touch down anytime after initiating chord motion without changing the selected chord. The user then need add the thumb to the surface when attempting rotation or scaling.

Finger chords which initially include the thumb can be reserved for one-shot command gestures, which only generate input events once for each slide of a chord rather than repeating transmission each time an additional unit of motion is detected. For example, the common editing commands cut, copy and paste can be intuitively allocated to a pinch hand scaling, chord tap, and anti-pinch hand scaling of the thumb and an opposing fingertip.

FIG. **41** shows the steps within the key layout definition and morphing process, which is part of the typing recognition module **12**. Step **700** retrieves at system startup a key layout which has been pre-specified by the user or manufacturer. The key layout consists of a set of key region data structures. Each region has associated with it the symbol or commands which should be sent to the host computer when the region is pressed and coordinates representing the location of the center of the region on the surface. In the preferred embodiment, arrangement of those key regions containing alphanumeric and punctuation symbols roughly corresponds to either the QWERTY or the Dvorak key layouts common on mechanical keyboards.

In some embodiments of the multi-touch surface apparatus it is advantageous to be able to snap or morph the key layout to the resting positions of the hands. This is especially helpful for multi-touch surfaces which are several times larger than the standard keyboard or key layout, such as one covering an entire desk. Fixing the key layout in one small fixed area of such a surface would be inconvenient and discourage use of the whole available surface area. To provide feedback to the user about changes in the position of the key layout, the position of the key symbols in these embodiments of the multi-touch surface would not be printed permanently on the surface. Instead, the position of the key symbols would be reprogrammably displayed on the surface by light emitting polymers, liquid crystal, or other dynamic visual display means embedded in the multi-touch surface apparatus along with the proximity sensor arrays.

Given such an apparatus, step **702** retrieves the current paths from both hands and awaits what will be known as a layout homing gesture. If decision diamond **704** decides with the help of, a hand's synchronization detector that all five of the hand's fingers have just been placed on the surface synchronously, step **706** will attempt to snap the key layout to the hand such that the hand's home row keys lie under the synchronized fingertips, wherever the hand is on the surface. Step **706** retrieves the measured hand offsets from the hand position estimator and translates all key regions which are normally typed by the given hand in proportion to the measured hand offsets. Note the currently measured rather than filtered estimates of offsets can be used because when all five fingers are down there is no danger of finger misidentification corrupting the measured offsets. This procedure assumes that the untranslated locations of the home row keys are the same as the default finger locations for the hand.

Decision diamond **708** checks whether the fingers appear to be in a neutral, partially closed posture, rather closed than outstretched or pinched together. If the posture is close to neutral, step **710** may further offset the keys normally typed by each finger, which for the most part are the keys in the same column of the finger by the measured finger offsets. Temporal filtering of these finger offsets over several layout homing gestures will tend to scale the spacing between columns of keys to the user's hand size. Spacing between rows is scaled down in proportion to the scaling between columns.

With the key layout for the hand's keys morphed to fit the size and current position of the resting hand, step **712** updates

the displayed position of the symbols on the surface, so that the user will see that the key layout has snapped to the position of his hand. From this stage the user can begin to type and the typing recognizer **718** will use the morphed key region locations to decide what key regions are being pressed. The layout will remain morphed this way until either the user performs another homing gesture to move it somewhere else on the surface, or until the user takes both hands off the surface for a while. Decision diamond **714** will eventually time out so that step **716** can reset the layout to its default position in readiness for another user or usage session.

For smaller multi-touch surfaces in which the key layout is permanently printed on the surface, it is advantageous to give the user tactile feedback about the positions of key regions. However, any tactile indicators placed on the surface must be carefully designed so as not to impede smooth sliding across the surface. For example, shallow depressions made in the surface near the center of each key mimicking the shallow depressions common on mechanical keyboard keycaps would cause a vibratory washboard effect as the hand slides across the surface. To minimize such washboard effects, in the preferred embodiment the multi-touch surface provides for the fingertips of each hand a single, continuous depression running from the default index fingertip location to the default pinky fingertip location. This corresponds on the QWERTY key layout to shallow, slightly arched channels along home row from the "J" key to the ";" key for the right hand, and from the "A" key to the "F" key for the left hand. Similarly, the thumbs can each be provided with a single oval-shaped depression at their default locations, slanted slightly from vertical to match the default thumb orientation. These would preferably correspond to "Space" and "BackSpace" key regions for the right and left thumbs, respectively. Such minimal depressions can tactilely guide users' hands back to home row of the key layout without requiring users to look down at the surface and without seriously disrupting finger chord slides and manipulations on the surface.

The positions of key regions off home row can be marked by other types of tactile indicators. Simply roughening the surface at key regions does not work well. Though humans easily differentiate textures when sliding fingers over them, most textures cannot be noticed during quick taps on a textured region. Only relatively abrupt edges or protrusions can be sensed by the users' fingertips under typing conditions. Therefore, a small raised dot like a Braille dot is formed on top of the surface at the center of each key region. The user receives feedback on the accuracy of their typing strokes from where on the fingertip a dot is felt. This feedback can be used to correct finger aim during future keypresses. Since single finger slides are ignored by the chord motion recognizer, the user can also slide a finger around the surface in tactile search of a particular key region's dot and then tap the key region when the dot is found, all without looking at the surface. Each dot should be just: large enough to be felt during tapping but not so large as to impede chord slides across the surface. Even if the dots are not large enough to impede sliding, they can still corrupt proximity and fingertip centroid measurements by raising the fingertip flesh near the dot off the surface thus locally separating the flesh from the underlying proximity sensing electrode. Therefore, in the preferred embodiment, the portion of each dot above the surface dielectric is made of a conductive material. This improves capacitive coupling between the raised fingertip flesh and the underlying electrodes.

FIG. **42** shows the steps within the keypress detection loop. Step **750** retrieves from the current identified path data **250** any paths which were recently created due to hand part touchdown or the surface. Decision diamond **752** checks whether the path proximity reached a keypress proximity thresh for the first time during the current sensor array scan. If the proximity has not reached the threshold yet or has already exceeded it previously, control returns to step **750** to try keypress detection on the next recent path. If the path just crossed the keypress proximity threshold decision diamond **754** checks whether the contact path has been identified as a finger rather than a palm. To give the users the freedom rest the palms anywhere on the surface, palm presses should not normally cause keypresses, and are therefore ignored. Assuming the path is a finger, decision diamond **756** checks whether the hand the identified finger comes from is currently performing a chord slide gesture or writing via the pen grip hand configuration. Asynchronous finger presses are ignored once these activities have started, as also indicated in step **660** of FIG. **40**A. Assuming such hand activities are not ongoing, decision diamond **757** proceeds with debounce tests which check that the finger has touched the surface for at least two sensor array scan cycles and that it had been off the surface for several scan cycles before touching down. The path tracking module (FIG. **22**) facilitates such liftoff debouncing by reactivating in step **334** a finger's old path if the finger lifts off and quickly touches back down over the same spot. Upon reactivation the time stamp of the last liftoff by the old path must be preserved for comparison with the time stamp of the new touchdown.

If all of these tests are passed, step **758** looks up the current path position ($P_x[n],P_y[n]$), and step **760** finds the key region whose reference position is closest to the fingertip centroid. Decision diamond **762** checks that the nearest region is within a reasonable distance of the finger, and if not causes the finger press to be ignored. Assuming a key region is close to the finger, step **764** creates a keypress element data structure containing the path, index identifier and finger identity, the closest key region, and a time stamp indicating when the finger crossed the keypress proximity threshold. Step **766** then appends this element data structure to the tail of a FIFO keypress queue. This accomplished, processing returns to step **750** to process or wait for touchdowns by other fingers.

The keypress queue effectively orders finger touchdowns by when they pass the keypress transmitted to the host. However, an element's key symbol is not assured transmission of the host once in the keypress queue. Any of a number of conditions such as being part of a synchronized subset of pressing fingers can cause it to be deleted from the queue before being transmitted to the host. In this sense the keypress queue should be considered a keypress candidate queue. Unlike the ordered lists of finger touchdowns and releases maintained for each hand separately in the synchronization detector, the keypress queue includes and orders the finger touchdowns from both hands.

FIG. **43**A shows the steps within the keypress acceptance and transmission loop. Step **770** picks the element at the head of the keypress queue, which represents the oldest finger touchdown which has neither been deleted from the queue as an invalid keypress candidate nor transmitted its associated key symbol. Decision diamond **772** checks whether the path is still identified as a finger. While waiting in the queue path proximity could have increased so much that the identification system decides the path is actually from a palm heel, in which case step **778** deletes the keypress element without transmitting to the host and step **770** advances processing to the next element. Decision diamond **774** also invalidates the element if its press happened synchronously with other fingers of the same hand. Thus decision diamond **774** follows through on deletion command steps **601**, **612**, **615**, **620** of the

synchronization detection process (FIG. **39**). Decision diamond **776** invalidates the keypress if too much lateral finger motion has occurred since touchdown, even if that lateral finger motion has not yet caused a chord slide to start. Because users may be touch typing on the surface, several millimeters of lateral motion are allowed to accommodate glancing fingertip motions which often occur when quickly reaching for keys. This is much more glancing tap motion than is tolerated by touchpads which employ a single finger slide for mouse cursor manipulation and a single finger tap for key or mouse button click emulation.

Decision diamond **780** checks whether the finger whose touchdown created the keypress element has since lifted off the surface. If so, decision diamond **782** checks whether it was lifted off soon enough to qualify as a normal key tap. If so, step **784** transmits the associated key symbol to the host and step **778** deletes it from the head of the queue. Note that a keypress is always deleted from the queue upon liftoff, but even though it may have stayed on the surface for a time exceeding the tap timeout, it may have still caused transmission as a modifier key, as an impulsive press with hand resting, or as a typematic press, as described below.

When a keypress is transmitted to the host it is advantageous for a sound generation device on the multi-touch surface apparatus or host computer to emit an audible click or beep as feedback to the user. Generation of audible click and beep feedback in response to keypresses is well known in commercial touchscreens, kiosks, appliance control panels and mechanical keyboards in which the keyswitch action is nearly silent and does not have a make force threshold which feels distinctive to the user. Feedback can also be provided as a light on the multi-touch surface apparatus which flashes each time a keypress is sent. Keypresses accompanied by modifier keypresses should cause longer flashes or tones to acknowledge that the key symbol includes modifiers.

If the finger has not yet lifted, decision diamond **786** checks whether its associated key region is a modifier such as <shift>, <ctrl>, or <alt>. If so, step **788** advances to the next element in the queue without deleting the head. Processing will continue at step **772** to see if the next element is a valid key tap. If the next element successfully reaches the transmission stage, step **784** will scan back toward the head of the queue for any modifier regions which are still pressed. Then step **784** can send the next element's key symbol along with the modifying symbols of any preceding modifier regions.

Decision diamond **782** requires that users touch the finger on the surface and lift back off within a few hundred milliseconds for a key to be sent. This liftoff timing requirement substitutes for the force activation threshold of mechanical keyswitches. Like the force threshold of mechanical keyswitches, the timing constraint provides a way for the user to rest the finger on the key surface without invoking a keypress. The synchronization detector **14** provides another way forefingers to rest on the surface without generating key symbols: they must touch down at the same time as at least one other finger. However, sometimes users will start resting by simultaneously placing the central fingertips on the surface, but then they follow asynchronously with the pinky a second later and the thumb a second after that. These latter presses are essentially asynchronous and will not be invalidated by the synchronization detector, but as long as they are not lifted within a couple hundred milliseconds, decision diamond **782** will delete them without transmission. But, while decision diamond **782** provides tolerance of asynchronous finger resting, its requirement that fingers quickly lift off, i.e., crisply tap, the surface to cause key generation makes it very difficult to keep most of the fingers resting on the surface to support the hands while tapping long sequences of symbols. This causes users to raise their hands off the surface and float them above the surface during fast typing sequences. This is acceptable typing posture except that the users arms will eventually tire if the user fails to rest the hands back on the surface between sequences.

To provide an alternative typing posture which does not encourage suspension of the hands above the surface, decision diamond **790** enables a second key acceptance mode which does not require quick finger liftoff after each press. Instead, the user must start with all five fingers of a hand resting on the surface. Then each time a finger is asynchronously raised off the surface and pressed on a key region, that key region will be transmitted regardless of subsequent liftoff timing. If the surface is hard such that fingertip proximity quickly saturates as force is applied, decision diamond **792** checks the impulsivity of the proximity profile for how quickly the finger proximity peaks. If the proximity profile increases to its peak very slowly over time, no key will be generated. This allows the user to gently set down a raised finger without generating a key in case the user lifts the finger with the intention of generating a key but then changes his mind. If the touch surface is compressible, decision diamond **792** can more directly infer finger force from the ratio of measured fingertip proximity to ellipse axis lengths. Then it can threshold the inferred force to distinguish deliberate key presses from gentle finger rests. Since when intending to generate a key the user will normally press down on the new key region quickly after lifting off the old key region, the impulsivity and force thresholds should increase with the time since the finger lifted off the surface.

Emulating typematic on a multi-touch surface presents special problems if finger resting force cannot be distinguished reliably from sustained holding force on a key region. In this case, the special touch timing sequence detected by the steps of FIG. **43**B supports reliable typematic emulation. Assuming decision diamond **798** finds that typematic has not started yet, decision diamond **794** checks whether the keypress queue element being processed represents the most recent finger touchdown on the surface. If any finger touchdowns have followed the touchdown represented by this element, typematic can never start from this queue element. Instead, decision diamond **796** checks whether the element's finger has been touching longer than the normal tap timeout. If the finger has been touching too long, step **778** should delete its keypress element because decision diamond **786** has determined it is not a modifier and decision diamond **794** has determined it can never start typematic. If decision diamond **794** determines that the keypress element does not represent the most recent touchdown, yet decision diamond **796** indicates the element has not exceeded the tap timeout, processing returns to step **770** to await either liftoff or timeout in a future sensor array scan. This allows finger taps to overlap in the sense that a new key region can be pressed by a finger before another finger lifts off the previous key region. However, either the press times or release times of such a pair of overlapping finger taps must be asynchronous to prevent the pair from being considered a chord tap.

Assuming the finger touchdown is the most recent, decision diamond **800** checks whether the finger has been touching for a typematic hold setup interval of between about half a second and a second. If not, processing returns to **770** to await either finger liftoff or the hold setup condition to be met during future scans of the sensor array. When the hold setup condition is met, decision diamond **802** checks whether all other fingers on the hand of the given finger keypress lifted off the surface more than a half second ago. If they did, step **804**

will initialize typematic for the given keypress element. The combination of decision diamonds **800** and **802** allow the user to have other fingers of the hand to be resting on the surface when a finger intended for typematic touches down. But typematic will not start unless the other fingers lift off the surface within half a second of the desired typematic finger's touchdown, and typematic will also not start until the typematic finger has a continued to touch the surface for at least half a second after the others lifted off the surface. If these stringent conditions are not met, the keypress element will not start typematic and will eventually be deleted through either tap timeout **782** when the finger lifts off or through tap timeout **796**) if another touches down after it.

Step **804** simply sets a flag which will indicate to decision diamond **798** during future scan cycles that typematic has already started for the element. Upon typematic initialization, step **810** sends out the key symbol for the first time to the host interface communication queue, along with any modifier symbols being held down by the opposite hand. Step **812** records the time the key symbol is sent for future reference by decision diamond **808**. Processing then returns to step **770** to await the next proximity image scan.

Until the finger lifts off or another taps asynchronously, processing will pass through decision diamond **798** to check whether the key symbol should be sent again. Step **806** computes the symbol repeat interval dynamically to be inversely proportional to finger proximity. Thus the key will repeat faster as the finger is pressed on the surface harder or a larger part of the fingertip touches the surface. This also reduces the chance that the user will cause more repeats than intended since as finger proximity begins to drop during liftoff the repeat interval becomes much longer. Decision diamond **808** checks whether the dynamic repeat interval since the last typematic symbol send has elapsed, and if necessary sends the symbol again in **810** and updates the typematic send time stamp **812**.

It is desirable to let the users rest the other fingers back onto the surface after typematic has initiated **804** and while typematic continues, but the user must do so without tapping. Decision diamond **805** causes typematic to be canceled and the typematic element deleted **778** if the user asynchronously taps another finger on the surface as if trying to hit another key. If this does not occur, decision diamond **182** will eventually cause deletion of the typematic element when its finger lifts off.

The typing recognition process described above thus allows the multi-touch surface to ergonomically emulate both the typing and hand resting capabilities of a standard mechanical keyboard. Crisp taps or impulsive presses on the surface generate key symbols as soon as the finger is released or decision diamond **792** verifies the impulse has peaked, ensuring prompt feedback to the user. Fingers intended to rest on the surface generate no keys as long as they are members of a synchronized finger press or release subset or are placed on the surface gently and remain there along with other fingers for a second or two. Once resting, fingers can be lifted and tapped or impulsively pressed on the surface to generate key symbols without having to lift other resting fingers. Typematic is initiated ether by impulsively pressing and maintaining distinguishable force on a key, or by holding a finger on a key while other fingers on the hand are lifted. Glancing motions of single fingers as they tap key regions are easily tolerated since most cursor manipulation must be initiated by synchronized slides of two or more fingers.

Other embodiments of the invention will be apparent to those skilled in the art from consideration of the specification and practice of the invention disclosed herein. It is intended that the specification and examples be considered as exemplary only, with a true scope and spirit of the invention being indicated by the following claims.

What is claimed is:

1. A method of processing input from a touch-sensitive surface, the method comprising:

receiving at least one proximity image representing a scan of a plurality of electrodes of the touch-sensitive surface;

segmenting each proximity image into one or more pixel groups that indicate significant proximity, each pixel group representing proximity of a distinguishable hand part or other touch object on or near the touch-sensitive surface; and

mathematically fitting an ellipse to at least one of the pixel groups.

2. The method of claim **1** further comprising transmitting one or more ellipse parameters as a control signal to an electronic or electromechanical device.

3. The method of claim **2** wherein the one or more ellipse parameters is selected from the group consisting of position, shape, size, orientation, eccentricity, major radius, minor radius, and any combination thereof.

4. The method of claim **3** wherein the one or more ellipse parameters are used to distinguish a pixel group associated with a fingertip from a pixel group associated with a thumb.

5. The method of claim **1** wherein fitting an ellipse to a group of pixels comprises computing one or more eigenvalues and one or more eigenvectors of a covariance matrix associated with the pixel group.

6. The method of claim **1** further comprising: tracking a path of at least one of the one or more pixel groups through a time-sequenced series of proximity images;

fitting an ellipse to the at least one of the one or more pixel groups in each of the time-sequenced series of proximity images; and

tracking a change in one or more ellipse parameters through the time-sequenced series of proximity images.

7. The method of claim **6** further comprising transmitting the change in the one or more ellipse parameters as a control signal to an electronic or electromechanical device.

8. The method of claim **7** wherein the change in the one or more ellipse parameters is selected from the group consisting of position, shape, size, orientation, eccentricity, major radius, minor radius, and any combination thereof.

9. The method of claim **6** wherein fitting an ellipse to the one pixel group comprises computing one or more eigenvalues and one or more eigenvectors of a covariance matrix associated with the pixel group.

10. A touch-sensing device comprising:

a substrate;

a plurality of touch-sensing electrodes arranged on the substrate;

electronic scanning hardware adapted to read the plurality of touch-sensing electrodes;

a calibration module operatively coupled to the electronic scanning hardware and adapted to construct a proximity image having a plurality of pixels corresponding to the touch-sensing electrodes; and

a contact tracking and identification module adapted to:

segment the proximity image into one or more pixel groups, each pixel group representing proximity of a distinguishable hand part or other touch object on or near the touch-sensitive surface;

and

mathematically fit an ellipse to at least one of the one or more pixel groups.

11. The touch-sensing device of claim **10** further comprising a host communication interface adapted to transmit one or more ellipse parameters as a control signal to an electronic or electromechanical device.

12. The touch-sensing device of claim **11** wherein the touch-sensing device is integral with the electronic or electromechanical device.

13. The touch-sensing device of claim **11** wherein the one or more ellipse parameters comprise one or more parameters selected from the group consisting of position, shape, size, orientation, eccentricity, major radius, minor radius, and any combination thereof.

14. The method of claim **13** wherein the one or more ellipse parameters are used to distinguish a pixel group associated with a fingertip from a pixel group associated with a thumb.

15. The touch-sensing device of claim **10** wherein the contact tracking and identification module is adapted to compute one or more eigenvalues and one or more eigenvectors to fit the ellipse.

16. The touch-sensing device of claim **10** wherein the contact tracking and identification module is further adapted to:

track a path of one or more pixel groups through a plurality of time-sequenced proximity images;

fit an ellipse to at least one of the one or more pixel groups in a first proximity image of the plurality of time-sequenced proximity images; and

track a change in one or more ellipse parameters associated with the fitted ellipse through two or more of the time-sequenced proximity images.

17. The touch-sensing device of claim **16** further comprising a host communication interface adapted to transmit the change in at least one of the one or more ellipse parameters as a control signal to an electronic or electromechanical device.

18. The touch-sensing device of claim **17** wherein the touch-sensing device is integral with the electronic or electromechanical device.

19. The touch-sensing device of claim **17** wherein the change in one or more ellipse parameters used as a control input to an electronic or electromechanical device comprises one or more parameters selected from the group consisting of position, shape, size, orientation, eccentricity, major radius, minor radius, and any combination thereof.

20. The touch-sensing device of claim **16** wherein the contact tracking and identification module is adapted to compute one or more eigenvalues and one or more eigenvectors to fit the ellipse.

21. The touch-sensing device of any one of claims **10-12** and **16-18** wherein the touch-sensing device is fabricated on or integrated with a display device.

22. The touch-sensing device of claim **21**, wherein the display device comprises a liquid crystal display (LCD) or a light-emitting polymer display (LPD).

23. A computer-readable medium having embodied thereon instructions executable by a machine to perform a method according to any of claims **1-9**.

24. A touch-sensing device comprising:

means for producing a proximity image representing a scan of a plurality of electrodes of a touch-sensitive surface, the proximity image having a plurality of pixels corresponding to the touch-sensing electrodes; and

means for segmenting the proximity image into one or more pixel groups, each pixel group representing a touch object on or near the touch-sensitive surface; and

means for fitting an ellipse to at least one of the pixel groups.

25. The touch-sensing device of claim **24** wherein the touch object comprises at least a portion of a hand.

26. The touch-sensing device of claim **24** wherein the touch object comprises at least a portion of one or more fingers.

27. The touch-sensing device of claim **24** wherein the touch object comprises at least a portion of a body part.

28. The touch-sensing device of claim **27** wherein the body part comprises one or more of a hand, a finger, an ear, or a cheek.

29. The touch-sensing device of claim **24** further comprising means for transmitting one or more ellipse parameters as a control signal to an electronic or electromechanical device.

30. The touch-sensing device of claim **27** wherein the touch-sensing device is integral with the electronic or electromechanical device.

31. The touch-sensing device of claim **24** further comprising:

means for tracking a path of one or more pixel groups through a plurality of time-sequenced proximity images;

means for fitting an ellipse to at least one of the pixel groups in a plurality successive proximity images; and

means for tracking a change in one or more ellipse parameters through a plurality of time-sequenced proximity images.

32. The touch-sensing device of claim **29** further comprising means for transmitting the change in the one or more ellipse parameters as a control signal to an electronic or electromechanical device.

33. The touch-sensing device of claim **32** wherein the touch-sensing device is integral with the electronic or electromechanical device.

34. The touch-sensing device of any one of claims **24** and **29-33** wherein the touch-sensing device is fabricated on or integrated with a display device.

35. The touch-sensing device of claim **34**, wherein the display device comprises a liquid crystal display (LCD) or a light-emitting polymer display (LPD).

* * * * *

Exhibit 2

US007669134B1

(12) **United States Patent**
Christie et al.

(10) **Patent No.: US 7,669,134 B1**
(45) **Date of Patent: Feb. 23, 2010**

(54) **METHOD AND APPARATUS FOR DISPLAYING INFORMATION DURING AN INSTANT MESSAGING SESSION**

(75) Inventors: **Gregory N. Christie**, San Jose, CA (US); **Peter T. Westen**, Menlo Park, CA (US); **Stephen O. Lemay**, San Francisco, CA (US); **Jens Alfke**, San Jose, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 686 days.

(21) Appl. No.: **10/428,523**

(22) Filed: **May 2, 2003**

(51) **Int. Cl.**
G06F 3/00 (2006.01)
G06F 9/00 (2006.01)
G06F 17/00 (2006.01)

(52) **U.S. Cl.** **715/758**; 715/762; 715/751; 715/753

(58) **Field of Classification Search** 715/715, 715/501.1, 707, 706, 751, 762, 763, 764, 715/760; 709/204
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,107,784	A	8/1978	Bemmelen	364/900
4,433,377	A	2/1984	Eustis et al.	364/200
4,485,439	A	11/1984	Rothstein	364/200
4,513,379	A	4/1985	Wilson et al.	364/474
4,586,158	A	4/1986	Brandle	364/900
4,642,790	A	2/1987	Minshull et al.	364/900
5,287,448	A	2/1994	Nicol et al.	395/159
5,428,731	A	6/1995	Powers, III	395/154
4,555,775	A	12/1995	Pike	395/158
5,617,539	A	4/1997	Ludwig et al.	395/200.02
5,758,079	A	5/1998	Ludwig et al.	395/200.34
5,854,893	A	12/1998	Ludwig et al.	395/200.34
5,884,039	A	3/1999	Ludwig et al.	395/200.57
5,896,500	A	4/1999	Ludwig et al.	395/200.34
6,177,931	B1 *	1/2001	Alexander et al.	725/52
6,237,025	B1	5/2001	Ludwig et al.	709/204
6,351,762	B1	2/2002	Ludwig et al.	709/204
6,434,604	B1 *	8/2002	Harada et al.	709/207
6,437,818	B1	8/2002	Ludwig et al.	348/14.09

(Continued)

OTHER PUBLICATIONS

Meyrowitz, Norman; Moser, Margaret; Bruwin: *An Adaptable Design Strategy for Window Manager/Virtual Terminal Systems; Department of Computer Science, Brown University* (1981); pp. 180-189.

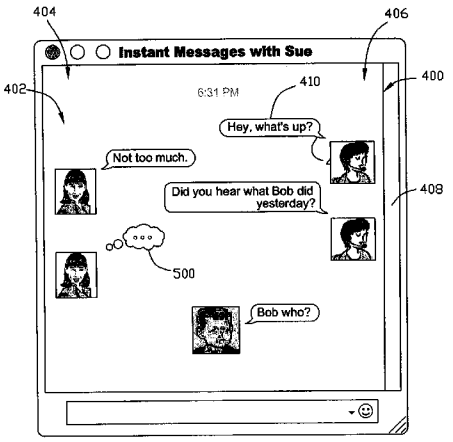
(Continued)

Primary Examiner—Tadesse Hailu
Assistant Examiner—Anita D Chaudhuri
(74) *Attorney, Agent, or Firm*—Williams, Morgan & Amerson, P.C.

(57) **ABSTRACT**

A method and an apparatus are provided for controlling a graphical user interface to display information related to a communication session. Information relating to data produced by a first participant to the communication session is displayed on a first display unit, wherein the information produced by the first participant is displayed at a first position on the first display unit. Data is received from a second participant to the communication session, and information relating to the data received from the second participant is displayed on the first display unit, wherein the information received from the second participant is displayed at a second position on the first display unit. The first and second positions are horizontally spaced apart.

28 Claims, 8 Drawing Sheets



U.S. PATENT DOCUMENTS

6,583,806	B2	6/2003	Ludwig et al.	
6,594,673	B1 *	7/2003	Smith et al.	707/104.1
6,594,688	B2	7/2003	Ludwig et al.	709/204
6,910,186	B2 *	6/2005	Kim	715/706
7,124,164	B1 *	10/2006	Chemtob	709/204
7,133,900	B1 *	11/2006	Szeto	709/206
7,152,093	B2	12/2006	Ludwig et al.	
7,206,809	B2	4/2007	Ludwig et al.	
7,412,470	B2	8/2008	Masuno et al.	
7,433,921	B2	10/2008	Ludwig et al.	
2003/0097408	A1 *	5/2003	Kageyama et al.	709/205
2003/0134678	A1 *	7/2003	Tanaka	463/42
2003/0187925	A1 *	10/2003	Inala et al.	709/204
2003/0208756	A1 *	11/2003	Macrae et al.	725/34
2003/0228909	A1 *	12/2003	Tanaka et al.	463/42
2004/0199663	A1	10/2004	Horvitz et al.	
2004/0223485	A1	11/2004	Arellano et al.	
2006/0026233	A1 *	2/2006	Tenembaum et al.	709/205

OTHER PUBLICATIONS

Scheifler, R.W.; *The X Window System; MIT Laboratory for Computer Science and Gettys, Jim Digital Equipment Corporation and MIT Project Athena; ACM Transactions on Graphics*, vol. 5, No. 2, Apr. 1986, pp. 79-109.

Wadlow, M.G.; *The Role of Human Interface Guidelines in the Design of Multimedia Applications, Carnegie Mellon University* (*To be Published in Current Psychology: Research and Reviews, Summer 1990*) pp. 1-22 (CMU-ITC-91-101).

Morris, James H. et al.; *Andrew: A Distributed Personal Computing Environment; Communications of the ACM*,(Mar. 1986); vol. 29 No. 3, pp. 184-201.

Howard, John H.; (*Abstract*) *An Overview of the Andrew File System; Information Technology Center, Carnegie Mellon University*; (CMU-ITC-88-062); pp. 1-6 (CMU-ITC-88-062) (To Appear in a future issue of the ACM Transactions on Computer Systems.

Borenstein, Nathaniel S.; Information Technology Center and Computer Science Department, Carnegie Mellon University; Thyberg, Chris A. Academic Computing, Carnegie Mellon University: *Cooperative Work in the Andrew Message System*; (1988) pp. 306-323.

Trowbridge, David: Center for Design of Educational Computing, Carnegie-Mellon University; *Using Andrew for Development of Educational Applications*; pp. 1-6 (CMU-ITC-85-065).

Palay, Andrew J., et al.; Information Technology Center, Carnegie-Mellon University; *The Andrew Toolkit: An Overview*; pp. 1-15.

Raper, Larry K., *The C-MU PC Server Project* (Dec. 1986) pp. 1-30 (CMU-ITC-86-051).

Gruhn, A.M. and Hohl, A.C.; *A Research Perspective on Computer-Assisted Office Work*; (1979) IBM Syst. J. vol. 18, No. 3, pp. 432-456.

Bleher, J.H., et al.; *A Graphic Interactive Application Monitor; IBM Syst. J.*, vol. 19, No. 3 (1980) pp. 382-402.

Gardner, P.C.; *A System for the Automated Office Environment; IBM Syst. J.*, vol. 20, No. 3, (1981) pp. 321-345.

Betts, B., et al.; *Goals and Objectives for User Interface Software; Computer Graphics*, vol. 21, No. 2, (Apr. 1987) pp. 73-78.

Conklin, Jeffrey; *A Survey of Hypertext*; MCC Software Technology Program (Dec. 1987); pp. 1-38.

Halbert, D.C.; *Programming by Example; Dept. Electrical Engineering and Comp. Sciences, University of California, Berkley*, (Nov. 1984) pp. 1-76.

Walker, B., et al.; *The LOCUS Distributed Operating System*[1] *; University of California Los Angeles* (1983) pp. 49-70.

Hill, R.D.; *Some Important Features and Issues in User Interface Management System; Dynamic Graphics Project*, University of Toronto, CSRI, vol. 21, No. 2 (Apr. 1987); pp. 116-120.

Croft, W. Bruce and Lefkowitz, Lawrence S.; *Task Support In an Office System; Computer and Information Science Department, University of Massachusetts*, (1984) pp. 22-24.

Fishner, S.S., et al.; *Virtual Environment Display System; Interactive 3D Graphics* (Oct. 23-24, 1986) pp. 77-87.

Ziegler, K., Jr.; *A Distributed Information System Study; IBM Syst. J.*, vol. 18, No. 3 (1979).

Meyer, Mike; *A Shell for Modern Personal Computers; University of California*; pp. 13-19.

Trigg, R.H. and Irish, P.M.; *Hypertext Habitats: Experiences of Writers in NoteCards; Hypertext '87 Papers*; Intelligent Systems Laboratory, Xerox Palo Alto Research Center; pp. 89-108.

Donahue, James; Widom, Jennifer; *Whiteboards: A Graphical Database Tool*; ACM Transactions on Office Information Systems, vol. 4, No. 1, Jan. 1986, pp. 24-41.

Rosenberg, J. et al.; *An Overview of the Andrew Message System; Information Technology Center Carnegie-Mellon University*; (Jul. 1987); pp. 99-108.

Morland, D. Verne; *Human Factors Guidelines for Terminal Interface Design; Communications of the ACM* (Jul. 1983) vol. 26, No. 7, pp. 484-494.

Good, Michael D., et al.; *Building a User-Derived Interface; Communications of the ACM*; (Oct. 1984) vol. 27, No. 10; pp. 1032-1043.

Smith, Sidney L. and Mosier, Jane D.; *Guidelines for Designing User Interface Software; UserLab, Inc.* (Aug. 1986); pp. 1-384.

"A Comparison of Application Sharing Mechanisms in Real-Time Desktop Conferencing Systems," by S. R. Ahuja, et al; AT&T Bell Laboratories; 1990.

"A Distributed Architecture for Multimedia Conference Control," by Eve Schooler, ISI Research Report (Nov. 1991).

"A Packet-Switched Multimedia Conferencing System," by Eve Schooler, et al; ACM Sigois Bulletin, vol. 1, No. 1, pp. 12-22 (Jan. 1989).

"An Architecture for Multimedia Connection Management," by Eve Schooler, et al; Proceedings IEE 4th COMSOC International Workshop on Multimedia Communications; MM '92, pp. 271-274 (Apr. 1992).

"Case Study: Multimedia Conference Control in a Packet-Switched Teleconferencing System," Journal of Internetworking: Research and Experience; vol. 4, pp. 99-120 (Jun. 1993).

"CERC Technical Report Series Research Note," by Kankanahalli Srinivas, et al; Concurrent Engineering Research Center; Feb. 1992.

"Collaboration Awareness in Support of Collaboration Transparency: Requirements for the Next Generation of Shared Window Systems," by J. Chris Lauwers, et al; 1990 CHI '90 Proceedings.

"CSCW '92 Demonstrations," by Michael Muller, et al; Nov. 1992.

"Design of a Multi-Media Vehicle for Social Browsing," by Robert Root; Bell Communications Research; 1988.

"Distributed Multiparty Desktop Conferencing System: Mermaid," by Kazuo Watabe, et al CSCW 90 Proceedings; Oct. 1990.

"Europarc's Integrated Interactive Intermedia Facility (IIIF): Early Experiences," by Bull Buxton, et al; Elsevier Science Publishers B.V. (North-Holland); 11-34.

"Explorations into Real-Time Multimedia Conferencing," by Harry Forsdick; Computer Message System—85; 1986.

"Kernal Support for Live Digital Audio and Video," by Kevin Jeffay, et al; In Proc. of the Second International Workshop on Network and Operating System Support for Digital Audio and Video, Nov. 1991.

"MM Conf: An Infrastructure for Building Shared Multimedia Applications," by Terrence Crowley, et al; CSCW 90 Proceedings (Oct. 1990).

"Multimedia—Principles, Systems and Applications," by Lars Kjelldahl, et al; Apr. 1991; Proceedings of the 1991 Eurographics Workshop on Multimedia Systems, Applications, and Interaction; Stockholm, Sweden.

"Multimedia Conferencing: Has it Come of Age?," by Eve Schooler, et al; Proceedings 24th Hawaii International Conference on System Sciences, vol. 2, pp. 707-716 (Jan. 1991).

"N-Way Conferencing With Packet Video," by Stephen Casner, et al; The Third International Workshop on Packet Video, Mar. 22-23, 1990.

"One is Not Enough: Multiple Views in a Media Space," by William Gaver, et al; Interchi; Apr. 24-29, 1993.

"Piloting of Multimedia Integrated Communications for European Researchers," by P.T. Kirstein, et al; Proc. INET '93.

"Portholes: Supporting Awareness in a Distributed Work Group," by Paul Dourish, et al; CHI 1992; May 3-8.

"Realizing a Video Environment: Europarc's Rave System," by William Gaver, et al; Ranx Xerox Cambridge Europarc; 1992.
"Rendevous: An Architecture for Synchronous Multi-User Applications," by John Patterson, et al; CSCW Proceedings; Oct. 1990.
"Replicated Architectures for Shared Window Systems: A Critique," by J. Chris Lauwers, et al; Olivetti Research California 1990.
"System Support for Integrated Desktop Video Conferencing," by Amy Pearl; Sunmicrosystems Laboratories; Dec. 1992.
"Team Work Station: Towards a Seamless Shared Workspace," by Hiroshi; NTT Human Interface Laboratories; CSCW 90 Proceedings; Oct. 1990.
"The Connection Contorl Protocol: Architecture Overview," by Eve Schooler, et al; USC/Inormation Sciences Institute (Jan. 28, 1992).
"The Impact of Scaling on a Multimedia Connection Architecture," by Eve Schooler; ACM Journal of Multimedia Systems; vol. 1; No. 1; pp. 2-9 (1993).
"The Inria Videoconferencing System (IVS)," by Thierry Turletti; October 1994.
"The Multi Media Workstation," by Dick Phillips; Siggraph '89 Panel Proceedings.
"Touring Machine: A Software Platform for Distributed Multimedia Applications," by M. Arango, et al; 1992 IFIP International Conference on Upper Layer Protocols, Architectures and Applications, May 1992.
"Towards a Universal Directory Service," by Keith Lantz, et al; Department of Computer Science and Electrical Engineering; Stanford University, 1985.
"Videoconferencing as a Technology to Support Group Work: A Review of Its Failure," by Carmen Egido; Bell Communications Research; 1988.

* cited by examiner

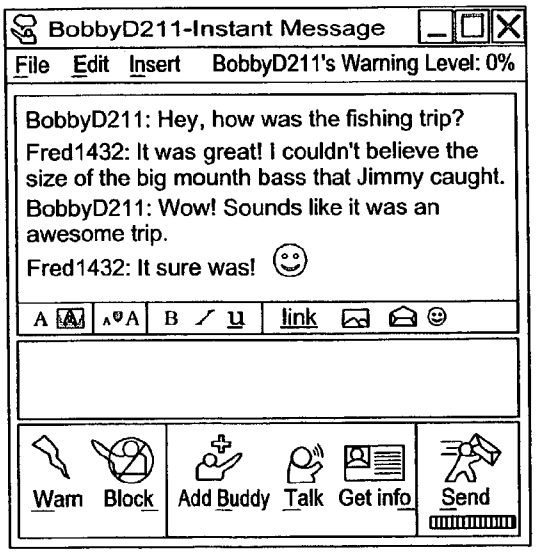


FIGURE 1
(prior art)

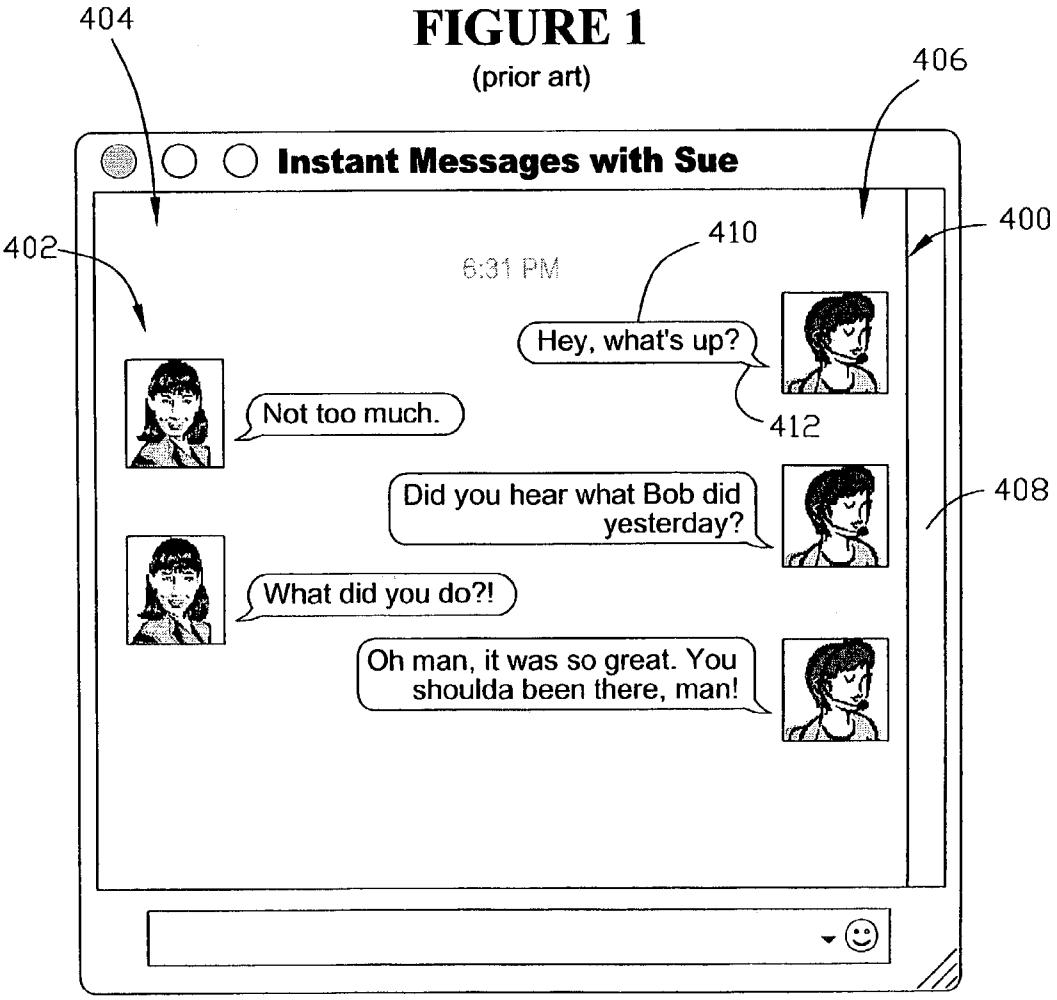


FIGURE 4

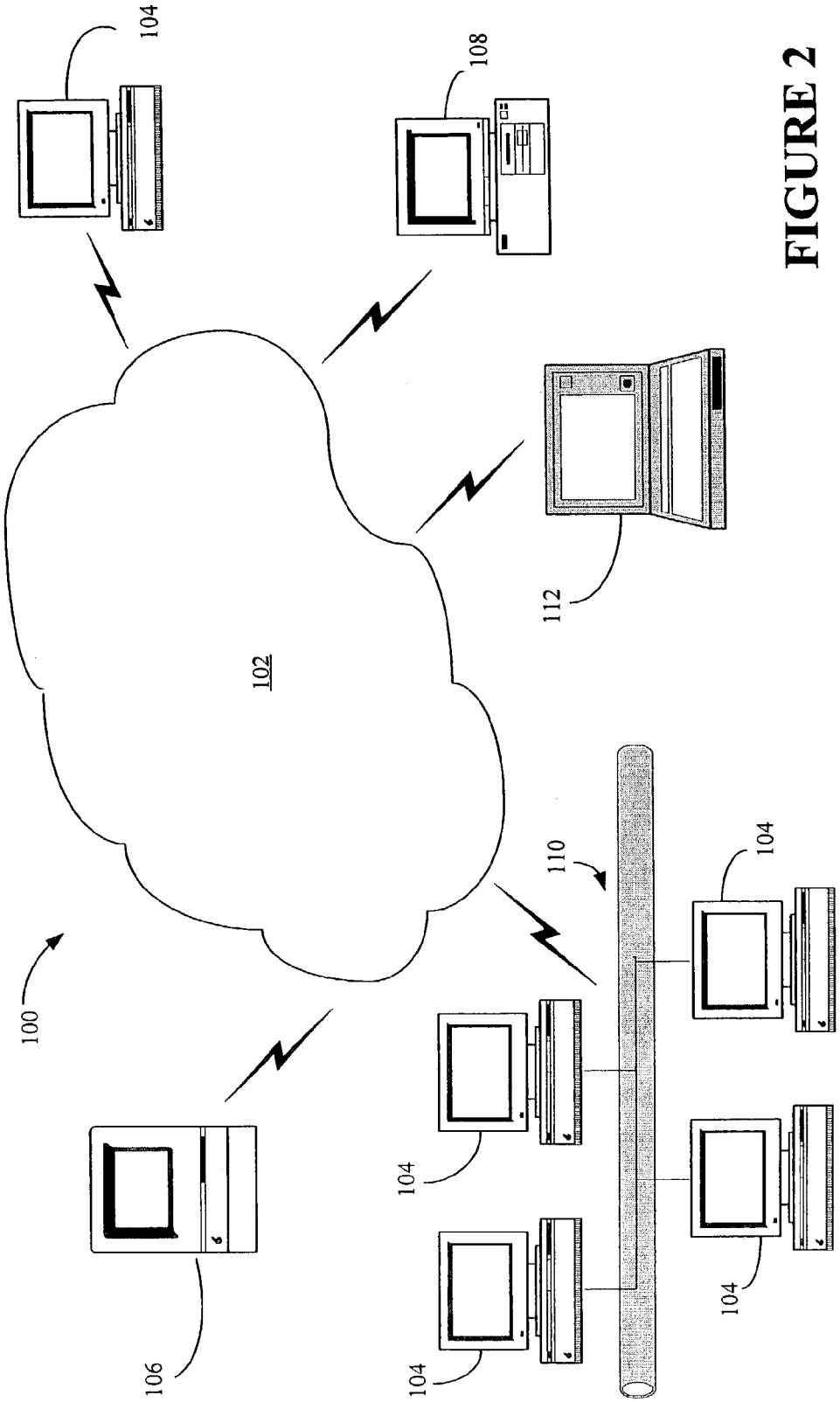


FIGURE 2

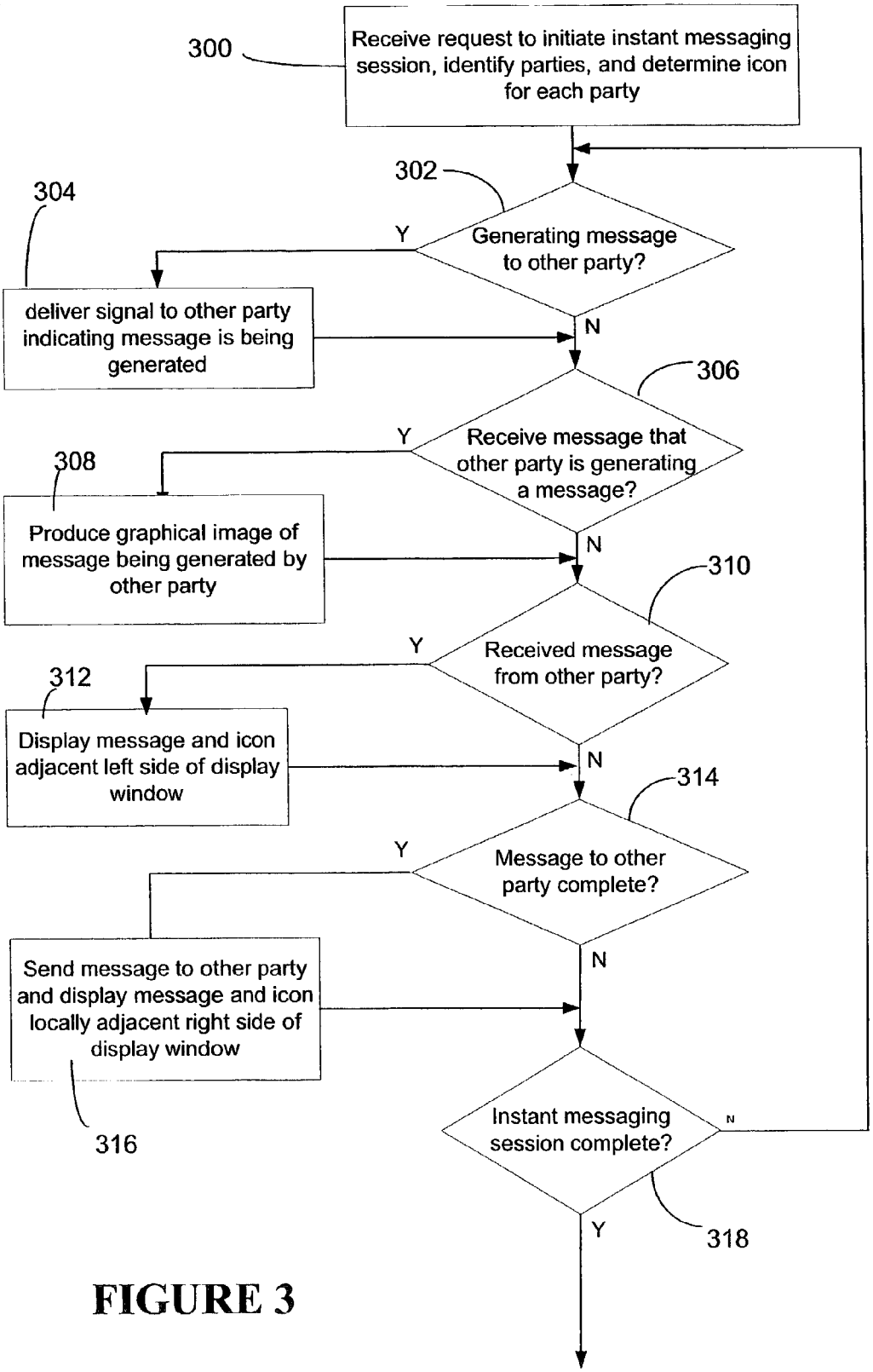

300
Receive request to initiate instant messaging session, identify parties, and determine icon for each party
302
Generating message to other party?
Y
N
304
deliver signal to other party indicating message is being generated
306
Receive message that other party is generating a message?
Y
N
308
Produce graphical image of message being generated by other party
310
Received message from other party?
Y
N
312
Display message and icon adjacent left side of display window
314
Message to other party complete?
Y
N
Send message to other party and display message and icon locally adjacent right side of display window
316
Instant messaging session complete?
N
Y
318

FIGURE 3

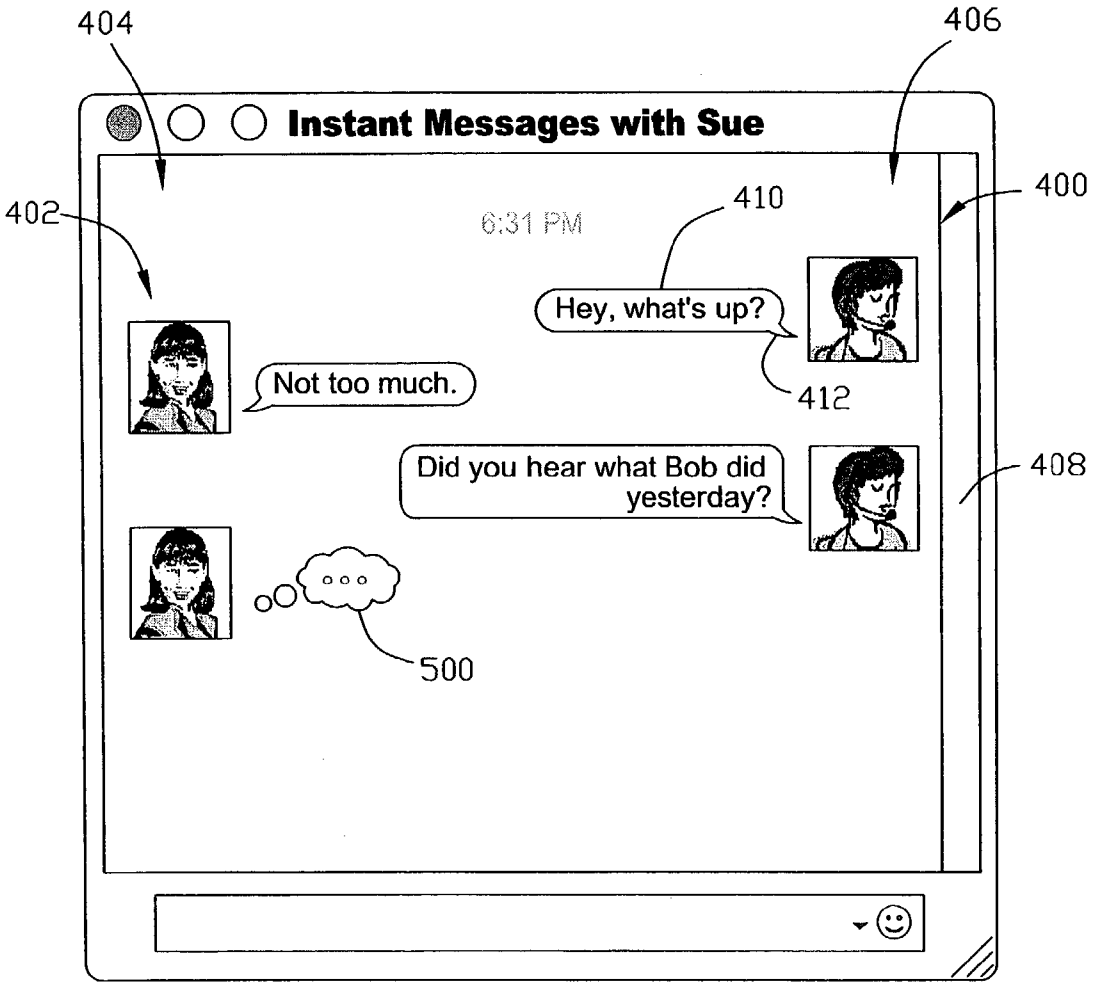

404
406
Instant Messages with Sue
400
402
410
6:31 PM
Hey, what's up?
Not too much.
412
Did you hear what Bob did yesterday?
408
500

FIGURE 5A

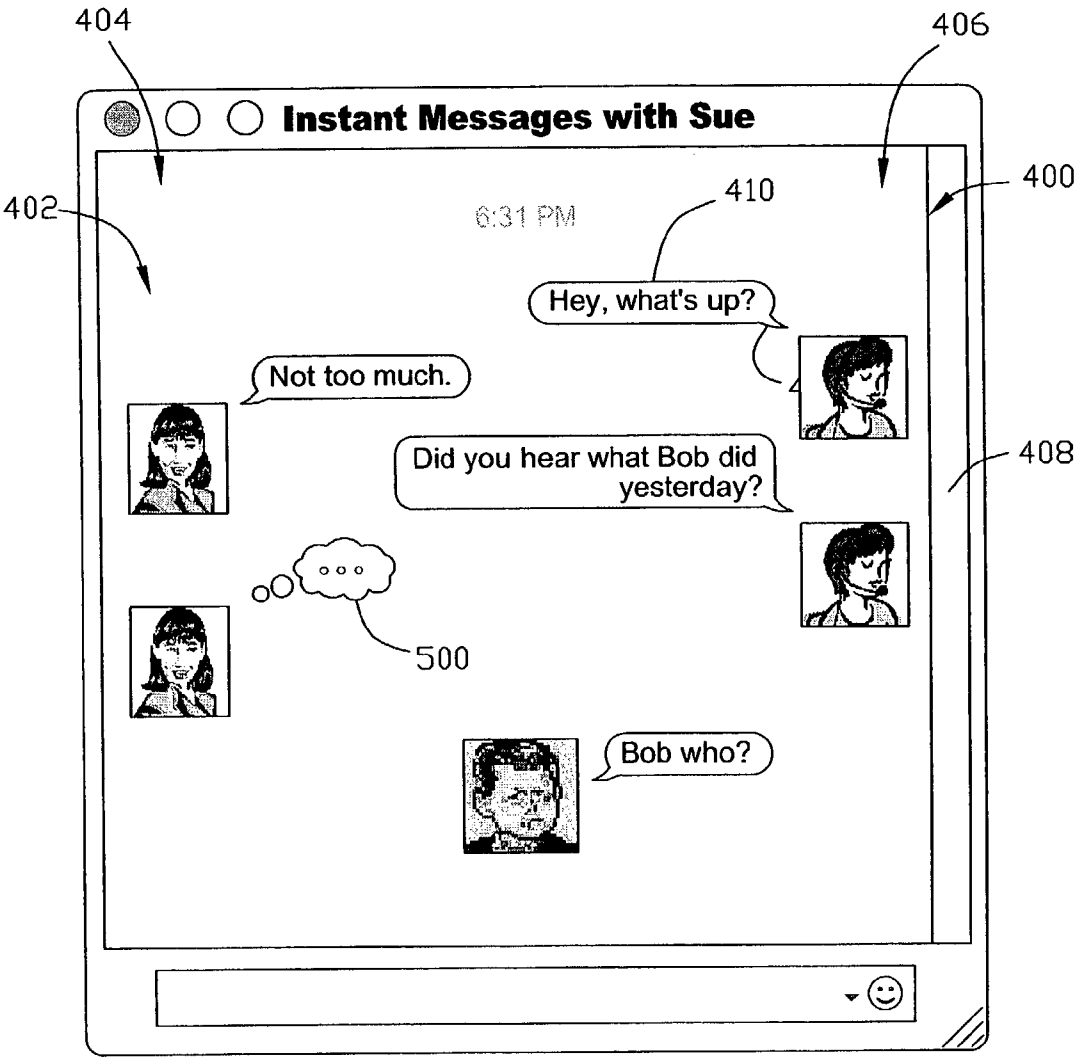

404
406
Instant Messages with Sue
400
402
410
6:31 PM
Hey, what's up?
Not too much.
Did you hear what Bob did yesterday?
408
500
Bob who?

FIGURE 5B

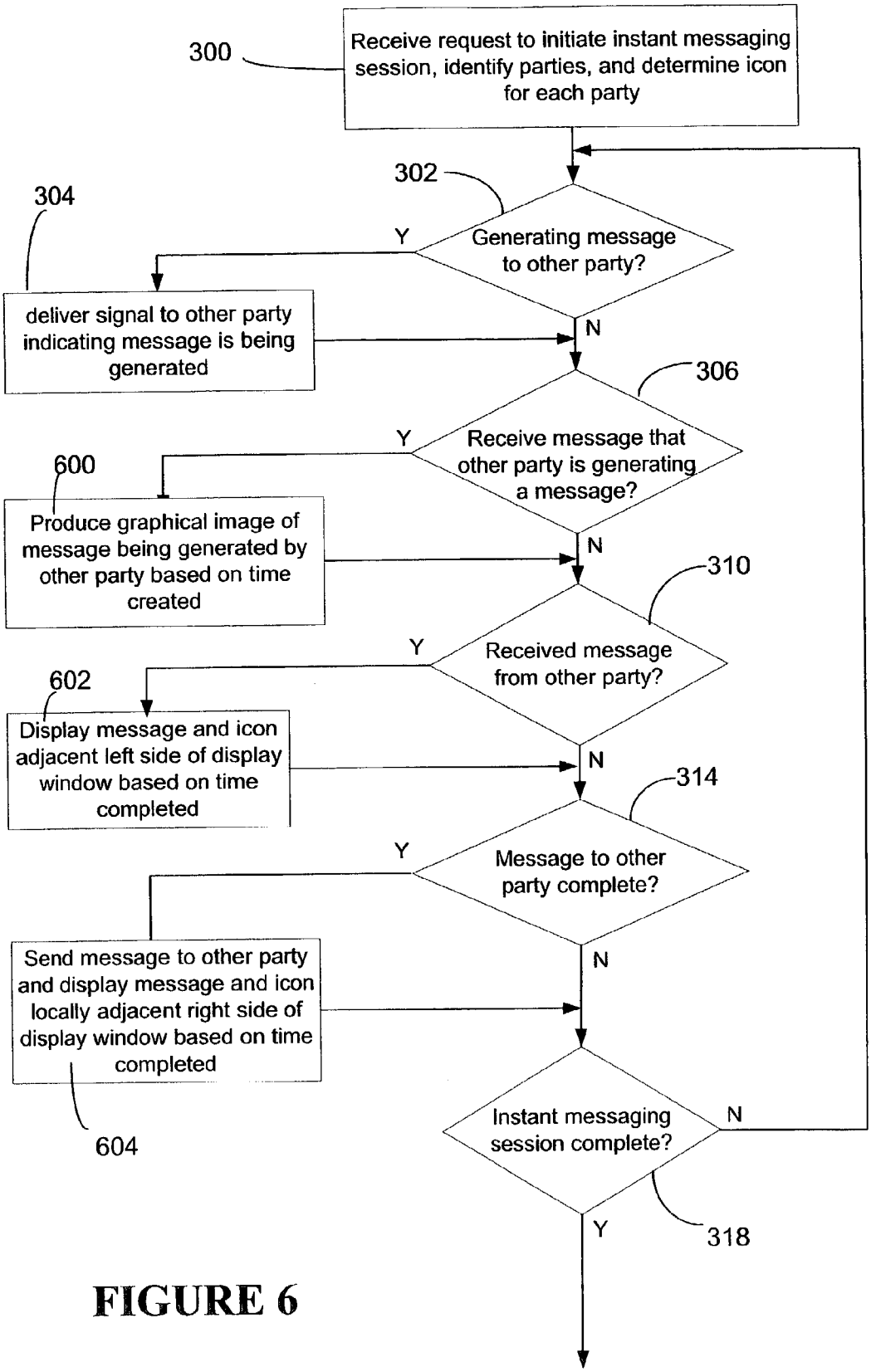

300
Receive request to initiate instant messaging session, identify parties, and determine icon for each party
302
Generating message to other party?
Y
N
304
deliver signal to other party indicating message is being generated
306
Receive message that other party is generating a message?
Y
N
600
Produce graphical image of message being generated by other party based on time created
310
Received message from other party?
Y
N
602
Display message and icon adjacent left side of display window based on time completed
314
Message to other party complete?
Y
N
Send message to other party and display message and icon locally adjacent right side of display window based on time completed
604
Instant messaging session complete?
N
Y
318

FIGURE 6

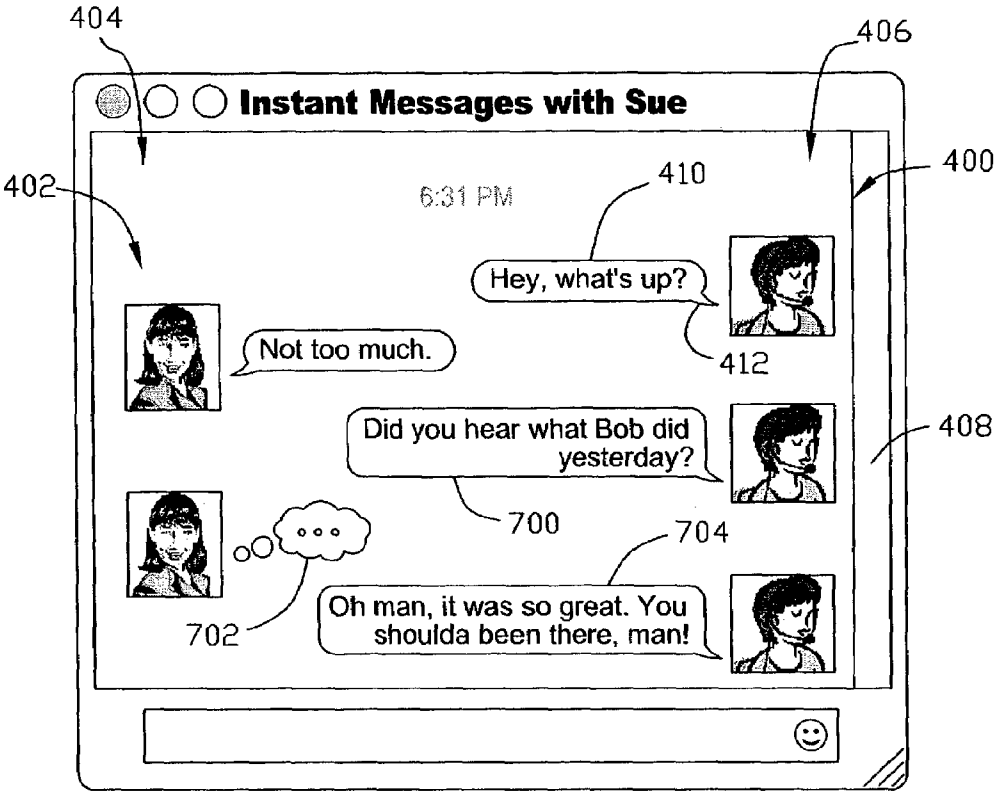


FIGURE 7

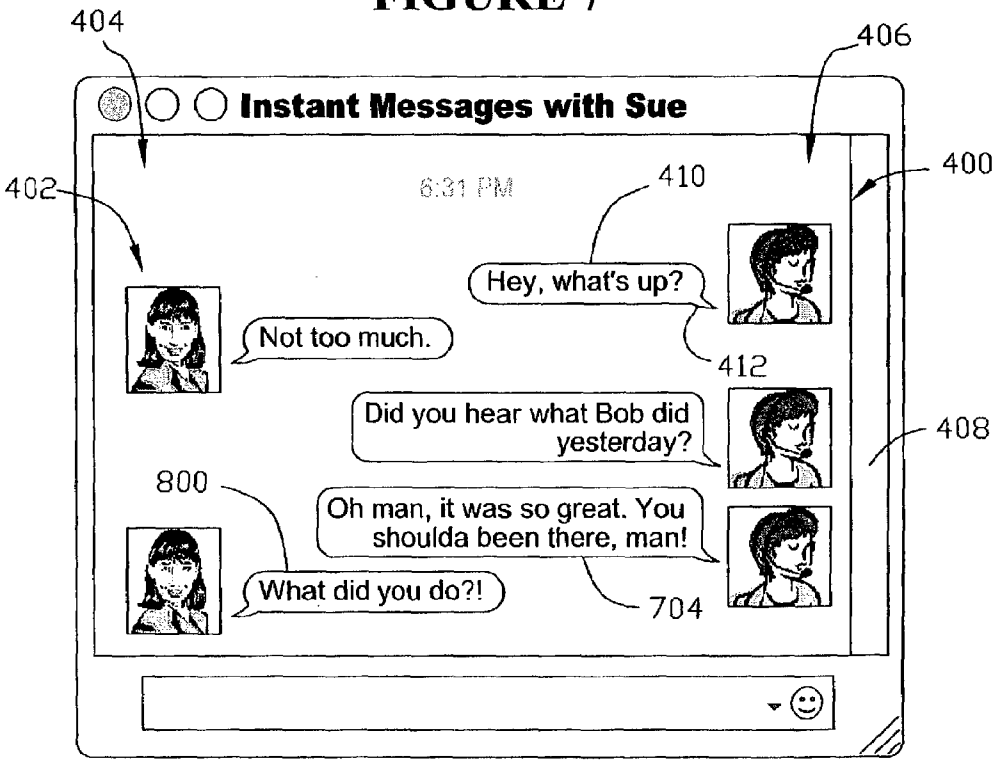


FIGURE 8

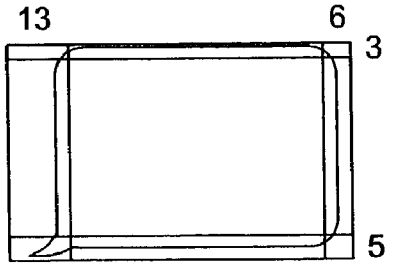


FIGURE 9A

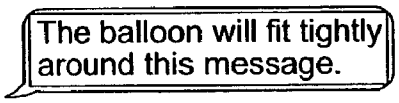


FIGURE 9B

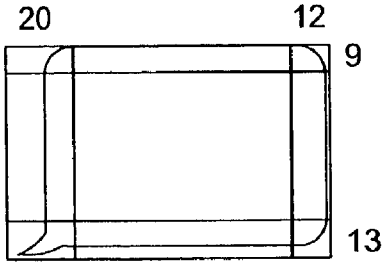


FIGURE 10A

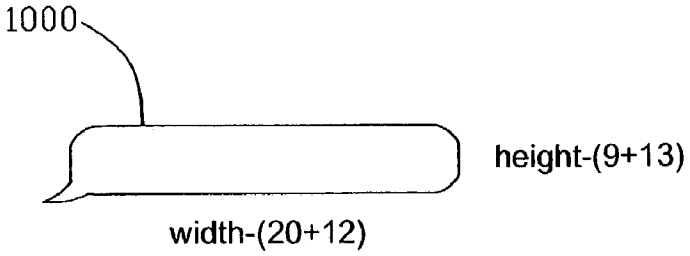


FIGURE 10B

METHOD AND APPARATUS FOR DISPLAYING INFORMATION DURING AN INSTANT MESSAGING SESSION

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to a user interface for displaying an exchange of messages during an instant messaging session, and, more particularly, to a method and apparatus for displaying instant message exchanges in a manner that graphically differentiates the participants in a conversation.

2. Description of the Related Art

Networks, such as the Internet, intranets, or other private or public networks, are ubiquitous. In fact, many computers are connected to one or more networks at the same time. For example, a business may have hundreds or even thousands of computers coupled to its own private network, which was, at least initially, used primarily for storage and exchange of computer files. At least some of these same business computers may also be coupled to the internet. Further, with the development of wireless devices, ad hoc networks may also be formed with properly configured portable devices. Even telephonic devices, such as cellular phones, pagers and the like, may be coupled to one or more of these networks. Small businesses and homes are also often connected in similar arrangements.

All of this connectivity has naturally led to communications between various users over these networks. For example, electronic mail (e-mail), because of its usefulness, is now commonplace. E-mail is now widely used by businesses and individuals, and in at least some instances has replaced more traditional forms of communications, such as mailed letters, facsimiles, telexes, and the like. However, e-mail has proven to be somewhat awkward when used to carry on an ongoing conversation.

Instant messaging, on the other hand, allows two or more users connected through these networks to carry on an interactive conversation. Exemplary instant messaging systems include Apple iChat, AOL Instant Messenger, Microsoft MSN Messenger, and the like. Typically, two or more users type in messages or select icons, which they send to one another. The receiving party(ies) may immediately respond with an appropriate message or icon. These instant messages are commonly all displayed in serial fashion, such as shown in FIG. **1**, usually scrolling the user's screen from top to bottom. Commonly, each message is preceded by a label, such as BobbyD211 and Fred1432 in FIG. **1**, indicating the identity of the author of the message. Heretofore, users have relied on these labels, or other limited indicia, to locate and identify messages from a particular party. Accordingly, it will be appreciated that the presentation of each message in substantially similar format makes it difficult to readily determine the authorship of one or more previous messages. Likewise, it is difficult to go back and quickly locate a previous message without reading through many previous messages.

The present invention is directed to overcoming or at least reducing one or more of the problems set forth above.

SUMMARY OF THE INVENTION

In one aspect of the present invention, a method is provided for displaying information related to a communication session. Information relating to data produced by a first participant to the communication session is displayed on a first display unit, wherein the information produced by the first participant is displayed at a first position on the first display unit. Data is received from a second participant to the communication session, and information relating to the data received from the second participant is displayed on the first display unit, wherein the information received from the second participant is displayed at a second position on the first display unit. The first and second positions are spatially distinct.

In another aspect of the present invention, a computer readable program storage device is provided and encoded with instructions that, when executed by a computer, performs a method. The method includes displaying information relating to data produced by a first participant to the communication session on a first display unit, wherein the information produced by the first participant is displayed at a first position on the first display unit. Data is received from a second participant to the communication session, and information relating to the data received from the second participant is displayed on the first display unit, wherein the information received from the second participant is displayed at a second position on the first display unit. The first and second positions are spatially distinct.

In still another aspect of the present invention, a graphical user interface for displaying information related to a communication session is provided. The interface is comprised of a first and a second spatially distinct region. The first region is adapted to display at least one message from a first participant to the instant messaging session. The second region is adapted to display at least one message from a second participant to the instant messaging session, and the first and second spatially distinct regions partially overlap and each include at least a portion that is free from overlapping.

In yet another aspect of the present invention, a method for displaying information related to a communication session is provided. Information relating to data produced by a participant to the communication session is received. The information received from the participant is then at least partially displayed within a speech balloon.

In still another aspect of the present invention, a graphical user interface for displaying information related to a communication session is provided. The graphical user interface comprises a first and second region, wherein the first region is adapted to display a speech balloon. The second region is adapted to display at least one message from a participant to the instant messaging session, wherein the second region is at least partially located within the first region.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention may be understood by reference to the following description taken in conjunction with the accompanying drawings, in which like reference numerals identify like elements, and in which:

FIG. **1** illustrates a view of a screen representative of a graphical user interface of a prior art instant messaging system;

FIG. **2** illustrates a top-level diagram of one embodiment of a hardware system on which the present invention may be implemented;

FIG. **3** illustrates a flowchart of an embodiment of a graphical user interface that may be executed by components within the system of FIG. **1** to produce the exemplary screens of FIGS. **4** and **5**;

FIG. **4** illustrates a first view of an exemplary screen representative of a graphical user interface;

FIGS. **5**A-**5**B illustrate a second and third view of exemplary screens representative of a graphical user interface;

FIG. **6** illustrates a flowchart of an alternative embodiment of a graphical user interface that may be executed by components within the system of FIG. **1**;

FIG. **7** illustrates a view of an exemplary screen representative of a graphical user interface;

FIG. **8** illustrates an alternative view of the exemplary screen of FIG. **7**;

FIG. **9** illustrates a typical format for a speech bubble; and

FIG. **10** illustrates a speech balloon that has been stretched or modified to accommodate a message.

While the invention is susceptible to various modifications and alternative forms, specific embodiments thereof have been shown by way of example in the drawings and are herein described in detail. It should be understood, however, that the description herein of specific embodiments is not intended to limit the invention to the particular forms disclosed, but on the contrary, the intention is to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the invention as defined by the appended claims.

DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

Illustrative embodiments of the invention are described below. In the interest of clarity, not all features of an actual implementation are described in this specification. It will of course be appreciated that in the development of any such actual embodiment, numerous implementation-specific decisions must be made to achieve the developers' specific goals, such as compliance with system-related and business-related constraints, which will vary from one implementation to another. Moreover, it will be appreciated that such a development effort might be complex and time-consuming, but would nevertheless be a routine undertaking for those of ordinary skill in the art having the benefit of this disclosure.

Turning now to FIG. **2**, a block diagram depicting a system **100** in accordance with embodiments of the present invention is illustrated. The system **100** includes a plurality of computing devices coupled together through one or more network connections. For example, a plurality of devices may be coupled together via a private or public network, such as a local area network (LAN) **102** or the Internet. The actual connection between the devices and the LAN **102** may take on one or more of any of a variety of forms, such as a network interface card (NIC), a modem, a digital subscriber line (DSL), a cable modem, a wireless connection, and the like. The devices coupled to the LAN **102** may include, for example, desktop computers, such as an Apple Macintosh® **104**, a classic Apple Mac® **106**, an IBM compatible personal computer (PC) **108**, and the like. Further, these desktop computers, such as the Apple Macintosh® **104**, may be coupled together via a smaller sub-LAN **110**, with the sub-LAN **110** being coupled to the LAN **102**. Portable devices, such as the Apple PowerBook® or iBook® **112**, may also be coupled to the LAN **102**, either directly or as part of the sub-LAN **110**. Further, other consumer devices, such as cell phones, personal data assistants (PDAs), network appliances, and other embedded devices may be connected to the LAN **102** so as to employ aspects of the instant invention.

While the invention has been illustrated herein as being useful in a network environment, it also has application in other connected environments. For example, two or more of the devices described above may be coupled together via device-to-device connections, such as by hard cabling, radio frequency signals (e.g., 802.11(a), 802.11(b), 802.11(g), Bluetooth, or the like), infrared coupling, telephone lines and modems, or the like. The instant invention may have application in any environment where two or more users are interconnected and capable of communicating with one another.

Those skilled in the art will appreciate that network connections may include a variety of other equipment, such as routers, switches, telephone modems, wireless devices, cable modems, digital subscriber lines, and the like. This type of equipment is not illustrated or discussed in detail herein so as to avoid unnecessarily obfuscating the instant invention. For purposes of understanding the instant invention, it is sufficient to recognize that additional conventional equipment of this type may be useful in establishing and maintaining communications between the various users.

At least two of the devices in the system **100** have software, such as an application program, installed thereon to allow an instant messaging session to be initiated and conducted. An instant messaging session may include real-time or near real-time communications. FIG. **3** illustrates a flowchart of a portion of the software associated with initiating the instant messaging session and controlling a graphical user interface (GUI) used by the participants to the instant messaging session. In particular, the process begins at block **300** in a conventional manner with one of the two parties sending the other party an invitation to initiate an instant messaging session. Assuming that the other party accepts the invitation, the software on each party's computer initiates the GUI, which opens a window where both parties' messages and other pertinent information and controls are displayed. An exemplary representation of the GUI is shown in FIG. **4** and may be referenced simultaneous with the discussion of FIG. **3** herein for a more complete understanding of the operation of the instant invention.

The messages exchanged by the participants may contain information regarding an icon to be used to represent each party. For example, party A may select an icon, such as "Mary" **400** as a graphical representation of party A. Party B may receive and store the icon and then display it adjacent a message delivered by party A. The icon makes it easier for party B to more quickly identify those messages associated with party A. An exemplary exchange of messages in which party A has selected the icon "Mary" **400** and party B has selected the icon "Sue" **402** is shown in FIG. **4**. Displaying unique graphical icons allows a user to readily identify the speaker with a quick glance. Additionally, displaying the icons adjacent each party's message allows the users to identify the speaker without looking away from the message region of the GUI. In an alternative embodiment, the user may elect to display not only the icon, but also the name associated with the author of the message. On the other hand, the user may elect to display only the name associated with the author of the message, preventing the icon from being displayed altogether, if desired.

Those skilled in the art will appreciate that the icons need not be delivered with each message. That is, party A may send an icon during the initial portion of the session, and party B will associate the icon with party A, store it locally, and then retrieve and display it each time a message is received from party A. Additionally, party A's icon may be overridden locally by party B. That is, party B may elect to display a different icon adjacent party A's messages, at least on the GUI viewed by party B. Party B may select any of a plurality of icons stored locally, and indicate through the local GUI, such as by pointing and clicking on various pull-down menus provided by the local GUI, that the selected icon should be used when displaying party A's messages.

The GUI may also use additional strategies to graphically differentiate the parties of the instant messaging session. For example, a sending party may send an indication of a color

scheme in which his/her messages should be displayed. The receiving party may then, at his/her discretion, display the messages from the sender in the requested color scheme.

Alternatively, the receiving party may elect to override the sending parties requested preference, and instead display each party's message in its own distinct color. That is, party A, during an initialization phase, may indicate through the local GUI that any message received from party B should be displayed with red letters and a white background, and that any messages generated by himself, should be displayed with a yellow background and black letters. In either case, the color distinction allows the party to visually determine the author of a message without the need to read and understand an identifying name, such as is illustrated in the prior art at FIG. **1** (e.g., BobbyD211).

Allowing the sender to select the color and style, however, may lead to some confusion in the event that another participant to the instant messaging sessions elects a similar style and/or font. Empowering the receiver of the message to override locally the style and color choices indicated by the sender may help to alleviate any confusion. That is, the receiving party may elect to display the message with a different color and style than indicated by the sending party, at least on the GUI viewed by the receiving party. The receiving party may select any of a plurality of colors and styles stored locally, and indicate through the local GUI, such as by pointing and clicking on various pull-down menus provided by the local GUI, that the selected color and style should be used when displaying the received messages. Alternatively, the GUI may be programmed to automatically assign a different color to each participant.

An additional graphical distinction may be accomplished by partitioning the GUI into spatially distinct regions and then directing the messages to a region based upon its authorship. For example, the exemplary GUI of FIG. **4** has been generally divided into two horizontal regions, a left region **404** and a right region **406**. For example, all messages generated by the local user (party A), represented by Mary **400**, are displayed in the right region **406**, and all messages generated by the remote user (party B), represented by Sue **402**, are displayed in the left region **404**. It should be appreciated that the assignment of left and right regions to parties A and B, respectively, may be reversed without departing from the spirit and scope of the instant invention. Moreover, it is anticipated that the various spatially distinct regions may overlap. That is, a message generated by Mary **400** may extend from the right region **406** and at least partially into the left region **404**. Similarly, a message generated by Sue **402** may extend from the left region **404** and at least partially into the right region **406**. Thus, the messages may at least partially overlap, depending on the length of the messages.

Further, depending upon the number of participants, it may be useful to define more than two spatially distinct regions. For example, where three participants are present, it may be useful to provide three horizontal regions.

In the exemplary GUI of FIG. **4**, the text of the messages associated with Sue **402** is displayed in the left region and is left justified. Similarly the text of the messages associated with Mary **400** is displayed in the right region and is right justified. Those skilled in the art will appreciate that other justification schemes may be used without departing from the spirit and scope of the instant invention.

In one embodiment of the instant invention, the order in which the messages appear on the GUI generally corresponds to the order in which they were received. For example, in the illustrated embodiment of FIG. **4**, each message is displayed below previously received messages so that the order of the conversation is preserved, with older messages appearing nearer the top of the GUI and newer messages appearing nearer the bottom of the GUI. As the display region of the GUI fills, old messages are scrolled up and out of view. A user may, however, activate a scrollbar mechanism **408** using conventional point and click techniques to alter the portion of the conversation presented in the GUI. For example, the user may move the scrollbar mechanism **408** upward to view an older portion of the conversation, or downward to view a more recent portion of the conversation.

To further enhance the readability and to provide further graphical identification of the author of each message appearing in the GUI, each message may be displayed in a speech balloon **410**. The balloon **410** includes a tail section **412**, which generally extends toward the icon associated with the author of the message. For example, each message from the user identified by the icon Mary **400** appears in a balloon **410** that has its tail section **412** extending generally toward the icon Mary **400**. In the event that an icon is not associated with the author of the message, the tail section **412** is still useful to graphically illustrate the author. That is, since the GUI is divided into left and right horizontal regions, **404**, **406** a speech balloon **410** located in the left horizontal region **404** with its tail section **412** extending toward the left will still provide a graphical indication of the author (e.g., Sue **402** in the embodiment of FIG. **4**).

The size of the balloon **410** is controlled according to the length of the message. That is, the GUI receives a message, determines the length of the message, determines the size (e.g., based on the number of lines of text to be displayed) of the balloon **410** required to display the message, and then draws the balloon **410** with text in the appropriate horizontal portion of the GUI using the colors, style, and icon associated with the author of the message. A more detailed discussion of the sizing aspect of the speech balloons may be found below in conjunction with FIGS. **9** and **10**.

During an instant messaging session it is often useful to indicate when a remote party is preparing a message to be sent. For example, after party A sends a message requesting a response, it is useful to know if party B is preparing the requested response. Knowing that the other party is about to respond allows a more natural flow to the conversation. For example, if party B does not answer a question from party A in a timely manner, party A may send a second, related request. Party B, however, may promptly respond to the first request, leaving party A to guess at whether the response applies to the first request, the second request, or both.

Accordingly, in the embodiment of the GUI shown in FIG. **3**, at block **302** the software determines whether a message is being generated, and in the event that a message is being prepared, the software at block **304** delivers a signal to the other party indicating that a message is being generated. One method for determining whether a message is being generated is for the remote terminal to monitor an input field for any characters having been entered, such as via the keyboard, and report back to the local terminal. For example, if the software detects that a keystroke has been entered or that emoticons have been selected, then it assumes that a message is being prepared to be sent to the other party, and the software sends a signal indicating that a possible responsive message is being prepared.

At block **306**, the software checks for a signal from the other remote party indicating that a message is being prepared. If such a signal is received, control transfers to block **308** where the GUI is activated to produce a graphical indication that a message is being prepared by the other party. An exemplary representation of the graphical indicator is shown

in the exemplary GUI of FIG. **5**. For example, a "thought bubble," such as is often used in comic strips to indicate that a character is thinking, is displayed along with the icon associated with the party who is preparing the message. In the exemplary embodiment of FIG. **5**, a "thought bubble" **500** provides a graphical indication that Sue **402** is currently preparing a message. For a number of reasons, the thought bubble **500** is particularly efficient for conveying the idea that the other party is preparing a response. First, the thought bubble **500** appears in the GUI in the same general region that a message would be displayed. Second, thought bubbles are common graphical representations familiar to many users. Third, because the thought bubble **500** is graphically similar to, but easily distinguishable from, the speech balloon **410**, the user may intuitively understand its function as a precursor to an actual message. Accordingly, even inexperienced users may readily understand the function and operation of the instant messaging system, and will be able to more quickly participate in an instant messaging session at a higher level of proficiency.

In an alternative embodiment, incomplete or partial messages are communicated to the recipient as an indication that a message is being prepared. In this alternative embodiment, the partial message is accompanied by a graphic indication that the message is not yet complete, such as by " . . . ." The partial messages are then periodically updated as more of the message is produced by the sender.

At block **310**, the software checks to determine if a message has been received from the other party. If so, control transfers to block **312** where the software displays the text message (or emoticon, or the like) along with the icon associated with the author. In this instance, any corresponding thought bubble is replaced by the corresponding speech balloon and its accompanying text. In the illustrated embodiments of FIGS. **4** and **5**, the messages received from the other, remote party are displayed on the left side **404** of a display window in the GUI. Additionally, the text message is presented in a speech balloon **410** and is left justified to further enhance its association with the other, remote party.

At block **314**, the software checks to determine if the message being prepared by the local party is complete. If so, control transfers to block **316** and the software delivers the message over the network connection to the other party. The message is then displayed in the speech balloon **410** in replacement of the thought balloon. Additionally, the software displays the text message (or emoticon, or the like) along with the icon associated with the author in the local GUI. In the illustrated embodiments of FIGS. **4** and **5**, the messages produced by the local party are displayed on the right side **406** of a display window in the GUI. Additionally, the text message is presented in a speech balloon **410** and is right justified to further enhance its association with the local party.

Those skilled in the art will appreciate that while the instant invention has been depicted in exemplary embodiments in which there are two participants to an instant messaging session, the instant invention may be readily employed in instant messaging sessions involving three or more participants. In one embodiment, all locally generated messages are presented on the right side **406** of the display window in the GUI, and all remotely generated messages are presented on the left side **404** of the display window. Thus, where there are two or more remote participants, each of their messages are presented on the left side **404** of the local participant's display window. In other embodiments, each remote participant's messages could be displayed in a spatially distinct region from that of the other participants. For example, messages from first, second and third remote participants could be displayed in first, second and third regions, respectively, wherein the first, second and third regions are spatially distinct. In the embodiment shown in FIG. **5**B, a third participant is assigned to a central region, with his/her icon appearing in the central region and the associated speech bubble extending generally therefrom.

Additionally, while the embodiments described herein have been shown with the GUI divided into spatially distinct horizontal regions, it is envisioned that other embodiments may spatially distinguish the various participants in other manners without departing from the spirit and scope of the instant invention. For example, the various participants may be distinguished by dividing the GUI into spatially distinct vertical regions with each participant being assigned a vertical portion of the GUI. For example, the local participant may be assigned the top region of the GUI, and the remote participant may be assigned the bottom region. Additional remote participants may be grouped together or, as described above, assigned an intermediate region, such as a vertical middle region. In a GUI with vertically distinct regions it may also be useful to allow the conversation to scroll horizontally, as opposed to the common vertical scrolling employed in many instant messaging systems. For example, more recent messages may be displayed to the right of older messages, with older messages scrolling off the left side of the GUI as the conversation advances.

If the messaging session is complete, such as by one or both of the parties logging off of the network or otherwise shutting down the software, then block **318** detects the ending of the session and transfers control out to another program responsible for a proper and orderly winding up of the program. Otherwise, if the instant messaging session continues, then control transfers back to block **302** where the process repeats.

Turning now to FIG. **6**, an alternative embodiment of at least a portion of the software shown in FIG. **3** is illustrated. In this embodiment of the software, provision is made to preserve the order of a conversation during those instances in which a first party is preparing a response but the second party nevertheless sends a response before the first party completes and sends its response. For example, consider the instant messaging session shown in FIG. **7** to illustrate an out-of-order presentation of messages in a conversation. In the embodiment illustrated in FIG. **7**, Sue **402** is in the process of preparing a response to a message **700** generated by Mary **400**. Accordingly, a thought bubble **702** is positioned adjacent the Sue icon **402** below the message **700**. Mary **400**, however, did not wait for Sue's response, but sent a message **704**. Thus, once Sue **402** completes and sends the response, the thought bubble **702** will be replaced by a speech balloon (not shown) containing the message. If the speech balloon (not shown) merely replaces the thought bubble with re-ordering, then the conversation will appear to have occurred in the order **700-702-704**, even though the speech bubble replacing the thought bubble **702** occurred after, not before the speech bubble **704**. This out-of-sequence ordering may give rise to confusion, particularly where the participants read the flow of the conversation at a subsequent time.

The flowchart of FIG. **6** illustrates one embodiment of a method useful in reordering the speech balloons so that they appear in the GUI in the order in which they actually occurred. Generally, the process set forth in FIG. **3** is substantially similar to that of FIG. **6**, with the exception of blocks **600**, **602**, and **604**. Generally, the order of the speech balloons is maintained based upon the time that the message was completed. Thought bubbles, on the other hand, are ordered based upon the time that they were created and are subse-

quently replaced by a speech balloon. Because a thought bubble may be created well before the corresponding speech balloon is completed, it is possible for other parties to complete messages in the intervening time. Thus, when the corresponding speech bubble is completed and replaces the corresponding thought bubble, the order of the speech balloons may vary.

At block **306**, the software checks for a signal from the other remote party indicating that a message is being prepared. If such a signal is received, control transfers to block **600** where the GUI is activated to produce a graphical indication that a message is being prepared by the other party. The order in which the graphical indication is displayed is based upon the time that the thought bubble was created. The time that the thought bubble was created may be determined from a time stamp provided by the remote user who is in the process of preparing the message.

Thereafter, at block **310**, the software checks to determine if a message has been received from the other party. If so, control transfers to block **602** where the software displays the text message (or emoticon, or the like) along with the icon associated with the author. In this instance, any corresponding thought bubble is removed and replaced by the corresponding speech balloon and its accompanying text. However, the speech balloon is ordered based upon the time completed. The time that the speech bubble was completed may be determined from a time stamp provided by the remote user who generated the message.

An exemplary instant messaging session that illustrates the results of this ordering policy is presented in FIGS. **7** and **8**. In the embodiment illustrated in FIG. **7**, Sue **402** is in the process of preparing a response to a message **700** generated by Mary **400**. Accordingly, a thought bubble **702** is positioned adjacent the Sue icon **402** below the message **700**. Mary **400**, however, did not wait for Sue's response, but sent a message **704**. Because the speech balloons **700**, **704** are ordered based on the time completed whereas the thought balloon **702** is ordered based on the time created, the order of the messages will remain as shown in FIG. **7**, until the message from Sue **402** is finally received.

Thus, as is shown in FIG. **8**, a speech balloon **800** has replaced the thought bubble **702**, but is located after (or below) the speech balloon **704**, as the speech balloon **800** was completed after the speech balloon **704**. In this manner, the actual order of the conversation is preserved.

A substantially similar process occurs with respect to displaying speech balloons associated with the local user. For example, at block **314**, the software checks to determine if the message being prepared by the local party is complete. If so, control transfers to block **604** and the software delivers the message over the network connection to the other party. The message is then displayed in a speech balloon in an order based on the time that the message was completed.

Turning now to FIGS. **9** and **10**, one exemplary method for formatting and sizing the speech balloon **410** is shown. In one embodiment, a text system, such as a standard text system used in Mac OS X is used to produce the text of the message. The text system provides information regarding the size of the text message to the GUI. The GUI uses this size information to construct a speech balloon of an appropriate size to contain the message. Certain rules regarding the formatting of the speech balloon affect the size of the speech balloon. For example, in one embodiment, upper, lower, left and right margins are established. In the exemplary embodiment illustrated in FIG. **9**A, the margins are selected as follows: upper—3; lower—5; left—13; and right—6. The text of the message is required to be positioned within the speech balloon and within these margins. Using these margins causes the speech balloon to fit tightly around the message, as illustrated in FIG. **9**B, so as to be aesthetically pleasing while not unnecessarily consuming large portions of the GUI. Reducing the size of the speech balloons allows more messages to appear on the GUI at one time.

The GUI uses the information regarding the size of the text message and the desired margins to produce a speech balloon of the appropriate size. The process involves dividing a template speech balloon into nine regions, such as is shown in FIG. **10**A. The nine regions are comprised of four corners, left and right edges, top and bottom edges, and a central region. Initially, the margins are added to the rectangular area taken up by the text to produce a destination rectangle having dimensions in which the balloon is to be drawn. The four corner regions are drawn directly into the corners of the destination rectangle without significant change to their shape or size. The top and bottom edges are tiled horizontally into rectangles of the appropriate width (and original height). Varying the width of the top and bottom edges has the desirable effect of altering the horizontal dimension of the speech balloon. The left and right edges are tiled vertically into rectangles of the appropriate height (and original width) to produce a stretched or modified speech bubble **1000**, as shown in FIG. **10**B. It should be noted that the destination rectangle can be smaller than the original template image in either or both its vertical and horizontal dimension. For example, in the embodiment illustrated in FIG. **10**B, the vertical dimension of the speech balloon is smaller that the vertical dimension of the template speech balloon of FIG. **10**A, and the horizontal dimension of the speech balloon is larger that the horizontal dimension of the template speech balloon of FIG. **10**A.

Once the speech balloon is appropriately sized, it is color filled according to the requirements of the sender or the recipient, as discussed above. Coloration and shading of the speech balloon is accomplished by alternative methodologies. In one embodiment, custom artwork is provided for each color to produce a desired variation across the surface of the speech balloon. For example, the color may be varied so that the coloration is lighter adjacent the bottom edge of the speech balloon. This coloration scheme has proven to be pleasing to users, providing the appearance of three dimensional qualities to the speech balloon. Alternatively, rather than developing custom artwork for each possible color, the speech balloon may be filled uniformly with the desired color. Thereafter, standard template shadows may be overlayed to produce a similar three dimensional effect.

Finally, while FIGS. **9** and **10** illustrate the speech balloon **410** drawn with its tail extending leftward, so as to be used in the left horizontal region of the GUI, the same processes described herein may be applied in drawing the speech balloon **410** with its tail extending rightward, so as to be used in the right horizontal region of the GUI. In particular, when the speech balloon **410** with a rightward extending tail is required, the speech balloon **410** with the leftward extending tail is designed to be of the proper size for the message, and then the speech balloon **410** is flipped horizontally or rotated about a central vertical axis to produce the appropriately sized speech balloon **410** with a rightward extending tail. In this manner, an economy of software coding is realized, as only a single routine for generating speech balloon with either leftward or rightward extending tails is required.

The particular embodiments disclosed above are illustrative only, as the invention may be modified and practiced in different but equivalent manners apparent to those skilled in the art having the benefit of the teachings herein. Further-

more, no limitations are intended to the details of construction or design herein shown, other than as described in the claims below. It is therefore evident that the particular embodiments disclosed above may be altered or modified and all such variations are considered within the scope and spirit of the invention. Accordingly, the protection sought herein is as set forth in the claims below.

What is claimed is:

1. A method for displaying information related to a communication session, comprising:

displaying information relating to data produced by a plurality of participants to the communication session on a first display unit, wherein the information produced by a first participant of the plurality of participants is displayed at a first position on the first display unit;

receiving data from at least one of the remaining participants of the plurality of participants by the communication session; and

displaying information relating to data received from the at least one of the remaining participants of the plurality of participants, on the first display unit, wherein the information received from the at least one of the remaining participants of the plurality of participants is displayed at a separate position for each said remaining participant of the plurality of participants on the first display unit, wherein the respective separate positions relating to each of the plurality of participants are all non-random, horizontal-spatially distinct and chronologically arranged.

2. A method, as set forth in claim **1**, wherein the respective positions relating to the plurality of participants are permitted to at least partially overlap.

3. A method, as set forth in claim **1**, wherein:

receiving data from the remaining participants of the plurality of participants includes receiving data from a second participant of the plurality of participants;

displaying information relating to data produced by the first participant further comprises displaying information relating to data produced by the first participant in a right justified format; and

displaying information relating to data received from the second participant further comprises displaying information relating to data received from the second participant in a left justified format.

4. A method, as set forth in claim **1**, wherein:

receiving data from the remaining participants of the plurality of participants includes receiving data from a second participant of the plurality of participants;

displaying information relating to data produced by the first participant further comprises displaying information relating to data produced by the first participant in a left justified format; and

displaying information relating to data received from the second participant further comprises displaying information relating to data received from the second participant in a right justified format.

5. A method, as set forth in claim **1**, wherein:

displaying information relating to data produced by the first participant further comprises displaying information relating to data produced by the first participant in a speech balloon.

6. A method, as set forth in claim **1**, wherein:

receiving data from the remaining participants of the plurality of participants includes receiving data from a second participant of the plurality of participants;

displaying information relating to data received from the second participant further comprises displaying information relating to data received from the second participant in a speech balloon.

7. A method, as set forth in claim **1**, further comprising:

receiving data from a second participant of the plurality of participants;

detecting data being entered by the second participant before the data is delivered to the first participant; and

displaying information relating to data being entered by the second participant on the first display unit.

8. A method, as set forth in claim **7**, wherein:

displaying information relating to data being entered by the second participant on the first display unit further comprises displaying a thought bubble on the first display unit.

9. A method, as set forth in claim **7**, wherein:

displaying information relating to data being entered by the second participant on the first display unit further comprises displaying the thought bubble on the first display unit at a second position on the first display unit.

10. A method, as set forth in claim **7**, wherein:

displaying information relating to data being entered by the second participant on the first display unit further comprises displaying preselected graphical information on the first display unit at a second position on the first display unit.

11. A method, as set forth in claim **7**, wherein:

displaying information relating to data being entered by the second participant on the first display unit further comprises displaying preselected graphical information on the first display unit at a second position on the first display unit; and

removing the preselected graphical information from the first display unit in response to receiving data from the second participant.

12. A method, as set forth in claim **11**, wherein:

displaying preselected graphical information on the first display unit at the second position on the first display unit further comprises displaying a thought bubble at the second position on the first display unit.

13. A computer readable program storage device encoded with instructions that, when executed by a computer, performs a method, comprising:

displaying information relating to data produced by a plurality of participants to the communication session on a first display unit, wherein the information produced by a first participant of the plurality of participants is displayed at a first position on the first display unit;

receiving data from at least one of the remaining participants of the plurality of participants by the communication session; and

displaying information relating to data received from the at least one of the remaining participants of the plurality of participants, on the first display unit, wherein the information received from the at least one of the remaining participants of the plurality of participants is displayed at a separate position for each said remaining participant of the plurality of participants on the first display unit, wherein the respective separate positions relating to each of the plurality of participants are all non-random, horizontal-spatially distinct and chronologically arranged.

14. A computer readable program storage device, as set forth in claim **13**, wherein the respective positions relating to the plurality of participants are permitted to at least partially overlap.

15. An apparatus for displaying information related to a communication session, comprising:

means for displaying information relating to data produced by a plurality of participants to the communication session on a first display unit, wherein the information produced by a first participant is displayed at a first position on the first display unit;

means for receiving data from at least one of the remaining participants of the plurality of participants by the communication session; and

means for displaying information relating to data received from the at least one of the remaining participants of the plurality of participants on the first display unit, wherein the information received from the at least one of the remaining participants of the plurality of participants is displayed at a separate position for each said remaining participant of the plurality of participants on the first display unit, wherein the respective separate positions relating to each of the plurality of participants are all non-random, horizontal-spatially distinct and chronologically arranged.

16. An apparatus, as set forth in claim **15**, wherein the respective positions relating to the plurality of participants are permitted to at least partially overlap.

17. A graphical user interface for displaying information related to a communication session, comprising:

a plurality of regions adapted to display messages from a plurality of participants to the instant messaging session;

a first region of the plurality of regions adapted to display at least one message from a first participant to the instant messaging session; and

the remaining regions of the plurality of regions adapted to display at least one message from the at least one of the remaining participants of the plurality of participants respectively to the instant messaging session, wherein the respective separate plurality of regions relating to each of the plurality of participants are all non-random, horizontal-spatially distinct and chronologically arranged.

18. A graphical user interface, as set forth in claim **17**, wherein the respective plurality of regions relating to the plurality of participants are permitted to at least partially overlap.

19. A graphical user interface, as set forth in claim **17**, wherein the first region is adapted to display an icon associated with the first participant, and a second region of the plurality of regions is adapted to display an icon associated with a second participant of the plurality of participants.

20. A graphical user interface, as set forth in claim **17**, wherein the first region is adapted to display text from the first participant in a speech balloon.

21. A graphical user interface, as set forth in claim **20**, wherein the first region is adapted to display the text in a left justified format.

22. A graphical user interface, as set forth in claim **17**, wherein a second region of the plurality of regions is adapted to display text from a second participant of the plurality of participants in a speech balloon in a right justified format.

23. A graphical user interface, as set forth in claim **17**, wherein the first region is adapted to display a thought bubble in response to receiving a signal from the first participant that a message is being prepared to be sent to a second participant of the plurality of participants.

24. A graphical user interface, as set forth in claim **23**, further comprising a second region of the plurality of regions, wherein the first and second regions are adapted to display the messages received from the first and second participants vertically in the order in which they were received.

25. A graphical user interface, as set forth in claim **24**, wherein the first and second regions are adapted to replace the thought bubble with a corresponding speech balloon when the speech balloon is received.

26. A graphical user interface, as set forth in claim **25**, wherein the first and second regions are adapted to display the corresponding speech balloon in a vertical position corresponding to the order in which it was received regardless of the vertical position in which the thought bubble was displayed.

27. A system, comprising: a first terminal; and

a second terminal, wherein the first and second terminals each include a graphical user interface for displaying information related to a communication session between the first and second terminals, the graphical user interface comprising:

a plurality of regions adapted to display messages from a plurality of participants;

a first region of the plurality of regions adapted to display at least one message from a first participant to the instant messaging session; and

the remaining regions of the plurality of regions adapted to display at least one message from the at least one of the remaining participants of the plurality of participants to the instant messaging session, wherein the respective separate plurality of regions relating to each of the plurality of participants are all non-random, horizontal-spatially distinct and chronologically arranged.

28. A system, as set forth in claim **27**, wherein the respective plurality of regions relating to the plurality of participants are permitted to at least partially overlap.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,669,134 B1
APPLICATION NO. : 10/428523
DATED : February 23, 2010
INVENTOR(S) : Christie et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

On the Title Page:

The first or sole Notice should read --

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 1143 days.

Signed and Sealed this

Seventh Day of December, 2010

David J. Kappos

David J. Kappos
Director of the United States Patent and Trademark Office

Exhibit 3

US006493002B1

(12) **United States Patent**
Christensen

(10) **Patent No.: US 6,493,002 B1**
(45) **Date of Patent: *Dec. 10, 2002**

(54) **METHOD AND APPARATUS FOR DISPLAYING AND ACCESSING CONTROL AND STATUS INFORMATION IN A COMPUTER SYSTEM**

(75) Inventor: **Steven W. Christensen**, Milpitas, CA (US)

(73) Assignee: **Apple Computer, Inc.**, Cupertino, CA (US)

(*) Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **08/821,004**

(22) Filed: **Mar. 20, 1997**

Related U.S. Application Data

(63) Continuation of application No. 08/316,237, filed on Sep. 30, 1994, now abandoned.

(51) **Int. Cl.**[7] **G06F 3/00**
(52) **U.S. Cl.** **345/779**; 345/772
(58) **Field of Search** 395/345, 350, 395/352, 354, 326, 339, 340, 341, 973, 974; 345/348, 350, 352, 354, 326, 339, 340, 341, 973, 974, 700, 764, 771–772, 776–779, 781, 784, 788, 791, 798–800, 833, 856

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,862,389	A	*	8/1989	Takagi	345/794
4,885,704	A	*	12/1989	Takagi et al.	345/166
4,896,291	A	*	1/1990	Gest et al.	345/841
4,931,957	A	*	6/1990	Takagi et al.	345/453
5,091,866	A	*	2/1992	Takagi	345/803
5,146,556	A		9/1992	Hullot et al.	345/790
5,202,961	A	*	4/1993	Mills et al.	345/720
5,416,895	A		5/1995	Anderson et al.	707/503
5,428,730	A	*	6/1995	Baker et al.	345/740
5,588,105	A	*	12/1996	Foster et al.	345/779
5,617,526	A	*	4/1997	Oran et al.	345/779
5,640,498	A	*	6/1997	Chew	345/790
5,644,334	A	*	7/1997	Jones et al.	345/419
5,659,693	A	*	8/1997	Hansen et al.	345/779
5,678,034	A	*	10/1997	Chew	345/520
5,757,371	A	*	5/1998	Oran et al.	345/779
5,825,357	A	*	10/1998	Malamud et al.	345/779
6,133,898	A		10/2000	Ludolph et al.	345/790

FOREIGN PATENT DOCUMENTS

EP	0 483 777 A2	10/1991
EP	483777 A	5/1992
EP	0 584 391 A1	8/1992
EP	584392 A	3/1994

OTHER PUBLICATIONS

EPO 0 584 392 A1, Cohausz, English Translation of the German patent document, Mar. 1992.*
"PCT Written Opinion", Oct. 4, 1996.
Steven Harris, et al.: "Inside WordPerfect 6 for Windows", 1994, USA, pp. 1104–1108.
Mark A. Benge, Matt Smith: "Designing Custom Controls", Spring 1993, USA, pp. 72–85.
International Search Report, PCT/US 95/11025, Jan. 03, 1996.

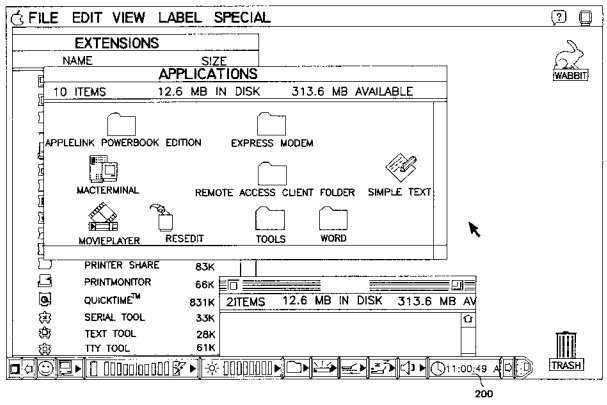
* cited by examiner

Primary Examiner—Crescelle N. dela Torre
(74) *Attorney, Agent, or Firm*—Blakely, Sokoloff, Taylor & Zafman LLP

(57) **ABSTRACT**

An interactive computer-controlled display system having a processor, a data display screen, a cursor control device for interactively positioning a cursor on the data display screen, and a window generator that generates and displays a window on a data display screen. The window region provides status and control information in one or more data display areas. The individiual data display areas may be controlled through the use of controls and indicators on the control strip itself using cursor control keys.

50 Claims, 13 Drawing Sheets



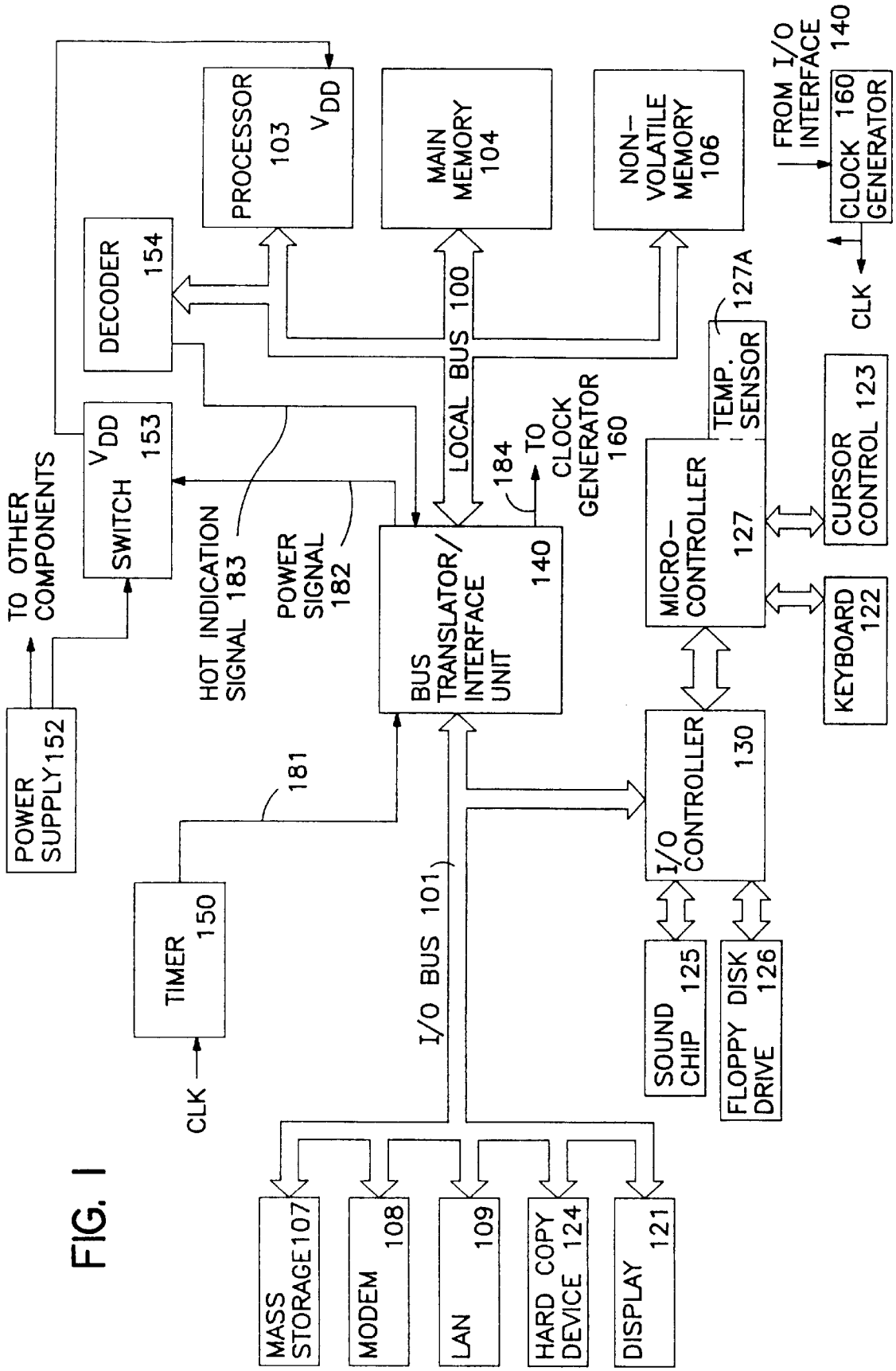

PROCESSOR 103
V_{DD}
MAIN MEMORY 104
NON-VOLATILE MEMORY 106
FROM I/O INTERFACE 140
CLOCK GENERATOR 160
CLK
DECODER 154
LOCAL BUS 100
127A
TEMP. SENSOR
CURSOR CONTROL 123
V_{DD} SWITCH 153
TO OTHER COMPONENTS
184
TO CLOCK GENERATOR 160
MICRO-CONTROLLER 127
HOT INDICATION SIGNAL 183
POWER SIGNAL 182
BUS TRANSLATOR/ INTERFACE UNIT 140
KEYBOARD 122
POWER SUPPLY 152
181
I/O CONTROLLER 130
TIMER 150
I/O BUS 101
SOUND CHIP 125
FLOPPY DISK DRIVE 126
CLK
MASS STORAGE 107
MODEM 108
LAN 109
HARD COPY DEVICE 124
DISPLAY 121

FIG. 1

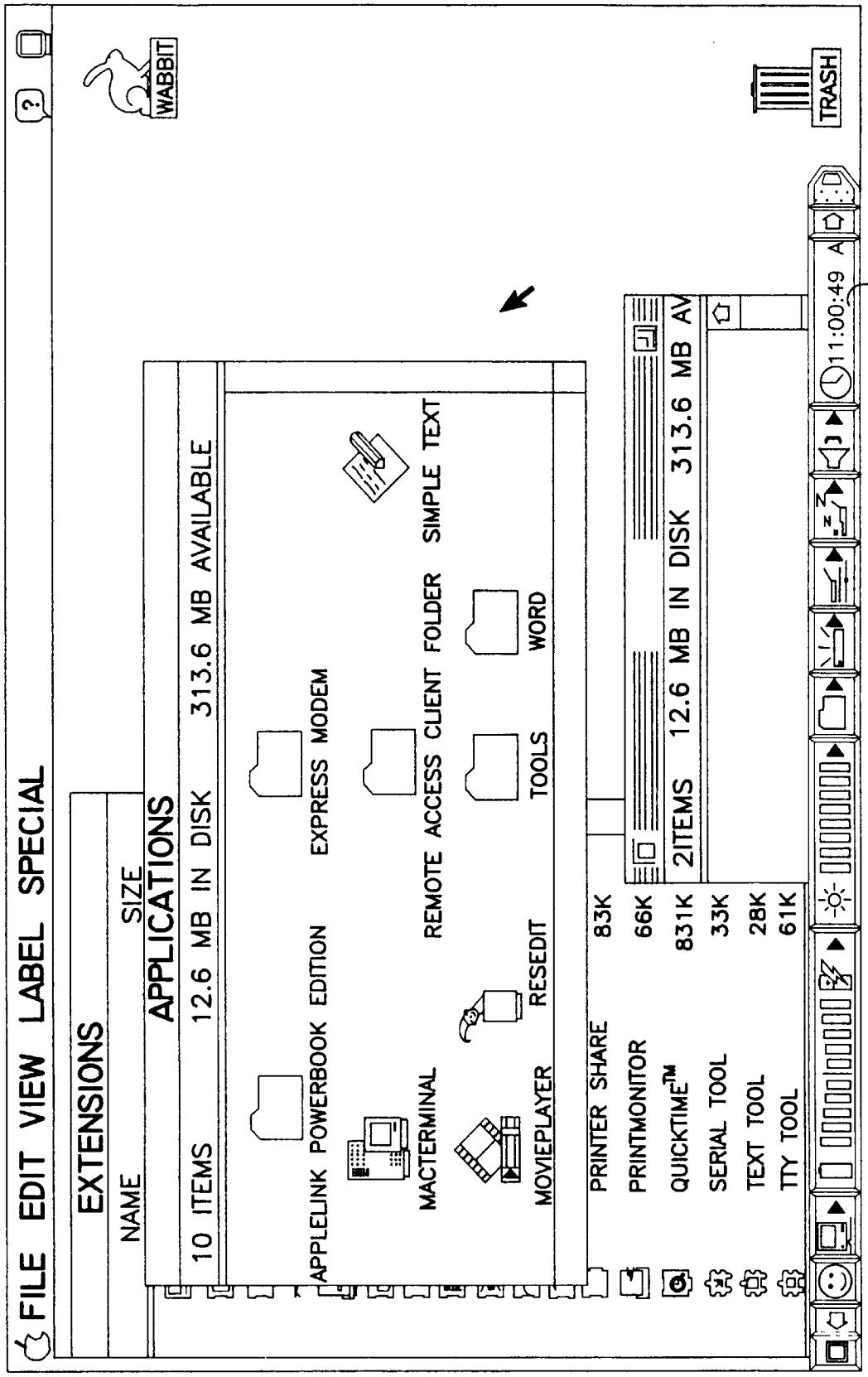

FILE EDIT VIEW LABEL SPECIAL
EXTENSIONS
NAME
SIZE
APPLICATIONS
10 ITEMS
12.6 MB IN DISK
313.6 MB AVAILABLE
APPLELINK
POWERBOOK EDITION
EXPRESS MODEM
MACTERMINAL
REMOTE ACCESS CLIENT
RESEDIT
MOVIEPLAYER
TOOLS
FOLDER
WORD
SIMPLE TEXT
PRINTER SHARE 83K
PRINTMONITOR 66K
QUICKTIME™ 831K
SERIAL TOOL 33K
TEXT TOOL 28K
TTY TOOL 61K
2ITEMS 12.6 MB IN DISK 313.6 MB AV
WABBIT
TRASH
11:00:49 A
200

FIG. 2A

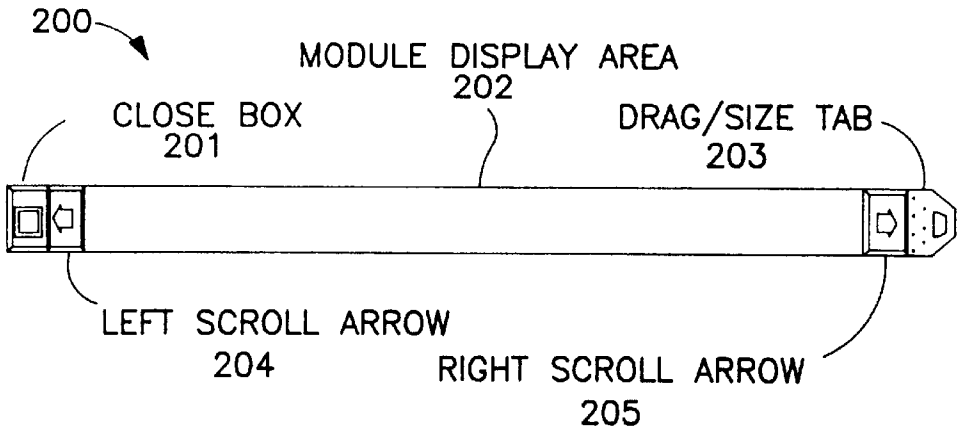


FIG. 2B

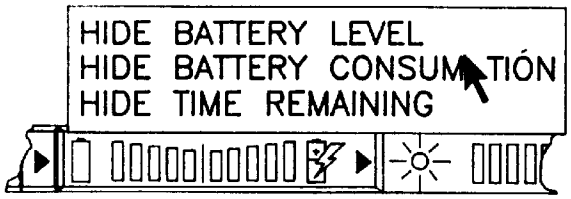


FIG. 2C

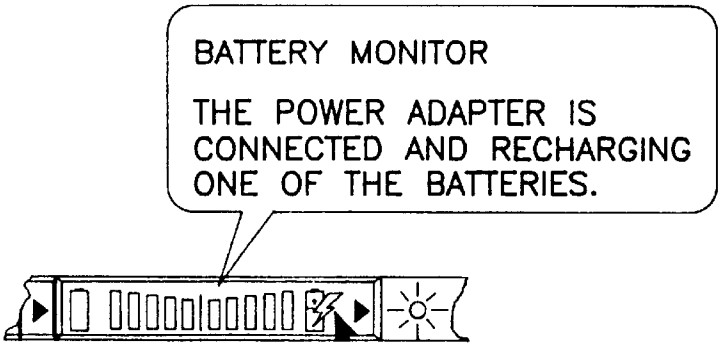


FIG. 2D

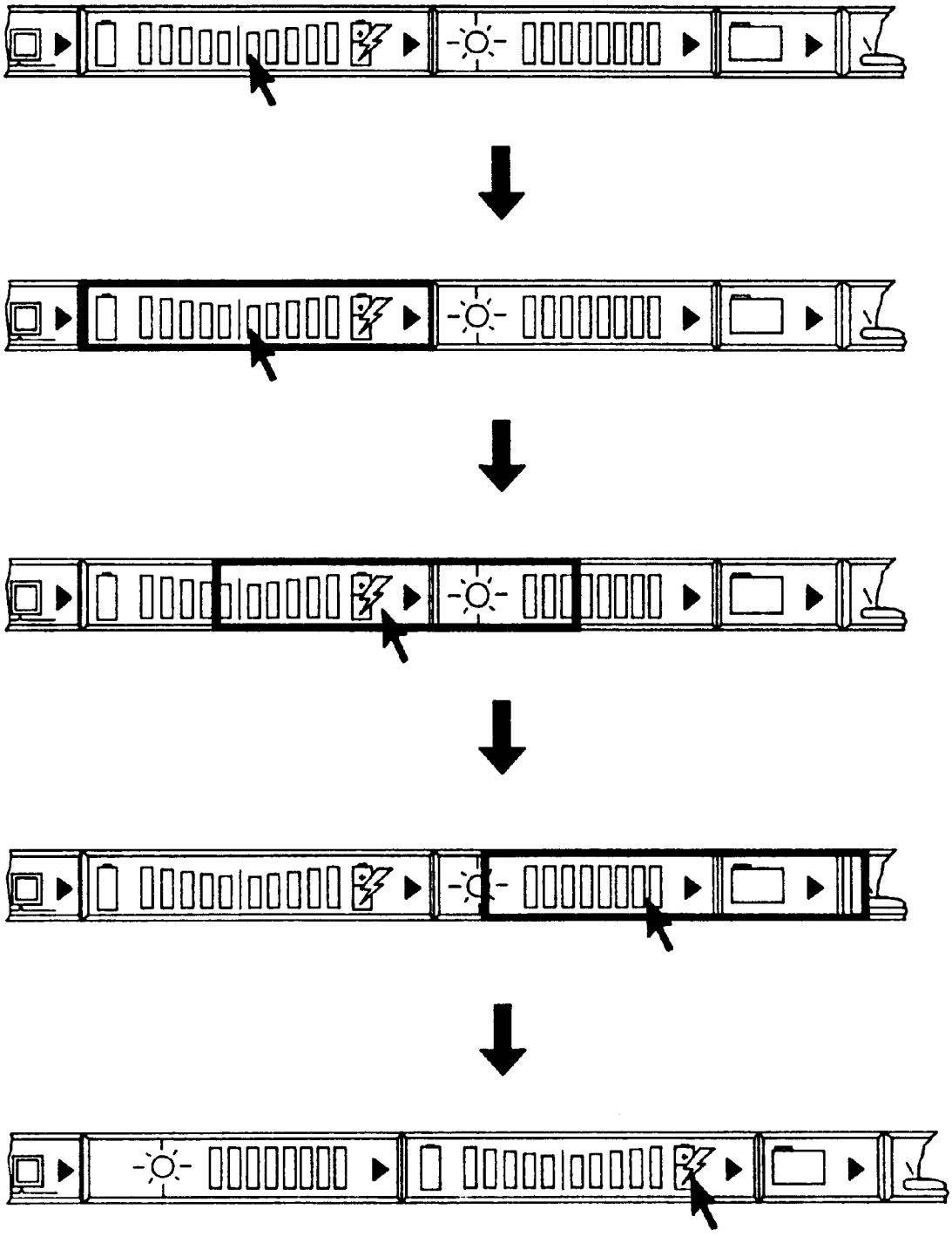


FIG. 2E

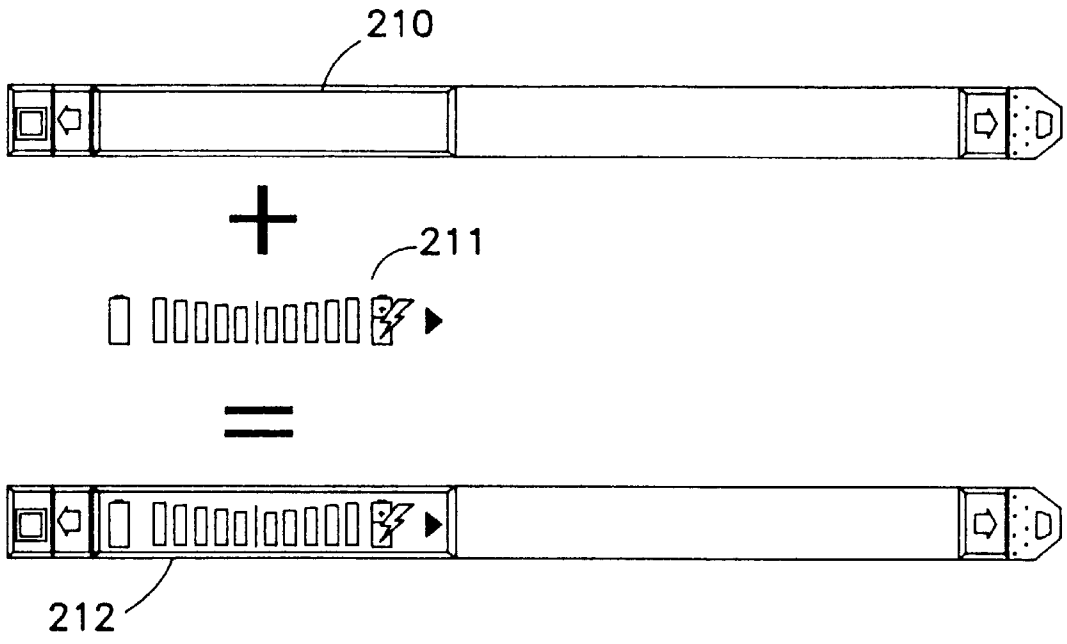

210
211
212

FIG. 2F

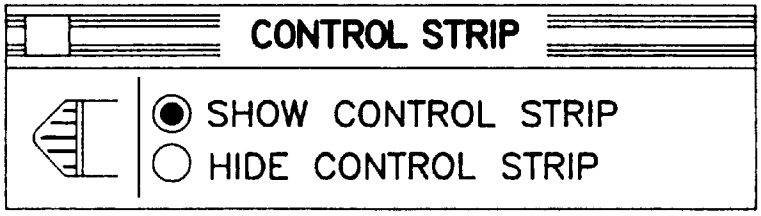

CONTROL STRIP
SHOW CONTROL STRIP
HIDE CONTROL STRIP

FIG. 3

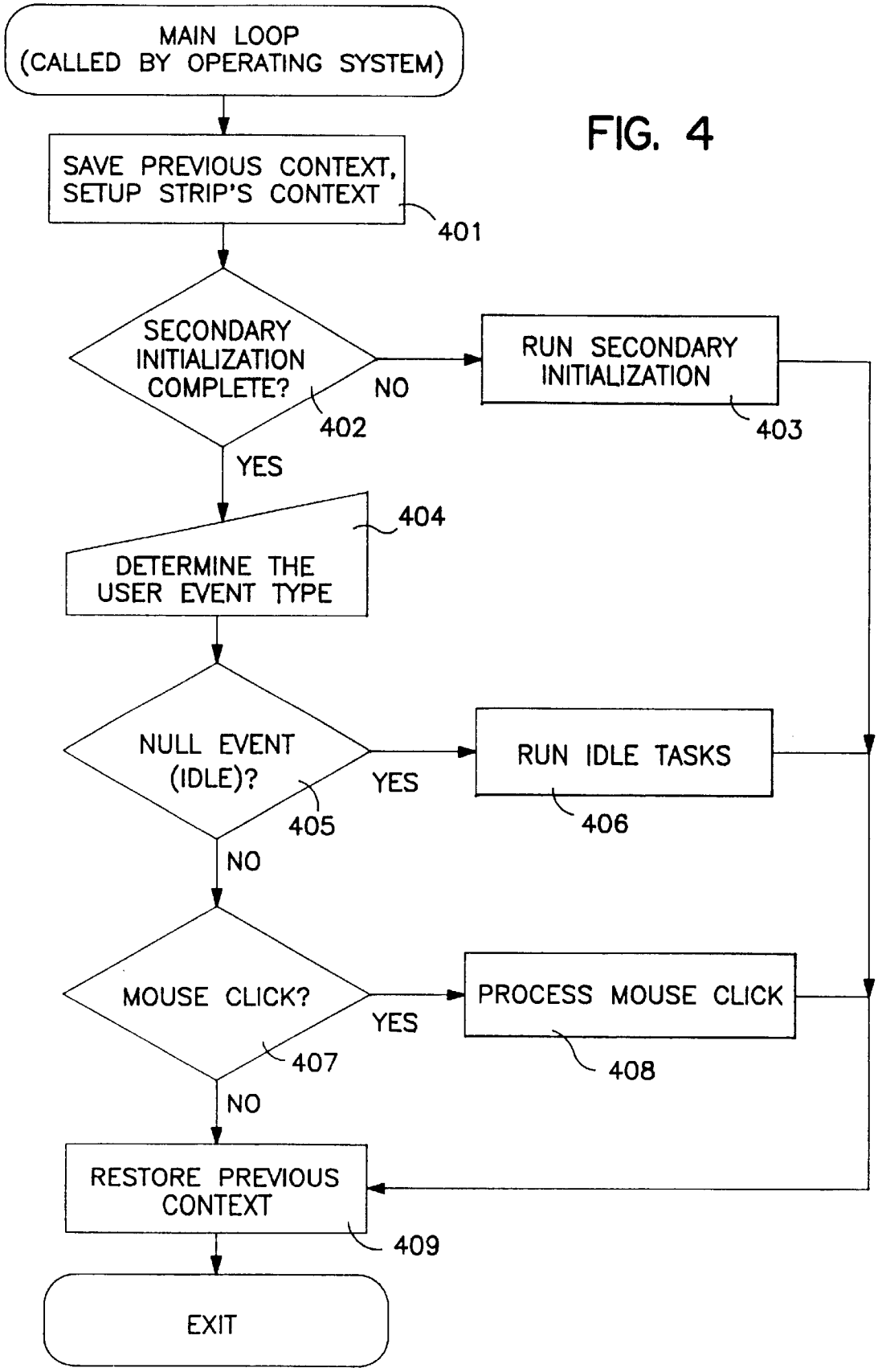

MAIN LOOP
(CALLED BY OPERATING SYSTEM)
SAVE PREVIOUS CONTEXT, SETUP STRIP'S CONTEXT
401
SECONDARY INITIALIZATION COMPLETE?
402
NO
RUN SECONDARY INITIALIZATION
403
YES
404
DETERMINE THE USER EVENT TYPE
NULL EVENT (IDLE)?
405
YES
RUN IDLE TASKS
406
NO
MOUSE CLICK?
407
YES
PROCESS MOUSE CLICK
408
NO
RESTORE PREVIOUS CONTEXT
409
EXIT

FIG. 4

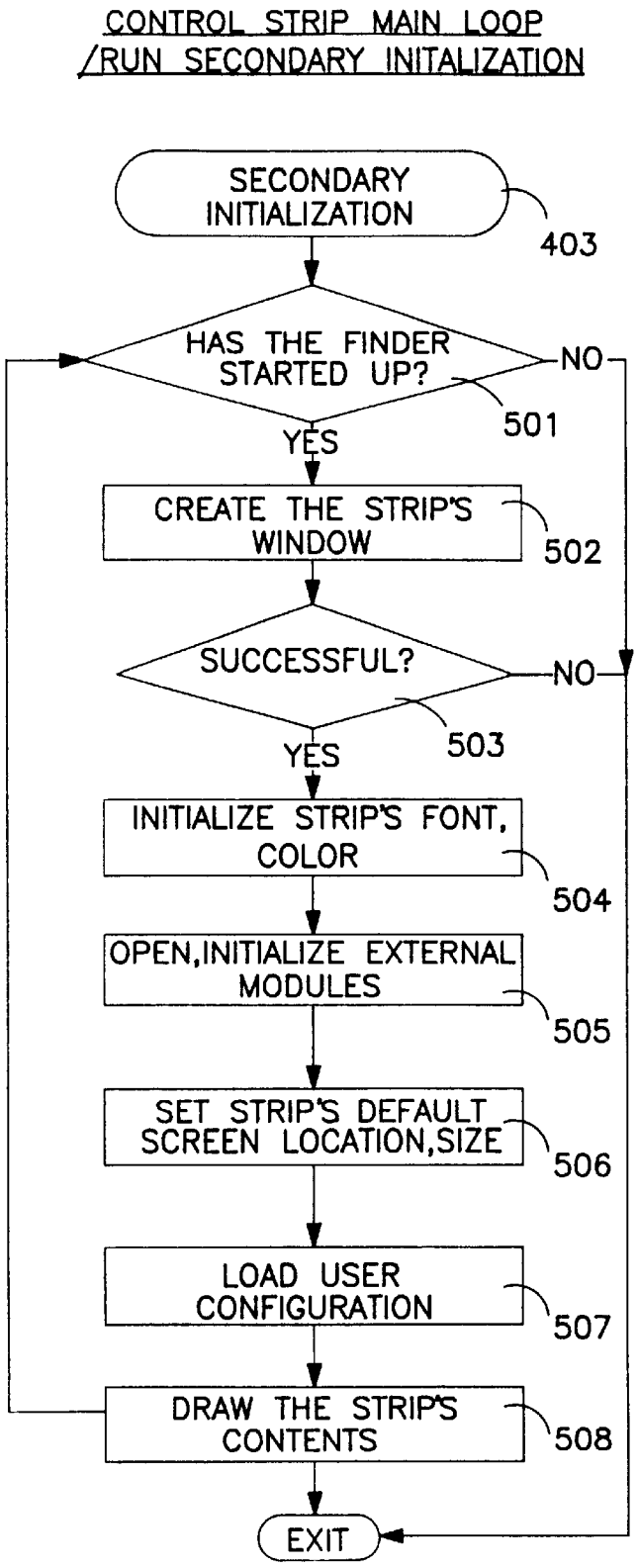

CONTROL STRIP MAIN LOOP
/RUN SECONDARY INITALIZATION
SECONDARY INITIALIZATION
403
HAS THE FINDER STARTED UP?
NO
501
YES
CREATE THE STRIP'S WINDOW
502
SUCCESSFUL?
NO
503
YES
INITIALIZE STRIP'S FONT, COLOR
504
OPEN,INITIALIZE EXTERNAL MODULES
505
SET STRIP'S DEFAULT SCREEN LOCATION,SIZE
506
LOAD USER CONFIGURATION
507
DRAW THE STRIP'S CONTENTS
508
EXIT

FIG. 5

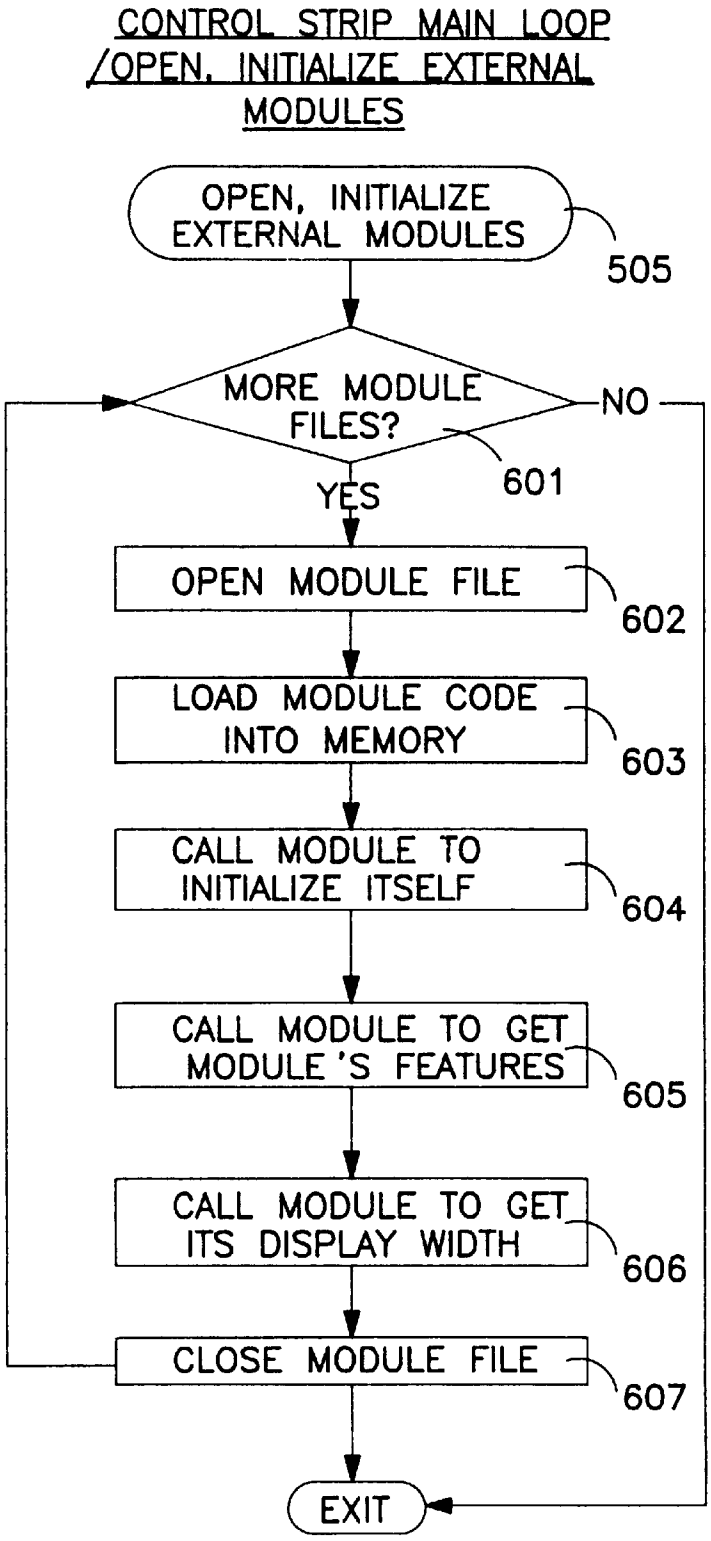

CONTROL STRIP MAIN LOOP /OPEN, INITIALIZE EXTERNAL MODULES
OPEN, INITIALIZE EXTERNAL MODULES
505
MORE MODULE FILES?
NO
YES
601
OPEN MODULE FILE
602
LOAD MODULE CODE INTO MEMORY
603
CALL MODULE TO INITIALIZE ITSELF
604
CALL MODULE TO GET MODULE'S FEATURES
605
CALL MODULE TO GET ITS DISPLAY WIDTH
606
CLOSE MODULE FILE
607
EXIT

FIG. 6

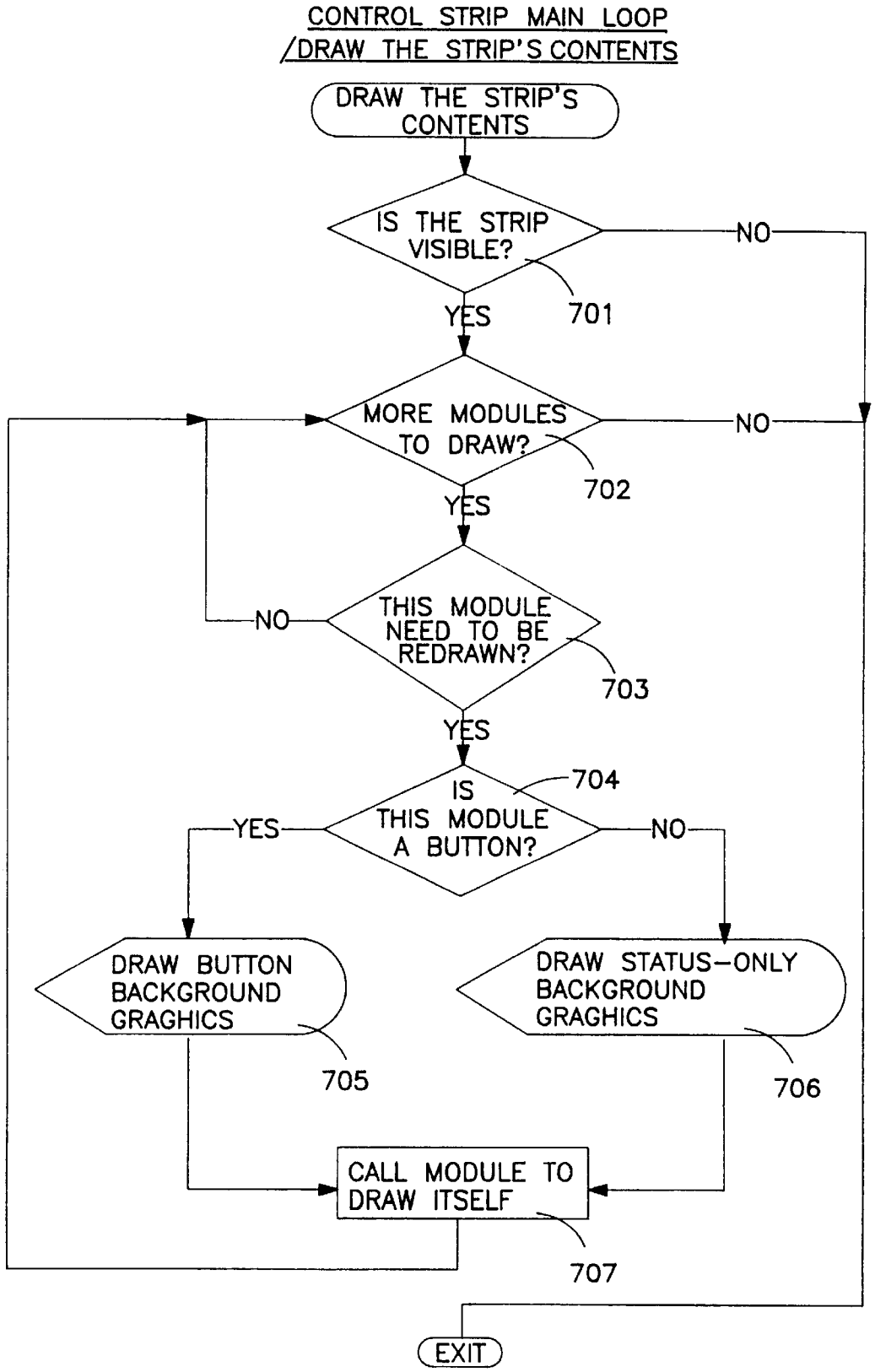

CONTROL STRIP MAIN LOOP
/DRAW THE STRIP'S CONTENTS
DRAW THE STRIP'S CONTENTS
IS THE STRIP VISIBLE?
NO
YES
701
MORE MODULES TO DRAW?
NO
YES
702
THIS MODULE NEED TO BE REDRAWN?
NO
YES
703
IS THIS MODULE A BUTTON?
704
YES
NO
DRAW BUTTON BACKGROUND GRAGHICS
705
DRAW STATUS-ONLY BACKGROUND GRAGHICS
706
CALL MODULE TO DRAW ITSELF
707
EXIT

FIG. 7

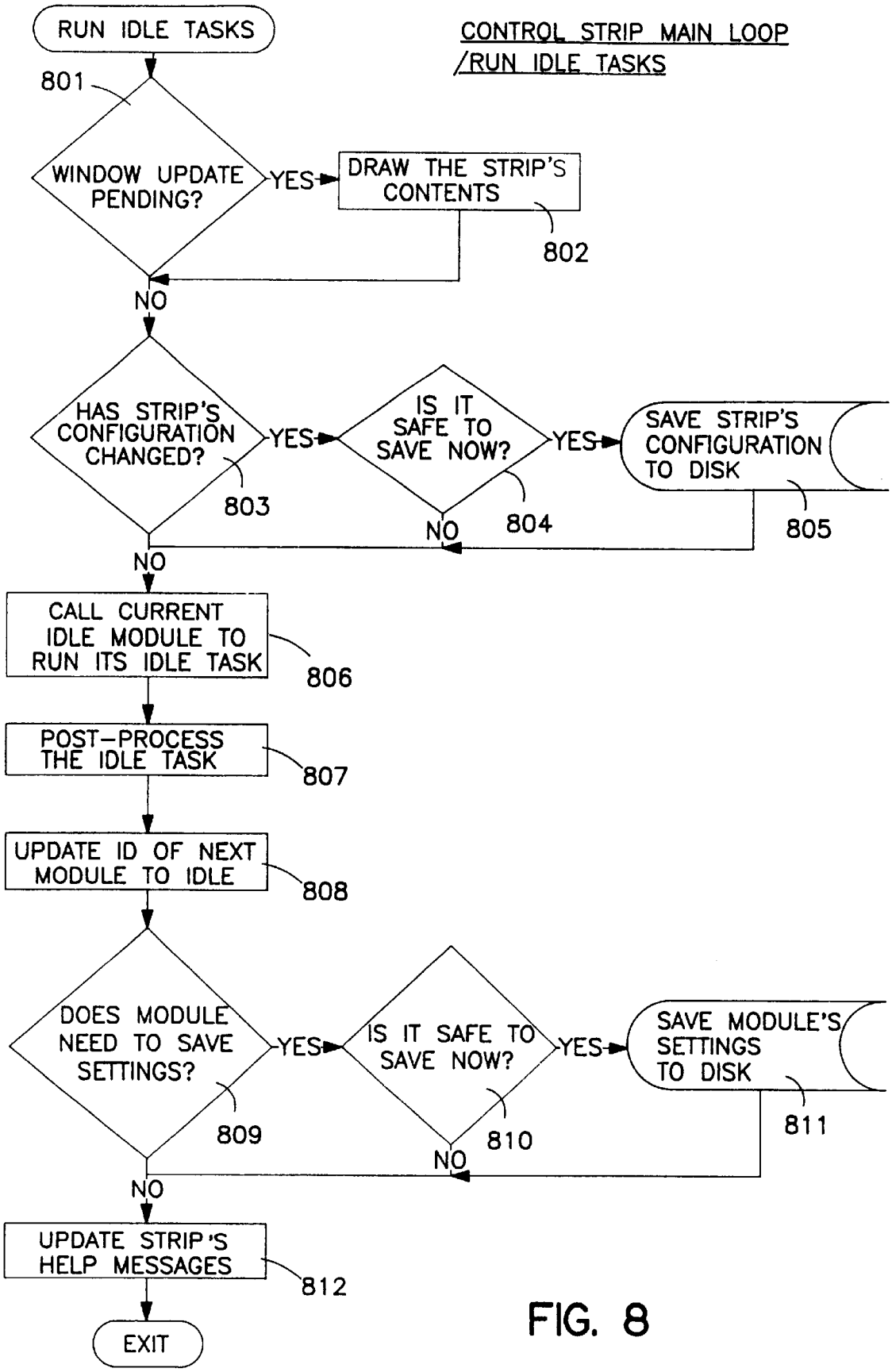

RUN IDLE TASKS
CONTROL STRIP MAIN LOOP /RUN IDLE TASKS
801
WINDOW UPDATE PENDING?
YES
DRAW THE STRIP'S CONTENTS
802
NO
HAS STRIP'S CONFIGURATION CHANGED?
YES
IS IT SAFE TO SAVE NOW?
YES
SAVE STRIP'S CONFIGURATION TO DISK
803
NO
804
805
NO
CALL CURRENT IDLE MODULE TO RUN ITS IDLE TASK
806
POST-PROCESS THE IDLE TASK
807
UPDATE ID OF NEXT MODULE TO IDLE
808
DOES MODULE NEED TO SAVE SETTINGS?
YES
IS IT SAFE TO SAVE NOW?
YES
SAVE MODULE'S SETTINGS TO DISK
809
NO
810
811
NO
UPDATE STRIP'S HELP MESSAGES
812
EXIT

FIG. 8

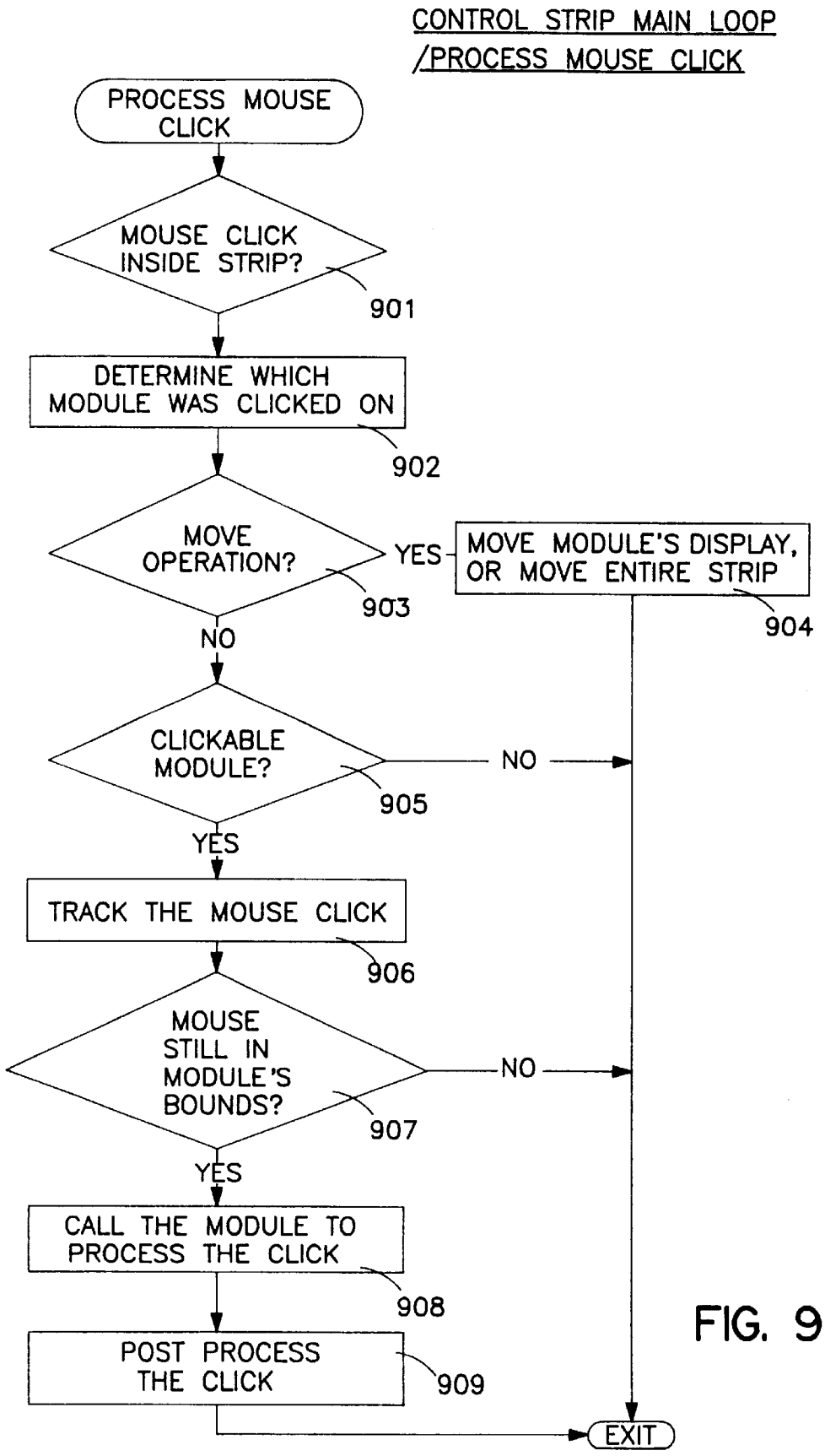

CONTROL STRIP MAIN LOOP
/PROCESS MOUSE CLICK
PROCESS MOUSE CLICK
MOUSE CLICK INSIDE STRIP?
901
DETERMINE WHICH MODULE WAS CLICKED ON
902
MOVE OPERATION?
YES
MOVE MODULE'S DISPLAY, OR MOVE ENTIRE STRIP
903
904
NO
CLICKABLE MODULE?
NO
905
YES
TRACK THE MOUSE CLICK
906
MOUSE STILL IN MODULE'S BOUNDS?
NO
907
YES
CALL THE MODULE TO PROCESS THE CLICK
908
POST PROCESS THE CLICK
909
EXIT

FIG. 9

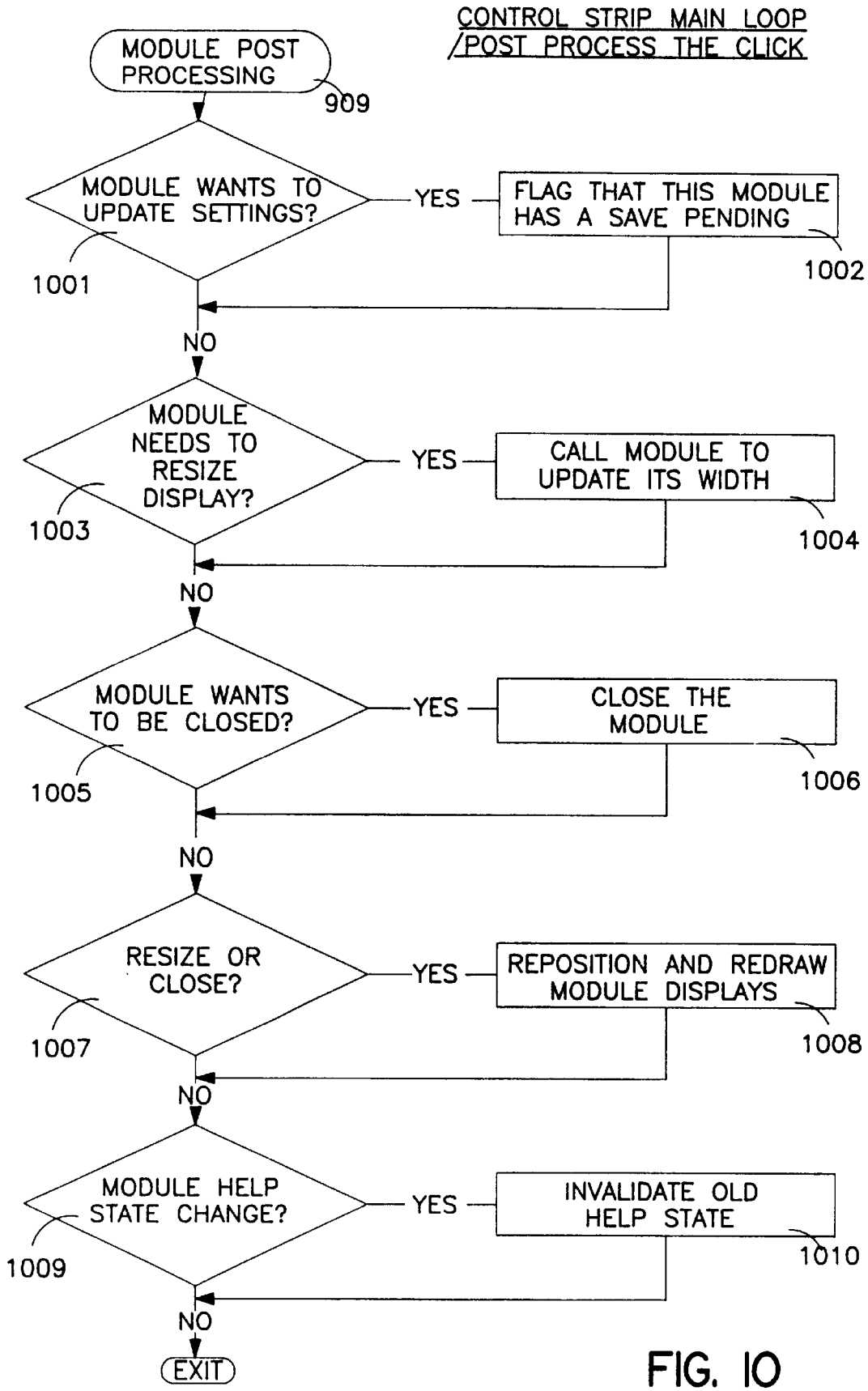

CONTROL STRIP MAIN LOOP
/POST PROCESS THE CLICK
MODULE POST PROCESSING
909
MODULE WANTS TO UPDATE SETTINGS?
1001
YES
FLAG THAT THIS MODULE HAS A SAVE PENDING
1002
NO
MODULE NEEDS TO RESIZE DISPLAY?
1003
YES
CALL MODULE TO UPDATE ITS WIDTH
1004
NO
MODULE WANTS TO BE CLOSED?
1005
YES
CLOSE THE MODULE
1006
NO
RESIZE OR CLOSE?
1007
YES
REPOSITION AND REDRAW MODULE DISPLAYS
1008
NO
MODULE HELP STATE CHANGE?
1009
YES
INVALIDATE OLD HELP STATE
1010
NO
EXIT

FIG. 10

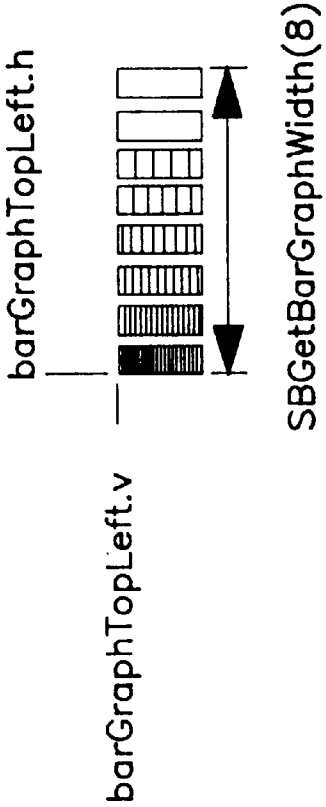

barGraphTopLeft.h
barGraphTopLeft.v
SBGetBarGraphWidth(8)

FIG. 11

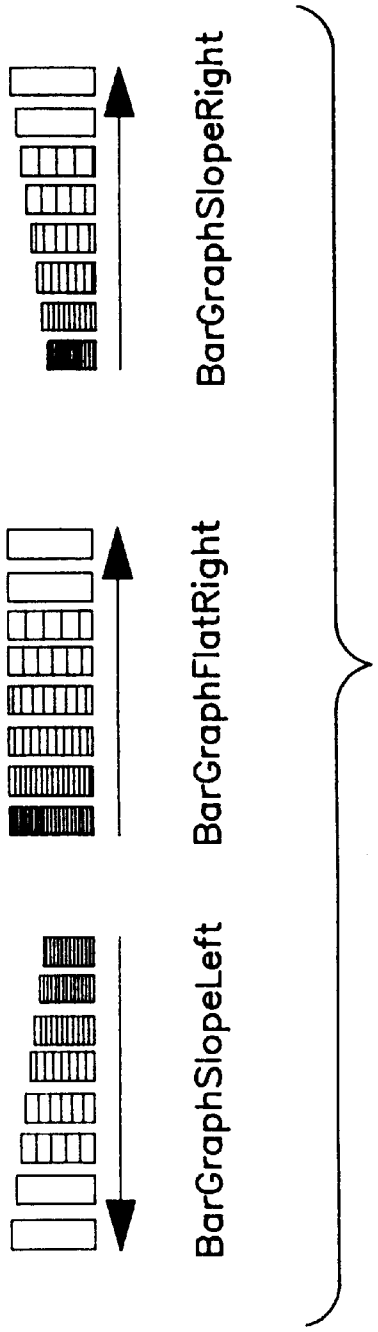

BarGraphSlopeRight
BarGraphFlatRight
BarGraphSlopeLeft

FIG. 12

METHOD AND APPARATUS FOR DISPLAYING AND ACCESSING CONTROL AND STATUS INFORMATION IN A COMPUTER SYSTEM

This is a continuation of application Ser. No. 08/316,237, filed Sep. 30, 1994, now abandoned.

FIELD OF THE INVENTION

The present invention relates to the field of computer systems; particularly, the present invention relates to displaying a status and control function bar or window to enable access of user selected indicia to a computer system user.

BACKGROUND OF THE INVENTION

Typically, a computer system contains a processor, a bus, and other peripheral devices. The processor is responsible for executing instructions using the data in the computer system. The bus is used by the processor and the peripheral devices for transferring information between one another. The information on the bus usually includes data, address and control signals. The peripheral devices comprise storage devices, input/output (I/O) devices, etc.

Computer systems also include information management systems that coordinate the display of information to the user. Currently, the art in computer display management provides the capability of displaying data in rectangular portions (commonly referred to as windows) of a display screen. Such information management systems include the Finder™ interface of the computer systems manufactured by Apple Computer, Inc. of Cupertino, Calif. Controls are typically provided to resize and move windows within the confines of the physical display boundaries.

Windows may be used to display information regarding application programs, as well as information produced by system programs, that are run on the computer system. Many of these system and control programs provide status and control information and functionality. Some of the system control programs also provide options with respect to the information they provide and the functions they perform. These options can be accessed and/or selected by moving a cursor at a predetermined point in the window and "clicking" a mouse or performing requisite key strokes. Access to these programs may require locating the program (e.g., locating and entering a folder) before execution. The time necessary to access such programs may be unduly long. It is desirable to provide a less obtrusive manner of accessing such system and control programs.

The computer system is often capable of displaying multiple windows or data areas on the display screen at the same time. Windows may overlap each other. The information contained in the portion of the window that is overlapped is not visible. The window that is entirely visible to the computer user is typically the active window. Therefore, a program, such as a system or control program may be running, while another program displaying information in another window is selected as active and thereafter covers, partially or completely, the windows or data areas displayed by the system/control program. Sometimes the user may wish to have an unobstructed view of the system/control data area, regardless of the window selected as active (even when the windows overlap each other). Thus, it is desirable at times to have windows that are always visible to the user. However, it is also desirable to be able to eliminate that window at times based on the user's requirements.

The present invention overcomes these problems by providing a status and control information display. The display of the present invention is in an easily accessible format. Also, the display may be configured to permanently display in a visible manner control and status indicia.

SUMMARY OF THE INVENTION

An interactive computer-controlled display system is described. In the present invention, the display system includes a processor, a data display screen, and a cursor control device for interactively positioning a cursor on the data display screen. The present invention also includes a window generator that generates and displays a window (e.g., a control strip) on a data display screen. In one embodiment, the window comprises a control and/or status window for display on the desktop of the computer system The window displays graphics depicting at least one display area of indicia. The individual data areas may be controlled through the use of controls and indicators in the window itself using cursor control keys.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

FIG. **1** is a block diagram of one embodiment of the computer system of the present invention.

FIG. **2**A displays a computer desktop illustrating the control strip of the present invention as well as opened windows.

FIG. **2**B illustrates one embodiment of the control strip of the present invention.

FIG. **2**C illustrates a pop-up menu displayed from the control strip of the present invention.

FIG. **2**D illustrates a help balloon displayed from the control strip of the present invention.

FIG. **2**E illustrates the process of moving a display area from one position to another in the control strip.

FIG. **2**F shows the control strip window graphics generated by processing logic being combined with graphics generated by a module to illustrate the creation of the resulting control strip.

FIG. **3** illustrates one embodiment of the control panel of the present invention.

FIG. **4** is a flow chart of one embodiment of the process for the control strip of the present invention.

FIG. **5** is a flow chart of one embodiment of the secondary initialization process of the present invention.

FIG. **6** is a flow chart of one embodiment of the open and initialization process for the external modules of the present invention.

FIG. **7** is a flow chart of one embodiment of the process for drawing the contents of the control strip of the present invention.

FIG. **8** is a flow chart of one embodiment of the process for running idle tasks in the sent invention.

FIG. **9** is a flow chart of one embodiment of the process for responding to a mouse click occurring in the control strip of the present invention.

FIG. **10** is a flow chart of one embodiment of the process for post processing a mouse click in the present invention.

FIG. **11** illustrates a bar graph for display in a data display area in the control strip of the present invention.

FIG. **12** illustrates a bar graph that results after using arrow direction icons.

DETAILED DESCRIPTION OF THE INVENTION

A method and apparatus for providing status and control indicia. In the following detailed description of the present invention numerous specific details are set forth, such as types of status indicia, instruction names, etc., in order to provide a thorough understanding of the present invention. However, it will be apparent to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in black diagram form, rather than in detail, in order to avoid obscuring the preset invention.

Some portions of the detailed descriptions which follow are presented in terms of algorithms and symbolic representations of operations on data bits within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like.

It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the following discussions, it is appreciated that throughout the present invention, discussions utilizing terms such as "processing" or "computing" or "calculating" or "determining" or "displaying" or the like, refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system's registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

The present invention also relates to apparatus for performing the operations herein. This apparatus may be specially constructed for the required purposes, or it may comprise a general purpose computer selectively activated or reconfigured by a computer program stored in the computer. The algorithms and displays presented herein are not inherently related to any particular computer or other apparatus. Various general purpose machines may be used with programs in accordance with the teachings herein, or it may prove convenient to construct more specialized apparatus to perform the required method steps. The required structure for a variety of these machines will appear from the description below. In addition, the present invention is not described with reference to any particular programming language. It will be appreciated that a variety of programming languages may be used to implement the teachings of the invention as described herein.

OVERVIEW OF THE COMPUTER SYSTEM OF THE PRESENT INVENTION

Referring to FIG. **1**, an overview of a computer system of the present invention is shown in block diagram form. The present invention may be implemented on a general purpose microcomputer, such as one of the members of the Apple family of personal computers, one of the members of the IBM personal computer family, or one of several other computing and assistant devices which are presently commercially available. Of course, the present invention may also be implemented on a multi-user system while encountering all of the costs, speed, and function advantages and disadvantages available with these machines. The preferred embodiment of the present invention is implemented on an Apple PowerBook™ computer system developed by the assignee of the present invention.

As illustrated in FIG. **1**, the computer system of the present invention generally comprises a local bus or other communication means **100** for communicating information, a processor **103** coupled with local bus **100** for processing information, a random access memory (RAM) or other dynamic storage device **104** (commonly referred to as a main memory) coupled with local bus **100** for storing information and instructions for processor **103**, and a read-only memory (ROM) or other non-volatile storage device **106** coupled with local bus **100** for storing non-volatile information and instructions for processor **103**.

The computer system of the present invention also includes an input/output (I/O) bus or other communication means **101** for communication information in the computer system. A data storage device **107**, such as a magnetic tape and disk drive, including its associated controller circuitry, is coupled to I/O bus **101** for storing information and instructions. A display device **121**, such as a cathode ray tube, liquid crystal display, etc., including its associated controller circuitry, is also coupled to I/O bus **101** for displaying information to the computer user, as well as a hard copy device **124**, such as a plotter or printer, including its associated controller circuitry for providing a visual representation of the computer images. Hard copy device **124** is coupled with processor **103**, main memory **104**, non-volatile memory **106** and mass storage device **107** through I/O bus **101** and bus translator/interface unit **140**. A modem **108** and an ethernet local area network **109** are also coupled to I/O bus **101**.

Bus interface unit **140** is coupled to local bus **100** and I/O bus **101** and acts as a gateway between processor **103** and the I/O subsystem. Bus interface unit **140** may also provide translation between signals being sent from units on one of the buses to units on the other bus to allow local bus **100** and I/O bus **101** to co-operate as a single bus.

An I/O controller **130** is coupled to I/O bus **101** and controls access to certain I/O peripherals in the computer system. For instance, I/O controller **130** is coupled to controller device **127** that controls access to an alpha-numeric input device **122** including alpha-numeric and other

keys, etc., for communicating information and command selections to processor **103**, a cursor control **123**, such as a trackball, stylus, mouse, or trackpad, etc., for controlling cursor movement, and a temperature sensor **127**A for measuring the internal system temperature. The system also includes a sound chip **125** coupled to I/O controller **130** for providing audio recording and play back. Sound chip **125** may include a sound circuit and its driver which are used to generate various audio signals from the computer system. I/O controller **130** may also provide access to a floppy disk and driver **126**. The processor **103** controls I/O controller **130** with its peripherals by sending commands to I/O controller **130** via local bus **100**, interface unit **140** and I/O bus **101**.

Batteries or other power supply **152** may also be included to provide power necessary to run the various peripherals and integrated circuits in the computer system. Power supply **152** is typically a DC power source that provides a constant DC power to various units, particularly processor **103**. Various units such as processor **103**, display **121**, etc., also receive clocking signals to synchronize operations within the computer systems. These clocking signals may be provided by a global clock generator or multiple clock generators, each dedicated to a portion of the computer system. Such a clock generator is shown as clock generator **160**. In one embodiment, clock generator **160** comprise a phase-locked loop (PLL) that provides clocking signals to processor **103**.

In one embodiment, processor **103** is a member of the 68000 family of processors, such as the 68040 processor manufactured by Motorola Corporation of Schaumberg, Ill. The memory in the computer system is initialized to store the operating system as well as other programs, such as file directory routines, control programs, system programs and application programs, and data inputted from I/O controller **130**. The operating system running on processor **103** takes care of basic tasks such as starting the system, handling interrupts, moving data to and from memory **104** and peripheral devices via input/output interface unit **140**, and managing the memory space in memory **104**. In one embodiment, the operating system is stored in ROM **106**, while RAM **104** is utilized as the internal memory for the computer system for accessing data and application programs.

Processor **103** accesses memory in the computer system via an address bus within bus **100**. Commands in connection with the operation of memory in the computer system are also sent from the processor to the memory using bus **100**. Bus **100** also includes a bi-directional data bus to communicate data in response to the commands provided by processor **103** under the control of the operating system running on it.

Of course, certain implementations and uses of the present invention may neither require nor include all of the above components. For example, in certain implementations a keyboard or cursor control device for inputting information to the system may not be required. Furthermore, the computer system may include additional processing units.

OVERVIEW OF THE PRESENT INVENTION

The present invention provides a control and/or status window for display on the desktop of the computer system. The control and status window will be referred to herein as the control strip. The control strip of the present invention is a window of graphics depicting one or more display areas for control and/or status indicia. In one embodiment, each of the display areas is individually and variably sized. The size of the control strip itself may also be variably sized. In one embodiment, the size may be adjusted such that none, all, or only a portion of the display areas within its boundaries are visible. The size of the control strip may also be varied such that only a portion of one display area is visible in the control strip. FIG. **2**A displays a computer desktop illustrating the control strip of the present invention as well as opened windows. FIG. **2**B illustrates one embodiment of the control strip of the present invention.

Each of the variably sized data areas may be sensitive to user input for control. That is, a user may interact with the individually display data areas. Different parts of the control strip either display information or act as buttons, or both. Note that buttons may display information on their surface. When the user clicks a button, it is highlighted. In one embodiment, buttons may also display additional elements such as pop-up menus (shown in FIG. **2**C) or help messages (e.g., balloons shown in FIG. **2**D). Thus, in one embodiment, control of the individual data areas is accomplished, in part, through the use of small button controls and indicators in the form of various icons.

Each of the display areas is associated with a programming module. Each of the modules provides a specific status or control function. In one embodiment, the module is represented by a disk file containing the code necessary for the module to interact with the control strip as well as other elements such as text, icons, pictures, etc. Modules may be designed to be responsive to selection from cursors via a mouse, trackpad, or cursor control keys, such as on a keyboard. Many of the modules are able to provide control to various system functionality, and may provide menus to do the same.

The control strip is a control panel that provides the operating environment for control strip modules. In one embodiment, the control strip runs on any Macintosh™ computer using a System 7.0 or later operating system. The control strip of the present invention may be designed to run on computer systems using other operating systems.

In one embodiment, the control strip is implemented in a private window layer that appears in front of the windows of all the application layers. That is, the control strip window appears on top of all application programming windows that may be generated as part of the execution of an application program. This prevents other windows from obscuring it. In one embodiment, processing logic in the computer system may maintain a list of windows ordered from the frontmost window on the screen being at the top of the list and the bottommost window being at the bottom of the list. Processing logic can maintain the control strip window at the top of the list.

The control strip of the present invention may include windowing configurations that are shown as being horizontal or vertical on the screen. Furthermore, the present invention is not limited to a single row or column of status and control data areas. In other words, multiple rows and columns of module data areas may be included in the window of the control strip.

CONTROL STRIP MANIPULATION

The control strip, such as shown in FIG. **2**B, may also be moved to different portions of the display screen. However, in one embodiment, the window for the control strip may be moved to any location on the display as long as the right and/or left edge of the strip is attached to the right or left edge, respectively, of the display. The user may also hold

down the option key and drag the tab **203** of the control strip **200** with the use of a cursor control device (e.g., trackpad, trackball, mouse) to move the control strip to a new position on the display.

In one embodiment, the user may adjust the size of the control strip window. Adjustments to the size of the window may comprise either an increase in the height of the window, the width of the window, or both. In one embodiment, only the width of the control strip window may altered. The definition and use of windows is well-known in the art. In one embodiment, the control strip **200** has a tab **203** on its unattached end. The user can drag tab **203** to adjust the length of the strip. By "clicking" on tab **203**, i.e. selection through the use of the trackpad, mouse, cursor control keys, etc., the user is able to shift from a minimal control strip size to a maximum control strip size, and vice versa. In its minimal size, the graphics of the modules in the control strip are not visible and only the tab is showing. In its maximum size, all of the modules in the control strip are showing. Recognizing cursor controlled selections through the use of trackpad, trackball, mouse, cursor control keys, etc., as well as the tracking of movements of the cursor made by the same are well-known in the art.

Scroll arrows, such as left scroll arrow **204** and right scroll arrow **205**, are provided on the control strip that enable the window of the control strip to be scrolled to the left or right, respectively. Use of scroll arrows with windows is well-known in the art.

The user may also hide the control strip. In one embodiment, to make the control strip disappear completely, the user can click the Hide button in the control strip control panel, as described later in conjunction with FIG. **3**. A close box **201** is also included in control strip **200** In one embodiment, by holding the option key and clicking a display area, the user can drag the display area to another position in the control strip. An example of the process of moving one display area to another position on the control strip is shown in FIG. **2**E. Referring to FIG. **2**E, the user selects one of the display areas by, for instance, positioning the cursor over the display area. When the user "clicks" the display area, its border becomes highlighted. While clicking, the display area is dragged to another location in the control strip module display area. When the user has moved the display area to the location of his choice, the user stops "clicking" The control strip display areas are then rearranged.

After the user rearranges the parts of the control strip, the new arrangement is saved. The saving operation may be deferred until resources, such as the hard disk is ready (e.g., spinning) or until just before the computer system is shut down or restarted. In other words, in computer systems in which the hard disk is not turned on all the time in order to save power, the saving operation may be deferred until the hard disk has been turned on by another.

EXEMPLARY CONTROL/STATUS INFORMATION

The control strip of the present invention provides a standard screen location for a collection of individual modules that provide status and control functions. In one embodiment, the control strip functions include a network switch that shows whether a network connection for the computer system, such as an AppleTalk™ network connection, is on or off and lets the user turn the network connection on or off without having to locate and execute other network connection software on the computer system (e.g., without having to open the Chooser™).

The control strip may also include a battery monitor that displays the status of the battery or batteries. In one embodiment, the battery monitor displays the current power drain in a manner similar to a car's miles per gallon (MPG) indicator. The needle for the power drain indicator indicates the drain relative to the maximum possible. The control strip of the present invention allows this display to be updated frequently so if the user increased the LCD display screen's brightness level, the needle would animate to denote the consequence of the action.

Another control strip module displays the state of File Sharing (e.g., on, off, or users connected) that may be currently employed on the computer system. The file sharing module also lets the user turn file sharing on or off and lets the user open a control panel to control processing to setup file sharing on the computer system.

The control strip of the present invention may also provide a module to allow the internal hard disk power to be turned off (to save power), and to indicate whether is currently on or off.

The control strip may also provide power settings that allow the user to select between maximum battery conservation or maximum computer performance without opening a control panel. In one embodiment, the power settings portion of the control strip also allow the user to open up the power savings control panel. The control strip of the present invention may also include a function that places the computer in sleep mode or allows the user to select the sound volume.

Other modules, for example, may provide time and/or date information, may list currently running programming applications, may indicate the amount of available memory, may control a CD drive, may provide access to audio controls and status information. Therefore, the control strip acts as a status and control function bar, or windowing area, that provides running modules to be displayed in an arrangement that is to be displayed, such an arrangement being modifiable such that the size of the window or bar may be changed.

In one embodiment, the control strip is controllable through a control panel. An exemplary display of such control panel is shown in FIG. **3**. Use of control panels is well-known in the art. Using the control panel in FIG. **3**, the user is able to hide or show the control strip by clicking the corresponding button in the control panel. Note that in one embodiment, the control panel may also be used to change the font and size of the text in the control strip window.

PROCESSING LOGIC FOR THE PRESENT INVENTION

The present invention includes computer processing logic for generating the control strip of the present invention. This processing logic is described, in part, in the flow charts shown in FIGS. **4–10**. In addition to the computer resources described earlier, the present invention relies upon the availability of an operating system and system functions capable of displaying windows, information in windows, characters, and cursor symbols on the display devices. System functions for interfacing with the cursor control devices and cursor function keys, including the tracking of cursor location within a window, are also required. These resources are standard processing components known in the art.

When the processor of the present invention is first powered up, the operating system logic obtains control and initializes the system components such as read/write

memory, the display device, the cursor control device, the cursor function keys, and keyboard. During this initialization process or in response to a user command, the operating system displays the control strip of the present invention.

In one embodiment, the control strip initialization is performed in two stages. The first stage begins by initially loading at least one routine at start up. Upon loading necessary routines, the operating system allocates storage for global variables use. Next, resources are loaded for use by the control strip processing logic. These resources include the visual components or indicia that is to appear in the control strip, such as pictures, icons, text, etc. The processing logic for the control strip is patched into the operating system.

Later, as a second stage of the initialization during the set up process, the control strip processing logic causes each of the module files to be opened one at a time. The code for the module is loaded. An initialization routine is run in response to a call, during which time, the module itself determines if it can run. This information is conveyed to the control strip. The processing logic then causes the window to be displayed and calls the modules to run themselves and appear in the control strip. FIG. **2F** shows the control strip window graphics generated by processing logic being combined with graphics generated by a module to illustrate the creation of the resulting control strip.

FIG. **4** is a flowchart of the processing logic responsible for generating the control strip of the present invention and processing events that occur involving the control strip. In one embodiment, the control strip main processing is called by the operating system. Referring to FIG. **4**, the processing begins by saving the previous context and sets up the context of the control strip (processing block **401**). The previous context refers to the state of the computer system prior to performing control strip processing. The previous context may correspond to an application program running immediately prior to the control strip processing being called. The context may include settings up its memory space, providing access to its global variables, etc.

Next, a test determines if the secondary initialization has been done (processing block **402**). If the secondary initialization has not been done, processing continues at processing block **403** where a secondary initialization process is run, and processing thereafter continues at processing block **409**. The secondary initialization process causes the processing logic to initialize the control strip. One embodiment of the secondary initialization process is described in FIG. **5**. On the other hand, if the window of the control strip is allocated, processing continues at processing block **404** where the processing logic awaits a user event and determines the type of such an event.

Then a test determines if the user event type is a null event (processing block **405**). That is, a test determines whether the user event type is idle or not. If the user event type is a null event, processing continues at processing block **406** where idle tasks are run, and processing thereafter continues at processing block **409**. Thus, during idle periods, tasks involved with the control strip window may be run as well as tasks of the modules. Examples of module tasks may include updating help messages (e.g., due to a help feature being enabled on the computer system) and saving updated state information (e.g., display area on screen moved to new location, display area resized, module made invisible; module indicates state is changed and that it must be saved). On the other hand, if the event type is not a null event, processing continues at processing block **407**.

At processing block **407**, a test determines if there has been a "click" of the mouse within the area defined by the control strip. If there has been a click of the mouse within the control strip, the mouse click is processed (processing block **408**), and processing continues at processing block **409**. The mouse click processing determines the location of the mouse click, which module in the controls strip was selected, or "clicked-on", if any, and any action to be taken based on that location. One embodiment of the mouse click processing is described in FIG. **9**. If a mouse click has not occurred within the control strip, processing continues directly to processing block **409**.

At processing block **409**, the previous context is restored and the processing logic exits to return control to the operating system.

One embodiment of the secondary initialization process called by the control strip main processing logic is described in a flowchart in FIG. **5**. Referring to FIG. **5**, the secondary initialization process begins by testing whether the Finder™ has started up (processing block **501**). If the Finder™ has not started up, the secondary initialization process ends. However, if the Finder™ has started up, the window of the control strip is created (processing block **502**).

Then a test determines whether the creation of the window of the control strip was successful (processing block **503**). If the creation of the window of the control strip was not successful, the process ends. The creation of the window may not be successful because, for instance, there is not enough memory, missing system resources, etc. On the other hand, if the creation of the window of the control strip was successful, the font and color of the control strip are initialized (processing block **504**). Then external modules are opened and initialized (processing block **505**), the default screen location and size of the control strip are set (processing block **506**), the user configuration is loaded (processing block **507**), and the contents of the control strip are drawn (processing block **508**). The user configuration may include screen location for the control strip, the saved display order of the modules, the window size of the control strip, etc. Then the secondary initialization process ends.

The default screen location and size of the control strip are stored in memory and accessed. In one embodiment, these values may be changed by the computer user, such as by interacting with the control strip itself. In another embodiment, the default values are determined and permanently set by a system designer. Note that specification of the font, color, default screen location and size may not be required in lieu of the user configuration. Likewise, by using solely the default settings, the user configuration is not required.

One embodiment of the process for opening and initializing external modules such as may be invoked by the secondary initialization process is shown in FIG. **6**. Referring to FIG. **6**, the processing logic begins by testing whether there are more module files to be opened (processing block **601**). If there are no more module files (e.g., all the modules have been opened and loaded), then the process ends. The modules are opened and initialized one at a time. If there are more module files, the processing logic opens the module file (processing block **602**) and loads the module code into memory (processing block **603**). The processing logic calls the module to initialize itself (processing **604**). The module is then also called by the processing logic to obtain the features of the module (processing block **605**) and to obtain the width of the module's area, as well as features of the module (processing block **606**). The features of the module

include help messages to be displayed when the module is "clicked on" with the cursor. Then the module file is closed (processing block **607**) and the processing loops back to processing block **601**. By looping back to processing block **601**, the processing logic is able to provide the initialization procedures to all the modules, such that when all the modules have been processed the process ends. When the process ends, it returns in a manner well-known in the art to the processing logic that called (e.g., initiated) it.

One embodiment of a process for drawing the contents of the control strip (processing block **508**), such as used at processing block **508** of FIG. **5**, is described in FIG. **7**. Referring to FIG. **7**, the processing logic initially determines if the control strip is visible (processing block **701**). If the control strip is not visible, processing ends. That is, if the user has hidden the control strip, the present invention will not draw its contents.

On the other hand, if the control strip is visible, processing continues at processing block **702** enters a looping structure where the processing logic tests whether there are more modules to draw. If there are no more modules to draw, processing ends and control returns to the process that called it. If there are more modules to draw, processing continues at processing block **703** where the processing logic tests whether the particular module needs to be redrawn. A module may need to be drawn when the information being displayed needs to be updated. For example, as the amount of energy in the battery is changing due to energy consumption from the computer system, an update to the battery indicia in the control strip must be made. If the module does not need to be redrawn, processing loops back to processing block **702** where the more modules test is repeated. On the other hand, if the module needs to be redrawn processing continues at processing block **704** where the processing logic determines whether the module is a button. If the module is a button, processing continues at processing block **705** where the background graphics of the button are drawn, and processing continues to processing block **707**. If the module is not a button, the status-only background graphics are drawn (processing block **706**) and processing continues at processing block **707**. Note that in one embodiment, the type of background graphics may be obtained using a message sent to the module requesting its features.

At processing block **707**, the processing logic makes a call to the module to draw itself. That is, it is the responsibility of the module itself to draw its status for control indicia. Thereafter processing loops back to processing block **702**.

One embodiment of the processing for running idle tasks in FIG. **4** is described in a flowchart in FIG. **8**. Referring to FIG. **8**, the processing logic begins by determining whether a window update is pending (processing block **801**). If a window update is pending, the processing continues at processing block **802** when the contents of the control strip are drawn. Window updates may be required due to a change in status in one of the modules. Changes may also be due to a reordering of the control strip entries. Note that one embodiment of the process to draw the control strip is shown in FIG. **7**. Thereafter processing continues at processing block **803**. If a window update is not pending processing continues directly to processing block **803**.

At processing **803**, processing logic tests whether the configuration of the control strip has changed. If the configuration of the control strip has not changed, processing continues at processing block **806**. If the configuration of the control strip has changed, processing continues at processing block **804** where a test determines whether it is safe to perform a save operation. This determination is based on whether the resources are available (i.e., H.D. is turned on) to perform the save operation. If it is not safe to perform a save operation, processing continues at processing block **806**. However, if it is safe to save control strip, processing continues at processing block **805** where the configuration of the control strip is saved to disk. Thereafter processing continues to processing block **806**.

At processing block **806**, the current idle module is called to run its idle task. In one embodiment, the processing logic of the present invention allows only one module to run its idle tasks during each a call to the processing of FIG. **8** (e.g., the currently designated module) to reduce overhead time. Identification of the current module is based on an ID associated with each of the modules.

Then the idle task undergoes post processing (processing block **807**), and the ID of the next module is updated to idle (processing block **808**). That is, the module designated as the current module for the next call to the processing of FIG. **8** will be the next module in the list of modules. An example of the post processing is shown in FIG. **10**.

Then, the processing logic determines whether the module needs to save settings for use later (processing block **809**). If the module does not need to save its settings, processing continues to processing block **812**. On the other hand, if the settings of the module are to be saved, the processing logic tests whether the settings may be saved at this time (processing block **810**). One reason the settings may not be saved is that the hard disk may be powered down or turned off. If the settings cannot be saved at this time, processing continues at processing block **812**. If the settings of the module may be saved at this time, the processing logic causes the module settings to be saved to disk (processing block **811**). Thereafter, processing continues at processing block **812**.

At processing block **812**, the help messages for the control strip are updated, and processing ends and returns to the control of the main processing logic.

One embodiment of the mouse click processing of the present invention, such as used in FIG. **4**, is shown in a flowchart in FIG. **9**. Referring to FIG. **9**, the processing logic determines whether a mouse click has occurred inside the control strip (processing block **901**). In one embodiment, this determination may be made by comparing the current location of the cursor with the location of the control strip (e.g., status bar). If a mouse click has not occurred inside the control strip, then processing loops back upon itself, retesting repeatedly until a mouse click does occur. When a mouse click occurs within the control strip, processing continues at processing block **902** where a determination is made as to upon which module the cursor was during the click.

The processing logic then determines whether a move operation is being selected by the mouse (processing block **903**). If a move operation has been chosen, the display of the module is moved or the entire control strip is moved (processing block **904**) and the processing logic ends the mouse click process and exits to control of the processing logic that called this procedure. The determination of whether to move a module or the entire strip is based on the user's keystrokes or mouse movements. On the other hand, if a move operation is not to occur, processing continues at processing block **905**.

At processing block **905** the processing logic determines whether the module on which the click occurred is a "clickable" module, as opposed to a status only module, (processing block **905**). That is, the processing logic tests

whether the module provides any additional functionality when a mouse moves the cursor to select an element in the control strip. If the module is not "clickable," processing ends. If the module is clickable, processing continues to process processing block **906** where the mouse click is tracked, i.e., the location of the cursor.

Next, a test determines if the mouse is still within the bounds of the module (processing block **907**). If the mouse is not within the bounds of the module, processing ends. However, if the mouse is within the boundaries of the module, the module is called to process the click (processing block **908**) and the click undergoes post processing (processing block **909**). Thereafter, the process ends.

One embodiment of the process for post processing the mouse click is shown in FIG. **10**. Referring to FIG. **10**, the processing logic begins by determining whether the module desires to update its settings (processing block **1001**). If the settings for the module are to be updated, processing continues at processing block **1002** where a flag is set to indicate that the module has a save pending, and processing continues to processing block **1003**. The settings for a module may have to be updated due to user interaction, such as in the case of an option for a module being turned off or a module acquiring data as part of its functionality. If the settings of a module do not have to be updated, processing continues directly to processing block **1003**. Setting may need to be updated when the module is displaying information that is changing frequently.

At processing block **1003**, a test determines whether the module needs to resize the display. If the display of the module must be resized, processing continues at processing block **1004** where the module is called to update its width. Then processing continues at processing block **1005**. On the other hand, if the display of the module does not need to be resized, processing continues directly to processing block **1005**.

At processing block **1005**, the processing logic determines whether the module desires to be closed. If the module desires to be closed, processing continues at processing block **1006** where the module is closed immediately. Then processing continues at processing block **1007**. If the module does not desire to be closed, processing continues to processing block **1007**.

At processing block **1007**, a test determines whether the control strip is to be resized or closed. If the control strip needs to be resized or closed, processing continues at processing block **1008** where the module displays are repositioned and redrawn, and then processing continues at processing block **1009**. If the control strip is not to be resized or closed, processing continues directly to processing block **1009**.

At processing block **1009**, a test determines whether the help state of the module is to be changed. The help state refers to help messages that the modules provide to users generally. If the help state of the module is to be changed, processing continues at processing block **1010** where the old help state of the module is invalidated and the process ends. If the module help state does not need to be changed, processing ends. Changes to the help state may occur due to a global change in the computer system, such as when a particular help feature (e.g., help balloons) is enabled.

ADDING CONTROL STRIP MODULES

In one embodiment, the control strip of the present invention operates as a shell with individual control and status modules added. Each module and its icons, pictures, etc., are contained in a file on a disk. The control strip processing logic draws the strip which acts as the background for the individual modules. Each module is responsible for drawing the icons and other objects that make up its user interface.

Contents of Module Files

In one embodiment, the module file includes only a single resource containing the code necessary for the module to interact with the control strip. A module file may contain more than one code resource if it is to provide multifunctional support. In that case, each module in the file is loaded and initialized separately and treated as an independent entity.

Module Interface

The interface of the module to the control strip comprises a code resource. In one embodiment, using the Macintosh™ computer, the type of the code resource is 'sdev'. This code is responsible for performing all of the functions required by the control strip as well as any functions that are custom to the module itself. The module's entry point is at the beginning of the resource and is defined as

```
pascal long ControlStripModule (long message, long
    params, Rect *statusRect, GraftPtr statusport);
```

Interactions between a module and the control strip are managed by passing messages to the module to tell it what to do or to obtain information about the module and its capabilities. In one embodiment, each module is required to observe Macintosh™ Pascal register saving conventions; that is, it may trash 680x0 processor registers D**0**, D**1**, D**2**, A**0**, and A**1**, but must preserve all other registers across its call. Note that other operating systems and implementations of the present invention may have different restrictions.

The message field comprises a message number from the list in the section "Control Strip Module Messages" that indicates to the module the action to perform.

The params field signifies the result returned by the initialize call to the module. This would typically be a pointer to a pointer (e.g., the handle) to the private variables to be used by the module since modules cannot have global variables. This result is passed to the module on all subsequent calls. Note that in embodiments where modules can have global variables, such a field may be eliminated.

The statusRect field comprises a pointer to a rectangle within the control strip defining the area that a module may draw within.

The statusPort field specifies a pointer to the graphics port of the control strip. The graphics port may be either a color or black-and-white graphics port, and depends on the computer system on which the control strip is running.

The result value returned by the module varies depending on the message sent to it. Results for each message are described below in the sections on the individual messages.

CONTROL STRIP MODULE REFERENCE

In one embodiment, control strip modules interact with the control strip processing logic in three ways: by accepting messages, by calling utility routines, and by calling the operating system manager (e.g., a call to Gestalt selectors). The next three sections describe each of those interactions.

Control Strip Module Messages

In one embodiment, all control strip modules respond to messages from the control strip processing logic, which is responsive to user interaction with the control strip displayed on the screen. The following messages have been defined:

Message name	Message No.	Description
sdevlnitModule	0	Initialize the module
sdevCloseModule	1	Clean up before being closed
sdevFeatures	2	Return the feature bits
sdevGetDisplayWidth	3	Return the current width of the module's display
sdevPeriodicTickle	4	Periodic tickle when nothing else is happening
sdevDrawStatus	5	Update the interface in the control strip
sdevMouseClick	6	User has clicked on the module's display area
sdevSavSettings	7	Save any changed settings in the module's preferences file
sdevShowBalloonHelp	8	Display a help balloon, if the module has one

sdevInitModule

The sdevInitModule message is the first message sent to a module after the module has been loaded from its file. Initialization allows the module to initialize its variables and to determine whether it can run on a particular machine. For example, if the function of the module is to display battery information, it may be only able to run on a portable computer, such as the Powerbook manufactured by Apple Computer.

In response to receiving the sdevInitModule message, the module loads and detaches any resources (e.g., text, code, icons, etc.) in its resource file that will be used. Also, space is allocated in the global variables for handles to those detached resources.

The sdevInitModule message returns a result depending on its success at installing itself. In one embodiment, a positive result (≧0) indicates successful installation. The processing logic passes this result value to the module on all subsequent calls. A negative result indicates an error condition, and installation of the module is aborted by the control strip processing logic. Also if a negative result occurs and installation has been aborted, the module does not receive a close message.

sdevCloseModule

The sDevCloseModule message is sent to a module when it should be closed. In one embodiment, the module itself decides when to be closed. A module may be closed when it no longer is required to be running, such as when a battery level indicator no longer needs to be running when the computer system is receiving its power from an outlet. When the module receives this message, it disposes of all the detached resources it loaded as well as its global storage. No result is expected.

sdevFeatures

The sdevFeatures message queries the module for the features it supports. This message returns as its result a bitmap consisting of 1 bits for supported features and 0 bits for unsupported features. In one embodiment, there are 32 bits returned. All undefined bits are reserved for future features, and, in one embodiment, are set to 0. The bits are defined as:

a) sdevWantMouseClicks **(0)**—If this bit is set, the control strip notifies the module of mouse down events. If this bit is not set, the control strip assumes that the module only displays status information with no user interaction.

b) sdevDontAutoTrack **(1)**—If this bit is set, the control strip highlights the display of the module and then calls the module to perform mouse tracking. In one embodiment, this bit is set when, for example, a module has a pop-up menu associated with it. If this bit is cleared, the control strip tracks the cursor until the mouse button is released, then sends an sdevMouseClick message, described below, to the module to notify it that there was a mouse-down event.

c) sdevHasCustomHelp **(2)**—If this bit is set, the module is responsible for displaying its own help messages. These help messages may be customized depending on its current state. If the bit is cleared, the control strip displays a generic help message when the cursor passes over the its display area and Balloon Help, or other help-based information provider, is on.

d) sdevKeepModuleLocked **(3)**—If this bit is set, the code of the module is kept locked and protected. In one embodiment, this bit is set only if the module is passing the address of one of its routines to a routine external to the module (e.g., installing itself in a queue).

sdevGetDisplayWidth

The sdevGetDisplayWidth message is sent to a module to determine how much horizontal space (in pixels) its display currently requires on the control strip. In response to the message, the module return the number of pixels as its result. In one embodiment, the returned width does not comprise the maximum width required for any configuration, but instead, reflects how much space it currently requires. Note that this useful because, in one embodiment, its possible for a module to request that its display be resized.

sdevPeriodicTickle

The sdevPeriodicTickle message is passed to the module periodically to allow the module to update its display due to changes in its state. In one embodiment, this message occurs at regular intervals, while in other embodiments, there is no minimum or maximum interval between "tickles." In response to the sdevPeriodicTickle message, the module returns, as its result, some bits that signal requests for actions from the control strip processing logic. In one embodiment, there are 32 bits returned. All undefined bits in the result are reserved for future use and, in one embodiment, are set to 0. The bits are defined as:

a) sdevResizeDisplay **(0)**—If this bit is set, the module resizes its display. The control strip processing logic sends a sdevGetDisplayWidth message to the module and then updates the control strip on the display.

b) sdevNeedToSave **(1)**—If this bit is set, the module needs to save changed settings to disk. The control strip processing logic marks the request but may defer the actual save operation to a better time (e.g., when the hard disk is spinning).

c) sdevHelpStateChange **(2)**—If this bit is set, the help message of the module needs to be updated due to a change in state. If a help balloon is being displayed for the module, the control strip processing logic removes the previous help balloon with a new help balloon for the current state.

d) sdevCloseNow **(3)**—If this bit is set, the module is requesting to be closed. The control strip processing logic calls the module to save its settings, then calls the module again to close itself by, for example, disposing of any loaded resources, disposing of private storage, etc.

sdevDrawStatus

The sdevDrawStatus message indicates that the module has to redraw its display to reflect the most recent state. In one embodiment, this message is sent when the user clicks on the display area of the module, when any of the display

of the module is resized, or when the control strip itself needs to be updated, perhaps in response to a screen saver deactivation.

The statusRect parameter points to a rectangle bounding the display area of the module, in local coordinates. All drawing done by a module within the bounds of the control strip is limited to the module's display rectangle. In other embodiment, drawing may extend outside the display rectange of the module. The clipping region of the control strip's window is set to the visible portion of the display rectangle of the module so that all the elements in the display may be drawn. If the clipping region is to be changed, the initial clipping region should be observed to avoid drawing over other items in the control strip.

sdevMouseClick

When the user clicks in a display area of the module, the control strip processing logic calls the module with the sdevMouseClick message if the sdevWantMouseClicks bit is set in the features of the module.

If the sdevDontAutoTrack bit is also set, the control strip processing logic draws the display of the module in its highlighted state and then sends the sdevMouseClick message to the module. If the sdevDontAutoTrack bit is not set, the control strip processing logic tracks the cursor until the mouse button is released. If the cursor is still within the display area of the module, the control strip processing logic sends the sdevMouseClick message to notify the module that a click occurred. In either case, the module can then perform the appropriate function in response to a mouse-down event.

This message returns the same result as the sdevPeriodicTickle message.

sdevSaveSettings

The sdevSaveSettings message is passed to the module when the control strip processing logic has determined that the configuration information may be saved to the disk (e.g., HD turned on, etc.). In one embodiment, the sdevSaveSettings message is sent only if the module had previously set the sdevNeedToSave bit in the result of a sdevPeriodicTickle or sdevMouseClick message. The call returns an error code (File Manager, Resource Manager, or the like) indicating the success of the save operation. The control strip processing logic continues to send this message to the module until the module returns a result of 0, indicating a successful save.

sdevShowBalloonHelp

The control strip processing logic calls the module with the sdevShowBalloonHelp message if Balloon Help is turned on, the module has previously set the sdevHasCustomHelp bit in its features, and the cursor is over the module's display area. In such a case, the module calls the Help Manager to display a help balloon describing the current state of the module. The module returns a value of 0 if successful or an appropriate error result if not.

UTILITY ROUTINES

In one embodiment, the control strip processing logic provides a set of utility routines that are available to control strip modules. They are provided to promote a consistent user interface within the control strip and to reduce the amount of duplicated code that each module would have to include to support common functions. Therefore, in an embodiment that does not include these utility routines, a portion or all of the modules may include duplicated code supporting common functions.

SBIsControlStripVisible

The SBIsControlStripVisible routine determines whether the control strip is visible. An exemplary call follows:

pascal Boolean SBIsControlStripVisible ( );

The SBIsControlStripVisible routine returns a Boolean value indicating whether or not the control strip is currently visible. It returns a value of "true" if the control strip is visible, or a value of "false" if it's hidden.

In one embodiment, the SBIsControlStripVisible call returns a value of "true" even when the control strip is not visible. That happens whenever the control strip is not accessible in the current environment. As soon as that condition changes, the control strip becomes visible again and the returned value correctly reflects the actual state.

SBShowHideControlStrip

The SBShowHideControlStrip routine shows or hides the control strip. An exemplary call follows:

pascal void SBShowHideControlStrip (Boolean showIt);

The SBShowHideControlStrip routine determines the visibility state for the control strip based on the value of the "showIt" parameter. Passing a value of "true" makes the control strip visible, and passing a value of "false" hides it. Modules may not need to call this routine. However, the SBShowHideControlStrip routine provides a means for other software to hide the control strip when it is in the way.

Calling the SBShowHideControlStrip routine with a "showIt" value of "true" may or may not show the control strip, depending on the current environment. If the control strip is not accessible, it does not become visible. If a "showIt" value of "true" is passed to this routine, then the control strip becomes visible when the environment changes.

SBSafeToAccessStartupDisk

The SBSafeToAccessStartupDisk routine determines whether the internal hard disk is turned on so that processing logic of the present invention can determine whether to make a disk access or postpone it until a time when the disk is already spinning. An exemplary call follows:

pascal Boolean SBSafeToAccesStartupDisk ( );

The SBSafeToAccessStartDisk routine returns a Boolean value of "true" if the disk is turned on and "false" if it is not.

SBOpenModuleResourceFile

The SBOpenModuleResourceFile routine opens a module resource file. An examplary call follows:

pascal short SBOpenModuleResourcFile (OSType fileCreator);

The SBOpenModuleResourceFile routine opens the resource fork of the module file whose creator is "fileCreator", and return the file's reference number as its result. If the file cannot be found or opened, the SBOpenMduleResourceFile routine returns a result of −1.

The SBOpenModuleResourceFile routine also provides a means for a module to load in large or infrequently used resources that it doesn't usually need, but that it requires for a particular operation.

SBLoadPreferences

The SBLoadPreferences routine loads a resource from a preferences file. An examplary call follows:

pascal OSErr SBLoadPreferences (ConstStr255Param prefsResourceName, Handle *preferences);

The SBLoadPreferences routine loads a resource containing a module's configuration information from the preferences file of the control strip. The PrefsResourceName parameter points to a Pascal string containing the name of the resource. The "Preferences" parameter points to a variable that holds a handle to the resource read from the file. The handle does not need to be preallocated.

If either prefsResourceName or preferences contains a nil pointer, the SBLoadPreferences routine does nothing and returns a result of paramErr. If the resource is successfully

loaded, the SBLoadPreferences routine returns a result of 0. The SBLoadPreferences routine also returns other Memory Manager and Resource Manager errors if it fails during some art of the process.

SBSavePreferences

The SBSavePreferences routine saves a resource to a preferences file. An exemplary call follows:

pascal OSErr SBSavePreferences (ConstStr255Param prefsResourceName, Handle preferences);

The SBSavePreferences routine saves a resource containing a module's configuration information to the preferences file of the control strip. The PrefsResourceName parameter points to a Pascal string containing the name of the resource. The "preferences" parameter contains a handle to a block of data which will be written to the file.

If either prefsResourceName or preferences has a nil value, the SBSavePreferences routine does nothing and returns a result of paramErr. if the resource is successfully saved, the SBSavePreferences routine returns a result of 0. The SBSavePreferences routine can also return other Memory Manager and Resource Manager errors if it fails during some part of the process.

SBGetDetachedString

The SBGetDetachedIndString routine obtains a string from a detached resource. An exemplary call follows:

pascal void SBGetDetachedIndString (StringPtr the String, Handle stringList, short whichString);

The SBGetDetachedIndString routine is the detached resource version of GetIndString. The parameter thestring points to a Pascal string; the stringList is a handle to a detached 'STR#' resource; and whichString is the index (1–n) into the array of Pascal strings contained in the detached resource. The SBGetDetachedIndString routine copies the string whose index is whichString into the space pointed to by theString. If whichString is out of range, the SBGetDetachedIndString routine returns a zero-length string.

SBGetDetachIconSuite

The SBGetDetachIconSuite routine sets up a detached icon suite. An exemplary call follows:

pascal OSErr SBGetDetachIconSuite (Handle *theIconSuite, short theResID, unsigned long selector);

The SBGetDetachIconSuite routine creates a new icon suite, loads all of the requested icons, and then detaches the icons. The parameter theIconSuite points to the location where the handle to the icon suite is stored; the parameter theResID is the resource ID of the icons that make up the icon suite; and the parameter "selector" indicates which icons are to be loaded into the suite. In one embodiment, the "selector" parameter contains one (or a combination of) the following values:

svAllLargeData	0x000000FF	load large 32-by-32-pixel icons ('ICN#', 'ic14', 'ic18')
svAllSmallData	x0000FF00	load small 16-by-16-pixel icons ('ics#', 'ics4', 'ics8')
svAllMiniData	0x00FF0000	load mini 12-by-12-pixel icons ('icm#', icm4', 'icm8')

These values may be ORed together to load combinations of icon sizes. The SBGetDetachIconSuite routine returns an appropriate error code if it's unsuccessful, or 0 if it was able to load the icon suite. Note that if none of the icons comprising the icon suite could be found, the call returns the error "resNotFound. In one embodiment, the SBGetDetachIconSuite routine is called only when the resource file of the module is open. This is typically the case during a module's initialization call.

SBTrackpopupMenu

The SBTrackpopupMenu routine manages a pop-up menu. An exemplary call follows:

pascal short SBTrackpopupMenu (const Rect *moduleRect, MenuHandle theMenu);

The SBTrackpopupMenu routine handles setting up and displaying a pop-up menu associated with a module. The module passes a pointer to its display rectangle and a handle to the menu to use. In one embodiment the menu is displayed immediately above and adjacent to the display rectangle of the module, yet this is not required. By doing so, the user is allowed to view the current configuration or to change the settings. The SBTrackpopupMenu routine returns an indication as to which menu item was selected, or 0 if no item was selected (e.g., because the user moved the cursor outside the menu's bounds).

SBTrackSlider

The SBTrackSlider routine displays and sets an arbitrary parameter. An exemplary call follows:

pascal short SBTrackSlider (const Rect *moduleRect, short ticksOnSlider, short initialValue);

The SBTrackSlider routine displays an unlabeled slider above the module's display rectangle. The slider may be used for displaying and setting the state of an arbitrary parameter. The parameter "ModuleRect" contains a pointer to the module's display rectangle; "ticksOnSlider' is the upper bounds of the value returned by the slider; and "initialValue" is the starting position (0 to ticksOnSlider −1). When the user releases the mouse button, the SBTrackSlider routine returns the final position.

SBShowHelpString

The SBShowHelpString routine displays a help balloon. An exemplary call follows:

pascal OSErr SBShowHelpString (const Rect *moduleRect, StringPtr helpstring);

The SBShowHelpString routine displays a module's help balloon. The module passes a pointer to its display rectangle and a pointer to a Pascal string, and the routine displays the balloon if possible. If the help dstring has a length of 0 or the Help Manager is unable to display a balloon, an error result is returned. If the SBShowHelpString routine successfully displays the help balloon, it returns a result of 0.

SBGetBarGraphWidth

The SBGetBarGraphWidth routine determines how wide a bar graph drawn by the SBDrawBarGraph routine (described below) will be so that a module can calculate its display width. An exemplary call follows:

pascal short SBGetBarGraphWidth (short barCount);

The SBGetBarGraphWidth routine returns the width of a bar graph containing barCount segments. If barCount has a value less than 0, the SBGetBarGraphWidth routine returns a width of 0.

SBDrawBarGraph

The SBDrawBarGraph routine draw as bar graph. An exemplary call follows below:

pascal void SBDrawBarGraph (short level, short barCount, short direction, Point barGraphTopLeft);

The SBDrawBarGraph routine draws a bar graph containing the number of segments specified by the barCount parameter in a module's display area. If the value of barCount is less than or equal to 0, the SBDrawBarGraph routine does nothing.

The bar graph is drawn relative to the location specified by barGraphTopLeft. FIG. **11** illustrates the manner in which the point barGraphTopLeft determines the position of the bar graph.

The "level" parameter determines how many segments are highlighted. The value of "level" should be in the range of 0 to barCount −1. If the value of "level" is less than 0, no segments in the bar graph are highlighted; if "level" is greater than or equal to barCount, all segments in the bar graph are highlighted.

The direction parameter specifies which way the bar graph will be drawn to show a larger level. In one embodiment, the direction parameter specifies one of the following values:

#define BarGraphSlopeLeft −1 //max end of sloping graph is on the left #define BarGraphFlatRight 0 //max end of flat graph is on the right #define BarGraphSlopeRight 1 //max end of sloping graph is on the right

FIG. **12** illustrates the resulting bar graph for each direction value. The arrows indicate which way an increasing level value is displayed. In one embodiment, for sloped versions of the bar graph, the number of segments specified by the barCount value may not be larger than 8. If a larger barCount value is passed, the SBDrawBarGraph routine draws nothing.

SBModalDialogInContext

The SBModalDialogInContext routine may be used in place of the ModalDialog routine to prevent background applications from being run while the modal dialog window is visible. An exemplary call is as follows:

pascal void SBModalDialogInContext (ModalFilterProcPtr filterProc, short *itemHit);

The SBModalDialogInContext routine is a special version of ModalDialog that doesn't allow background applications to be run while a modal dialog window is visible. The SBModalDialogInContext routine is used when the occurence of context switching is not desired.

GESTALT SELECTOR

The control strip processing logic installs two "Gestalt" selectors to return information to locations external to the computer system. One selector returns software attributes, and the other returns the current version of the processing logic (e.g., software).

gestaltControlStripAttr

The selector "gestaltControlStripAttr ('sdev') return 32 bits describing the attributes of the current version of the control strip processing logic. In one embodiment, only the following bit is defined:

gestaltControlStripExists 0 1=control strip is installed

gestaltControlStripVersion

The selector gestaltControlStripVersion ('csvr') returns the version of control strip processing logic that is installed. The format of the returned version is the same as that of the numeric part of a Macintosh™ computer system resource, that is:

Bits 31–24	Major part of the version, in BCD
Bits 23–20	Minor part of the version, in BCD
Bits 19–16	Bug release version, in BCD
Bits 15–8	Release stage: $80=final $60=beta $40=alpha $20=development
Bits 7–0	Revision level of nonreleased version, in binary

Whereas many alterations and modifications of the present invention will no doubt become apparent to a person of ordinary skill in the art after having read the foregoing description, it is to be understood that the particular embodiment shown and described by way of illustration is in no way intended to be considered limiting. Therefore, references to details of the preferred embodiment are not intended to limit the scope of the claims which in themselves recite only those features regarded as essential to the invention.

Thus, a method and apparatus for generating a window displaying control and status indicia has been described.

I claim:

1. An interactive computer-controlled display system comprising:

a processor;

a data display screen coupled to the processor;

a cursor control device coupled to said processor for positioning a cursor on said data display screen;

a window generation and control logic coupled to the processor and data display screen to create an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions, wherein the window generation and control logic generates and displays a first window region having a plurality of display areas on said data display screen, wherein the first window region is independently displayed and independently active of any application program, and wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated; and

an indicia generation logic coupled to the data display screen to execute at least one of the plurality of individual programming modules to generate information for display in one of the plurality of display areas in the first window region, wherein at least one of the plurality of display areas and its associated programming module is sensitive to user input, and further wherein the window generation and control logic and the indicia generation logic use message-based communication to exchange information to coordinate activities of the indicia generation logic to enable interactive display activity.

2. The display system defined in claim **1** wherein the first window region comprises a control strip.

3. The display system defined in claim **1** wherein said at least one display area is variably sized.

4. The display system defined in claim **1** wherein size of the first window region is variable.

5. The display system defined in claim **4** wherein the first window region is sized such that none of the plurality of display areas is visible.

6. The display system defined in claim **4** wherein the first window region is sized such that all of the plurality of display areas are visible.

7. The display system defined in claim **4** wherein the first window region is sized such that a portion of the plurality of display areas is visible.

8. The display system defined in claim **1** wherein at least one of the plurality of the display areas only displays information.

9. The display system defined in claim **1** wherein at least one of the display areas acts to provide access to control information when selected.

10. The display system defined in claim **9** wherein said at least one of the plurality of display areas displays an additional display element.

11. The display system defined in claim 1 wherein each of the plurality of display areas is individually and variably sized.

12. The display system defined in claim 1 wherein the first window region always appears in front of application windows.

13. The display system defined in claim 1 wherein the first window region is implemented in a private window layer that appears in front of windows for all applications layers.

14. An interactive computer-controlled display system comprising:

a processor;

a data display screen coupled to the processor;

a cursor control device coupled to said processor for positioning a cursor on said data display screen;

window generation and control logic coupled to the processor and data display screen to create an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions, wherein the window generation and control logic generates and displays a first window-region having a plurality of display areas on said data display screen, wherein the first window region is independently displayed and independently active of any application program, and wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated; and

at least one indicia graphics generation logic coupled to the processor and the window generation and control logic, wherein said at least one indicia graphics generation logic generates user sensitive graphics for display in at least one data display area by executing at least one of the plurality of individual programming modules;

wherein the window generation and control logic determines when said at least one data display area has been selected by the user and signals said at least one indicia graphics generation logic in response to user selection, and further wherein said at least one indicia graphics generation logic initiates a response from said at least one of the plurality of programming modules.

15. The display system defined in claim 14 wherein the first window region is always visible to the user.

16. The display system defined in claim 14 wherein the first window region comprises a control strip.

17. The display system defined in claim 14 wherein said at least one display area is variably sized.

18. The display system defined in claim 14 wherein each of the plurality of display areas is individually and variably sized.

19. The display system defined in claim 14 wherein the first window region always appears in front of application windows.

20. The display system defined in claim 14 wherein the first window region is implemented in a private window layer that appears in front of windows for all applications layers.

21. A method for generating control information comprising:

creating an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions;

generating a first window sized to accommodate a plurality of display areas for indicia resulting from executing at least one of the plurality of individual programming modules, wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, and wherein the first window is independently displayed and independently active of any application program, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated;

displaying the indicia in each of said plurality of display areas by executing one of a plurality of individual programming modules corresponding to each indicia;

selecting one of the indicia, wherein the selecting comprises a first programming module determining which of said plurality of display areas is selected and sending a message to a programming module of said plurality of individual programming modules responsible for generating a display of a selected indicia;

said programming module performing a function in response to a selection.

22. The method defined in claim 21 wherein one of said plurality of indicia comprises status information.

23. The method defined in claim 21 wherein one of said plurality of indicia comprises control information.

24. The method defined in claim 21 further comprising:

the first programming module requesting a set of features supported by said programming module, wherein said requesting comprises sending a first message to said programming module; and

said programming module returning a second message indicative of features supported by said programming module, such that said first programming module interacts with said programming module in response to user interaction with the first programming module based on indicated features as set forth by said programming module.

25. A system comprising:

a window generation and control logic to create an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions, wherein the window generation and control logic generates and displays a first window region having a plurality of display areas, wherein the first window region is independently displayed and independently active of any application program, and wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated;

an indicia generation logic coupled to the data display screen to execute at least one of the plurality of individual programming modules to generate information for display in one of the plurality of display areas in the first window region, wherein at least one of the plurality of display areas and its associated programming module is sensitive to user input, and further wherein the window generation and control logic and the indicia generation logic use message-based com-

munication to exchange information to coordinate activities of the indicia generation logic to enable interactive display activity.

26. An interactive computer-controlled display system comprising:

a means for positioning a cursor on a data display screen;

a means for creating an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions, wherein a first window region is displayed having a plurality of display areas on said data display screen, wherein the first window region is independently displayed and independently active of any application program, and wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated; and

a means for executing at least one of the plurality of individual programming modules to generate information for display in one of the plurality of display areas in the first window region, wherein at least one of the plurality of display areas and its associated programming module is-sensitive to user input, wherein an interactive display activity is enabled.

27. The display system defined in claim **26** wherein the first window region comprises a control strip.

28. The display system defined in claim **26** wherein said at least one of the plurality of display areas is variably sized.

29. The display system defined in claim **26** wherein size of the first window region is variable.

30. The display system defined in claim **29** wherein the first window region is sized such that none of the plurality of display areas is visible.

31. The display system defined in claim **29** wherein the first window regions is sized such that all of the plurality of display areas are visible.

32. The display system defined in claim **29** wherein the first window regions is sized such that a portion of the plurality of display areas is visible.

33. The display system defined in claim **26** wherein said at least one of the plurality of display areas only displays information.

34. The display system defined in claim **26** wherein said at least one of the plurality of display areas acts to provide access to control information when selected.

35. The display system defined in claim **34** wherein said at least one of the data areas display an additional display element.

36. The display system defined in claim **26** wherein each of the plurality of display areas is individually and variably sized.

37. The display system defined in claim **26** wherein the first window region always appears in front of application windows.

38. The display system defined in claim **26** wherein the first window region is implemented in a private window layer that appears in front of windows for all application layers.

39. An interactive computer-controlled display system comprising:

a means for positioning a cursor on said data display screen;

a means for creating an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions, wherein a first window region is displayed having a plurality of display areas on said data display screen, wherein the first window region is independently displayed and independently active of any application program, and wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated;

a means for generating user sensitive graphics for display in at least one data display area;

a means for determining when said at least one data display area has been selected by the user; and

a means for initiating a response from said at least one of the plurality of programming modules.

40. The display system defined in claim **39** wherein the first window region is always visible to the user.

41. The display system defined in claim **39** wherein the first window region comprises a control strip.

42. The display system defined in claim **39** wherein said at least one data display area is variably sized.

43. The display system defined in claim **39** wherein each of the plurality of display areas is individually and variably sized.

44. The display system defined in claim **39** wherein the first window region always appears in front of application windows.

45. The display system defined in claim **39** wherein the first window region is implemented in a private window layer that appears in front of windows for all applications layers.

46. A computer readable medium containing executable computer program instructions, which when executed by a data processing system, cause the data processing system to perform a method for generating control information comprising:

creating an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions;

generating a first window sized to accommodate a plurality of display areas for indicia resulting from executing at least one of the plurality of individual programming modules, wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, and wherein the first window is independently displayed and independently active of any application program, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated;

displaying the indicia in each of the plurality of display areas by executing one of a plurality of individual programming modules corresponding to each indicia; and

selecting one of the indicia, wherein the selecting comprises a first programming module determining which of the plurality of display areas is selected and sending a message to a programming module of the plurality of individual programming modules responsible for generating a display of a selected indicia, and the programming module performing a function in response to a selection.

47. The computer readable medium as set forth in claim **46** wherein ones of the indicia comprises status information.

48. The computer readable medium as set forth in claim **46** wherein one of the indicia comprises control information.

49. The computer readable medium as set forth in claim **46** further comprising:

the first programming module requesting a set of features supported by said programming module, wherein requesting comprises sending a first message to said programming module; and

said programming module returning a second message indicative of features supported by said programming module, such that said first programming module interacts with said programming module in response to user interaction with the first programming module based on indicated features as set forth by said programming module.

50. A system comprising:

a means for window generation and control to create an operating environment for a plurality of individual programming modules associated with different application programs that provide status and/or control functions, wherein the means for window generation and control generates and displays a first window region having a plurality of display areas, wherein the first window region is independently displayed and independently active of any application program, and wherein each of the plurality of display areas is associated with one of the plurality of individual programming modules, the first window region and the plurality of independent display areas implemented in a window layer that appears on top of application programming windows that may be generated;

a means for indicia generation coupled to the data display screen to execute at least one of the plurality of individual programming modules to generate information for display in one of the plurality of display areas in the first window region, wherein at least one of the plurality of display areas and its associated programming module is sensitive to user input, and further wherein the means for window generation and control and the means for indicia generation use message-based communication to exchange information to coordinate activities of the means for indicia generation to enable interactive display activity.

* * * * *

Exhibit 4

US007469381B2

(12) **United States Patent**
Ording

(10) **Patent No.: US 7,469,381 B2**
(45) **Date of Patent: Dec. 23, 2008**

(54) **LIST SCROLLING AND DOCUMENT TRANSLATION, SCALING, AND ROTATION ON A TOUCH-SCREEN DISPLAY**

(75) Inventor: **Bas Ording**, San Francisco, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **11/956,969**

(22) Filed: **Dec. 14, 2007**

(65) **Prior Publication Data**

US 2008/0168404 A1 Jul. 10, 2008

Related U.S. Application Data

(60) Provisional application No. 60/937,993, filed on Jun. 29, 2007, provisional application No. 60/946,971, filed on Jun. 28, 2007, provisional application No. 60/945,858, filed on Jun. 22, 2007, provisional application No. 60/879,469, filed on Jan. 8, 2007, provisional application No. 60/883,801, filed on Jan. 7, 2007, provisional application No. 60/879,253, filed on Jan. 7, 2007.

(51) **Int. Cl.**
G06F 3/01 (2006.01)

(52) **U.S. Cl.** **715/702**; 715/764; 715/863; 715/864; 715/769

(58) **Field of Classification Search** 715/764, 715/769, 702, 863, 864
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,495,566	A	2/1996	Kwatinetz	395/157
5,844,547	A	12/1998	Minakuchi et al.	345/173
5,867,158	A	2/1999	Murasaki et al.	345/341
6,034,688	A	3/2000	Greenwood et al.	345/353
6,489,951	B1	12/2002	Wong et al.	345/173
6,567,102	B2	5/2003	Kung	345/660

(Continued)

FOREIGN PATENT DOCUMENTS

EP 0 635 779 A1 1/1995

(Continued)

OTHER PUBLICATIONS

Microsoft Word 2003 Screen Shots.*

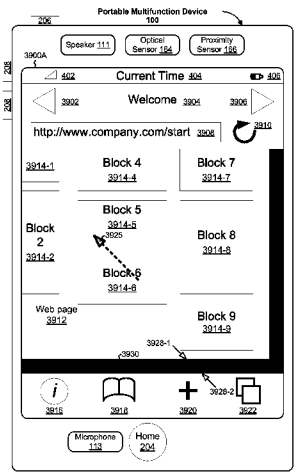
(Continued)

Primary Examiner—Boris Pesin
(74) *Attorney, Agent, or Firm*—Morgan, Lewis & Bockius LLP

(57) **ABSTRACT**

In accordance with some embodiments, a computer-implemented method for use in conjunction with a device with a touch screen display is disclosed. In the method, a movement of an object on or near the touch screen display is detected. In response to detecting the movement, an electronic document displayed on the touch screen display is translated in a first direction. If an edge of the electronic document is reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display, an area beyond the edge of the document is displayed. After the object is no longer detected on or near the touch screen display, the document is translated in a second direction until the area beyond the edge of the document is no longer displayed.

20 Claims, 38 Drawing Sheets



U.S. PATENT DOCUMENTS

6,661,409	B2	12/2003	Demartines et al.	345/173
6,690,387	B2 *	2/2004	Zimmerman et al.	345/684
6,707,449	B2	3/2004	Hinckley et al.	345/173
6,809,724	B1	10/2004	Shiraishi et al.	345/169
6,907,575	B2	6/2005	Duarte	715/784
6,912,462	B2	6/2005	Ogaki	701/208
6,972,776	B2	12/2005	Davis et al.	345/684
6,975,306	B2	12/2005	Hinckley et al.	345/173
7,009,599	B2	3/2006	Pihlaja	345/173
7,046,230	B2	5/2006	Zadesky et al.	345/156
7,075,512	B1	7/2006	Fabre et al.	345/156
7,102,626	B2	9/2006	Denny, III	345/179
7,154,534	B2	12/2006	Seki et al.	348/207.1
7,155,048	B2	12/2006	Ohara	382/132
7,181,373	B2	2/2007	Le Cocq et al.	703/1
7,184,796	B2	2/2007	Karidis et al.	455/566
7,240,291	B2	7/2007	Card et al.	715/776
2003/0095135	A1	5/2003	Kaasila et al.	345/613
2003/0095697	A1	5/2003	Wood et al.	382/131
2003/0184525	A1	10/2003	Tsai	345/173
2004/0021676	A1	2/2004	Chen et al.	345/684
2004/0027398	A1 *	2/2004	Jaeger	345/863
2004/0080541	A1	4/2004	Saiga et al.	345/805
2004/0155888	A1	8/2004	Padgitt et al.	345/619
2004/0263486	A1	12/2004	Seni	345/173
2005/0012723	A1	1/2005	Pallakoff	345/173
2005/0145807	A1	7/2005	Lapstun et al.	250/566
2005/0168488	A1	8/2005	Montague	345/659
2005/0198588	A1	9/2005	Lin et al.	715/784
2005/0237308	A1	10/2005	Autio et al.	345/173
2005/0270269	A1	12/2005	Tokkonen	345/156
2005/0275618	A1	12/2005	Juh et al.	345/156
2006/0025218	A1	2/2006	Hotta	463/37
2006/0026521	A1	2/2006	Hotelling et al.	715/702
2006/0048073	A1	3/2006	Jarrett et al.	715/784
2006/0055669	A1	3/2006	Das	345/156
2006/0061551	A1	3/2006	Fateh	345/158
2006/0077544	A1	4/2006	Stark	359/448
2006/0082549	A1	4/2006	Hoshino et al.	345/157
2006/0094502	A1	5/2006	Katayama et al.	463/31
2006/0156249	A1	7/2006	Blythe et al.	715/781
2006/0181510	A1	8/2006	Faith	345/158
2006/0187215	A1	8/2006	Rosenberg et al.	345/173
2006/0197753	A1	9/2006	Hotelling	345/173
2006/0238495	A1	10/2006	Davis	345/156
2007/0008066	A1	1/2007	Fukuda	340/5.52
2007/0024646	A1	2/2007	Saarinen et al.	345/660
2007/0046646	A1	3/2007	Kwon et al.	345/173
2007/0067745	A1	3/2007	Choi et al.	715/863
2007/0109275	A1	5/2007	Chuang	345/173
2007/0120835	A1	5/2007	Sato	345/173
2007/0150826	A1	6/2007	Anzures et al.	715/772
2007/0150842	A1	6/2007	Chaudhri et al.	715/863
2007/0152978	A1	7/2007	Kocienda et al.	345/173
2007/0152979	A1	7/2007	Jobs et al.	345/173
2007/0155434	A1	7/2007	Jobs et al.	455/565
2007/0156364	A1	7/2007	Rothkopf	702/117
2008/0104544	A1 *	5/2008	Collins et al.	715/846

FOREIGN PATENT DOCUMENTS

EP	0 701 220	A	3/1996
EP	0 880 091	A2	11/1998
WO	WO 02/01338	A1	1/2002
WO	WO 03/060622	A2	7/2003
WO	WO 2005/052773	A2	6/2005
WO	WO 2006/003591	A	1/2006
WO	WO 2006/020305	A2	2/2006

OTHER PUBLICATIONS

Photo Mesa 3.1.2 2006 Screen Shots.*

International Search Report and Written Opinion for International Application No. PCT/US2008/050292, mailed Sep. 19, 2008.

IBM, "Scroll Control Box," IBM Technical Disclosure Bulletin, vol. 38, No. 04, Apr. 1993, pp. 399-403.

Tidwell et al., "Magnetism," Designing Interfaces, Nov. 2005, Section 85.

Invitation to Pay Additional Fees for International Application No. PCT/US2008/050292, mailed Jul. 18, 2008.

Miller, D., "PersonalJava Application Environment," Sun Microsystems, http://java.sun.com/products/personaljava/touchable/, Jun. 8, 1999, 13 pages.

* cited by examiner

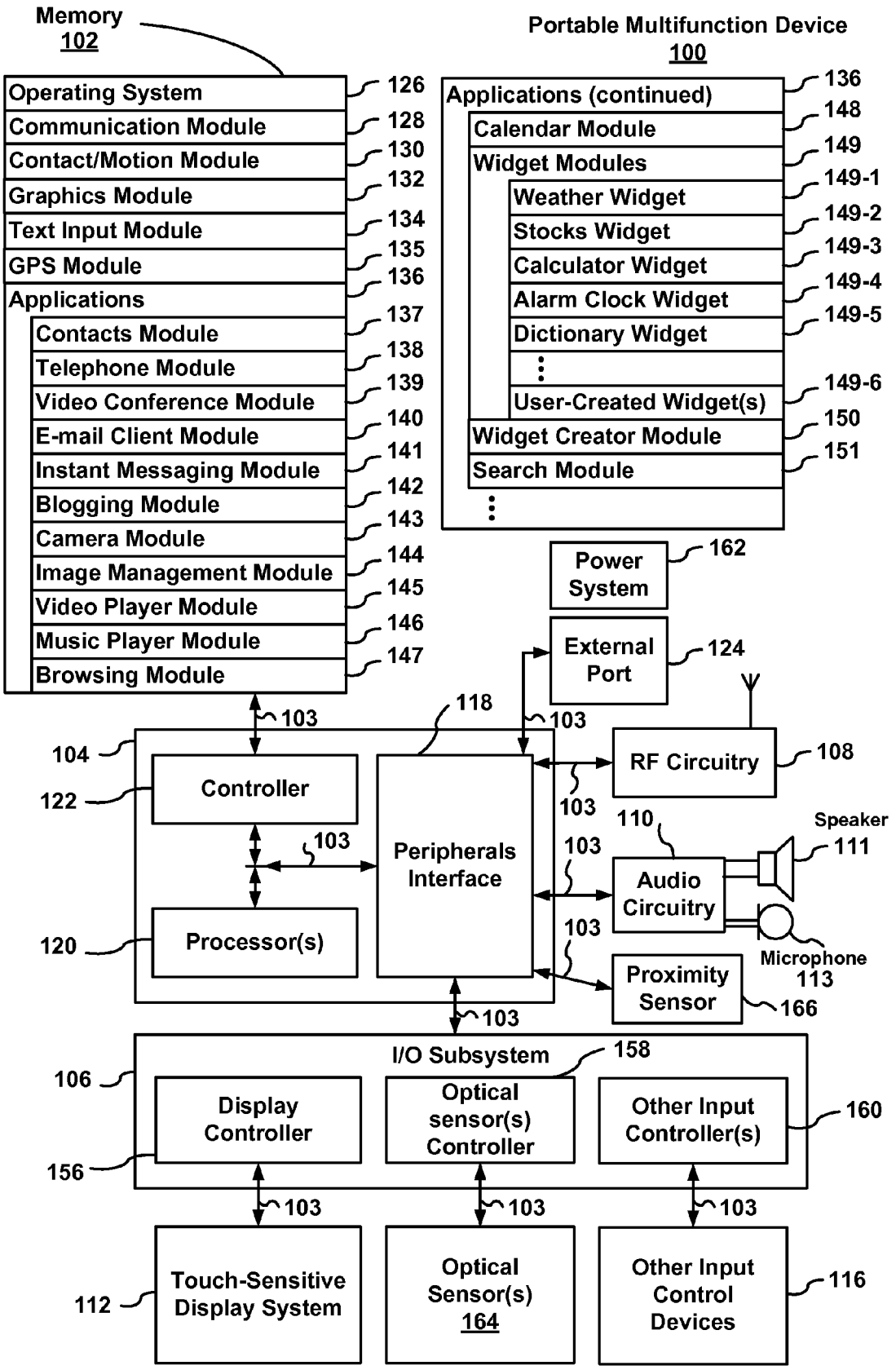

Memory
102
Operating System
126
Communication Module
128
Contact/Motion Module
130
Graphics Module
132
Text Input Module
134
GPS Module
135
Applications
136
Contacts Module
137
Telephone Module
138
Video Conference Module
139
E-mail Client Module
140
Instant Messaging Module
141
Blogging Module
142
Camera Module
143
Image Management Module
144
Video Player Module
145
Music Player Module
146
Browsing Module
147
Portable Multifunction Device
100
Applications (continued)
136
Calendar Module
148
Widget Modules
149
Weather Widget
149-1
Stocks Widget
149-2
Calculator Widget
149-3
Alarm Clock Widget
149-4
Dictionary Widget
149-5
User-Created Widget(s)
149-6
Widget Creator Module
150
Search Module
151
Power System
162
External Port
124
103
118
104
Controller
122
Peripherals Interface
RF Circuitry
108
110
Speaker
111
Audio Circuitry
Processor(s)
120
Microphone
113
Proximity Sensor
166
I/O Subsystem
158
106
Display Controller
Optical sensor(s) Controller
Other Input Controller(s)
160
156
Touch-Sensitive Display System
112
Optical Sensor(s)
164
Other Input Control Devices
116

Figure 1

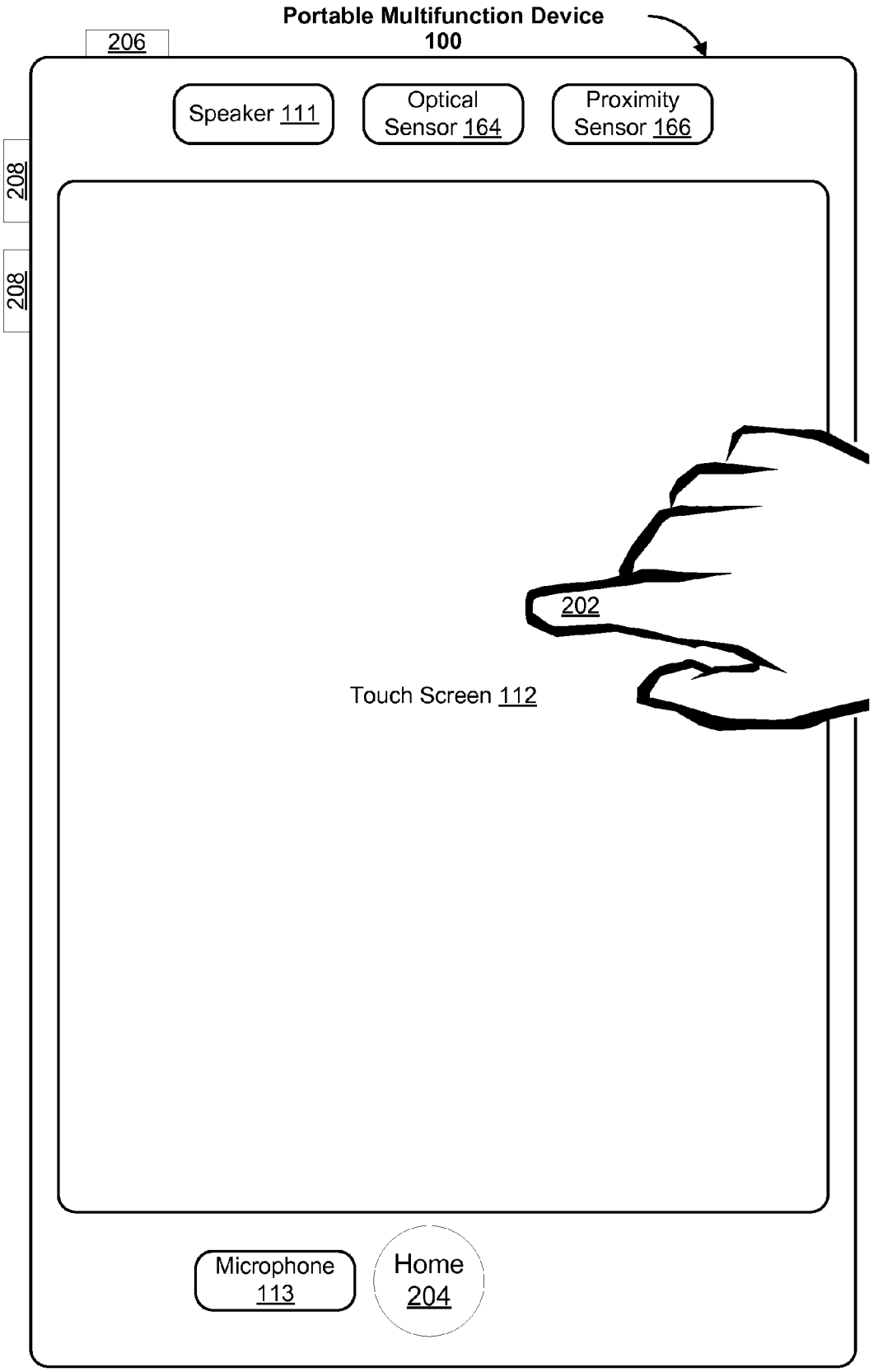

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
208
208
202
Touch Screen 112
Microphone 113
Home 204

Figure 2

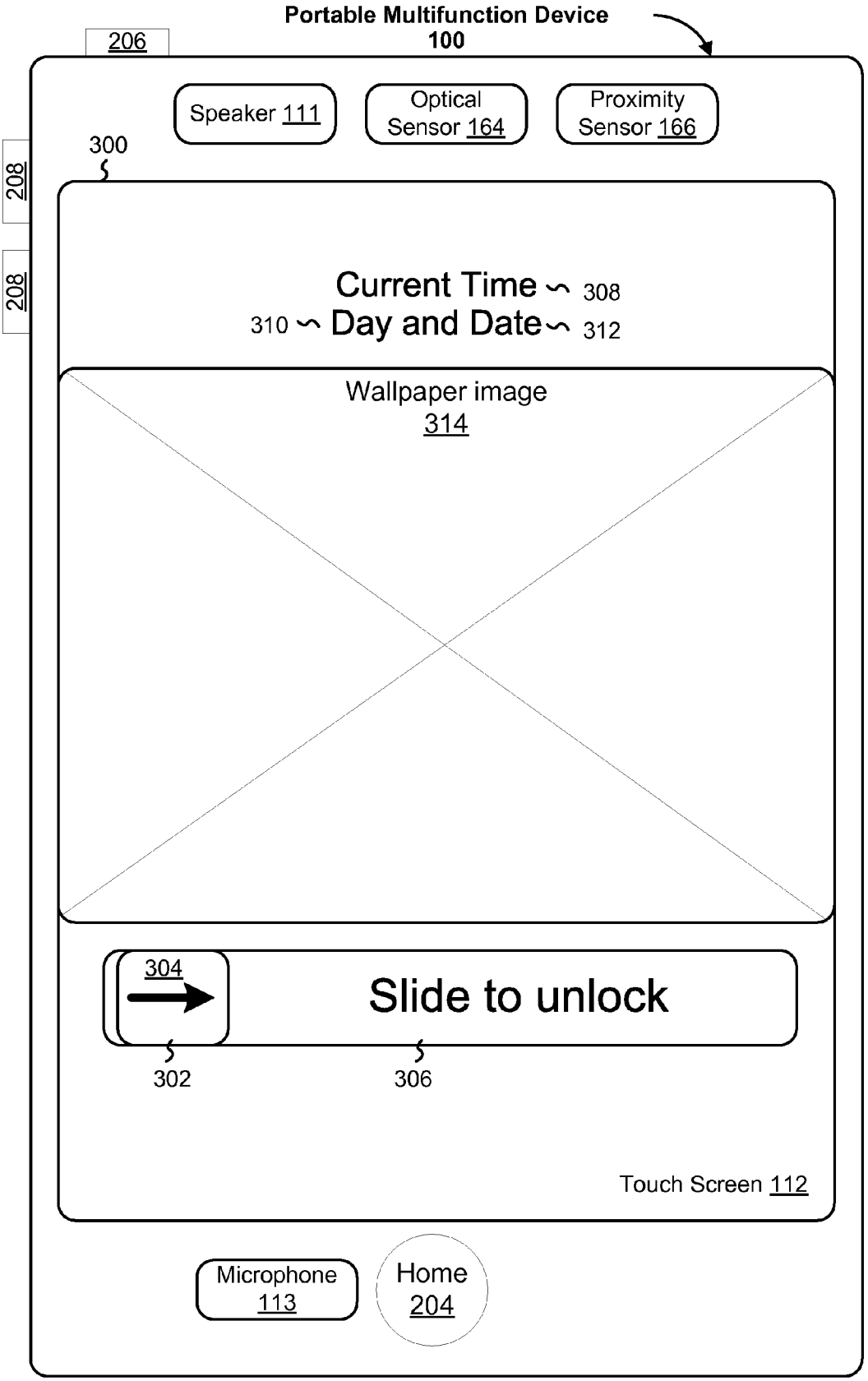

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
300
208
208
Current Time 308
310 Day and Date 312
Wallpaper image
314
304
Slide to unlock
302
306
Touch Screen 112
Microphone 113
Home 204

Figure 3

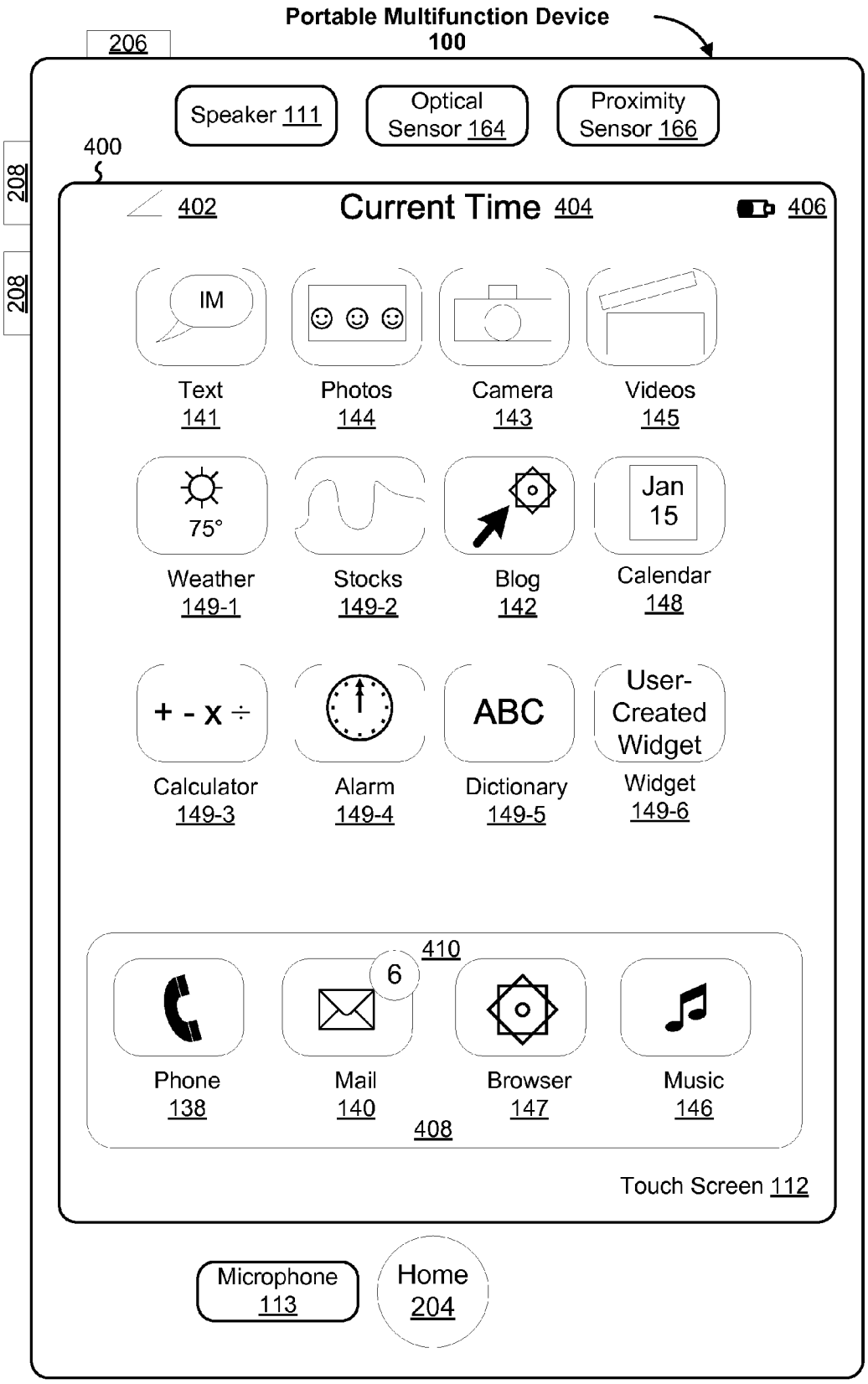

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
400
208
208
402
Current Time 404
406
IM
Text 141
Photos 144
Camera 143
Videos 145
75°
Weather 149-1
Stocks 149-2
Blog 142
Jan 15
Calendar 148
+ - x ÷
Calculator 149-3
Alarm 149-4
ABC
Dictionary 149-5
User-Created Widget
Widget 149-6
410
6
Phone 138
Mail 140
Browser 147
Music 146
408
Touch Screen 112
Microphone 113
Home 204

Figure 4

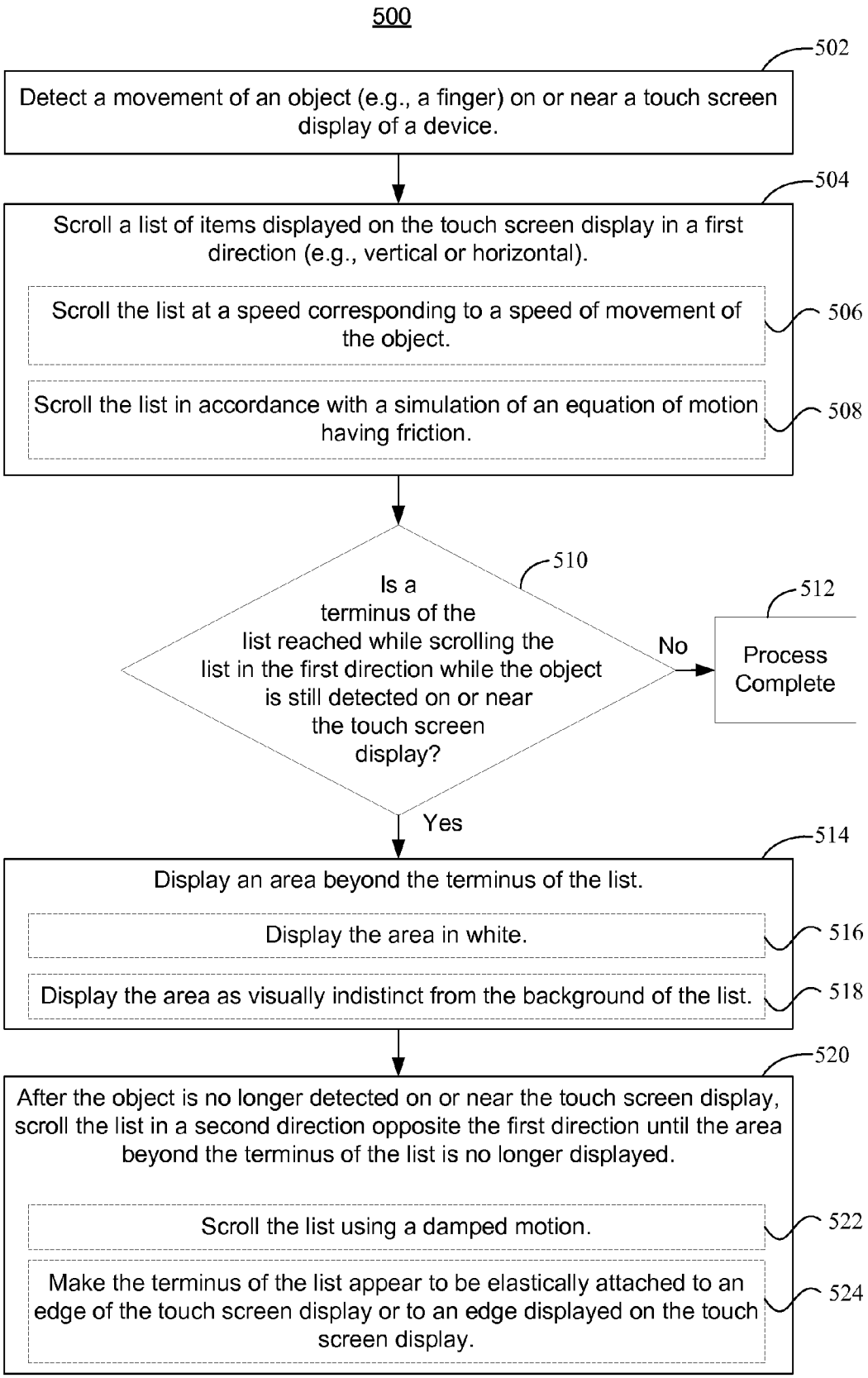

500
502
Detect a movement of an object (e.g., a finger) on or near a touch screen display of a device.
504
Scroll a list of items displayed on the touch screen display in a first direction (e.g., vertical or horizontal).
Scroll the list at a speed corresponding to a speed of movement of the object.
506
Scroll the list in accordance with a simulation of an equation of motion having friction.
508
510
Is a terminus of the list reached while scrolling the list in the first direction while the object is still detected on or near the touch screen display?
No
512
Process Complete
Yes
514
Display an area beyond the terminus of the list.
Display the area in white.
516
Display the area as visually indistinct from the background of the list.
518
520
After the object is no longer detected on or near the touch screen display, scroll the list in a second direction opposite the first direction until the area beyond the terminus of the list is no longer displayed.
Scroll the list using a damped motion.
522
Make the terminus of the list appear to be elastically attached to an edge of the touch screen display or to an edge displayed on the touch screen display.
524

Figure 5

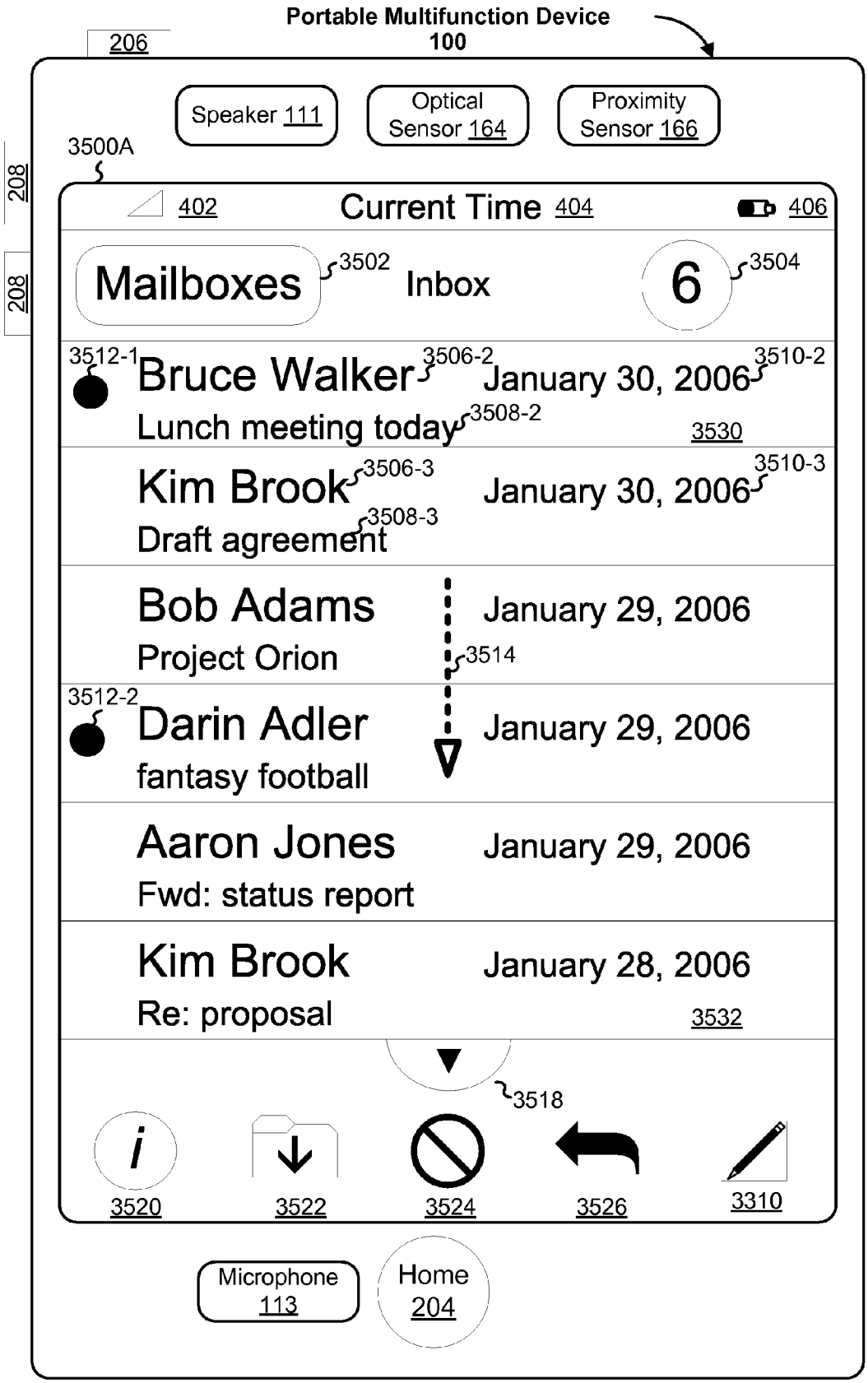

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3500A
208
208
402
Current Time 404
406
Mailboxes
3502
Inbox
6
3504
3512-1
Bruce Walker
3506-2
January 30, 2006
3510-2
Lunch meeting today
3508-2
3530
Kim Brook
3506-3
January 30, 2006
3510-3
Draft agreement
3508-3
Bob Adams
January 29, 2006
Project Orion
3514
3512-2
Darin Adler
January 29, 2006
fantasy football
Aaron Jones
January 29, 2006
Fwd: status report
Kim Brook
January 28, 2006
Re: proposal
3532
3518
3520
3522
3524
3526
3310
Microphone 113
Home 204

Figure 6A

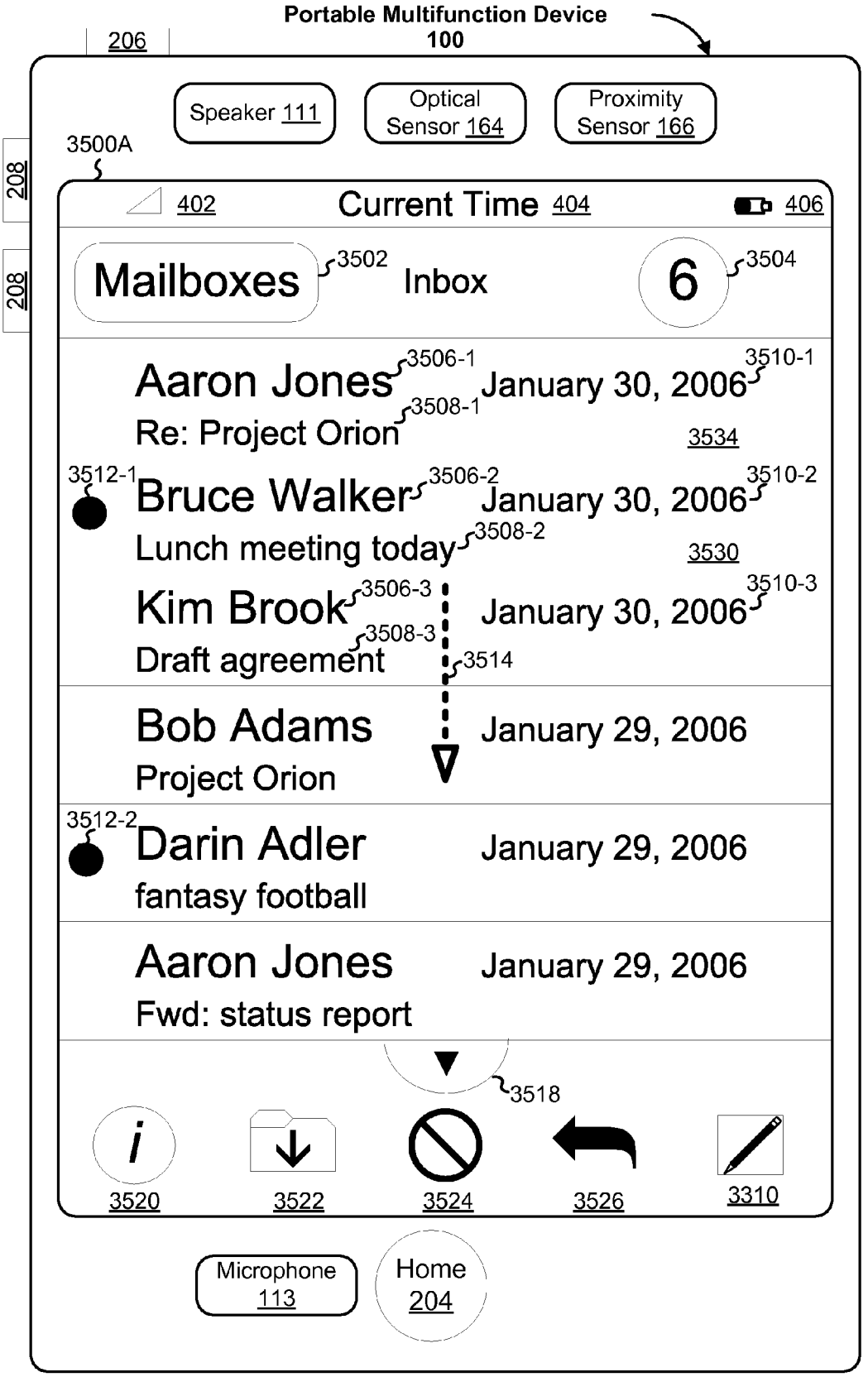

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3500A
208
208
402
Current Time 404
406
Mailboxes
3502
Inbox
6
3504
Aaron Jones
3506-1
January 30, 2006
3510-1
Re: Project Orion
3508-1
3534
3512-1
Bruce Walker
3506-2
January 30, 2006
3510-2
Lunch meeting today
3508-2
3530
Kim Brook
3506-3
January 30, 2006
3510-3
Draft agreement
3508-3
3514
Bob Adams
January 29, 2006
Project Orion
3512-2
Darin Adler
January 29, 2006
fantasy football
Aaron Jones
January 29, 2006
Fwd: status report
3518
3520
3522
3524
3526
3310
Microphone 113
Home 204

Figure 6B

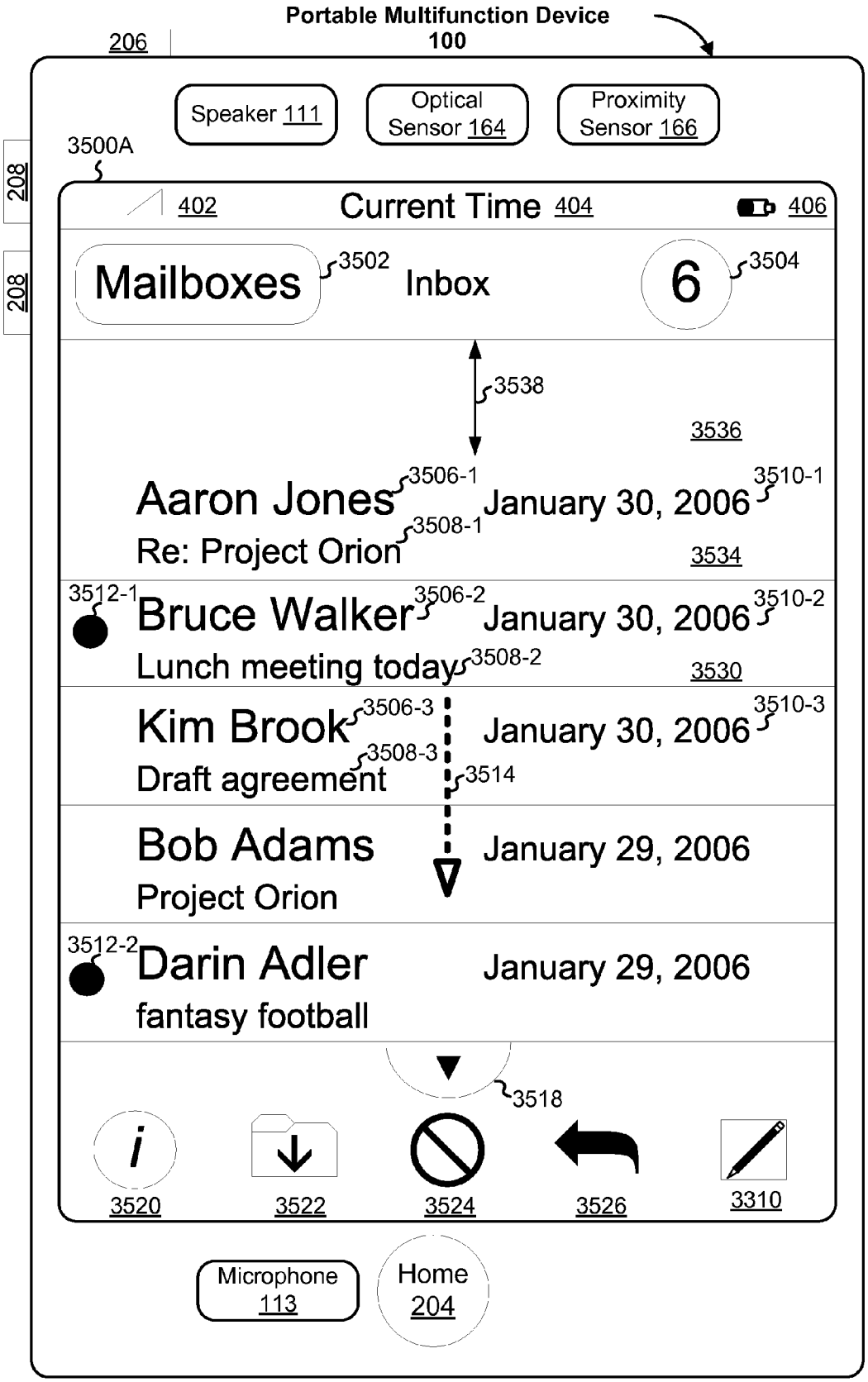

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3500A
208
208
402
Current Time 404
406
Mailboxes
3502
Inbox
6
3504
3538
3536
Aaron Jones
3506-1
January 30, 2006
3510-1
Re: Project Orion
3508-1
3534
3512-1
Bruce Walker
3506-2
January 30, 2006
3510-2
Lunch meeting today
3508-2
3530
Kim Brook
3506-3
January 30, 2006
3510-3
Draft agreement
3508-3
3514
Bob Adams
January 29, 2006
Project Orion
3512-2
Darin Adler
January 29, 2006
fantasy football
3518
3520
3522
3524
3526
3310
Microphone 113
Home 204

Figure 6C

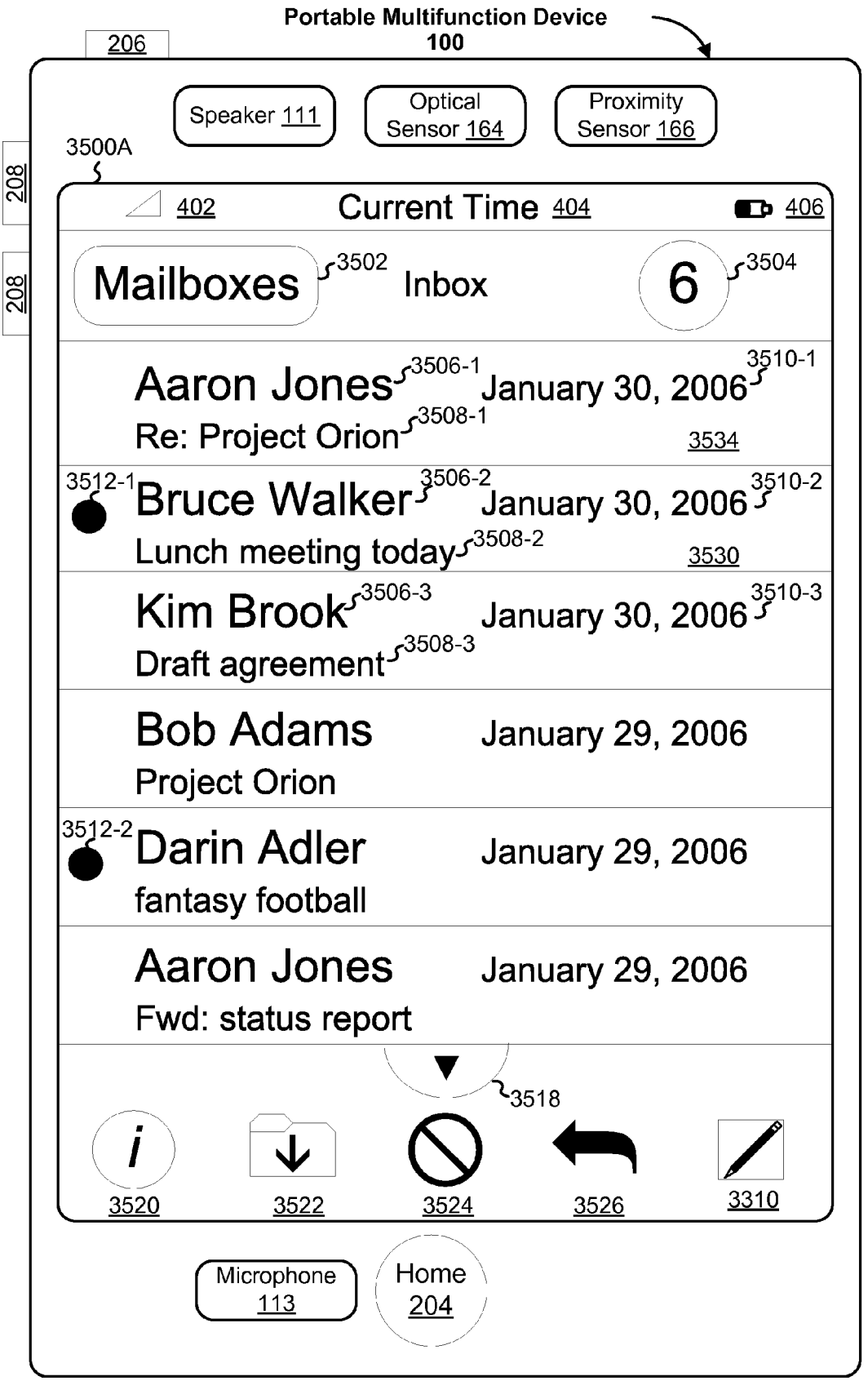

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3500A
208
208
402
Current Time 404
406
Mailboxes
3502
Inbox
6
3504
Aaron Jones
3506-1
January 30, 2006
3510-1
Re: Project Orion
3508-1
3534
3512-1
Bruce Walker
3506-2
January 30, 2006
3510-2
Lunch meeting today
3508-2
3530
Kim Brook
3506-3
January 30, 2006
3510-3
Draft agreement
3508-3
Bob Adams
January 29, 2006
Project Orion
3512-2
Darin Adler
January 29, 2006
fantasy football
Aaron Jones
January 29, 2006
Fwd: status report
3518
3520
3522
3524
3526
3310
Microphone 113
Home 204

Figure 6D

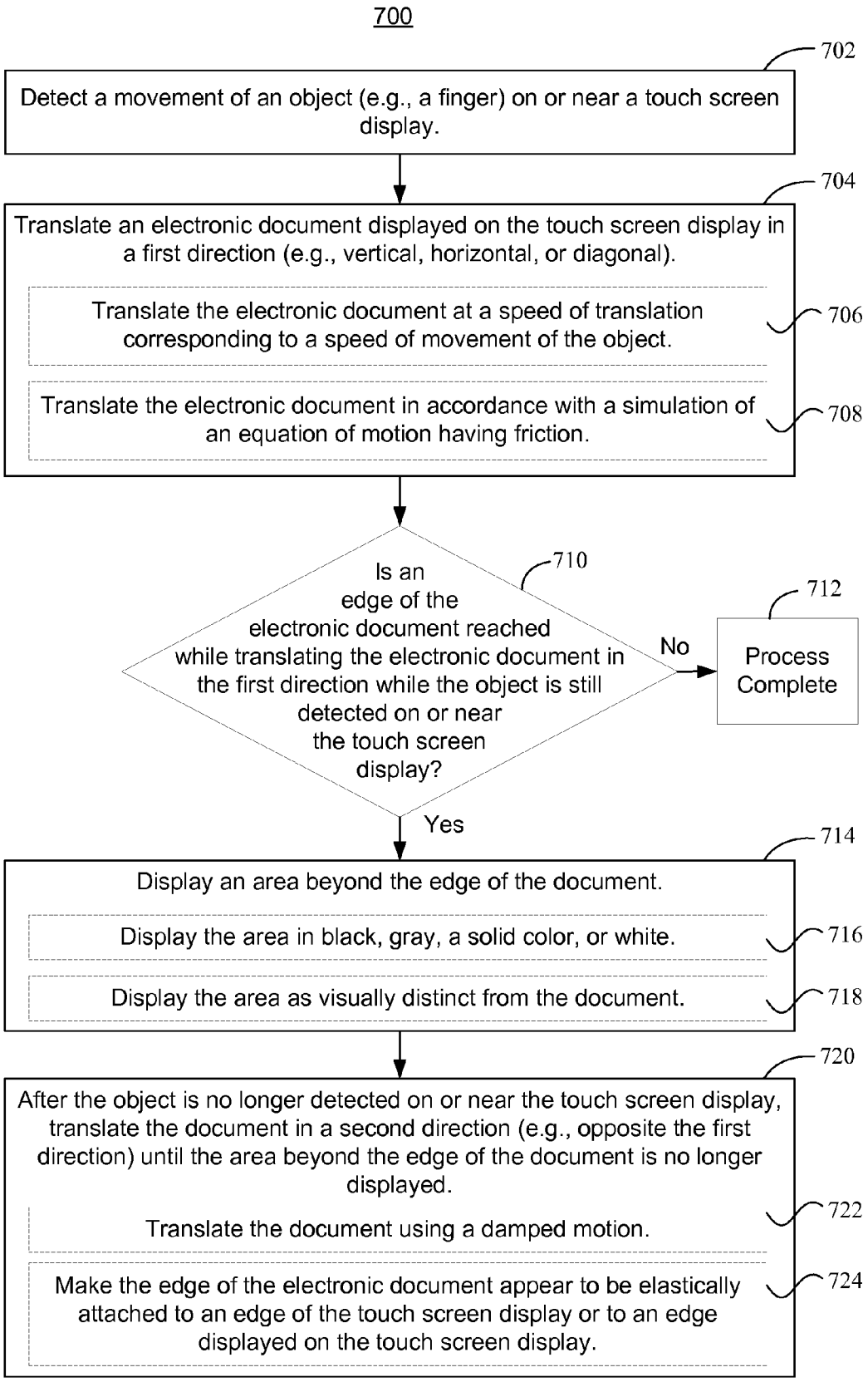

700
702
Detect a movement of an object (e.g., a finger) on or near a touch screen display.
704
Translate an electronic document displayed on the touch screen display in a first direction (e.g., vertical, horizontal, or diagonal).
Translate the electronic document at a speed of translation corresponding to a speed of movement of the object.
706
Translate the electronic document in accordance with a simulation of an equation of motion having friction.
708
710
Is an edge of the electronic document reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display?
No
712
Process Complete
Yes
714
Display an area beyond the edge of the document.
Display the area in black, gray, a solid color, or white.
716
Display the area as visually distinct from the document.
718
720
After the object is no longer detected on or near the touch screen display, translate the document in a second direction (e.g., opposite the first direction) until the area beyond the edge of the document is no longer displayed.
Translate the document using a damped motion.
722
Make the edge of the electronic document appear to be elastically attached to an edge of the touch screen display or to an edge displayed on the touch screen display.
724

Figure 7

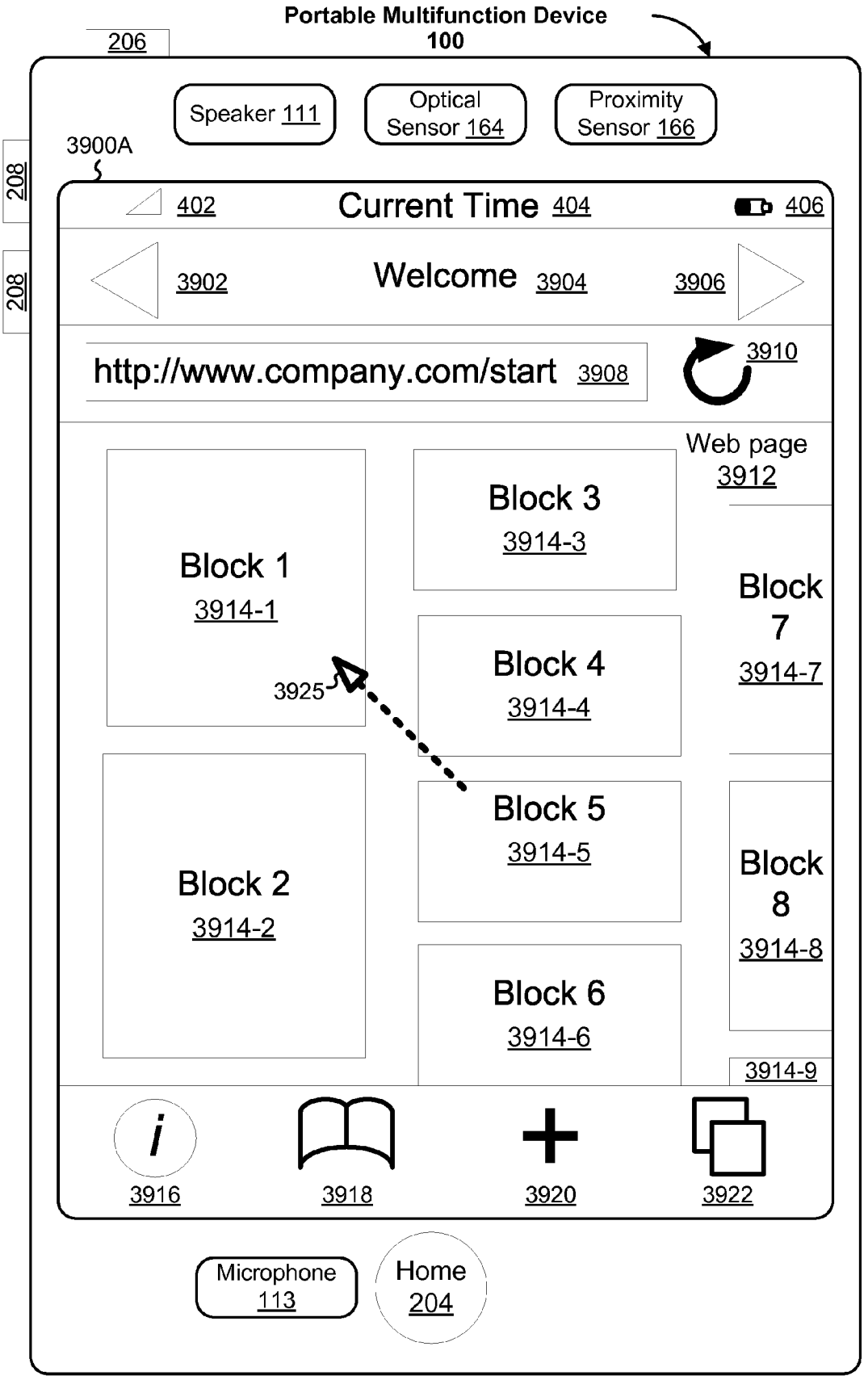

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Web page
3912
Block 1
3914-1
Block 3
3914-3
Block 7
3914-7
Block 4
3914-4
3925
Block 5
3914-5
Block 2
3914-2
Block 8
3914-8
Block 6
3914-6
3914-9
3916
3918
3920
3922
Microphone 113
Home 204

Figure 8A

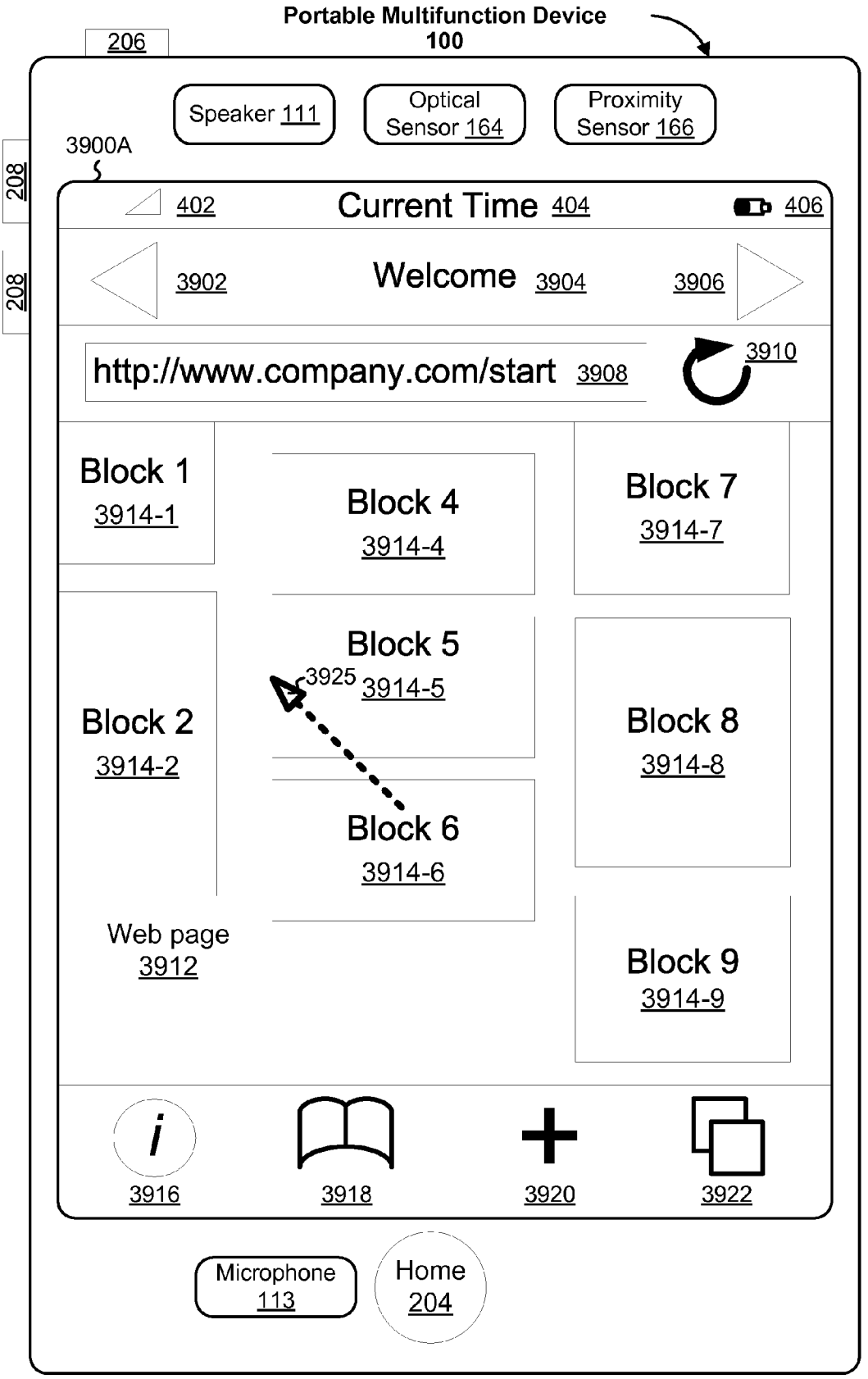

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Block 1
3914-1
Block 4
3914-4
Block 7
3914-7
Block 2
3914-2
Block 5
3925
3914-5
Block 8
3914-8
Block 6
3914-6
Web page
3912
Block 9
3914-9
3916
3918
3920
3922
Microphone
113
Home
204

Figure 8B

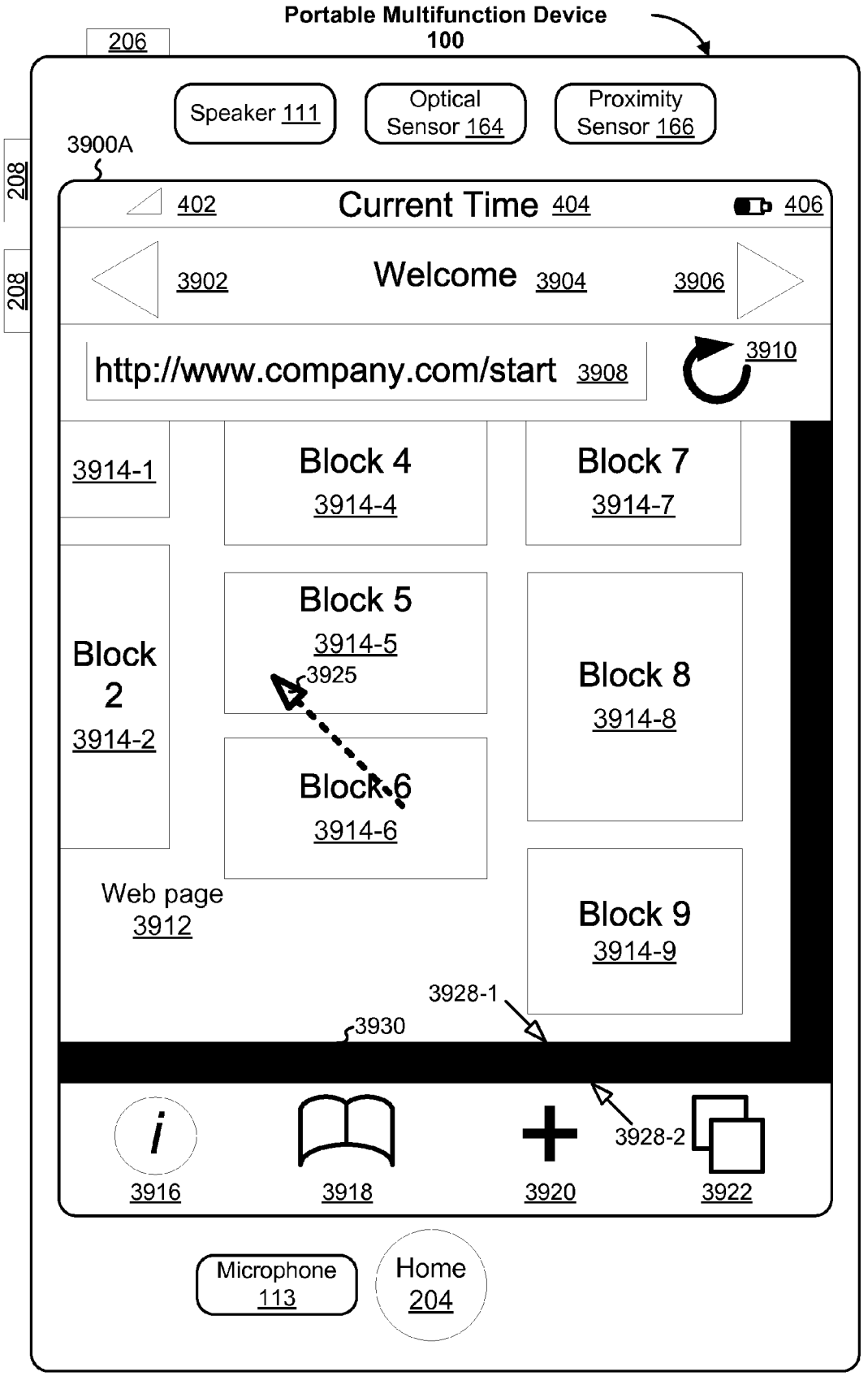

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
3914-1
Block 4
3914-4
Block 7
3914-7
Block 2
3914-2
Block 5
3914-5
3925
Block 8
3914-8
Block 6
3914-6
Web page
3912
Block 9
3914-9
3928-1
3930
3928-2
3916
3918
3920
3922
Microphone
113
Home
204

Figure 8C

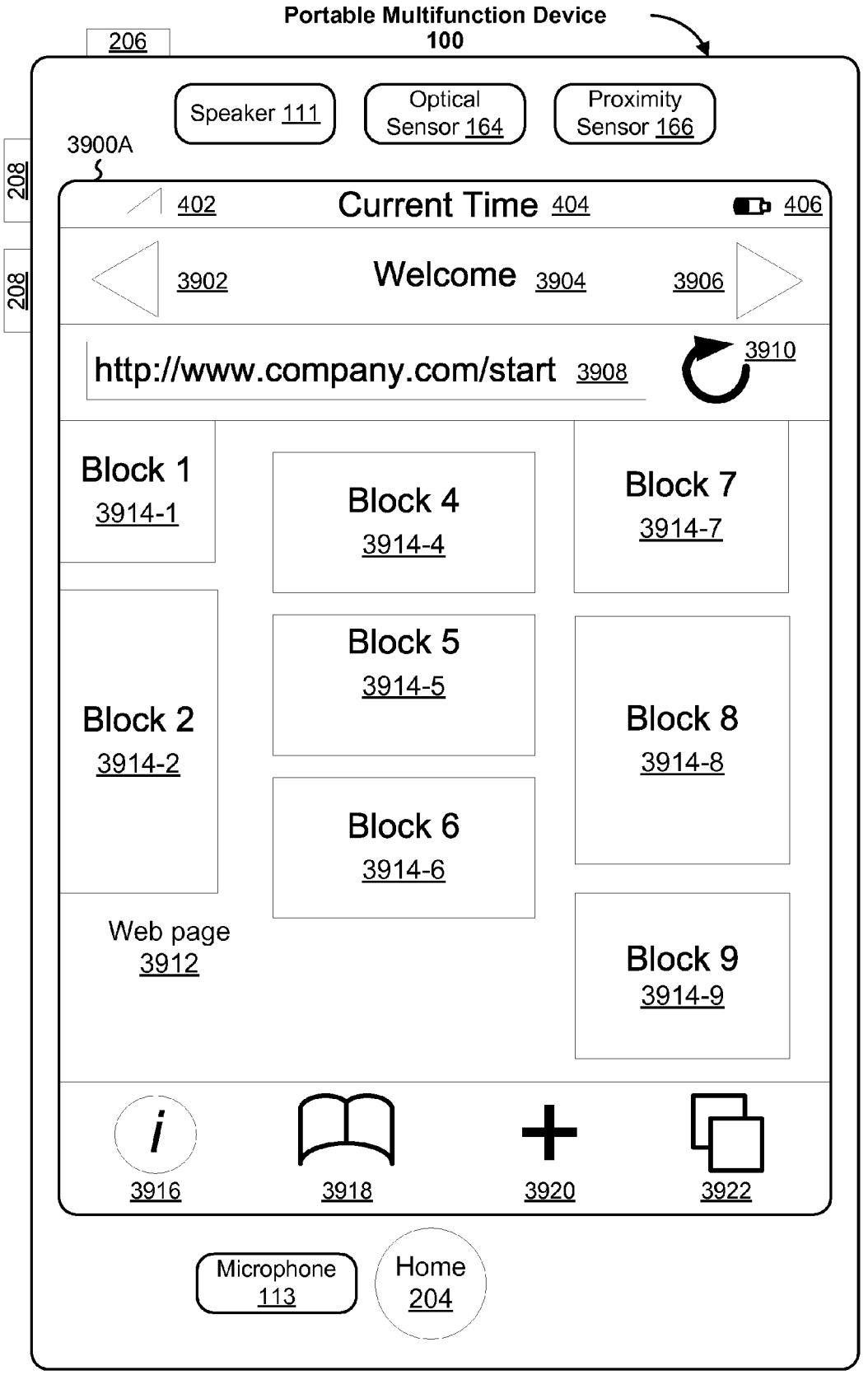

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Block 1
3914-1
Block 4
3914-4
Block 7
3914-7
Block 2
3914-2
Block 5
3914-5
Block 8
3914-8
Block 6
3914-6
Web page
3912
Block 9
3914-9
3916
3918
3920
3922
Microphone
113
Home
204

Figure 8D

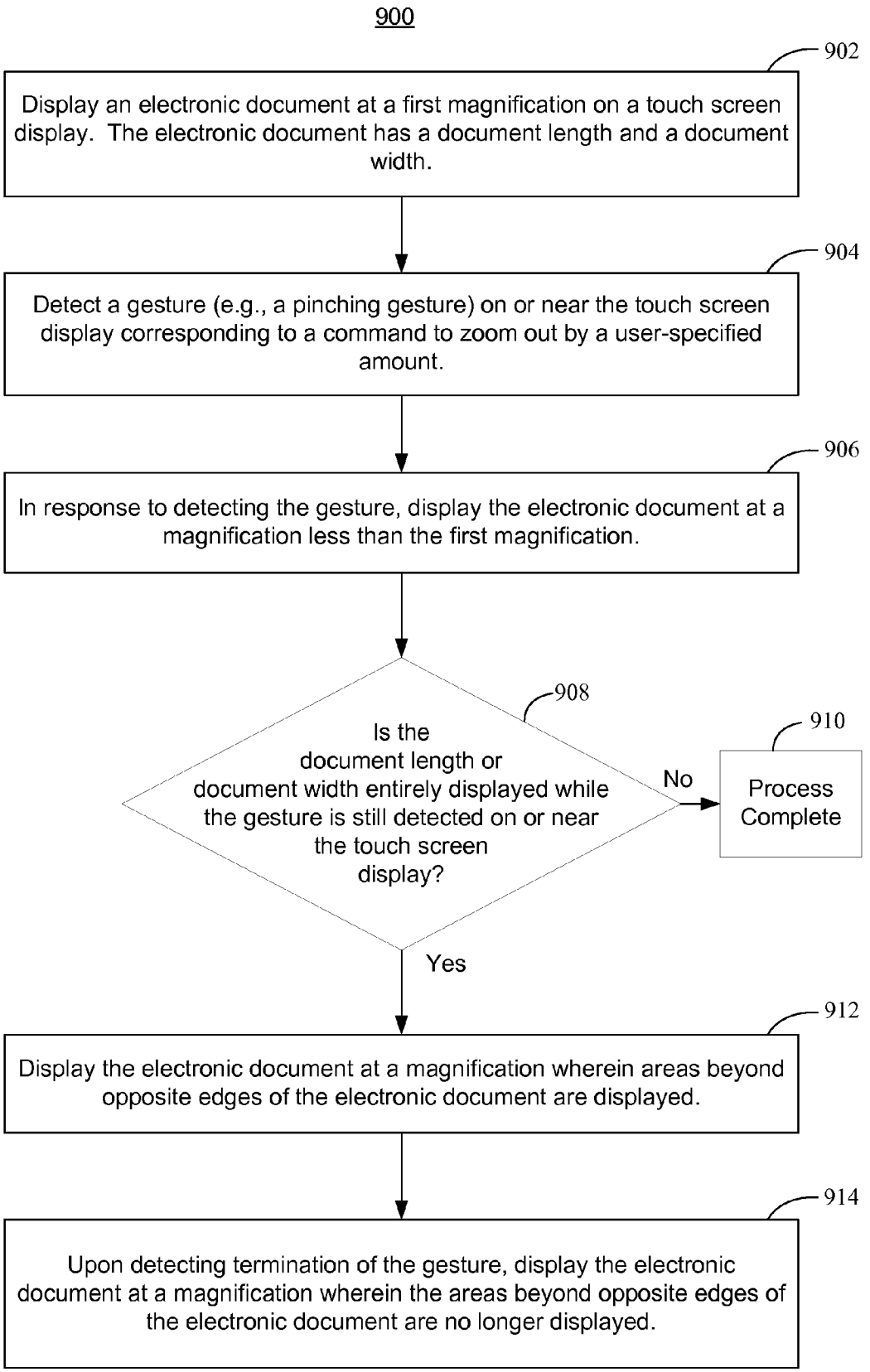

900
902
Display an electronic document at a first magnification on a touch screen display. The electronic document has a document length and a document width.
904
Detect a gesture (e.g., a pinching gesture) on or near the touch screen display corresponding to a command to zoom out by a user-specified amount.
906
In response to detecting the gesture, display the electronic document at a magnification less than the first magnification.
908
Is the document length or document width entirely displayed while the gesture is still detected on or near the touch screen display?
No
910
Process Complete
Yes
912
Display the electronic document at a magnification wherein areas beyond opposite edges of the electronic document are displayed.
914
Upon detecting termination of the gesture, display the electronic document at a magnification wherein the areas beyond opposite edges of the electronic document are no longer displayed.

Figure 9

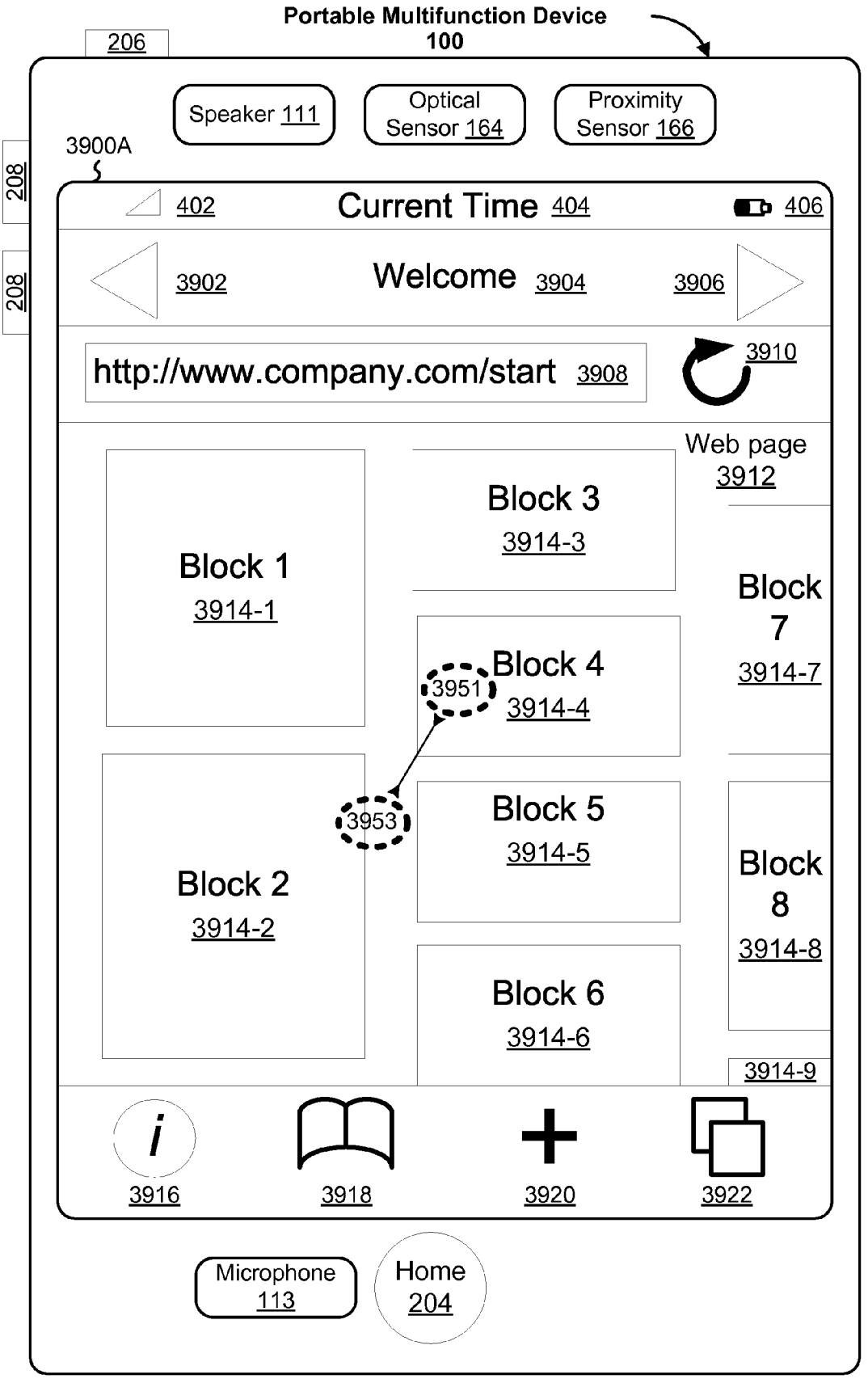

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Web page
3912
Block 1
3914-1
Block 3
3914-3
Block 4
3951
3914-4
Block 7
3914-7
3953
Block 5
3914-5
Block 2
3914-2
Block 8
3914-8
Block 6
3914-6
3914-9
3916
3918
3920
3922
Microphone
113
Home
204

Figure 10A

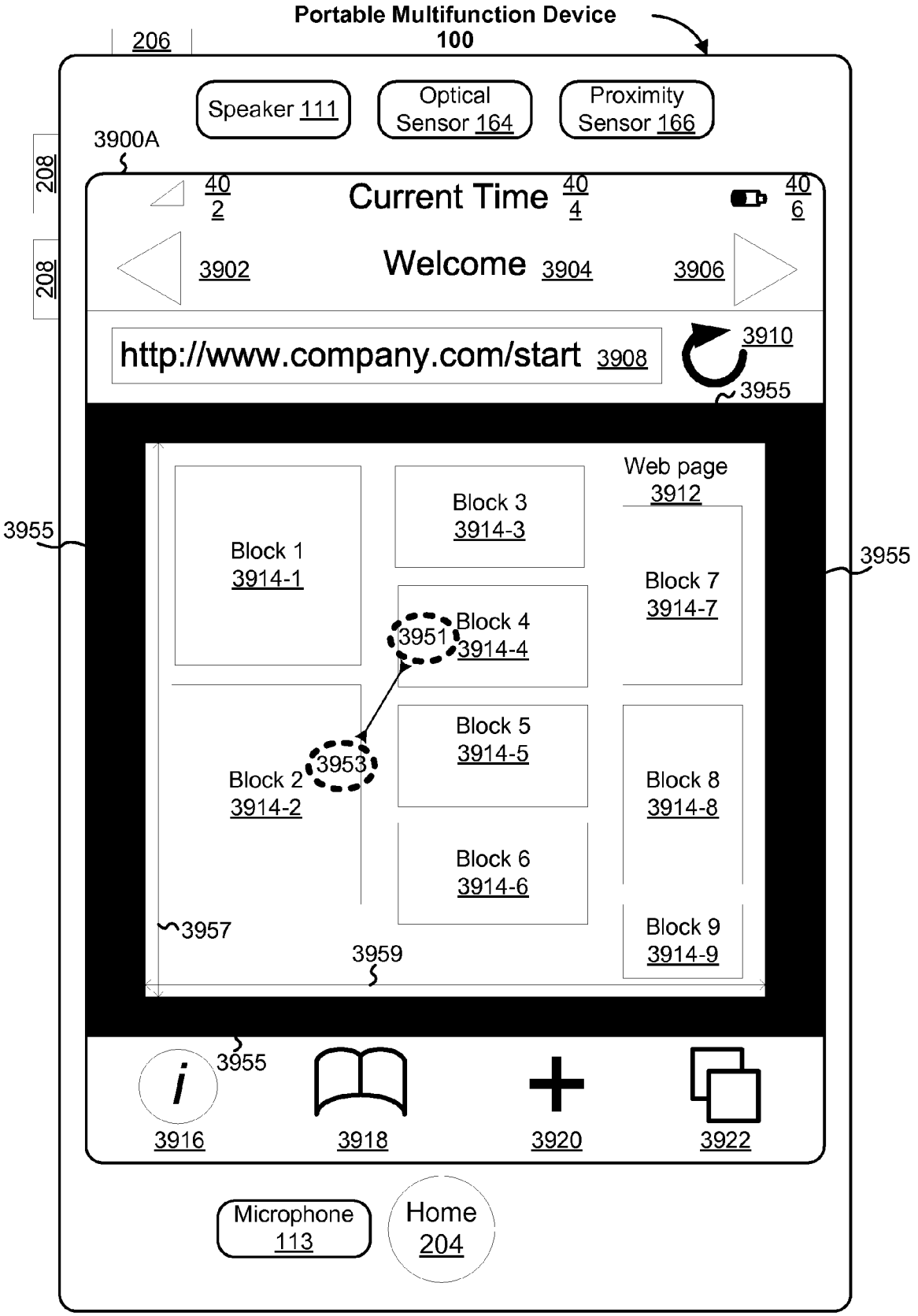

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
3955
3955
3955
Web page
3912
Block 1
3914-1
Block 3
3914-3
Block 7
3914-7
3951
Block 4
3914-4
Block 5
3914-5
3953
Block 2
3914-2
Block 8
3914-8
Block 6
3914-6
Block 9
3914-9
3957
3959
3955
3916
3918
3920
3922
Microphone
113
Home
204

Figure 10B

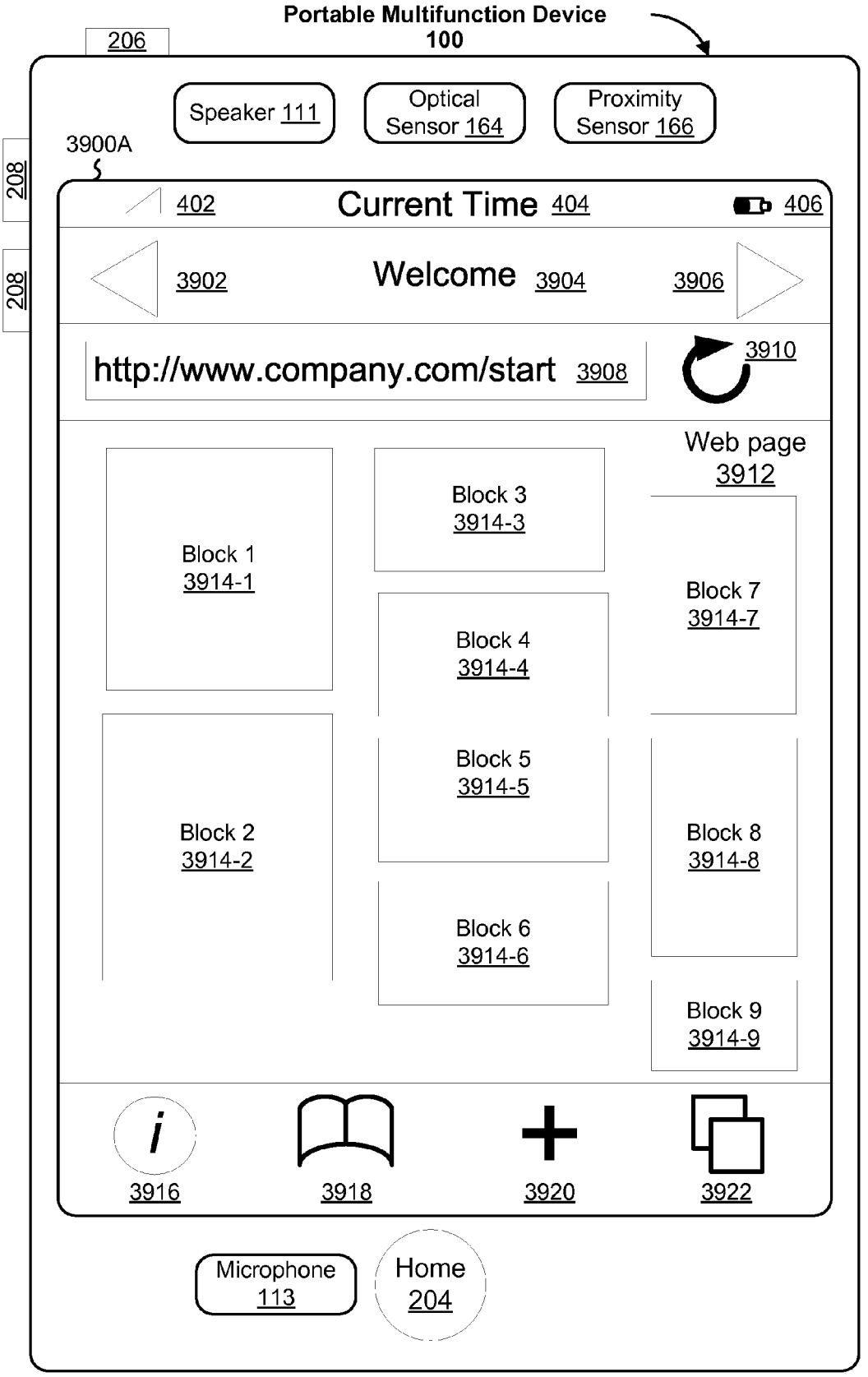

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Web page
3912
Block 1
3914-1
Block 2
3914-2
Block 3
3914-3
Block 4
3914-4
Block 5
3914-5
Block 6
3914-6
Block 7
3914-7
Block 8
3914-8
Block 9
3914-9
3916
3918
3920
3922
Microphone
113
Home
204

Figure 10C

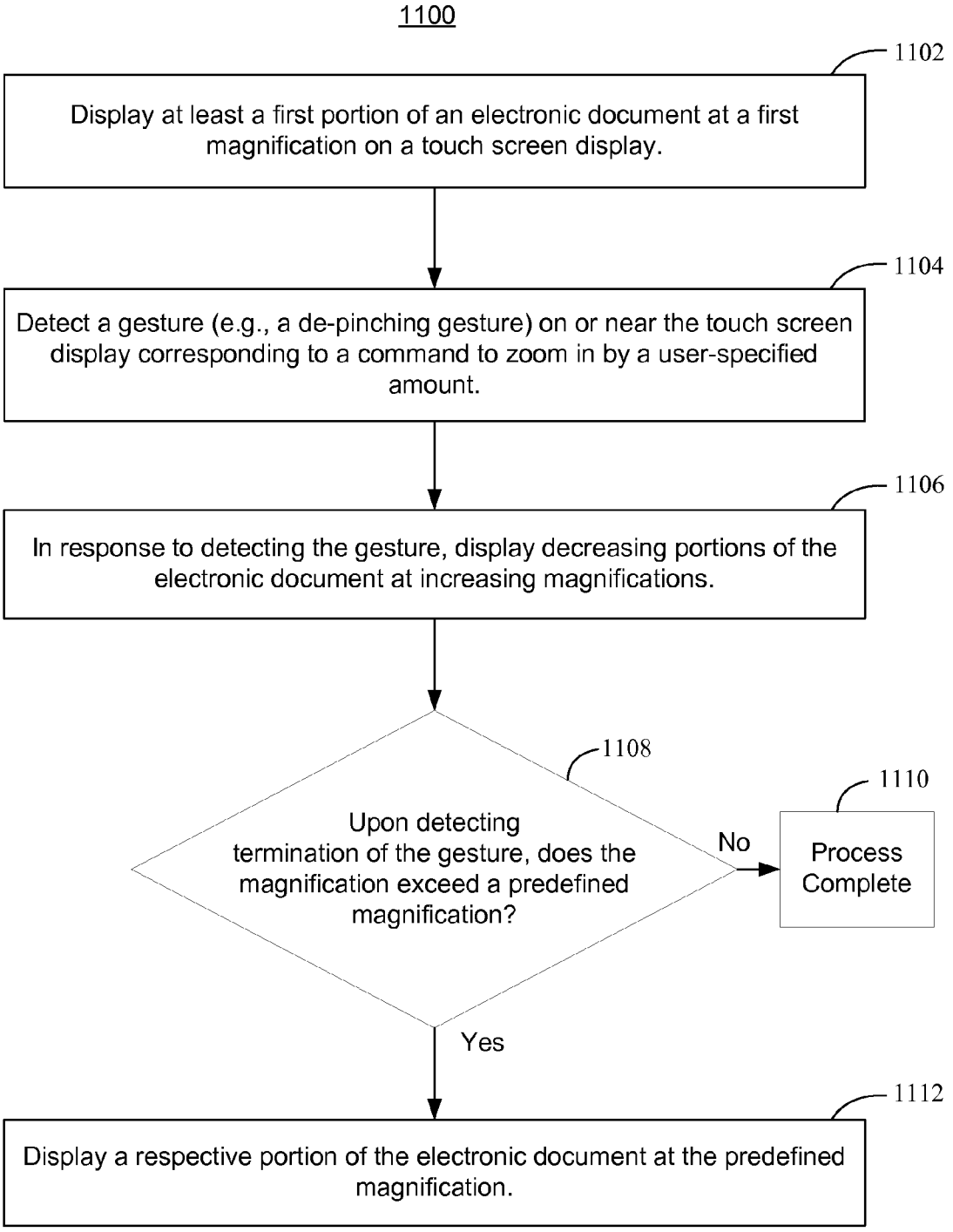

1100
1102
Display at least a first portion of an electronic document at a first magnification on a touch screen display.
1104
Detect a gesture (e.g., a de-pinching gesture) on or near the touch screen display corresponding to a command to zoom in by a user-specified amount.
1106
In response to detecting the gesture, display decreasing portions of the electronic document at increasing magnifications.
1108
Upon detecting termination of the gesture, does the magnification exceed a predefined magnification?
No
1110
Process Complete
Yes
1112
Display a respective portion of the electronic document at the predefined magnification.

Figure 11

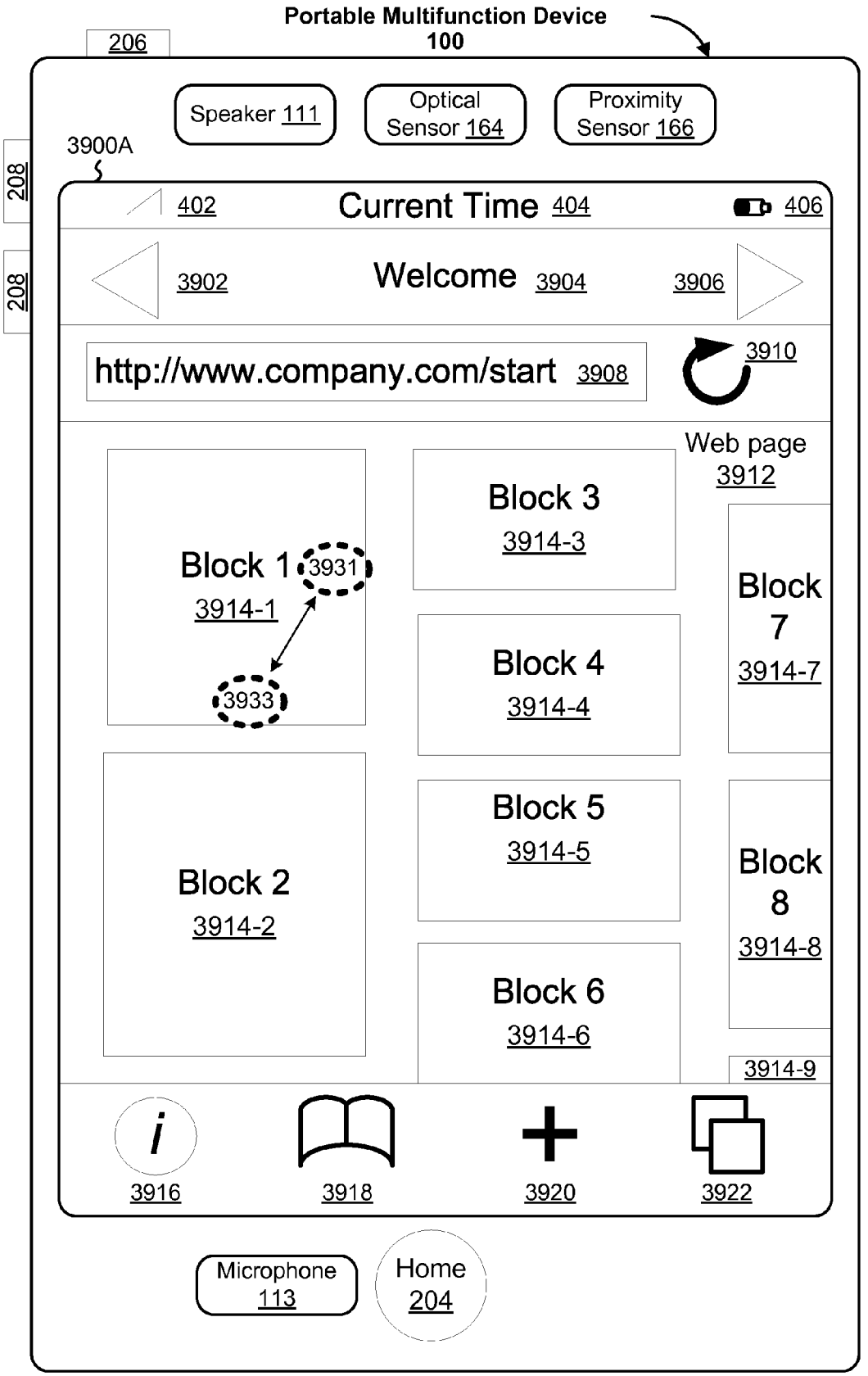

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Web page
3912
Block 1
3914-1
3931
3933
Block 3
3914-3
Block 4
3914-4
Block 7
3914-7
Block 2
3914-2
Block 5
3914-5
Block 8
3914-8
Block 6
3914-6
3914-9
3916
3918
3920
3922
Microphone
113
Home
204

Figure 12A

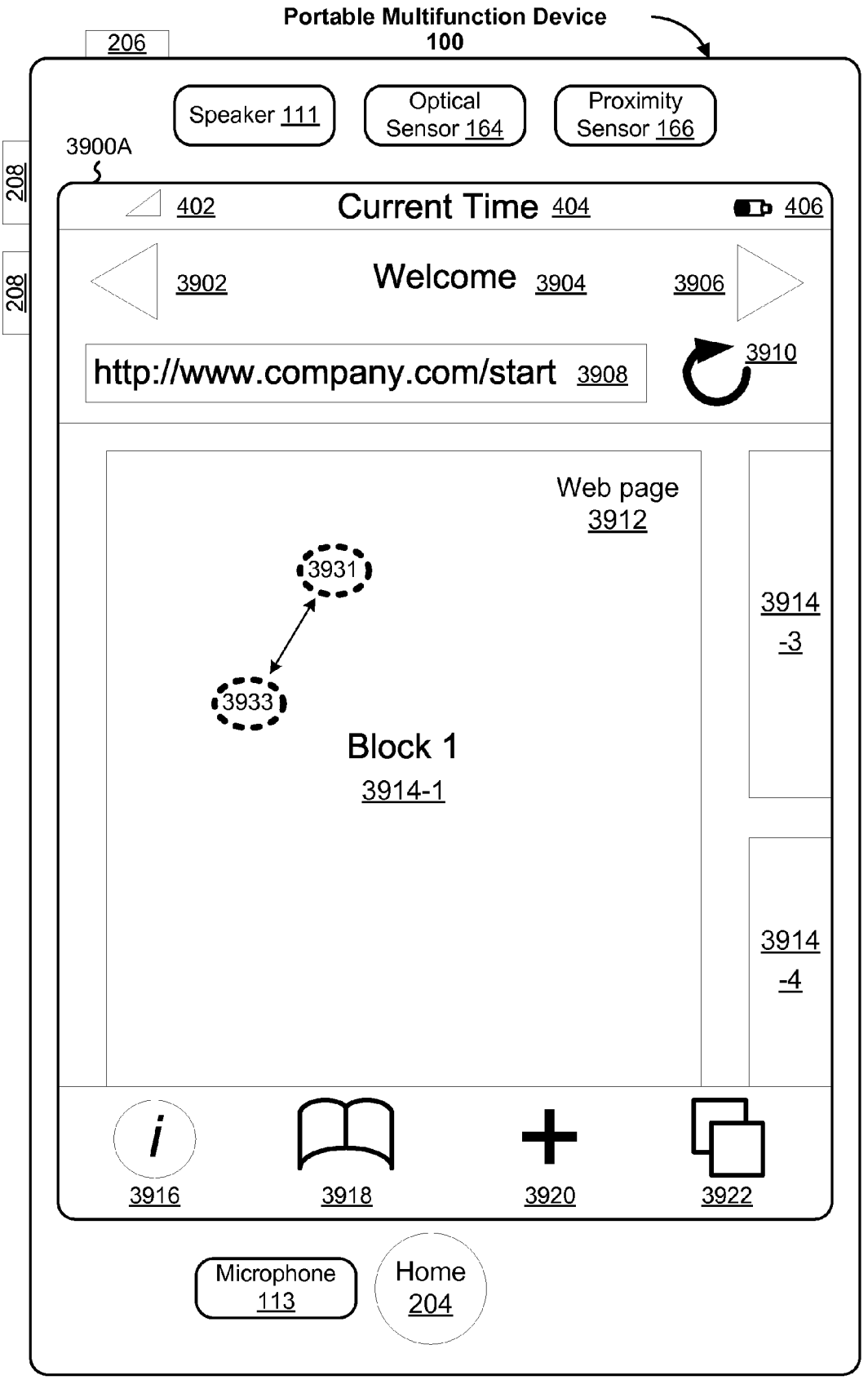

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Web page
3912
3931
3933
Block 1
3914-1
3914
-3
3914
-4
3916
3918
3920
3922
Microphone
113
Home
204

Figure 12B

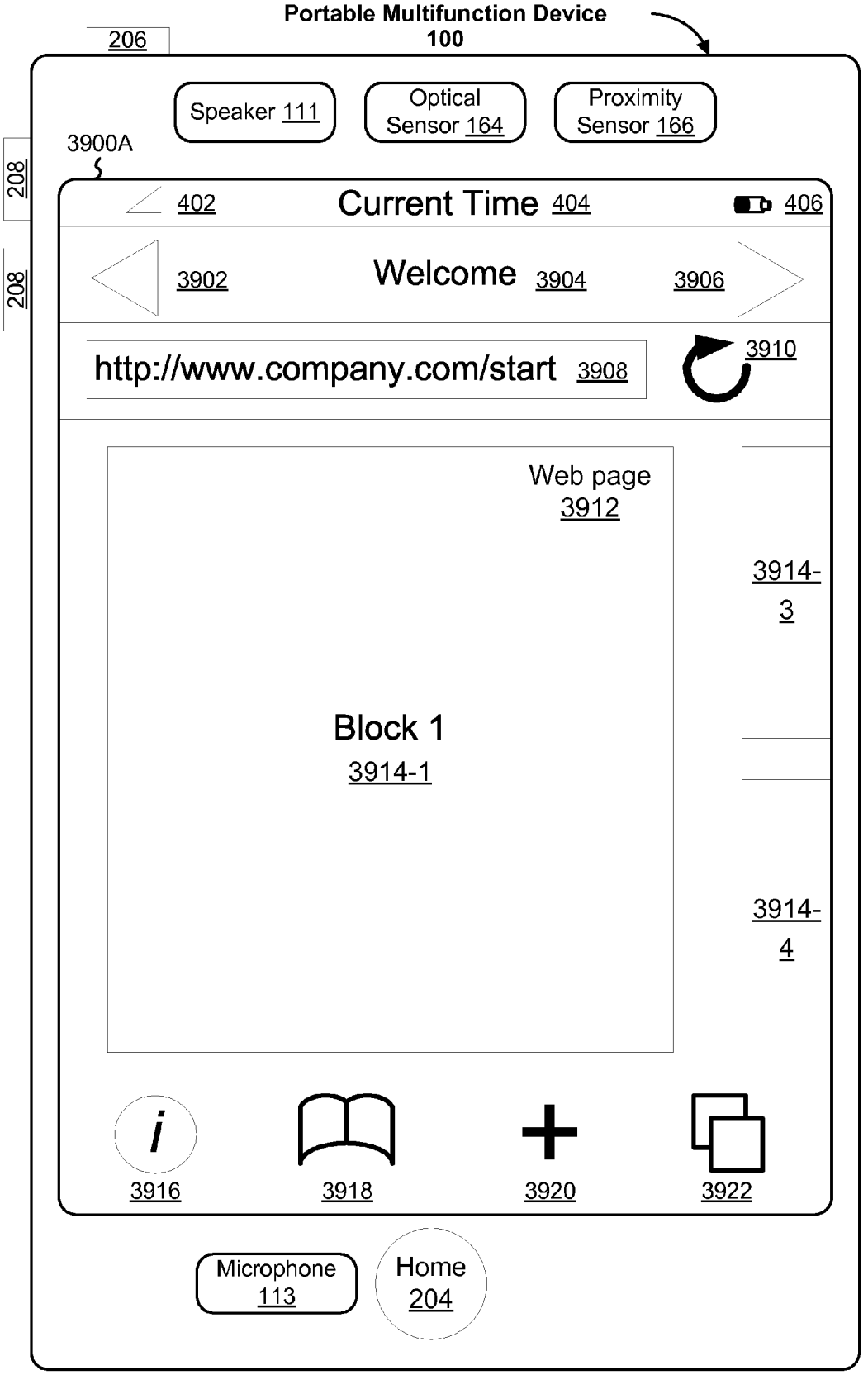

Portable Multifunction Device
100
206
Speaker 111
Optical Sensor 164
Proximity Sensor 166
3900A
208
208
402
Current Time 404
406
3902
Welcome 3904
3906
http://www.company.com/start 3908
3910
Web page
3912
3914-3
Block 1
3914-1
3914-4
3916
3918
3920
3922
Microphone 113
Home 204

Figure 12C

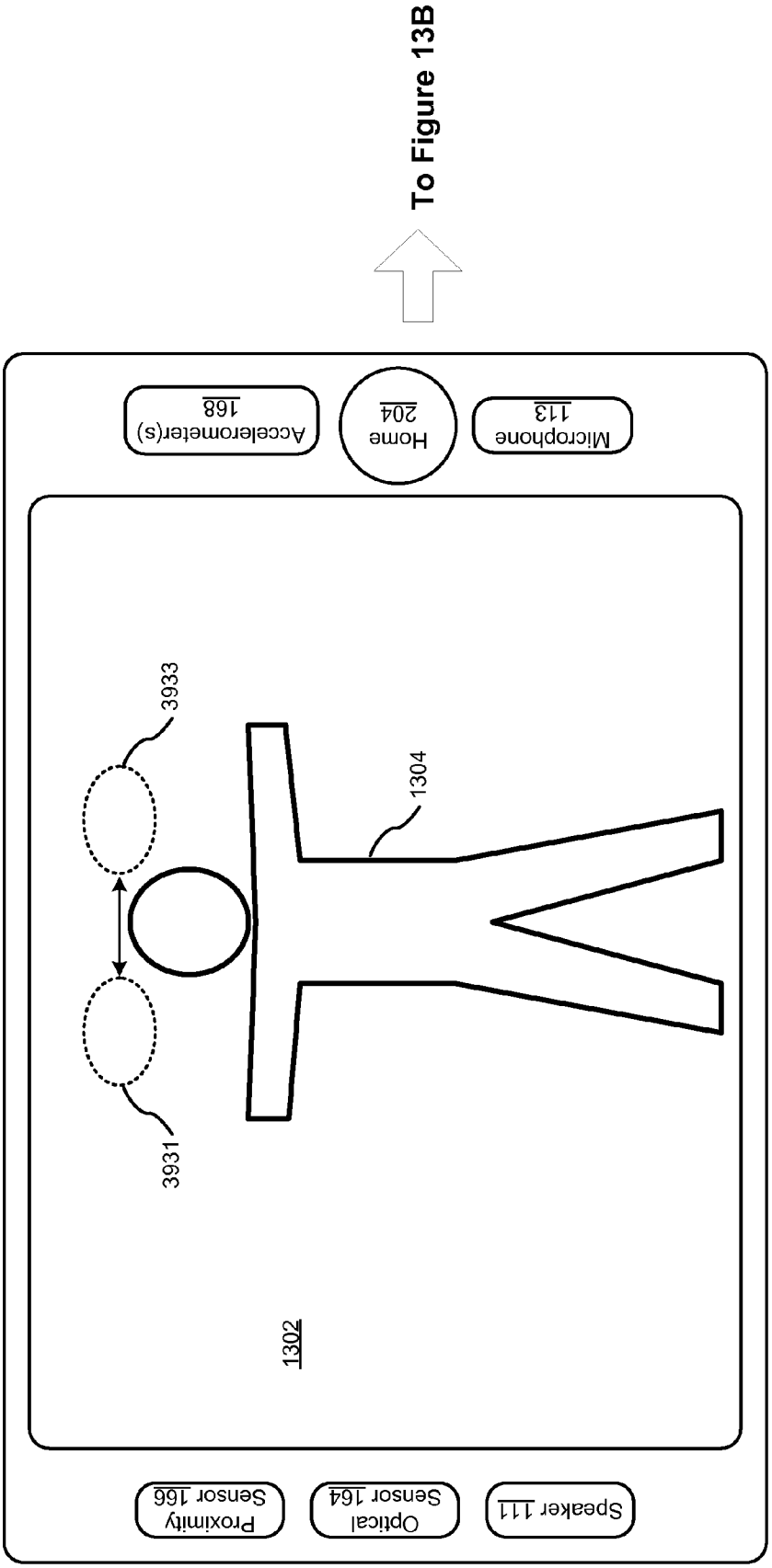

To Figure 13B
Microphone 113
Home 204
Accelerometer(s) 168
3933
1304
3931
1302
Speaker 111
Optical Sensor 164
Proximity Sensor 166

Figure 13A

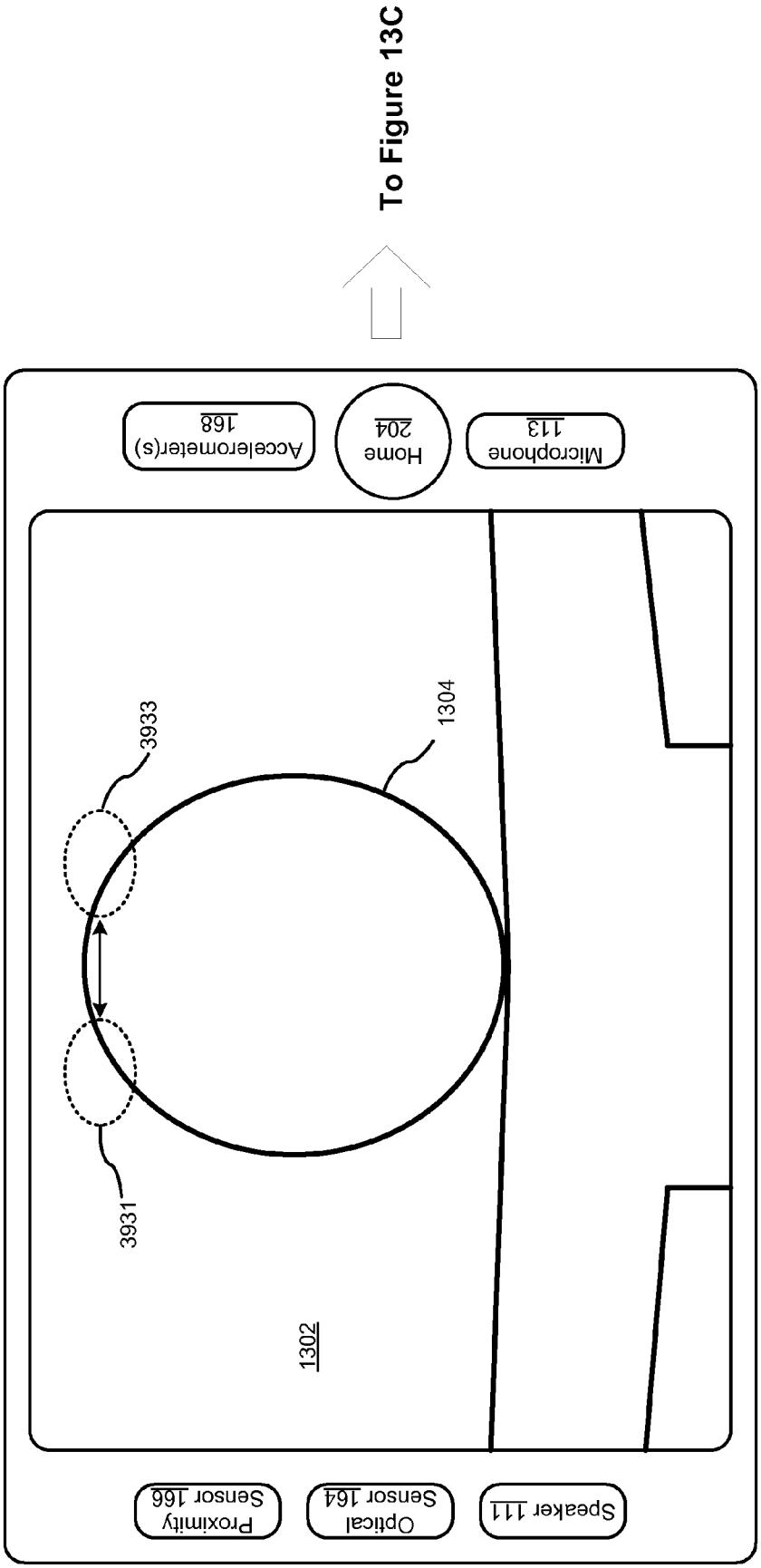

To Figure 13C
Speaker 111
Optical Sensor 164
Proximity Sensor 166
1302
3931
3933
1304
Microphone 113
Home 204
Accelerometer(s) 168

Figure 13B

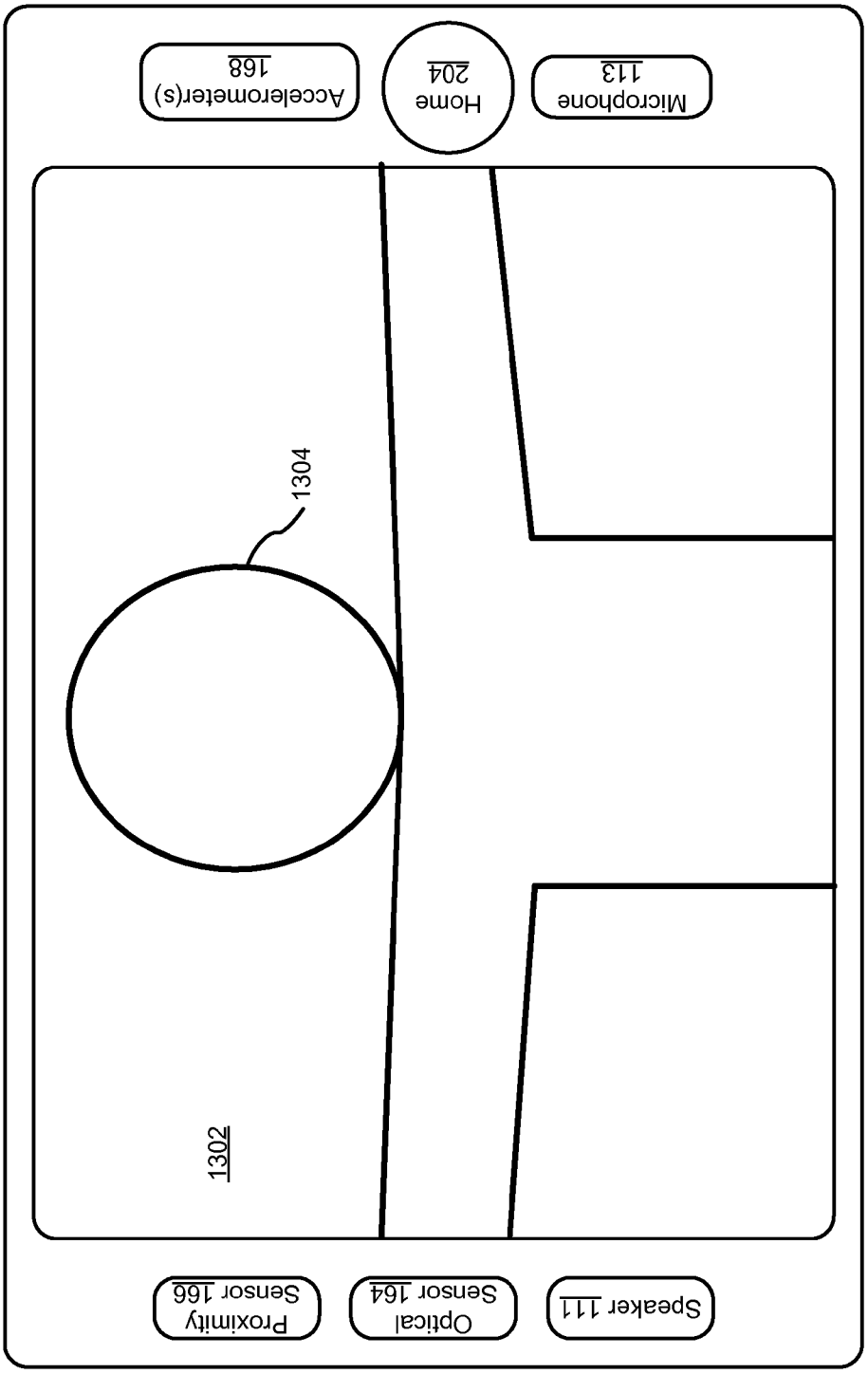

Speaker 111
Optical Sensor 164
Proximity Sensor 166
1302
1304
Microphone 113
Home 204
Accelerometer(s) 168

Figure 13C

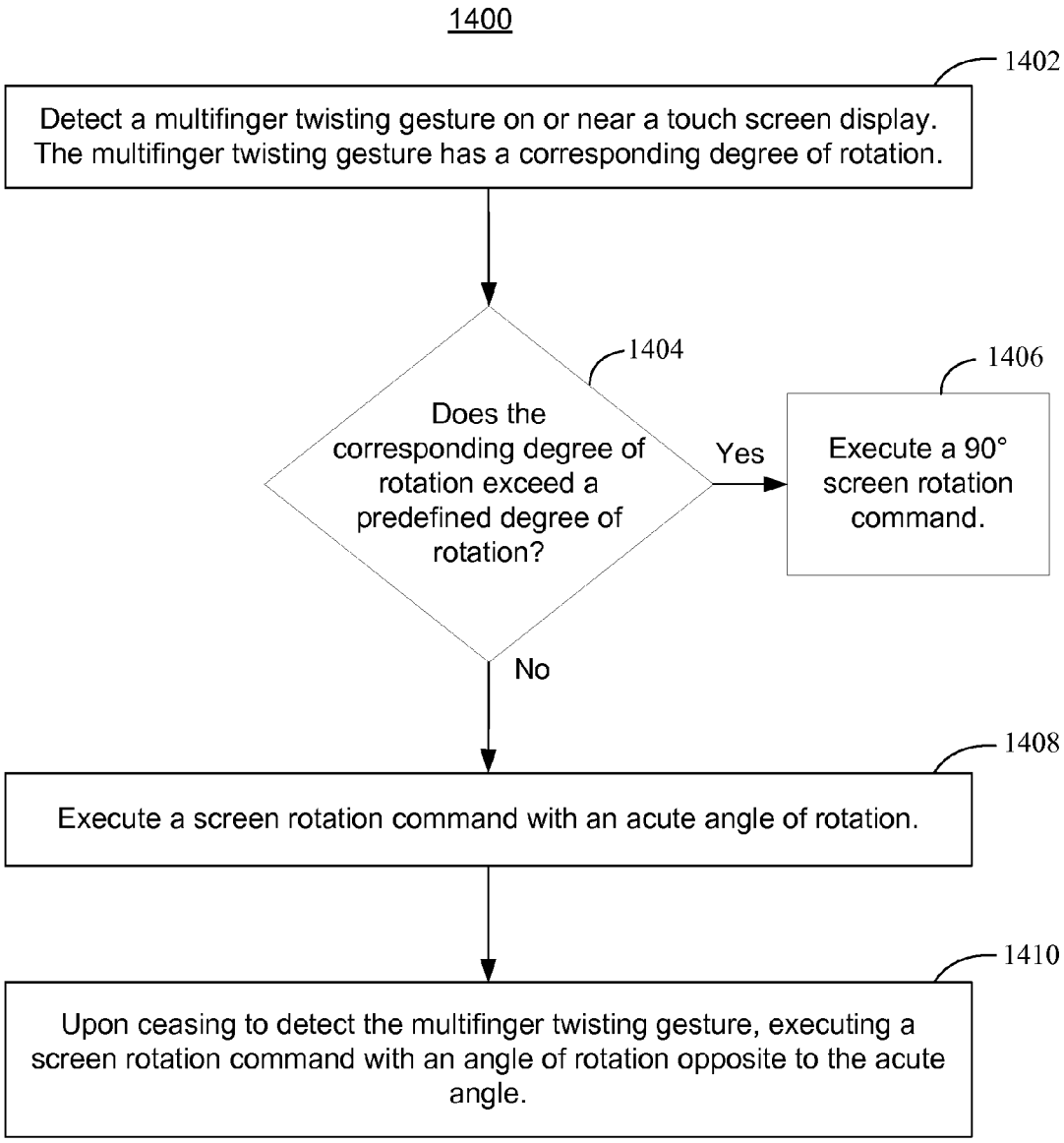

1400
1402
Detect a multifinger twisting gesture on or near a touch screen display. The multifinger twisting gesture has a corresponding degree of rotation.
1404
Does the corresponding degree of rotation exceed a predefined degree of rotation?
Yes
1406
Execute a 90° screen rotation command.
No
1408
Execute a screen rotation command with an acute angle of rotation.
1410
Upon ceasing to detect the multifinger twisting gesture, executing a screen rotation command with an angle of rotation opposite to the acute angle.

Figure 14

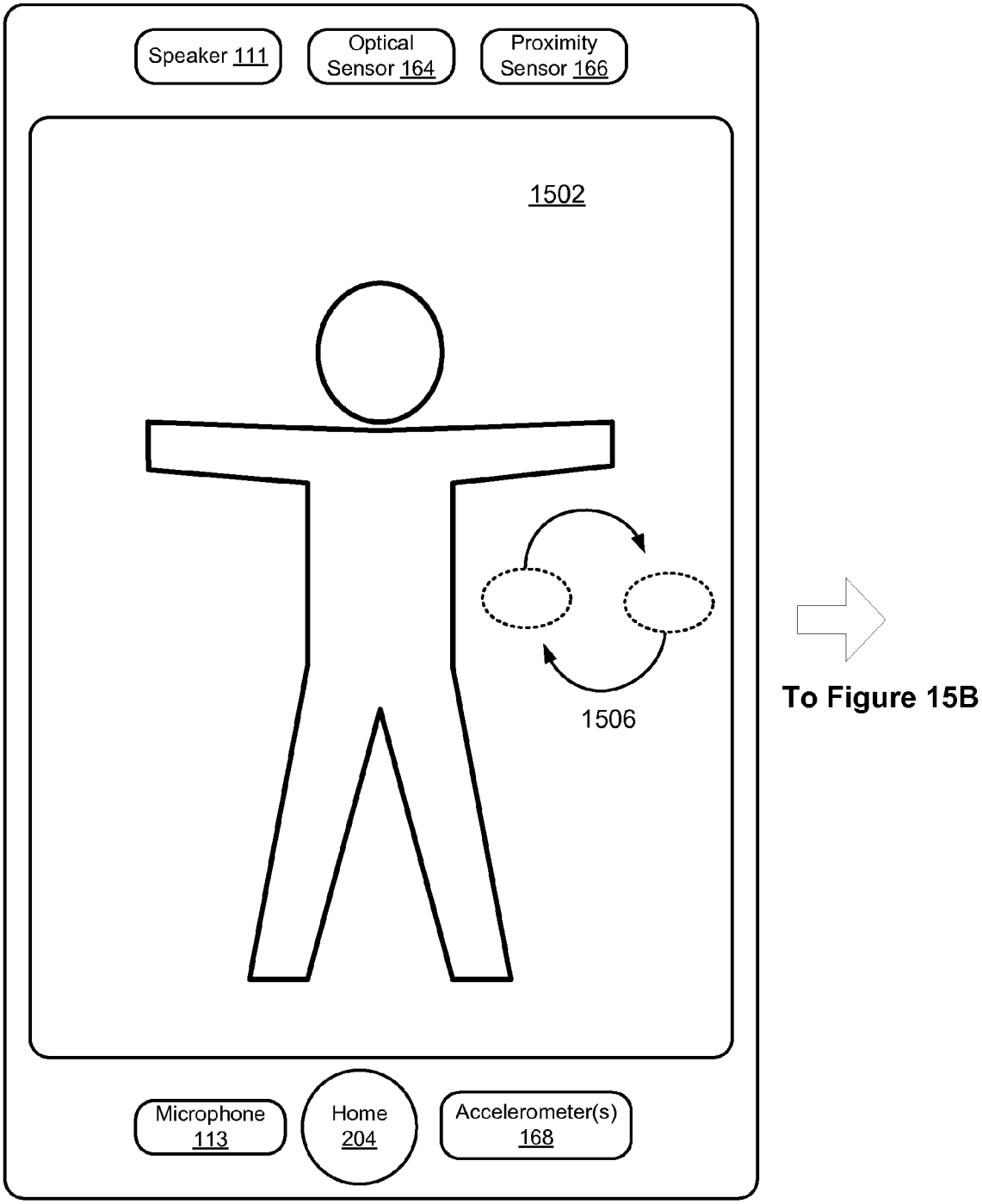

Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
1506
Microphone 113
Home 204
Accelerometer(s) 168
To Figure 15B

Figure 15A

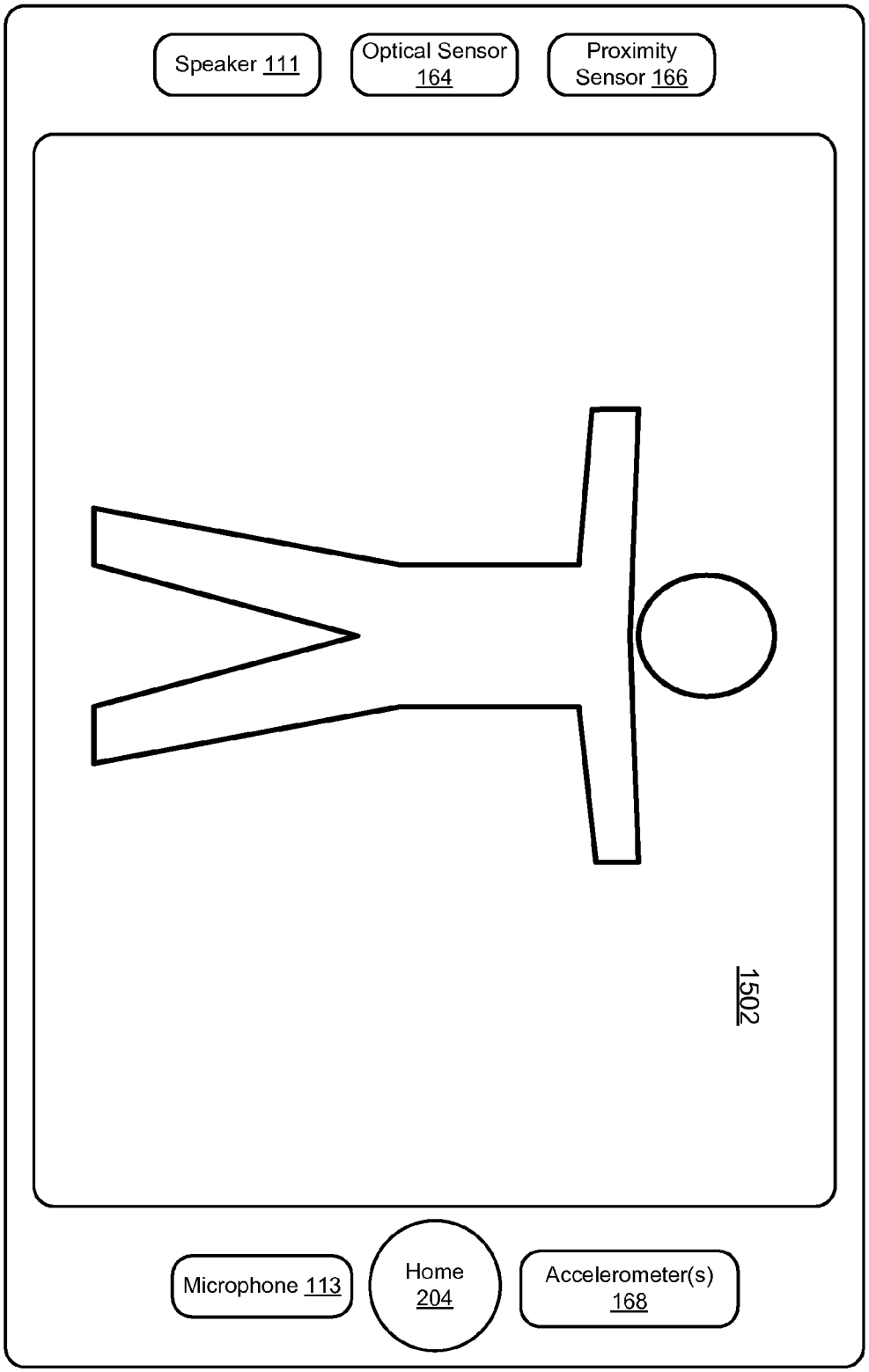

Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
Microphone 113
Home 204
Accelerometer(s) 168

Figure 15B

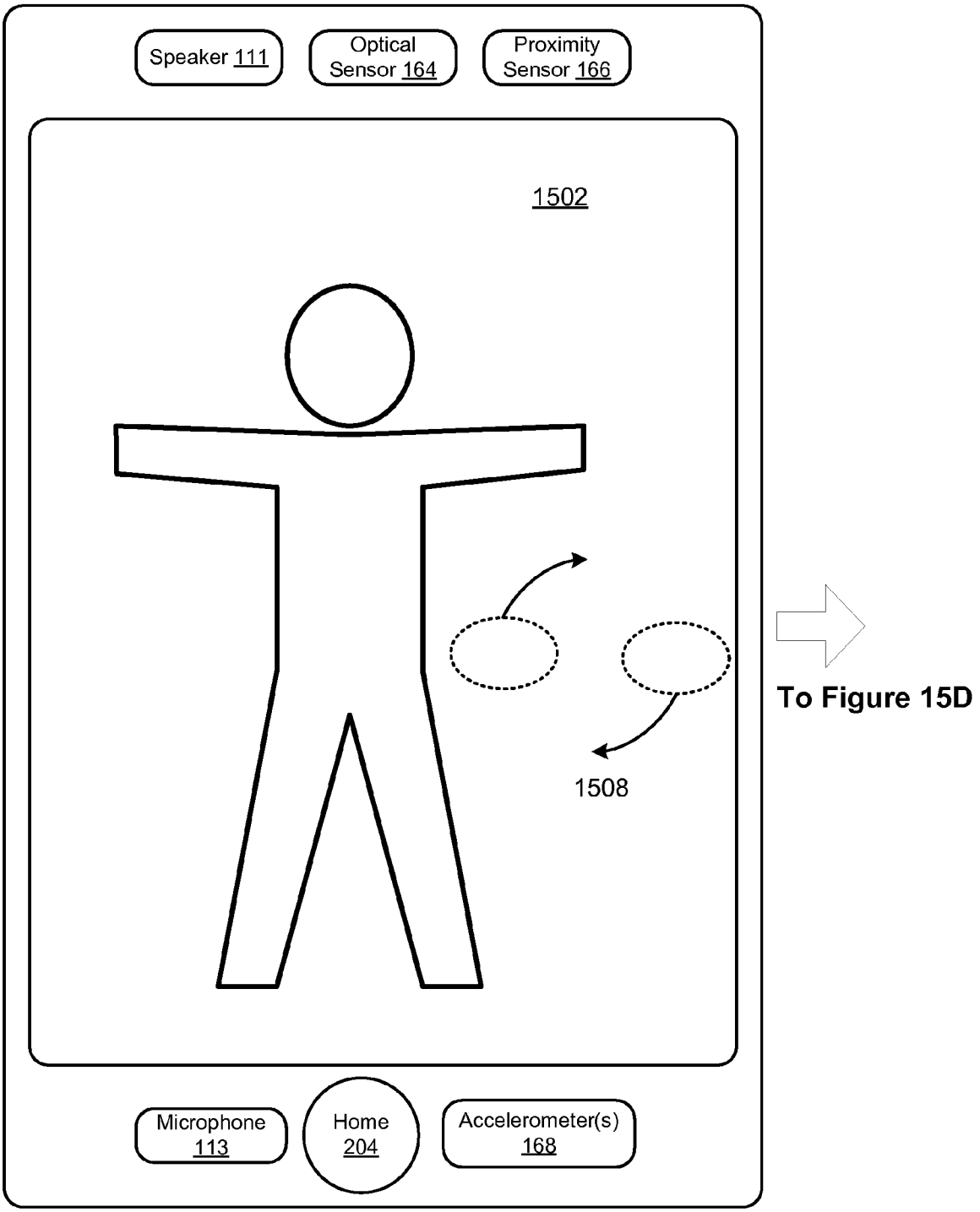

Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
1508
To Figure 15D
Microphone 113
Home 204
Accelerometer(s) 168

Figure 15C

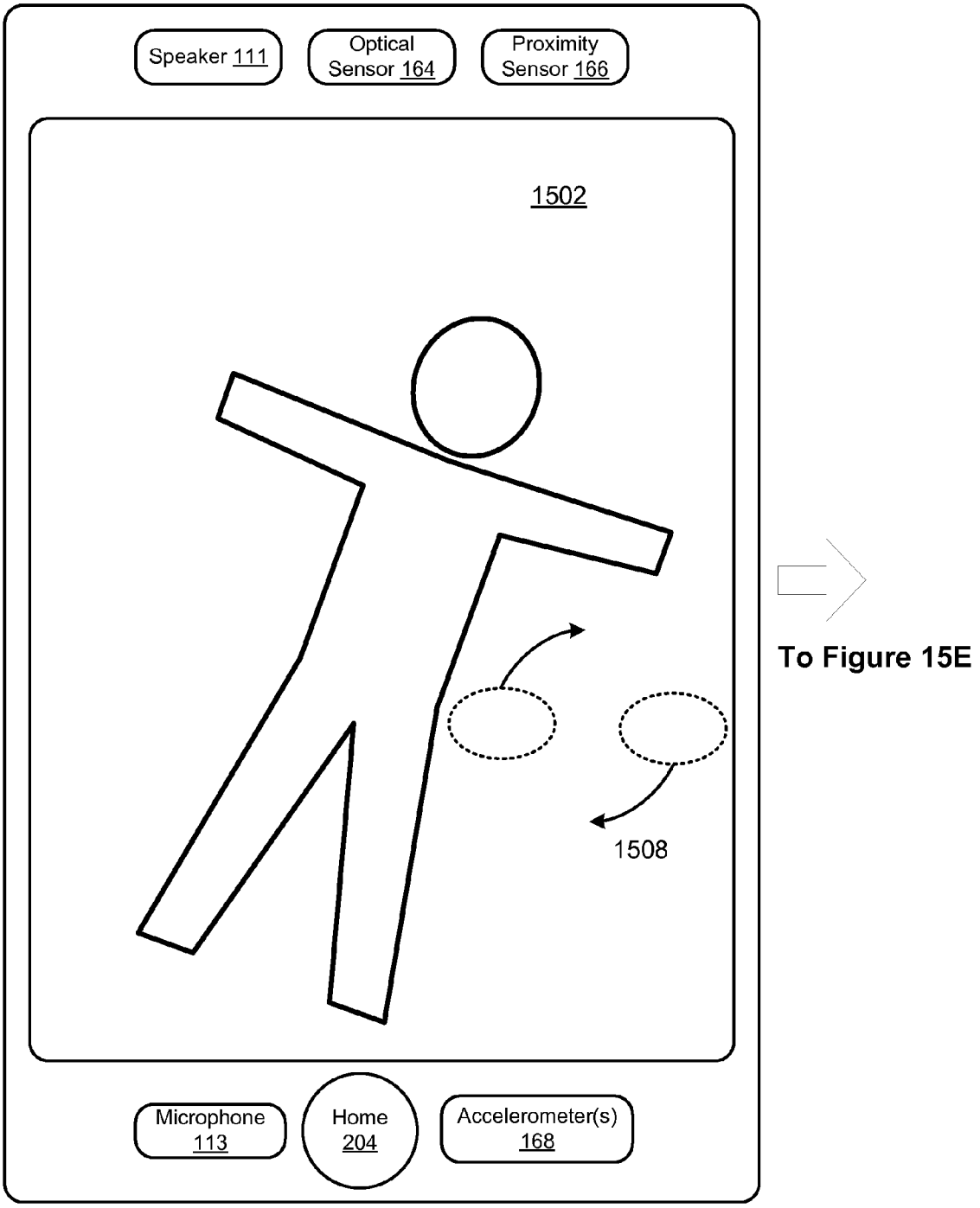

Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
1508
Microphone 113
Home 204
Accelerometer(s) 168
To Figure 15E

Figure 15D

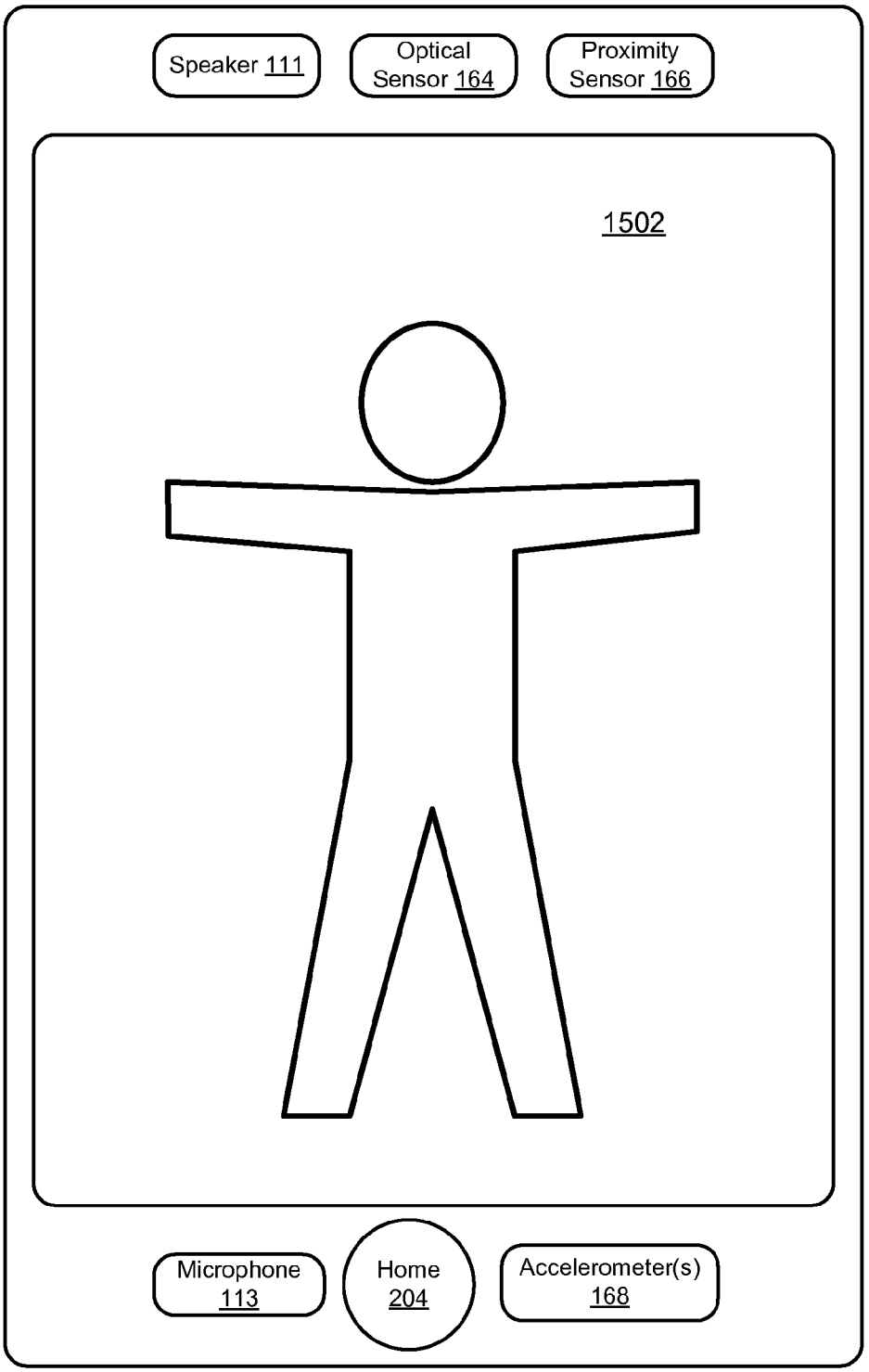

Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
Microphone 113
Home 204
Accelerometer(s) 168

Figure 15E

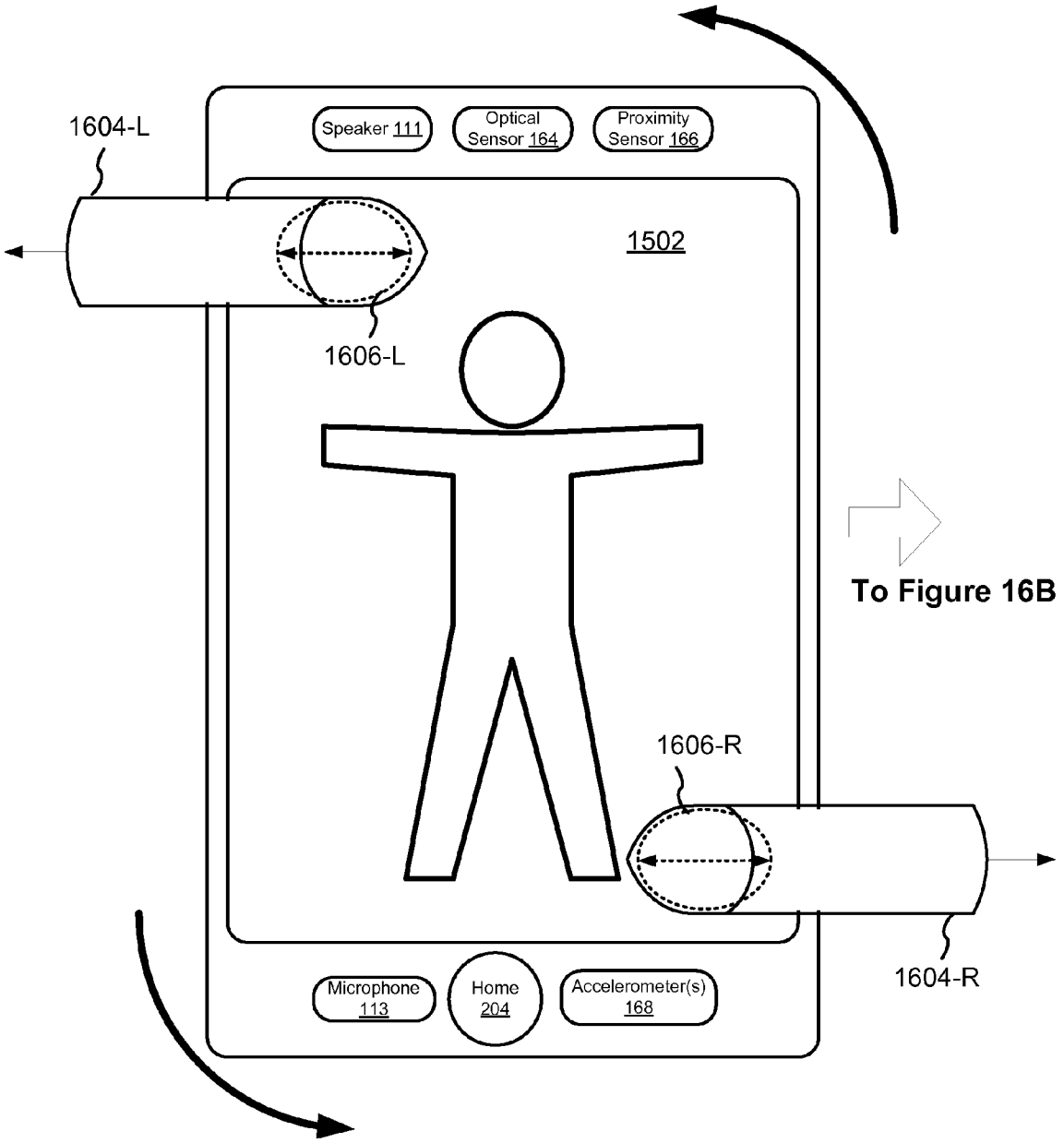

1604-L
Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
1606-L
To Figure 16B
1606-R
1604-R
Microphone 113
Home 204
Accelerometer(s) 168

Figure 16A

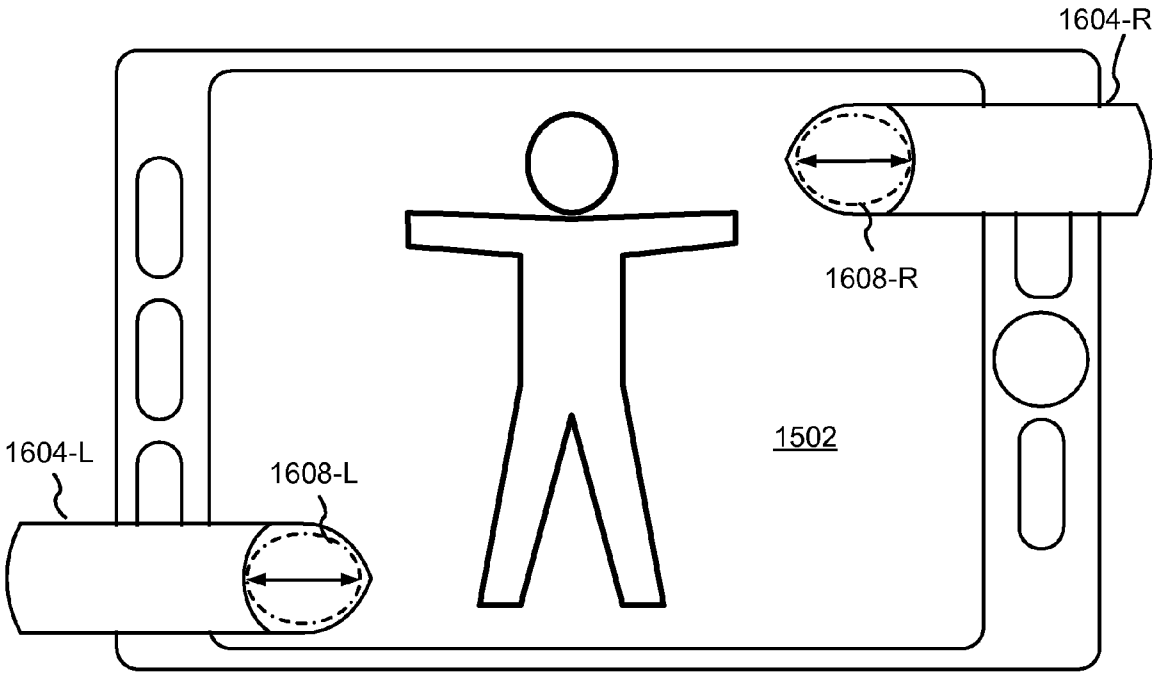


Figure 16B

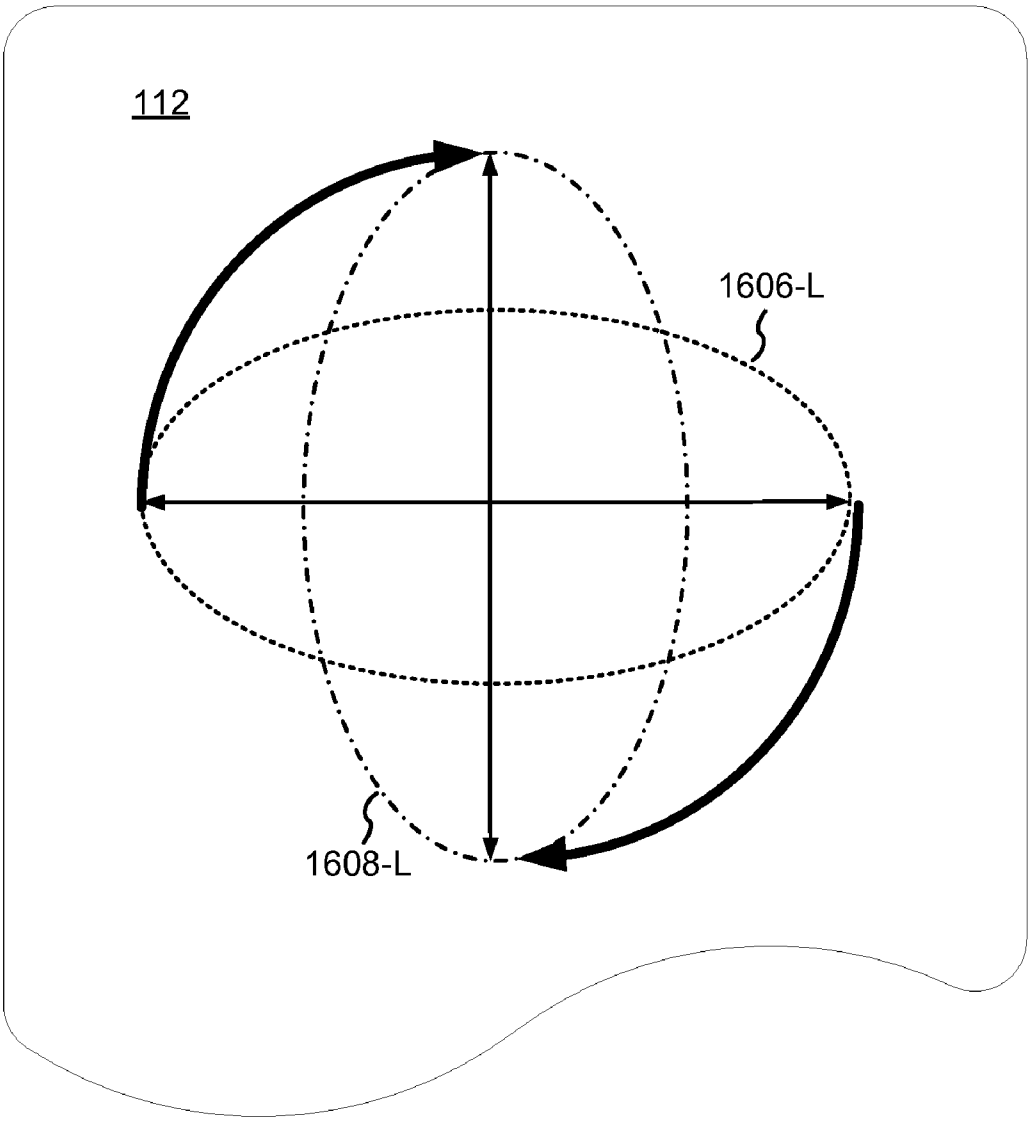


Figure 16C

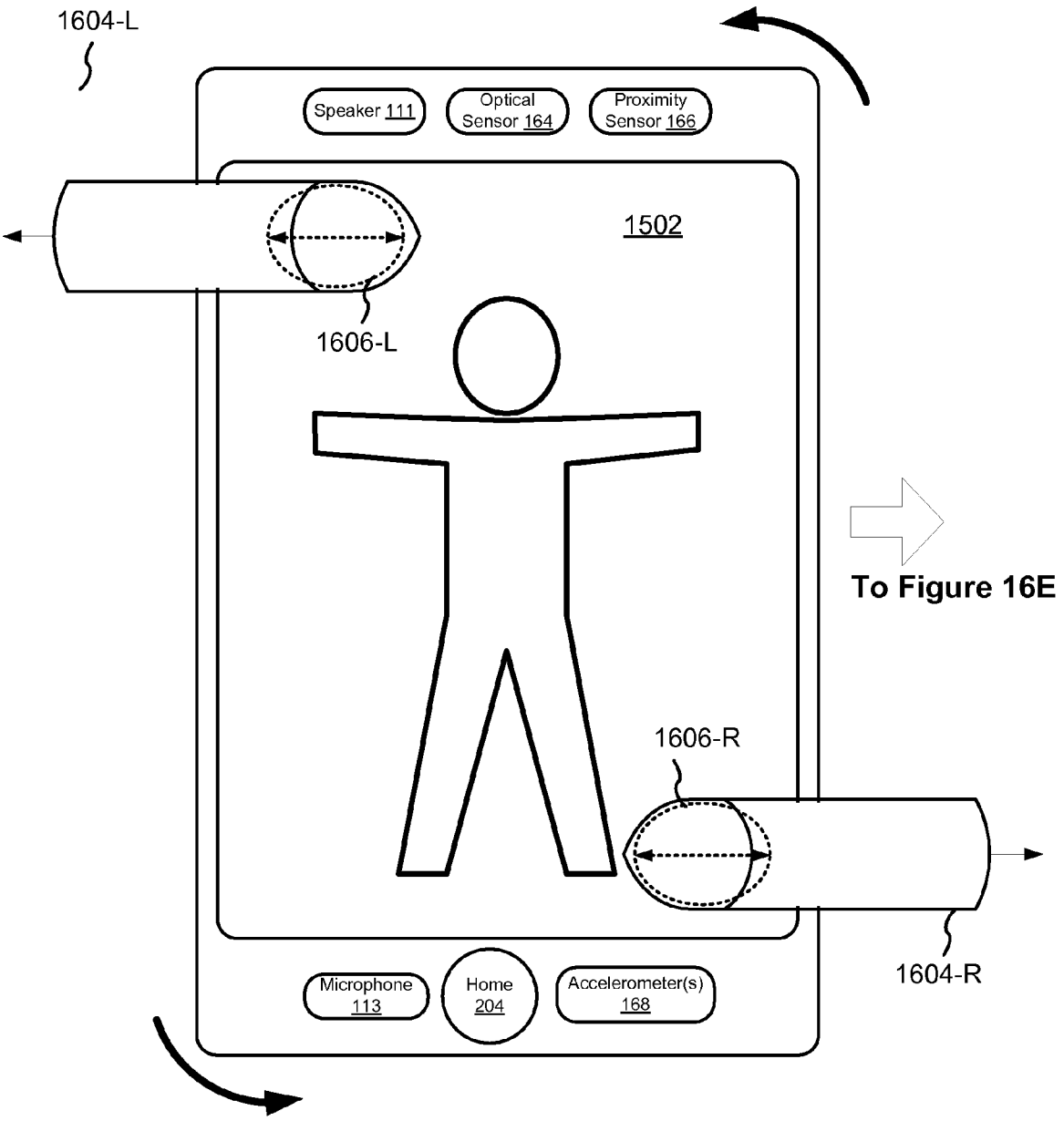

1604-L
Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
1606-L
To Figure 16E
1606-R
1604-R
Microphone 113
Home 204
Accelerometer(s) 168

Figure 16D

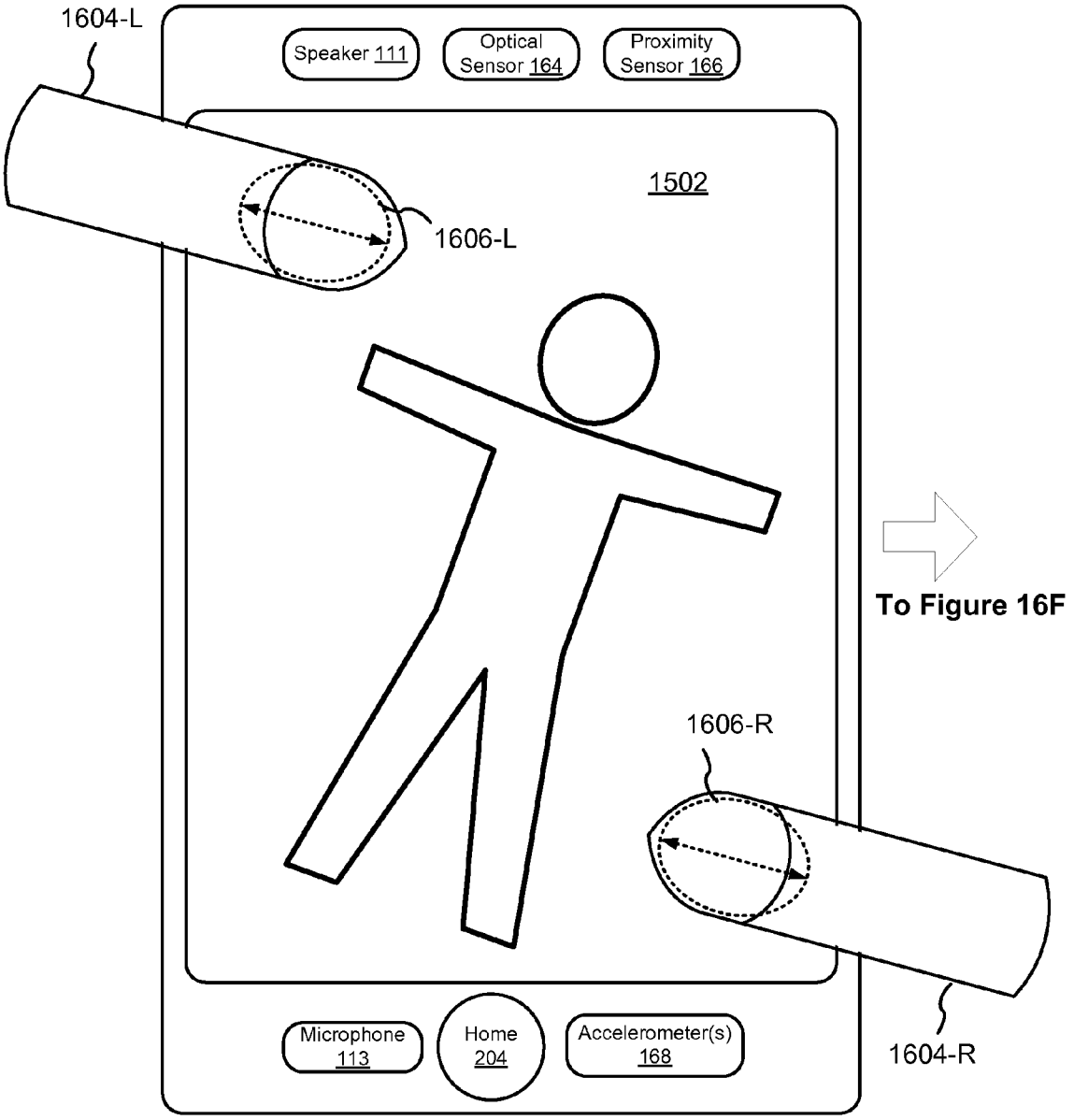

1604-L
Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
1606-L
To Figure 16F
1606-R
Microphone 113
Home 204
Accelerometer(s) 168
1604-R

Figure 16E

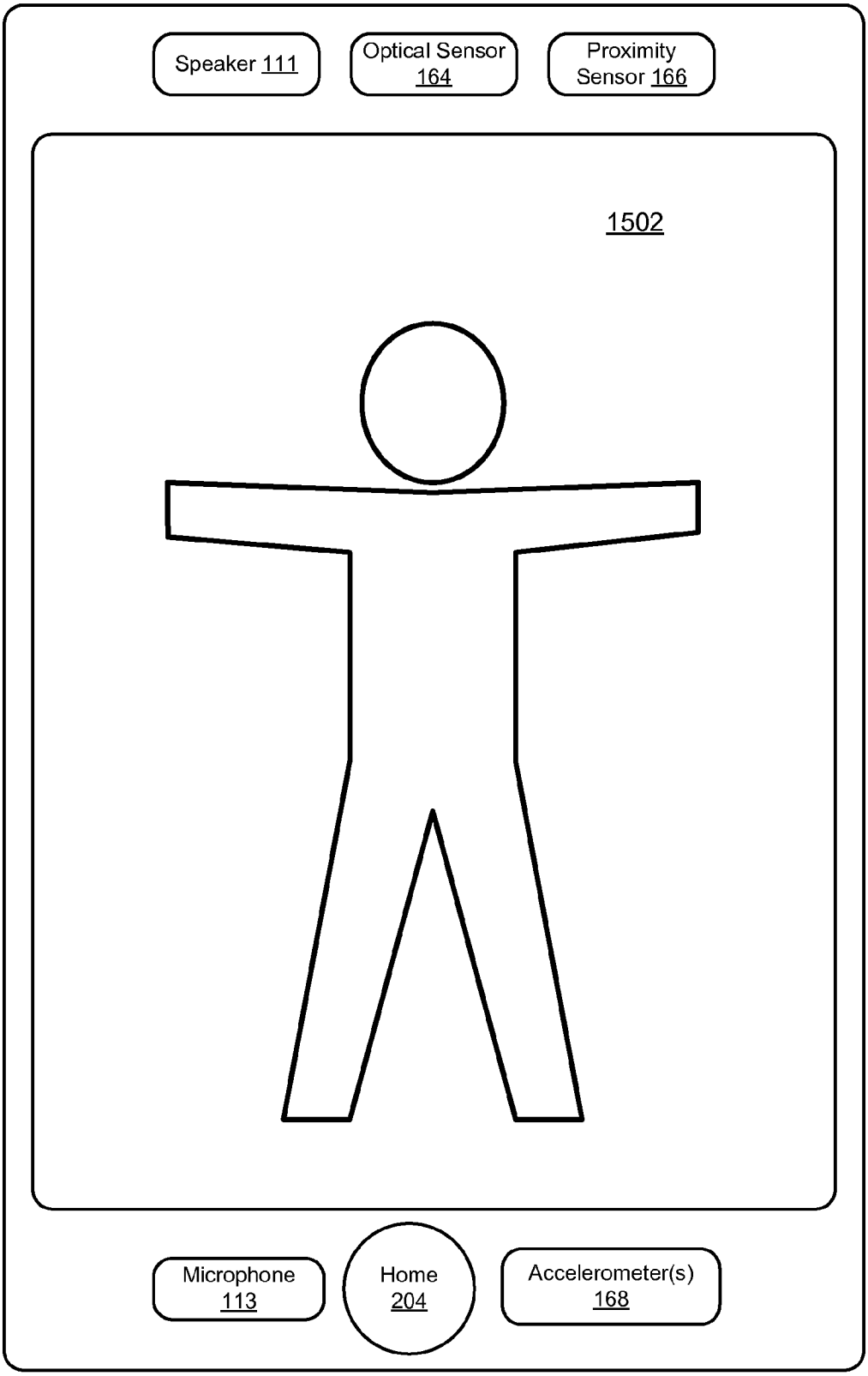

Speaker 111
Optical Sensor 164
Proximity Sensor 166
1502
Microphone 113
Home 204
Accelerometer(s) 168

Figure 16F

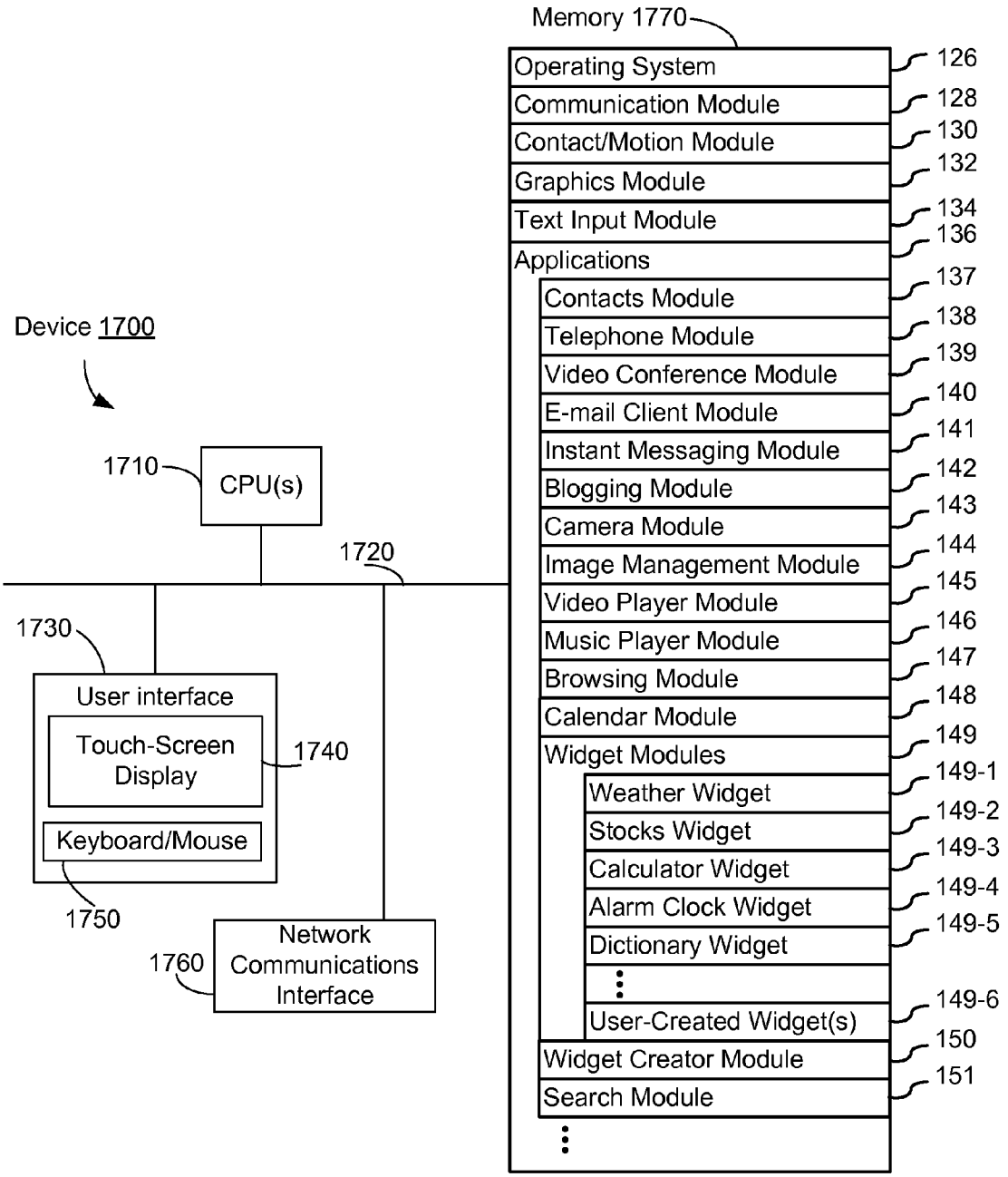

Memory 1770
Operating System
126
Communication Module
128
Contact/Motion Module
130
Graphics Module
132
Text Input Module
134
Applications
136
Contacts Module
137
Telephone Module
138
Video Conference Module
139
E-mail Client Module
140
Instant Messaging Module
141
Blogging Module
142
Camera Module
143
Image Management Module
144
Video Player Module
145
Music Player Module
146
Browsing Module
147
Calendar Module
148
Widget Modules
149
Weather Widget
149-1
Stocks Widget
149-2
Calculator Widget
149-3
Alarm Clock Widget
149-4
Dictionary Widget
149-5
User-Created Widget(s)
149-6
Widget Creator Module
150
Search Module
151
Device 1700
1710
CPU(s)
1720
1730
User interface
Touch-Screen Display
1740
Keyboard/Mouse
1750
1760
Network Communications Interface

Figure 17

LIST SCROLLING AND DOCUMENT TRANSLATION, SCALING, AND ROTATION ON A TOUCH-SCREEN DISPLAY

RELATED APPLICATIONS

This application claims priority to U.S. Provisional Patent Application Nos. 60/937,993, "Portable Multifunction Device," filed Jun. 29, 2007; 60/946,971, "List Scrolling and Document Translation, Scaling, and Rotation on a Touch-Screen Display," filed Jun. 28, 2007; 60/945,858, "List Scrolling and Document Translation on a Touch-Screen Display," filed Jun. 22, 2007; 60/879,469, "Portable Multifunction Device," filed Jan. 8, 2007; 60/883,801, "List Scrolling and Document Translation on a Touch-Screen Display," filed Jan. 7, 2007; and 60/879,253, "Portable Multifunction Device," filed Jan. 7, 2007. All of these applications are incorporated by reference herein in their entirety.

This application is related to the following applications: (1) U.S. patent application Ser. No. 10/188,182, "Touch Pad For Handheld Device," filed on Jul. 1, 2002; (2) U.S. patent application Ser. No. 10/722,948, "Touch Pad For Handheld Device," filed on Nov. 25, 2003; (3) U.S. patent application Ser. No. 10/643,256, "Movable Touch Pad With Added Functionality," filed on Aug. 18, 2003; (4) U.S. patent application Ser. No. 10/654,108, "Ambidextrous Mouse," filed on Sep. 2, 2003; (5) U.S. patent application Ser. No. 10/840,862, "Multipoint Touchscreen," filed on May 6, 2004; (6) U.S. patent application Ser. No. 10/903,964, "Gestures For Touch Sensitive Input Devices," filed on Jul. 30, 2004; (7) U.S. patent application Ser. No. 11/038,590, "Mode-Based Graphical User Interfaces For Touch Sensitive Input Devices" filed on Jan. 18, 2005; (8) U.S. patent application Ser. No. 11/057,050, "Display Actuator," filed on Feb. 11, 2005; (9) U.S. Provisional Patent Application No. 60/658,777, "Multi-Functional Hand-Held Device," filed Mar. 4, 2005; (10) U.S. patent application Ser. No. 11/367,749, "Multi-Functional Hand-Held Device," filed Mar. 3, 2006; and (11) U.S. Provisional Patent Application No. 60/824,769, "Portable Multifunction Device," filed Sep. 6, 2006. All of these applications are incorporated by reference herein in their entirety.

TECHNICAL FIELD

The disclosed embodiments relate generally to devices with touch-screen displays, and more particularly to scrolling lists and to translating, rotating, and scaling electronic documents on devices with touch-screen displays.

BACKGROUND

As portable electronic devices become more compact, and the number of functions performed by a given device increases, it has become a significant challenge to design a user interface that allows users to easily interact with a multifunction device. This challenge is particularly significant for handheld portable devices, which have much smaller screens than desktop or laptop computers. This situation is unfortunate because the user interface is the gateway through which users receive not only content but also responses to user actions or behaviors, including user attempts to access a device's features, tools, and functions. Some portable communication devices (e.g., mobile telephones, sometimes called mobile phones, cell phones, cellular telephones, and the like) have resorted to adding more pushbuttons, increasing the density of push buttons, overloading the functions of pushbuttons, or using complex menu systems to allow a user

to access, store and manipulate data. These conventional user interfaces often result in complicated key sequences and menu hierarchies that must be memorized by the user.

Many conventional user interfaces, such as those that include physical pushbuttons, are also inflexible. This is unfortunate because it may prevent a user interface from being configured and/or adapted by either an application running on the portable device or by users. When coupled with the time consuming requirement to memorize multiple key sequences and menu hierarchies, and the difficulty in activating a desired pushbutton, such inflexibility is frustrating to most users.

As a result of the small size of display screens on portable electronic devices and the potentially large size of electronic files, frequently only a portion of a list or of an electronic document of interest to a user can be displayed on the screen at a given time. Users thus will frequently need to scroll displayed lists or to translate displayed electronic documents. Users also will need to rotate and to scale (i.e., magnify or de-magnify) displayed electronic documents. However, the limitations of conventional user interfaces can cause these actions to be awkward to perform.

Furthermore, scrolling displayed lists and translating electronic documents can be awkward on both portable and non-portable electronic devices with touch-screen displays. A user may become frustrated if the scrolling or translation does not reflect the user's intent. Similarly, a user may become frustrated if rotation and scaling of electronic documents does not reflect the user's intent.

Accordingly, there is a need for devices with touch-screen displays with more transparent and intuitive user interfaces for scrolling lists of items and for translating, rotating, and scaling electronic documents that are easy to use, configure, and/or adapt.

SUMMARY

The above deficiencies and other problems associated with user interfaces for portable devices and devices with touch-sensitive displays are reduced or eliminated by the disclosed device. In some embodiments, the device has a touch-sensitive display (also known as a "touch screen") with a graphical user interface (GUI), one or more processors, memory and one or more modules, programs or sets of instructions stored in the memory for performing multiple functions. In some embodiments, the user interacts with the GUI primarily through finger contacts and gestures on the touch-sensitive display. In some embodiments, the functions may include telephoning, video conferencing, e-mailing, instant messaging, blogging, digital photographing, digital videoing, web browsing, digital music playing, and/or digital video playing. Instructions for performing these functions may be included in a computer program product configured for execution by one or more processors.

In accordance with some embodiments, a computer-implemented method for use in conjunction with a device with a touch screen display is disclosed. In the method, a movement of an object on or near the touch screen display is detected. In response to detecting the movement, an electronic document displayed on the touch screen display is translated in a first direction. If an edge of the electronic document is reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display, an area beyond the edge of the document is displayed. After the object is no longer detected on or near the touch

screen display, the document is translated in a second direction until the area beyond the edge of the document is no longer displayed.

In accordance with some embodiments, a graphical user interface on a device with a touch screen display is disclosed, comprising a portion of an electronic document displayed on the touch screen display and an area beyond an edge of the document. In the graphical user interface, in response to detecting a movement of an object on or near the touch screen display, the electronic document is translated in a first direction. If the edge of the electronic document is reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display, the area beyond the edge of the document is displayed. After the object is no longer detected on or near the touch screen display, the document is translated in a second direction until the area beyond the edge of the document is no longer displayed.

In accordance with some embodiments, a device is disclosed, comprising a touch screen display, one or more processors, memory, and one or more programs. The one or more programs are stored in the memory and configured to be executed by the one or more processors. The one or more programs include instructions for detecting a movement of an object on or near the touch screen display and instructions for translating an electronic document displayed on the touch screen display in a first direction, in response to detecting the movement. The one or more programs also include instructions for displaying an area beyond an edge of the electronic document if the edge of the electronic document is reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display. The one or more programs further include instructions for translating the document in a second direction until the area beyond the edge of the document is no longer displayed, after the object is no longer detected on or near the touch screen display.

In accordance with some embodiments, a computer-program product is disclosed, comprising a computer readable storage medium and a computer program mechanism embedded therein. The computer program mechanism comprises instructions, which when executed by a device with a touch screen display, cause the device to detect a movement of an object on or near the touch screen display and to translate an electronic document displayed on the touch screen display in a first direction, in response to detecting the movement. The instructions also cause the device to display an area beyond an edge of the electronic document if the edge of the electronic document is reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display. The instructions further cause the device to translate the document in a second direction until the area beyond the edge of the document is no longer displayed, after the object is no longer detected on or near the touch screen display.

In accordance with some embodiments, a device with a touch screen display is disclosed. The device comprises means for detecting a movement of an object on or near the touch screen display and means for translating an electronic document displayed on the touch screen display in a first direction, in response to detecting the movement. The device also comprises means for displaying an area beyond an edge of the electronic document if the edge of the electronic document is reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display. The device further comprises means for translating the document in a second direction until the area beyond the edge of the document is no longer displayed, after the object is no longer detected on or near the touch screen display.

In accordance with some embodiments, a computer-implemented method for use in conjunction with a device with a touch screen display is disclosed. In the method, a movement of an object on or near the touch screen display is detected. In response to detecting the movement, a list of items displayed on the touch screen display is scrolled in a first direction. If a terminus of the list is reached while scrolling the list in the first direction while the object is still detected on or near the touch screen display, an area beyond the terminus of the list is displayed. After the object is no longer detected on or near the touch screen display, the list is scrolled in a second direction opposite the first direction until the area beyond the terminus of the list is no longer displayed.

In accordance with some embodiments, a graphical user interface on a device with a touch screen display is disclosed, comprising a portion of a list of items displayed on the touch screen display and an area beyond a terminus of the list. In response to detecting a movement of an object on or near the touch screen display, the list is scrolled in a first direction. If the terminus of the list is reached while scrolling the list in the first direction while the object is still detected on or near the touch screen display, the area beyond the terminus of the list is displayed. After the object is no longer detected on or near the touch screen display, the list is scrolled in a second direction opposite the first direction until the area beyond the terminus of the list is no longer displayed.

In accordance with some embodiments, a device is disclosed, comprising a touch screen display, one or more processors, memory, and one or more programs. The one or more programs are stored in the memory and configured to be executed by the one or more processors. The one or more programs include instructions for detecting a movement of an object on or near the touch screen display and instructions for scrolling a list of items displayed on the touch screen display in a first direction in response to detecting the movement. The one or more programs also include instructions for displaying an area beyond a terminus of the list if the terminus of the list is reached while scrolling the list in the first direction while the object is still detected on or near the touch screen display. The one or more programs further include instructions for scrolling the list in a second direction opposite the first direction until the area beyond the terminus of the list is no longer displayed, after the object is no longer detected on or near the touch screen display.

In accordance with some embodiments, a computer-program product is disclosed, comprising a computer readable storage medium and a computer program mechanism embedded therein. The computer program mechanism comprises instructions, which when executed by a device with a touch screen display, cause the device to detect a movement of an object on or near the touch screen display and to scroll a list of items displayed on the touch screen display in a first direction in response to detecting the movement. The instructions also cause the device to display an area beyond a terminus of the list if the terminus of the list is reached while scrolling the list in the first direction while the object is still detected on or near the touch screen display. The instructions further cause the device to scroll the list in a second direction opposite the first direction until the area beyond the terminus of the list is no longer displayed, after the object is no longer detected on or near the touch screen display.

In accordance with some embodiments, a device with a touch screen display is disclosed. The device comprises means for detecting a movement of an object on or near the

touch screen display and means for scrolling a list of items displayed on the touch screen display in a first direction in response to detecting the movement. The device also comprises means for displaying an area beyond a terminus of the list if the terminus of the list is reached while scrolling the list in the first direction while the object is still detected on or near the touch screen display. The device further comprises means for scrolling the list in a second direction opposite the first direction until the area beyond the terminus of the list is no longer displayed, after the object is no longer detected on or near the touch screen display.

In accordance with some embodiments, a computer-implemented method for use at a device with a touch screen display includes detecting a multifinger twisting gesture on or near the touch screen display. The multifinger twisting gesture has a corresponding degree of rotation. If the corresponding degree of rotation exceeds a predefined degree of rotation, a 90° screen rotation command is executed. If the corresponding degree of rotation is less than the predefined degree of rotation, a screen rotation command with an acute angle of rotation is executed and, upon ceasing to detect the multifinger twisting gesture, a screen rotation command with an angle of rotation opposite to the acute angle is executed.

In accordance with some embodiments, a device includes a touch screen display, one or more processors, memory, and one or more programs. The one or more programs are stored in the memory and configured to be executed by the one or more processors. The one or more programs include: instructions for detecting a multifinger twisting gesture on or near the touch screen display, wherein the multifinger twisting gesture has a corresponding degree of rotation; instructions for executing a 90° screen rotation command, if the corresponding degree of rotation exceeds a predefined degree of rotation; and instructions for executing a screen rotation command with an acute angle of rotation and for executing, upon ceasing to detect the multifinger twisting gesture, a screen rotation command with an angle of rotation opposite to the acute angle, if the corresponding degree of rotation is less than the predefined degree of rotation.

In accordance with some embodiments, a computer-program product includes a computer readable storage medium and a computer program mechanism embedded therein. The computer program mechanism includes instructions, which when executed by a device with a touch screen display, cause the device to: detect a multifinger twisting gesture on or near the touch screen display, wherein the multifinger twisting gesture has a corresponding degree of rotation; execute a 90° screen rotation command, if the corresponding degree of rotation exceeds a predefined degree of rotation; and execute a screen rotation command with an acute angle of rotation and, upon ceasing to detect the multifinger twisting gesture, execute a screen rotation command with an angle of rotation opposite to the acute angle, if the corresponding degree of rotation is less than the predefined degree of rotation.

In accordance with some embodiments, a device with a touch screen display includes: means for detecting a multifinger twisting gesture on or near the touch screen display, wherein the multifinger twisting gesture has a corresponding degree of rotation; means for executing a 90° screen rotation command, if the corresponding degree of rotation exceeds a predefined degree of rotation; and means for executing a screen rotation command with an acute angle of rotation and, upon ceasing to detect the multifinger twisting gesture, for executing a screen rotation command with an angle of rotation opposite to the acute angle, if the corresponding degree of rotation is less than the predefined degree of rotation.

In accordance with some embodiments, a computer-implemented method of displaying an electronic document having a document length and a document width, for use at a device with a touch screen display, includes displaying the electronic document at a first magnification and detecting a gesture on or near the touch screen display corresponding to a command to zoom out by a user-specified amount. In response to detecting the gesture, the electronic document is displayed at a magnification less than the first magnification. If the document length or document width is entirely displayed while the gesture is still detected on or near the touch screen display, the electronic document is displayed at a magnification wherein areas beyond opposite edges of the electronic document are displayed, and upon detecting termination of the gesture, the electronic document is displayed at a magnification wherein the areas beyond opposite edges of the electronic document are no longer displayed.

In accordance with some embodiments, a graphical user interface on a device with a touch screen display includes an electronic document having a document length and a document width, to be displayed on the touch screen display at multiple magnifications including a first magnification, and areas beyond opposite edges of the electronic document. In response to detecting a gesture on or near the touch screen display corresponding to a command to zoom out by a user-specified amount, wherein the gesture is detected while displaying the electronic document at the first magnification, the electronic document is displayed at a magnification less than the first magnification. If the document length or document width is entirely displayed while the gesture is still detected on or near the touch screen display, the electronic document is displayed at a magnification wherein the areas beyond opposite edges of the electronic document are displayed, and upon detecting termination of the gesture, the electronic document is displayed at a magnification wherein the areas beyond opposite edges of the electronic document are no longer displayed.

In accordance with some embodiments, a device includes a touch screen display, one or more processors, memory, and one or more programs. The one or more programs are stored in the memory and configured to be executed by the one or more processors. The one or more programs include: instructions for displaying an electronic document at a first magnification; instructions for detecting a gesture on or near the touch screen display corresponding to a command to zoom out by a user-specified amount; instructions for displaying the electronic document at a magnification less than the first magnification, in response to detecting the gesture; instructions for displaying the electronic document at a magnification wherein areas beyond opposite edges of the electronic document are displayed, if a document length or a document width is entirely displayed while the gesture is still detected on or near the touch screen display; and instructions for displaying the electronic document at a magnification wherein the areas beyond opposite edges of the electronic document are no longer displayed, upon detecting termination of the gesture.

In accordance with some embodiments, a computer-program product includes a computer readable storage medium and a computer program mechanism embedded therein. The computer program mechanism includes instructions, which when executed by a device with a touch screen display, cause the device to: display an electronic document at a first magnification; detect a gesture on or near the touch screen display corresponding to a command to zoom out by a user-specified amount; display the electronic document at a magnification less than the first magnification, in response to detecting the

gesture; display the electronic document at a magnification wherein areas beyond opposite edges of the electronic document are displayed, if a document length or a document width is entirely displayed while the gesture is still detected on or near the touch screen display; and display the electronic document at a magnification wherein the areas beyond opposite edges of the electronic document are no longer displayed, upon detecting termination of the gesture.

In accordance with some embodiments, a device with a touch screen display includes: means for displaying an electronic document at a first magnification; means for detecting a gesture on or near the touch screen display corresponding to a command to zoom out by a user-specified amount; means for displaying the electronic document at a magnification less than the first magnification, in response to detecting the gesture; means for displaying the electronic document at a magnification wherein areas beyond opposite edges of the electronic document are displayed, if a document length or a document width is entirely displayed while the gesture is still detected on or near the touch screen display; and means for displaying the electronic document at a magnification wherein the areas beyond opposite edges of the electronic document are no longer displayed, upon detecting termination of the gesture.

In accordance with some embodiments, a computer-implemented method of displaying an electronic document, for use at a device with a touch screen display, includes displaying at least a first portion of the electronic document at a first magnification and detecting a gesture on or near the touch screen display corresponding to a command to zoom in by a user-specified amount. In response to detecting the gesture, decreasing portions of the electronic document are displayed at increasing magnifications. Upon detecting termination of the gesture, if the magnification exceeds a predefined magnification, a respective portion of the electronic document is displayed at the predefined magnification.

In accordance with some embodiments, a graphical user interface on a device with a touch screen display includes decreasing portions of an electronic document, to be displayed on the touch screen display at increasing magnifications. The decreasing portions of the electronic document include a first portion. In response to detecting a gesture on or near the touch screen display corresponding to a command to zoom in by a user-specified amount, wherein the gesture is detected while displaying at least the first portion of an electronic document at a first magnification, the decreasing portions of the electronic document are displayed at the increasing magnifications. Upon detecting termination of the gesture, if the magnification exceeds a predefined magnification, a respective portion of the electronic document is displayed at the predefined magnification.

In accordance with some embodiments, a device includes a touch screen display, one or more processors, memory, and one or more programs. The one or more programs are stored in the memory and configured to be executed by the one or more processors. The one or more programs include: instructions for displaying at least a first portion of an electronic document at a first magnification; instructions for detecting a gesture on or near the touch screen display corresponding to a command to zoom in by a user-specified amount; instructions for displaying decreasing portions of the electronic document at increasing magnifications, in response to detecting the gesture; and instructions for displaying a respective portion of the electronic document at a predefined magnification if, upon detecting termination of the gesture, the magnification exceeds the predefined magnification.

In accordance with some embodiments, a computer-program product includes a computer readable storage medium and a computer program mechanism embedded therein. The computer program mechanism includes instructions, which when executed by a device with a touch screen display, cause the device to: display at least a first portion of an electronic document at a first magnification; detect a gesture on or near the touch screen display corresponding to a command to zoom in by a user-specified amount; display decreasing portions of the electronic document at increasing magnifications, in response to detecting the gesture; and display a respective portion of the electronic document at a predefined magnification if, upon detecting termination of the gesture, the magnification exceeds a predefined magnification.

In accordance with some embodiments, a device with a touch screen display includes means for displaying at least a first portion of an electronic document at a first magnification; means for detecting a gesture on or near the touch screen display corresponding to a command to zoom in by a user-specified amount; means for displaying decreasing portions of the electronic document at increasing magnifications, in response to detecting the gesture; and means for displaying a respective portion of the electronic document at a predefined magnification if, upon detecting termination of the gesture, the magnification exceeds the predefined magnification.

The disclosed embodiments provide for easy and intuitive scrolling of lists and translating of electronic documents on a device with a touch screen display, and for easy and intuitive rotation and scaling of electronic documents on a device with a touch screen display.

BRIEF DESCRIPTION OF THE DRAWINGS

For a better understanding of the aforementioned embodiments of the invention as well as additional embodiments thereof, reference should be made to the Description of Embodiments below, in conjunction with the following drawings in which like reference numerals refer to corresponding parts throughout the figures.

FIG. **1** is a block diagram illustrating a portable multifunction device with a touch-sensitive display in accordance with some embodiments.

FIG. **2** illustrates a portable multifunction device having a touch screen in accordance with some embodiments.

FIG. **3** illustrates an exemplary user interface for unlocking a portable electronic device in accordance with some embodiments.

FIG. **4** illustrates an exemplary user interface for a menu of applications on a portable multifunction device in accordance with some embodiments.

FIG. **5** is a flow diagram illustrating a method of scrolling through a list in accordance with some embodiments.

FIGS. **6**A-**6**D illustrate an exemplary user interface for managing an inbox in accordance with some embodiments.

FIG. **7** is a flow diagram illustrating a method of translating an electronic document in accordance with some embodiments.

FIGS. **8**A-**8**D illustrate an exemplary user interface for a browser in accordance with some embodiments.

FIG. **9** is a flow diagram illustrating a process of displaying an electronic document at multiple magnifications in accordance with some embodiments.

FIGS. **10**A-**10**C illustrate the display of an electronic document at multiple magnifications in accordance with some embodiments.

FIG. **11** is a flow diagram illustrating a process of displaying an electronic document at multiple magnifications in accordance with some embodiments.

FIGS. **12**A-**12**C illustrate the display of an electronic document at multiple magnifications in accordance with some embodiments.

FIGS. **13**A-**13**C illustrate the display of an electronic document at multiple magnifications in accordance with some embodiments.

FIG. **14** is a flow diagram illustrating a process of executing a screen rotation command in accordance with some embodiments.

FIGS. **15**A-**15**E illustrate rotating the display of an electronic document or other digital object in accordance with some embodiments.

FIGS. **16**A-**16**F illustrate an exemplary screen rotation gesture in accordance with some embodiments.

FIG. **17** is a block diagram illustrating a device with a touch-screen display in accordance with some embodiments.

DESCRIPTION OF EMBODIMENTS

Reference will now be made in detail to embodiments, examples of which are illustrated in the accompanying drawings. In the following detailed description, numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be apparent to one of ordinary skill in the art that the present invention may be practiced without these specific details. In other instances, well-known methods, procedures, components, circuits, and networks have not been described in detail so as not to unnecessarily obscure aspects of the embodiments.

Embodiments of a portable multifunction device, user interfaces for such devices, and associated processes for using such devices are described. In some embodiments, the device is a portable communications device such as a mobile telephone that also contains other functions, such as PDA and/or music player functions.

The user interface may include a physical click wheel in addition to a touch screen or a virtual click wheel displayed on the touch screen. A click wheel is a user-interface device that may provide navigation commands based on an angular displacement of the wheel or a point of contact with the wheel by a user of the device. A click wheel may also be used to provide a user command corresponding to selection of one or more items, for example, when the user of the device presses down on at least a portion of the wheel or the center of the wheel. Alternatively, breaking contact with a click wheel image on a touch screen surface may indicate a user command corresponding to selection. For simplicity, in the discussion that follows, a portable multifunction device that includes a touch screen is used as an exemplary embodiment. It should be understood, however, that some of the user interfaces and associated processes may be applied to other devices, such as personal computers and laptop computers, that may include one or more other physical user-interface devices, such as a physical click wheel, a physical keyboard, a mouse and/or a joystick.

The device supports a variety of applications, such as a telephone application, a video conferencing application, an e-mail application, an instant messaging application, a blogging application, a digital camera application, a digital video camera application, a web browsing application, a digital music player application, and/or a digital video player application.

The various applications that may be executed on the device may use at least one common physical user-interface device, such as the touch screen. One or more functions of the touch screen as well as corresponding information displayed on the device may be adjusted and/or varied from one application to the next and/or within a respective application. In this way, a common physical architecture (such as the touch screen) of the device may support the variety of applications with user interfaces that are intuitive and transparent.

The user interfaces may include one or more soft keyboard embodiments. The soft keyboard embodiments may include standard (QWERTY) and/or non-standard configurations of symbols on the displayed icons of the keyboard, such as those described in U.S. patent applications Ser. No. 11/459,606, "Keyboards For Portable Electronic Devices," filed Jul. 24, 2006, and Ser. No. 11/459,615, "Touch Screen Keyboards For Portable Electronic Devices," filed Jul. 24, 2006, the contents of which are hereby incorporated by reference herein in their entirety. The keyboard embodiments may include a reduced number of icons (or soft keys) relative to the number of keys in existing physical keyboards, such as that for a typewriter. This may make it easier for users to select one or more icons in the keyboard, and thus, one or more corresponding symbols. The keyboard embodiments may be adaptive. For example, displayed icons may be modified in accordance with user actions, such as selecting one or more icons and/or one or more corresponding symbols. One or more applications on the portable device may utilize common and/or different keyboard embodiments. Thus, the keyboard embodiment used may be tailored to at least some of the applications. In some embodiments, one or more keyboard embodiments may be tailored to a respective user. For example, based on a word usage history (lexicography, slang, individual usage) of the respective user. Some of the keyboard embodiments may be adjusted to reduce a probability of a user error when selecting one or more icons, and thus one or more symbols, when using the soft keyboard embodiments.

Attention is now directed towards embodiments of the device. FIG. **1** is a block diagram illustrating a portable multifunction device **100** with a touch-sensitive display **112** in accordance with some embodiments. The touch-sensitive display **112** is sometimes called a "touch screen" for convenience. The device **100** may include a memory **102** (which may include one or more computer readable storage mediums), a memory controller **122**, one or more processing units (CPU's) **120**, a peripherals interface **118**, RF circuitry **108**, audio circuitry **110**, a speaker **111**, a microphone **113**, an input/output (I/O) subsystem **106**, other input or control devices **116**, and an external port **124**. The device **100** may include one or more optical sensors **164**. These components may communicate over one or more communication buses or signal lines **103**.

It should be appreciated that the device **100** is only one example of a portable multifunction device **100**, and that the device **100** may have more or fewer components than shown, may combine two or more components, or a may have a different configuration or arrangement of the components. The various components shown in FIG. **1** may be implemented in hardware, software or a combination of both hardware and software, including one or more signal processing and/or application specific integrated circuits.

Memory **102** may include high-speed random access memory and may also include non-volatile memory, such as one or more magnetic disk storage devices, flash memory devices, or other non-volatile solid-state memory devices. Access to memory **102** by other components of the device **100**, such as the CPU **120** and the peripherals interface **118**, may be controlled by the memory controller **122**.

The peripherals interface **118** couples the input and output peripherals of the device to the CPU **120** and memory **102**. The one or more processors **120** run or execute various software programs and/or sets of instructions stored in memory **102** to perform various functions for the device **100** and to process data.

In some embodiments, the peripherals interface **118**, the CPU **120**, and the memory controller **122** may be implemented on a single chip, such as a chip **104**. In some other embodiments, they may be implemented on separate chips.

The RF (radio frequency) circuitry **108** receives and sends RF signals, also called electromagnetic signals. The RF circuitry **108** converts electrical signals to/from electromagnetic signals and communicates with communications networks and other communications devices via the electromagnetic signals. The RF circuitry **108** may include well-known circuitry for performing these functions, including but not limited to an antenna system, an RF transceiver, one or more amplifiers, a tuner, one or more oscillators, a digital signal processor, a CODEC chipset, a subscriber identity module (SIM) card, memory, and so forth. The RF circuitry **108** may communicate with networks, such as the Internet, also referred to as the World Wide Web (WWW), an intranet and/or a wireless network, such as a cellular telephone network, a wireless local area network (LAN) and/or a metropolitan area network (MAN), and other devices by wireless communication. The wireless communication may use any of a plurality of communications standards, protocols and technologies, including but not limited to Global System for Mobile Communications (GSM), Enhanced Data GSM Environment (EDGE), wideband code division multiple access (W-CDMA), code division multiple access (CDMA), time division multiple access (TDMA), Bluetooth, Wireless Fidelity (Wi-Fi) (e.g., IEEE 802.11a, IEEE 802.11b, IEEE 802.11g and/or IEEE 802.11n), voice over Internet Protocol (VoIP), Wi-MAX, a protocol for email, instant messaging, and/or Short Message Service (SMS)), or any other suitable communication protocol, including communication protocols not yet developed as of the filing date of this document.

The audio circuitry **110**, the speaker **111**, and the microphone **113** provide an audio interface between a user and the device **100**. The audio circuitry **110** receives audio data from the peripherals interface **118**, converts the audio data to an electrical signal, and transmits the electrical signal to the speaker **111**. The speaker **111** converts the electrical signal to human-audible sound waves. The audio circuitry **110** also receives electrical signals converted by the microphone **113** from sound waves. The audio circuitry **110** converts the electrical signal to audio data and transmits the audio data to the peripherals interface **118** for processing. Audio data may be retrieved from and/or transmitted to memory **102** and/or the RF circuitry **108** by the peripherals interface **118**. In some embodiments, the audio circuitry **110** also includes a headset jack (not shown). The headset jack provides an interface between the audio circuitry **110** and removable audio input/output peripherals, such as output-only headphones or a headset with both output (e.g., a headphone for one or both ears) and input (e.g., a microphone).

The I/O subsystem **106** couples input/output peripherals on the device **100**, such as the display system **112** and other input/control devices **116**, to the peripherals interface **118**. The I/O subsystem **106** may include a display controller **156** and one or more input controllers **160** for other input or control devices. The one or more input controllers **160** receive/send electrical signals from/to other input or control devices **116**. The other input/control devices **116** may include physical buttons (e.g., push buttons, rocker buttons, etc.), dials, slider switches, joysticks, click wheels, and so forth. In some alternate embodiments, input controller(s) **160** may be coupled to any (or none) of the following: a keyboard, infrared port, USB port, and a pointer device such as a mouse. The one or more buttons (e.g., **208**, FIG. **2**) may include an up/down button for volume control of the speaker **111** and/or the microphone **113**. The one or more buttons may include a push button (e.g., **206**, FIG. **2**). A quick press of the push button may disengage a lock of the touch screen **112** or begin a process that uses gestures on the touch screen to unlock the device, as described in U.S. patent application Ser. No. 11/322,549, "Unlocking a Device by Performing Gestures on an Unlock Image," filed Dec. 23, 2005, which is hereby incorporated by reference herein in its entirety. A longer press of the push button (e.g., **206**) may turn power to the device **100** on or off. The user may be able to customize a functionality of one or more of the buttons. The touch screen **112** is used to implement virtual or soft buttons and one or more soft keyboards.

The touch-sensitive display system **112** provides an input interface and an output interface between the device and a user. The display controller **156** receives and/or sends electrical signals from/to the display system **112**. The display system **112** displays visual output to the user. The visual output may include graphics, text, icons, video, and any combination thereof (collectively termed "graphics"). In some embodiments, some or all of the visual output may correspond to user-interface objects, further details of which are described below.

A touch screen in display system **112** is a touch-sensitive surface that accepts input from the user based on haptic and/or tactile contact. The display system **112** and the display controller **156** (along with any associated modules and/or sets of instructions in memory **102**) detect contact (and any movement or breaking of the contact) on the display system **112** and converts the detected contact into interaction with user-interface objects (e.g., one or more soft keys, icons, web pages or images) that are displayed on the touch screen. In an exemplary embodiment, a point of contact between a touch screen in the display system **112** and the user corresponds to a finger of the user.

The touch screen in the display system **112** may use LCD (liquid crystal display) technology, or LPD (light emitting polymer display) technology, although other display technologies may be used in other embodiments. The touch screen in the display system **112** and the display controller **156** may detect contact and any movement or breaking thereof using any of a plurality of touch sensing technologies now known or later developed, including but not limited to capacitive, resistive, infrared, and surface acoustic wave technologies, as well as other proximity sensor arrays or other elements for determining one or more points of contact with a touch screen in the display system **112**. A touch-sensitive display in some embodiments of the display system **112** may be analogous to the multi-touch sensitive tablets described in the following U.S. Pat. Nos. 6,323,846 (Westerman et al.), 6,570,557 (Westerman et al.), and/or 6,677,932 (Westerman), and/or U.S. Patent Publication 2002/0015024A1, each of which is hereby incorporated by reference herein in their entirety. However, a touch screen in the display system **112** displays visual output from the portable device **100**, whereas touch sensitive tablets do not provide visual output. The touch screen in the display system **112** may have a resolution in excess of 100 dpi. In an exemplary embodiment, the touch screen in the display system has a resolution of approximately 168 dpi. The user may make contact with the touch screen in the display system **112** using any suitable object or append-

age, such as a stylus, a finger, and so forth. In some embodiments, the user interface is designed to work primarily with finger-based contacts and gestures, which are much less precise than stylus-based input due to the larger area of contact of a finger on the touch screen. In some embodiments, the device translates the rough finger-based input into a precise pointer/cursor position or command for performing the actions desired by the user.

A touch-sensitive display in some embodiments of the display system **112** may be as described in the following applications: (1) U.S. patent application Ser. No. 11/381,313, "Multipoint Touch Surface Controller," filed on May 2, 2006; (2) U.S. patent application Ser. No. 10/840,862, "Multipoint Touchscreen," filed on May 6, 2004; (3) U.S. patent application Ser. No. 10/903,964, "Gestures For Touch Sensitive Input Devices," filed on Jul. 30, 2004; (4) U.S. patent application Ser. No. 11/048,264, "Gestures For Touch Sensitive Input Devices," filed on Jan. 31, 2005; (5) U.S. patent application Ser. No. 11/038,590, "Mode-Based Graphical User Interfaces For Touch Sensitive Input Devices," filed on Jan. 18, 2005; (6) U.S. patent application Ser. No. 11/228,758, "Virtual Input Device Placement On A Touch Screen User Interface," filed on Sep. 16, 2005; (7) U.S. patent application Ser. No. 11/228,700, "Operation Of A Computer With A Touch Screen Interface," filed on Sep. 16, 2005; (8) U.S. patent application Ser. No. 11/228,737, "Activating Virtual Keys Of A Touch-Screen Virtual Keyboard," filed on Sep. 16, 2005; and (9) U.S. patent application Ser. No. 11/367,749, "Multi-Functional Hand-Held Device," filed on Mar. 3, 2006. All of these applications are incorporated by reference herein in their entirety.

In some embodiments, in addition to the touch screen, the device **100** may include a touchpad (not shown) for activating or deactivating particular functions. In some embodiments, the touchpad is a touch-sensitive area of the device that, unlike the touch screen, does not display visual output. The touchpad may be a touch-sensitive surface that is separate from the touch screen in the display system **112** or an extension of the touch-sensitive surface formed by the touch screen.

In some embodiments, the device **100** may include a physical or virtual click wheel as an input control device **116**. A user may navigate among and interact with one or more graphical objects (henceforth referred to as icons) displayed in the display system **112** by rotating the click wheel or by moving a point of contact with the click wheel (e.g., where the amount of movement of the point of contact is measured by its angular displacement with respect to a center point of the click wheel). The click wheel may also be used to select one or more of the displayed icons. For example, the user may press down on at least a portion of the click wheel or an associated button. User commands and navigation commands provided by the user via the click wheel may be processed by an input controller **160** as well as one or more of the modules and/or sets of instructions in memory **102**. For a virtual click wheel, the click wheel and click wheel controller may be part of the display system **112** and the display controller **156**, respectively. For a virtual click wheel, the click wheel may be either an opaque or semitransparent object that appears and disappears on the touch screen display in response to user interaction with the device. In some embodiments, a virtual click wheel is displayed on the touch screen of a portable multifunction device and operated by user contact with the touch screen.

The device **100** also includes a power system **162** for powering the various components. The power system **162** may include a power management system, one or more power sources (e.g., battery, alternating current (AC)), a recharging system, a power failure detection circuit, a power converter or inverter, a power status indicator (e.g., a light-emitting diode (LED)) and any other components associated with the generation, management and distribution of power in portable devices.

The device **100** may also include one or more optical sensors **164**. FIG. **1** shows an optical sensor coupled to an optical sensor controller **158** in I/O subsystem **106**. The optical sensor **164** may include charge-coupled device (CCD) or complementary metal-oxide semiconductor (CMOS) phototransistors. The optical sensor **164** receives light from the environment, projected through one or more lens, and converts the light to data representing an image. In conjunction with an imaging module **143**, the optical sensor **164** may capture still images or video. In some embodiments, an optical sensor is located on the back of the device **100**, opposite the touch screen display **112** on the front of the device, so that the touch screen display may be used as a viewfinder for either still and/or video image acquisition. In some embodiments, an optical sensor is located on the front of the device so that the user's image may be obtained for videoconferencing while the user views the other video conference participants on the touch screen display. In some embodiments, the position of the optical sensor **164** can be changed by the user (e.g., by rotating the lens and the sensor in the device housing) so that a single optical sensor **164** may be used along with the touch screen display for both video conferencing and still and/or video image acquisition.

The device **100** may also include one or more proximity sensors **166**. FIG. **1** shows a proximity sensor **166** coupled to the peripherals interface **118**. Alternately, the proximity sensor **166** may be coupled to an input controller **160** in the I/O subsystem **106**. The proximity sensor **166** may perform as described in U.S. patent application Ser. Nos. 11/241,839, "Proximity Detector In Handheld Device," filed Sep. 30, 2005, and 11/240,788, "Proximity Detector In Handheld Device," filed Sep. 30, 2005, which are hereby incorporated by reference herein in their entirety. In some embodiments, the proximity sensor turns off and disables the touch screen **112** when the multifunction device is placed near the user's ear (e.g., when the user is making a phone call). In some embodiments, the proximity sensor keeps the screen off when the device is in the user's pocket, purse, or other dark area to prevent unnecessary battery drainage when the device is a locked state.

In some embodiments, the software components stored in memory **102** may include an operating system **126**, a communication module (or set of instructions) **128**, a contact/motion module (or set of instructions) **130**, a graphics module (or set of instructions) **132**, a text input module (or set of instructions) **134**, a Global Positioning System (GPS) module (or set of instructions) **135**, and applications (or set of instructions) **136**.

The operating system **126** (e.g., Darwin, RTXC, LINUX, UNIX, OS X, WINDOWS, or an embedded operating system such as VxWorks) includes various software components and/or drivers for controlling and managing general system tasks (e.g., memory management, storage device control, power management, etc.) and facilitates communication between various hardware and software components.

The communication module **128** facilitates communication with other devices over one or more external ports **124** and also includes various software components for handling data received by the RF circuitry **108** and/or the external port **124**. The external port **124** (e.g., Universal Serial Bus (USB), FIREWIRE, etc.) is adapted for coupling directly to other

devices or indirectly over a network (e.g., the Internet, wireless LAN, etc.). In some embodiments, the external port is a multi-pin (e.g., 30-pin) connector that is the same as, or similar to and/or compatible with the 30-pin connector used on iPod (trademark of Apple Computer, Inc.) devices.

The contact/motion module **130** may detect contact with the touch screen in the display system **112** (in conjunction with the display controller **156**) and other touch sensitive devices (e.g., a touchpad or physical click wheel). The contact/motion module **130** includes various software components for performing various operations related to detection of contact, such as determining if contact has occurred, determining if there is movement of the contact and tracking the movement across the touch screen in the display system **112**, and determining if the contact has been broken (i.e., if the contact has ceased). Determining movement of the point of contact may include determining speed (magnitude), velocity (magnitude and direction), and/or an acceleration (a change in magnitude and/or direction) of the point of contact. These operations may be applied to single contacts (e.g., one finger contacts) or to multiple simultaneous contacts (e.g., "multi-touch"/multiple finger contacts). In some embodiments, the contact/motion module **130** and the display controller **156** also detects contact on a touchpad. In some embodiments, the contact/motion module **130** detects movement of one or more objects on or near the touch screen and/or the touchpad. In some embodiments, the contact/motion module **130** and the controller **160** detects contact on a click wheel **116**.

The graphics module **132** includes various known software components for rendering and displaying graphics on the display system **112**, including components for changing the intensity of graphics that are displayed. As used herein, the term "graphics" includes any object that can be displayed to a user, including without limitation text, web pages, icons (such as user-interface objects including soft keys), digital images, videos, animations and the like.

The text input module **134**, which may be a component of graphics module **132**, provides soft keyboards for entering text in various applications (e.g., contacts **137**, e-mail **140**, IM **141**, blogging **142**, browser **147**, and any other application that needs text input).

The GPS module **135** determines the location of the device and provides this information for use in various applications (e.g., to telephone **138** for use in location-based dialing, to camera **143** and/or blogger **142** as picture/video metadata, and to applications that provide location-based services such as weather widgets, local yellow page widgets, and map/navigation widgets).

The applications **136** may include the following modules (or sets of instructions), or a subset or superset thereof:

- a contacts module **137** (sometimes called an address book or contact list);
- a telephone module **138**;
- a video conferencing module **139**;
- an e-mail client module **140**;
- an instant messaging (IM) module **141**;
- a blogging module **142**;
- a camera module **143** for still and/or video images;
- an image management module **144**;
- a video player module **145**;
- a music player module **146**;
- a browser module **147**;
- a calendar module **148**;
- widget modules **149**, which may include weather widget **149-1**, stocks widget **149-2**, calculator widget **149-3**, alarm clock widget **149-4**, dictionary widget **149-5**, and other widgets obtained by the user, as well as user-created widgets **149-6**;
- widget creator module **150** for making user-created widgets **149-6**; and/or search module **151**.

Examples of other applications **136** that may be stored in memory **102** include memo pad and other word processing applications, JAVA-enabled applications, encryption, digital rights management, voice recognition, and voice replication.

In conjunction with display system **112**, display controller **156**, contact module **130**, graphics module **132**, and text input module **134**, the contacts module **137** may be used to manage an address book or contact list, including: adding name(s) to the address book; deleting name(s) from the address book; associating telephone number(s), e-mail address(es), physical address(es) or other information with a name; associating an image with a name; categorizing and sorting names; providing telephone numbers or e-mail addresses to initiate and/or facilitate communications by telephone **138**, video conference **139**, e-mail **140**, or IM **141**; and so forth.

In conjunction with RF circuitry **108**, audio circuitry **110**, speaker **111**, microphone **113**, display system **112**, display controller **156**, contact module **130**, graphics module **132**, and text input module **134**, the telephone module **138** may be used to enter a sequence of characters corresponding to a telephone number, access one or more telephone numbers in the address book **137**, modify a telephone number that has been entered, dial a respective telephone number, conduct a conversation and disconnect or hang up when the conversation is completed. As noted above, the wireless communication may use any of a plurality of communications standards, protocols and technologies.

In conjunction with RF circuitry **108**, audio circuitry **110**, speaker **111**, microphone **113**, display system **112**, display controller **156**, optical sensor **164**, optical sensor controller **158**, contact module **130**, graphics module **132**, text input module **134**, contact list **137**, and telephone module **138**, the videoconferencing module **139** may be used to initiate, conduct, and terminate a video conference between a user and one or more other participants.

In conjunction with RF circuitry **108**, display system **112**, display controller **156**, contact module **130**, graphics module **132**, and text input module **134**, the e-mail client module **140** may be used to create, send, receive, and manage e-mail. In conjunction with image management module **144**, the e-mail module **140** makes it very easy to create and send e-mails with still or video images taken with camera module **143**.

In conjunction with RF circuitry **108**, display system **112**, display controller **156**, contact module **130**, graphics module **132**, and text input module **134**, the instant messaging module **141** may be used to enter a sequence of characters corresponding to an instant message, to modify previously entered characters, to transmit a respective instant message (for example, using a Short Message Service (SMS) or Multimedia Message Service (MMS) protocol), to receive instant messages and to view received instant messages. In some embodiments, transmitted and/or received instant messages may include graphics, photos, audio files, video files and/or other attachments as are supported in a MMS and/or an Enhanced Messaging Service (EMS).

In conjunction with RF circuitry **108**, display system **112**, display controller **156**, contact module **130**, graphics module **132**, text input module **134**, image management module **144**, and browsing module **147**, the blogging module **142** may be used to send text, still images, video, and/or other graphics to a blog (e.g., the user's blog).

In conjunction with display system **112**, display controller **156**, optical sensor(s) **164**, optical sensor controller **158**, contact module **130**, graphics module **132**, and image management module **144**, the camera module **143** may be used to capture still images or video (including a video stream) and store them into memory **102**, modify characteristics of a still image or video, or delete a still image or video from memory **102**.

In conjunction with display system **112**, display controller **156**, contact module **130**, graphics module **132**, text input module **134**, and camera module **143**, the image management module **144** may be used to arrange, modify or otherwise manipulate, label, delete, present (e.g., in a digital slide show or album), and store still and/or video images.

In conjunction with display system **112**, display controller **156**, contact module **130**, graphics module **132**, audio circuitry **110**, and speaker **111**, the video player module **145** may be used to display, present or otherwise play back videos (e.g., on the touch screen or on an external, connected display via external port **124**).

In conjunction with display system **112**, display system controller **156**, contact module **130**, graphics module **132**, audio circuitry **110**, speaker **111**, RF circuitry **108**, and browser module **147**, the music player module **146** allows the user to download and play back recorded music and other sound files stored in one or more file formats, such as MP3 or AAC files. In some embodiments, the device **100** may include the functionality of an MP3 player, such as an iPod (trademark of Apple Computer, Inc.).

In conjunction with RF circuitry **108**, display system **112**, display system controller **156**, contact module **130**, graphics module **132**, and text input module **134**, the browser module **147** may be used to browse the Internet, including searching, linking to, receiving, and displaying web pages or portions thereof, as well as attachments and other files linked to web pages.

In conjunction with RF circuitry **108**, display system **112**, display system controller **156**, contact module **130**, graphics module **132**, text input module **134**, e-mail module **140**, and browser module **147**, the calendar module **148** may be used to create, display, modify, and store calendars and data associated with calendars (e.g., calendar entries, to do lists, etc.).

In conjunction with RF circuitry **108**, display system **112**, display system controller **156**, contact module **130**, graphics module **132**, text input module **134**, and browser module **147**, the widget modules **149** are mini-applications that may be downloaded and used by a user (e.g., weather widget **149-1**, stocks widget **149-2**, calculator widget **149-3**, alarm clock widget **149-4**, and dictionary widget **149-5**) or created by the user (e.g., user-created widget **149-6**). In some embodiments, a widget includes an HTML (Hypertext Markup Language) file, a CSS (Cascading Style Sheets) file, and a JavaScript file. In some embodiments, a widget includes an XML (Extensible Markup Language) file and a JavaScript file (e.g., Yahoo! Widgets).

In conjunction with RF circuitry **108**, display system **112**, display system controller **156**, contact module **130**, graphics module **132**, text input module **134**, and browser module **147**, the widget creator module **150** may be used by a user to create widgets (e.g., turning a user-specified portion of a web page into a widget).

In conjunction with display system **112**, display system controller **156**, contact module **130**, graphics module **132**, and text input module **134**, the search module **151** may be used to search for text, music, sound, image, video, and/or other files in memory **102** that match one or more search criteria (e.g., one or more user-specified search terms).

Each of the above identified modules and applications correspond to a set of instructions for performing one or more functions described above. These modules (i.e., sets of instructions) need not be implemented as separate software programs, procedures or modules, and thus various subsets of these modules may be combined or otherwise re-arranged in various embodiments. In some embodiments, memory **102** may store a subset of the modules and data structures identified above. Furthermore, memory **102** may store additional modules and data structures not described above.

In some embodiments, the device **100** is a device where operation of a predefined set of functions on the device is performed exclusively through a touch screen in the display system **112** and/or a touchpad. By using a touch screen and/or a touchpad as the primary input/control device for operation of the device **100**, the number of physical input/control devices (such as push buttons, dials, and the like) on the device **100** may be reduced.

The predefined set of functions that may be performed exclusively through a touch screen and/or a touchpad includes navigation between user interfaces. In some embodiments, the touchpad, when touched by the user, navigates the device **100** to a main, home, or root menu from any user interface that may be displayed on the device **100**. In such embodiments, the touchpad may be referred to as a "menu button." In some other embodiments, the menu button may be a physical push button or other physical input/control device instead of a touchpad.

FIG. **2** illustrates a portable multifunction device **100** having a touch screen **112** in accordance with some embodiments. The touch screen may display one or more graphics. In this embodiment, as well as others described below, a user may select one or more of the graphics by making contact or touching the graphics, for example, with one or more fingers **202** (not drawn to scale in the figure). In some embodiments, selection of one or more graphics occurs when the user breaks contact with the one or more graphics. In some embodiments, the contact may include a gesture, such as one or more taps, one or more swipes (from left to right, right to left, upward and/or downward) and/or a rolling of a finger (from right to left, left to right, upward and/or downward) that has made contact with the device **100**. In some embodiments, inadvertent contact with a graphic may not select the graphic. For example, a swipe gesture that sweeps over an application icon may not select the corresponding application when the gesture corresponding to selection is a tap.

The device **100** may also include one or more physical buttons, such as "home" or menu button **204**. As described previously, the menu button **204** may be used to navigate to any application **136** in a set of applications that may be executed on the device **100**. Alternatively, in some embodiments, the menu button is implemented as a soft key in a GUI in touch screen **112**.

In one embodiment, the device **100** includes a touch screen **112**, a menu button **204**, a push button **206** for powering the device on/off and locking the device, and volume adjustment button(s) **208**. The push button **206** may be used to turn the power on/off on the device by depressing the button and holding the button in the depressed state for a predefined time interval; to lock the device by depressing the button and releasing the button before the predefined time interval has elapsed; and/or to unlock the device or initiate an unlock process. In an alternative embodiment, the device **100** also may accept verbal input for activation or deactivation of some functions through the microphone **113**.

Attention is now directed towards embodiments of user interfaces ("UI") and associated processes that may be imple-

mented on a portable multifunction device **100** and/or on a device **1700** with a touch-screen display (FIG. **17**).

FIG. **3** illustrates an exemplary user interface for unlocking a portable electronic device in accordance with some embodiments. In some embodiments, user interface **300** includes the following elements, or a subset or superset thereof:

- Unlock image **302** that is moved with a finger gesture to unlock the device;
- Arrow **304** that provides a visual cue to the unlock gesture;
- Channel **306** that provides additional cues to the unlock gesture;
- Time **308**;
- Day **310**;
- Date **312**; and
- Wallpaper image **314**.

In some embodiments, the device detects contact with the touch-sensitive display (e.g., a user's finger making contact on or near the unlock image **302**) while the device is in a user-interface lock state. The device moves the unlock image **302** in accordance with the contact. The device transitions to a user-interface unlock state if the detected contact corresponds to a predefined gesture, such as moving the unlock image across channel **306**. Conversely, the device maintains the user-interface lock state if the detected contact does not correspond to the predefined gesture. As noted above, processes that use gestures on the touch screen to unlock the device are described in U.S. patent application Ser. No. 11/322,549, "Unlocking a Device by Performing Gestures on an Unlock Image," filed Dec. 23, 2005, which is hereby incorporated by reference herein in its entirety.

FIG. **4** illustrates an exemplary user interface for a menu of applications on a portable multifunction device in accordance with some embodiments. In some embodiments, user interface **400** includes the following elements, or a subset or superset thereof:

- Signal strength indicator **402** for wireless communication;
- Time **404**;
- Battery status indicator **406**;
- Tray **408** with icons for frequently used applications, such as one or more of the following:
  - Phone **138**;
  - E-mail client **140**, which may include an indicator **410** of the number of unread e-mails;
  - Browser **147**; and
  - Music player **146**; and
- Icons for other applications, such as one or more of the following:
  - IM **141**;
  - Image management **144**;
  - Camera **143**;
  - Video player **145**;
  - Weather **149-1**;
  - Stocks **149-2**;
  - Blog **142**;
  - Calendar **148**;
  - Calculator **149-3**;
  - Alarm clock **149-4**;
  - Dictionary **149-5**; and
  - User-created widget **149-6**.

In some embodiments, UI **400** displays all of the available applications **136** on one screen so that there is no need to scroll through a list of applications (e.g., via a scroll bar or via a swipe gesture). In some embodiments, as the number of applications increases, the icons corresponding to the applications may decrease in size so that all applications may be displayed on a single screen without scrolling. In some embodiments, having all applications on one screen and a menu button enables a user to access any desired application with at most two inputs, such as activating the menu button **204** and then activating the desired application (e.g., by a tap or other finger gesture on the icon corresponding to the application).

In some embodiments, UI **400** provides integrated access to both widget-based applications and non-widget-based applications. In some embodiments, all of the widgets, whether user-created or not, are displayed in UI **400**. In other embodiments, activating the icon for user-created widget **149-6** may lead to another UI (not shown) that contains the user-created widgets or icons corresponding to the user-created widgets.

In some embodiments, a user may rearrange the icons in UI **400**, e.g., using processes described in U.S. patent application Ser. No. 11/459,602, "Portable Electronic Device With Interface Reconfiguration Mode," filed Jul. 24, 2006, which is hereby incorporated by reference herein in its entirety. For example, a user may move application icons in and out of tray **408** using finger gestures.

In some embodiments, UI **400** includes a gauge (not shown) that displays an updated account usage metric for an account associated with usage of the device (e.g., a cellular phone account), as described in U.S. patent application Ser. No. 11/322,552, "Account Information Display For Portable Communication Device," filed Dec. 23, 2005, which is hereby incorporated by reference herein in its entirety.

As discussed above, UI **400** may display all of the available applications **136** on one screen so that there is no need to scroll through a list of applications. However, in some embodiments a touch-sensitive display may include a GUI with one or more windows that display only a portion of a list of items (e.g., information items) or of an electronic document. In response to detecting a movement of an object on or near the touch-sensitive display, the list may be scrolled or the electronic document may be translated. Detecting the movement of the object may include determining speed (magnitude), velocity (magnitude and direction), and/or an acceleration (including magnitude and/or direction) of the object. Scrolling through the list or translating the document may be accelerated in response to an accelerated movement of the object. In some embodiments, the scrolling and acceleration of the scrolling, or translation and acceleration of the translation, may be in accordance with a simulation of a physical device having friction, i.e., damped motion. For example, the scrolling or translation may correspond to a simulation of a force law or equation of motion having a mass or inertial term, as well as a dissipative term. In some embodiments, the simulation may correspond to a cylinder rotating about its axis.

In some embodiments, accelerated movement of the detected object may include an accelerated movement of a point of contact followed by a breaking of the point of contact. For example, the user may make contact with the touch-sensitive display, swipe or sweep one or more of his or her fingers along the display (i.e., move and/or accelerate the point of contact), and optionally, break the point of contact with the display, i.e., move the one or more fingers away from the display. The swipe or sweep may be along a predefined axis of the touch-sensitive display or may be within a predetermined angle of a predefined direction on the touch-sensitive display. In other embodiments, the accelerated movement of the point of contact may include a first user gesture oriented along a predefined axis of the touch-sensitive display or oriented within a predetermined angle of a predefined direction on the touch-sensitive display.

Scrolling through the list of items or translating the electronic document may be further accelerated in response to detection of a second movement of an object on or near the touch-sensitive display, such as a second sweeping motion of the point of contact along the predefined axis or within the predetermined angle of a predefined direction on the touch-sensitive display and/or a second user gesture oriented along the predefined axis or within the predetermined angle of a predefined direction on the touch-sensitive display. For example, the user may swipe one or more of his or her fingers along the touch-sensitive display two or more times.

The scrolling through the list of items or the translation of the electronic document may be stopped in accordance with the user breaking the point of contact and then establishing a substantially stationary point of contact with the touch-sensitive display for at least a pre-determined period of time. For example, after swiping one or more of his or her fingers along the touch-sensitive display and breaking the point of contact, the user may touch the display and hold the one or more fingers that are touching the display stationary (or approximately stationary) for one or more seconds, or fractions of a second.

The direction of scrolling or translation may be reversed in response to intersecting a virtual boundary corresponding to a terminus of the list or an edge of the electronic document. The scrolling reversal or translation reversal may correspond to a damped motion. For example, during scrolling, a displayed portion of the list of items may appear to bounce off of a boundary of the window in the touch-sensitive display when a beginning or an end of the list of items is reached. Similarly, during translation, a displayed portion of the electronic document may appear to bounce off of a boundary of the window in the touch-sensitive display when an edge of the document is reached. The apparent bounce may correspond to a simulation of a viscous or elastic ball having momentum in a first direction striking an immovable and/or inelastic object, such as a wall. The subsequent motion of the document (the motion of which corresponds to the ball in the aforementioned analogy) may be damped, for example, by including a friction or dissipative term in the simulation. A parameter corresponding to the friction term in the simulation may be adjustable, allowing the document to reach equilibrium in contact with the virtual boundary, or displaced from the virtual boundary.

In some embodiments movement of the point of contact by the user over an index on the touch-sensitive display may be determined. In some embodiments, the index may be displayed in a first region or a first window of the touch-sensitive display while the list of items or information items during the scrolling may be displayed in a second region or a second window of the touch-sensitive display. The displayed index may have a sequence of index items. In an exemplary embodiment, the sequence of index items may include letters in the alphabet, i.e., the index may include an alphabetical index. The list of information items may include an alphabetically ordered list of information items. The alphabetically ordered list of information items may include contact information, for example, in a user's contact list or address book.

In response to movement of the user's point of contact over a displayed index, the list of information items on the touch-sensitive display may be scrolled. The list of information items may include a sequence of information item subsets corresponding to the sequence of index items. The subsets may include one or more categories. For example, a respective category may include contact information for one or more individuals whose first and/or last names begin with one or more respective letters, such as the letter 's'. In an exemplary embodiment, there is a subset corresponding to each letter in the alphabet that has one or more entries. In some embodiments, the scrolling may be in accordance with a simulation of an equation of motion having friction.

The scrolling may include scrolling through a respective information item subset if the point of contact moves over a corresponding respective index item in the index items. The scrolling may have an associated scroll speed based on a speed of movement of the point of contact over the respective index item and the number of items in the information item subset corresponding to the respective index item. For example, the scroll speed may be faster for subsets that have more entries than subsets with fewer entries. The scrolling may include scrolling through all items in a plurality of the information item subsets in response to the point of contact moving over the corresponding index items in the displayed index.

If it is determined that the point of contact with the index corresponds to a respective index item in the index, the list of information items may be scrolled to a corresponding subset of the list of information items. For example, if the user selects an index item, such as the letter 'R', in the set of index symbols, the list of items may be smoothly scrolled to the corresponding subset for the letter 'R' in the list of items. Alternatively, the displayed list of information items jump directly from a current scroll position to a scroll position in which information items corresponding to the index item 'R' are displayed.

In the present document, the term "if" may be construed to mean "when," or "upon," or "in response to determining," or "in response to detecting," depending on the context Similarly, the phrase "if it is determined" or "if [a stated condition or event] is detected" may be construed to mean "upon determining" or "in response to determining" or "upon detecting" the stated condition or event, or "in response to detecting" the stated condition or event, depending on the context.

If the point of contact with the touch-sensitive display corresponds to a user selection of a respective information item in the list of information items, information corresponding to the respective information item may be displayed on the touch-sensitive display. For example, if the user selects a respective name, the corresponding contact information may be displayed.

While scrolling through respective information subsets, an index symbol may displayed in conjunction with each respective information item subset. In some embodiments, respective index symbols may be displayed adjacent to corresponding subsets (such as displayed text) of the list of information items. In some embodiments, a respective index symbol may be displayed at an upper edge of a window containing the displayed text of the respective information item subset.

The index symbol corresponding to a respective information subset may be translucently displayed over the respective information item subset. The translucently displayed index symbol may have a different font color than that used to display text in the information item subset, and/or it may be displayed using a larger font than the font used to display text in the information item subset.

If the list of information items contains no items for a particular index symbol, i.e., no entries for a particular subset, a first index symbol preceding a particular index symbol and a second index symbol following the index symbol may be displayed in conjunction with scrolling through the list of information items from the information subset corresponding to the first index symbol to the information subset corresponding to the second index symbol. The particular index symbol may not be displayed in conjunction with the displayed text of the list of information items during the scroll

through. For example, display of a respective index symbol may be skipped when the list of information items contains no items for the particular index symbol.

In some embodiments, the list scrolling described here operates without displaying a scroll bar. Similarly, in some embodiments, the translation of electronic documents described here operates without displaying scroll bars. The user's sweeping motion on the touch-sensitive display operation may be performed directly on top of the displayed list or displayed electronic document, and may include a sweeping or gliding motion, near or in contact with the display's surface, along a path anywhere within a display window in which the list or electronic document is displayed. While a scroll bar could potentially be displayed in conjunction with the displayed list, the scrolling or translation described here can be independent of any such scroll bar. In some embodiments, if a scroll bar is used, then an upward movement of a point of contact on the scroll bar may cause earlier entries in the list to be displayed, whereas a downward movement of the point of contact on the scroll bar may cause later entries in the list to be displayed.

In some embodiments, scrolling or translation may be in accordance with a speed of movement of a detected object, such as a speed of movement of a point of contact. The speed may be a time average of values determined during several time intervals. In an exemplary embodiment, the speed, velocity and/or acceleration may be determined over five time intervals, where a respective time interval corresponds to an inverse of a frame rate, such as 0.0167 s, of a display. In some embodiments, the speed, velocity and/or acceleration may be determined even when a variable frame rate is used, such as when one or more frames are skipped or not displayed. In these embodiments, the speed, velocity, and/or acceleration may be determined two or more times for the respective time interval and/or may be projected based on values determined in a preceding and/or a subsequent time interval.

In some embodiments, the scrolling or translation after a user optionally breaks the contact may be in accordance with the change in the acceleration and the speed or the velocity in one or more time intervals prior to the breaking of the contact. For example, the velocity v_f of scrolling or translation one or more time intervals after breaking contact may be determined using

$$v_f = v_o + \alpha\Delta t,$$

where v_o is a current value of the velocity when the contact is broken, a is a current value of the acceleration when the contact is broken and Δt is an elapsed time, such as one time interval. The velocities and/or acceleration in such a calculation may be projected along an axis or direction of the scrolling or translation. In some embodiments, in subsequent time intervals following the determination of the velocity based on the acceleration and/or the velocity in one or more time intervals prior to the breaking of the contact, the velocity of the scrolling or translation may be tapered. For example, in each successive time interval the velocity may be decreased by 5%. When the velocity crosses a lower threshold, it may be set to zero.

FIG. **5** is a flow diagram illustrating a method **500** of scrolling through a list in accordance with some embodiments. The method **500** provides a simple visual indicator to a user that a terminus of a list has been reached.

Movement of an object is detected on or near a touch screen display of a device (**502**). In some embodiments, the object is a finger. In some embodiments, the device is a portable multifunction device.

In response to detecting the movement, a list of items displayed on the touch screen display is scrolled in a first direction (**504**). In some embodiments, the list is a list of email messages, as illustrated in FIGS. **6**A-**6**D. In some embodiments, the list of items is a list of instant message conversations, a list of favorite phone numbers, a list of contact information (sometimes called a contact list or address book list), a list of labels, a list of email folders, a list of email addresses, a list of physical addresses, a list of ringtones, a list of album names, or a list of bookmarks. In some embodiments, the first direction is a vertical direction; in some other embodiments, the first direction is a horizontal direction. In some embodiments, scrolling the list in the first direction prior to reaching a terminus of the list has an associated scrolling speed corresponding to a speed of movement of the object (**506**). In some embodiments, the list is scrolled in accordance with a simulation of an equation of motion having friction (**508**).

If a terminus of the list is reached (e.g., upon reaching the terminus of the list) while scrolling the list in the first direction while the object is still detected on or near the touch screen display, an area beyond the terminus of the list is displayed (**510**-Yes, **514**). In some embodiments, the list has a first item and a last item and the terminus is either the first item or the last item. For example, in FIG. **6**B the email **3534** from Aaron Jones is the first item and thus the terminus of the corresponding list of emails. In some embodiments, the area beyond the terminus of the list is white (**516**). In some embodiments, the list of items has a background and the area beyond the terminus of the list is visually indistinct from the background (**518**). For example, in FIG. **6**C both the area **3536** and the background of the listed emails are white.

After the object is no longer detected on or near the touch screen display, the list of items is scrolled in a second direction opposite the first direction until the area beyond the terminus of the list is no longer displayed (**520**). In some embodiments, the list is scrolled in the second direction using a damped motion (**522**). In some embodiments, the change from scrolling the list in the first direction to scrolling the list in the second direction until the area beyond the terminus of the list is no longer displayed makes the terminus of the list appear to be elastically attached to an edge of the touch screen display or to an edge displayed on the touch screen display (**524**).

In some embodiments, scrolling in the first direction prior to reaching the terminus of the list has a first associated scrolling distance that corresponds to a distance of movement of the object prior to reaching the terminus of the list. For example, a scrolling distance prior to reaching the terminus of the list shown in FIGS. **6**A-**6**D may correspond to a distance traversed on the touch screen display by the swipe gesture **3514** before the terminus is reached. Displaying an area beyond the terminus of the list includes scrolling the list in the first direction for a second associated scrolling distance that is less than a distance of movement of the object after the terminus is reached. For example, in FIG. **6**C, after the terminus is reached the list is scrolled for a distance **3538**, which may be less than a distance traversed on the touch screen display by the swipe gesture **3514** after the terminus is reached.

In some embodiments, scrolling in the first direction prior to reaching a terminus of the list has a first associated scrolling speed that corresponds to a speed of movement of the object. For example, a scrolling speed prior to reaching the terminus of the list shown in FIGS. **6**A-**6**D may correspond to a speed on the touch screen display of the swipe gesture **3514** before the terminus is reached. Displaying an area beyond the

terminus of the list includes scrolling the list in the first direction at a second associated scrolling speed. The second associated scrolling speed is slower than the first associated scrolling speed. For example, in FIG. **6**C, displaying the area **3536** beyond the terminus of the list may include scrolling the list at a speed that is slower than the scrolling speed before the terminus is reached. In some embodiments, the second associated speed is a fraction (e.g., one-half or one-third) of the first associated speed. In some embodiments, the second associated speed is the square root of the first associated speed.

If a terminus of the list is not reached while scrolling the list in the first direction while the object is still detected on or near the touch screen display, the process **500** is complete (**510**-No, **512**). The process **500** may be re-initiated upon subsequent detection of another movement of an object on or near the touch screen display (**502**).

FIGS. **6**A-**6**D illustrate the scrolling of a list of items to a terminus of the list, at which point an area beyond the terminus is displayed and the list is then scrolled in an opposite direction until the area beyond the terminus is no longer displayed, in accordance with some embodiments. While FIGS. **6**A-**6**D illustrate this scrolling in the context of a portable multifunction device **100**, this scrolling is not limited to portable multifunction devices. In the example of FIGS. **6**A-**6**D, the listed items are email messages; FIGS. **6**A-**6**D illustrate an exemplary user interface **3500**A for managing an inbox in accordance with some embodiments. An analogous user interface may be used to display and manage other mailboxes (e.g., drafts, sent, trash, personal, etc.). In addition, other types of lists are possible, including but not limited to lists of instant message conversations, favorite phone numbers, contact information, labels, email folders, email addresses, physical addresses, ringtones, album names or bookmarks.

In some embodiments, user interface **3500**A include the following elements, or a subset or superset thereof:

- **402**, **404**, and **406**, as described above;
- a create email icon **3310** that when activated (e.g., by a finger tap on the icon) initiates display of a UI to create a new email message;
- mailboxes icon **3502** that when activated (e.g., by a finger tap on the icon) initiates the display of a UI listing email mailboxes (i.e., folders);
- unread messages icon **3504** that displays the number of unread messages in the inbox;
- names **3506** of the senders of the email messages;
- subject lines **3508** for the email messages;
- dates **3510** of the email messages;
- unread message icons **3512** that indicate messages that have not been opened;
- preview pane separator **3518** that separates the list of messages from a preview of a selected message in the list;
- settings icon **3520** that when activated (e.g., by a finger tap on the icon) initiates the display of a UI to modify settings;
- move message icon **3522** that when activated (e.g., by a finger tap on the icon) initiates the display of a UI to move messages;
- Delete symbol icon **3524** that when activated (e.g., by a finger tap on the icon) initiates display of a UI to confirm that the user wants to delete the selected email;
- Reply/Forward icon **3526** that when activated (e.g., by a finger tap on the icon) initiates display of a UI to select how to reply or forward the selected email;

If the list of emails fills more than the allotted screen area, the user may scroll through the emails using vertically upward and/or vertically downward swipe gestures on the touch screen. In the example of FIG. **6**A, a portion of a list of emails is displayed in the screen area, including a top displayed email **3530** from Bruce Walker and a bottom displayed email **3532** from Kim Brook. A user performs a vertically downward swipe gesture **3514** to scroll toward the top of the list. The vertically downward gesture **3514**, which may be a finger gesture, corresponds to the movement of an object on or near the touch screen that is detected in operation **502** of process **500** (FIG. **5**). The vertically downward gesture **3514** need not be exactly vertical; a substantially vertical gesture is sufficient. In some embodiments, a gesture within a predetermined angle of being perfectly vertical results in vertical scrolling. In one embodiment, a gesture within 27 degrees of being perfectly vertical results in vertical scrolling.

As a result of detecting the vertically downward gesture **3514**, in FIG. **6**B the displayed emails have shifted down, such that the previous bottom displayed email **3532** from Kim Brook is no longer displayed, the previous top displayed email **3530** from Bruce Walker is now second from the top, and the email **3534** from Aaron Jones, which was not displayed in FIG. **6**A, is now displayed at the top of the list. This shifting of emails is an example of the scrolling described in operation **504** of process **500** (FIG. **5**).

In this example, the email **3534** from Aaron Jones is the first email in the list and thus is the terminus of the list. Upon reaching this email **3534**, in response to continued detection of the vertically downward gesture **3514**, an area **3536** (FIG. **6**C) above the first email **3534** (i.e., beyond the terminus of the list) is displayed, as described in operation **514** of process **500** (FIG. **5**). In some embodiments, the area displayed beyond the terminus of the list is visually indistinct from the background of the list, as described in operation **518** of process **500** (FIG. **5**). In FIG. **6**C, both the area **3536** and the background of the emails (e.g., emails **3534** and **3530**) are white and thus are visually indistinct.

Once vertically downward gesture **3514** is complete, such that a corresponding object is no longer detected on or near the touch screen display, the list is scrolled in an opposite direction until the area **3536** is no longer displayed. FIG. **6**D illustrates the result of this scrolling in the opposite direction, which corresponds to operation **520** of process **500** (FIG. **5**): the email **3534** from Aaron Jones is now displayed at the top of the screen area allotted to the list and the area **3536** is not displayed.

In the example of FIGS. **6**A-**6**D, a vertically downward gesture resulted in display of an area beyond the first item in the list. Similarly, a vertically upward gesture may result in display of an area beyond the last item of the list, if the vertically upward gesture continues once the list has been scrolled to the last item. The last item may be considered a terminus of the list, similar to the first item. As discussed above, the gesture need not be exactly vertical to result in vertical scrolling; a gesture within a predefined range of angles from perfectly vertical is sufficient.

In some embodiments, instead of scrolling a list of items in one dimension, a user may desire to translate an electronic document in two dimensions. If the electronic document fills more than the screen area allotted to display the document, the screen will only display a portion of the document. The user may translate the electronic document to view portions of the document that are not initially displayed.

FIG. **7** is a flow diagram illustrating a method **700** of translating an electronic document in accordance with some embodiments. The method **700** provides a simple visual indicator to a user that one or more edges of an electronic document are being displayed.

Movement of an object is detected on or near a touch screen display of a device (**702**). In some embodiments, the object is a finger. In some embodiments, the device is a portable multifunction device.

In response to detecting the movement, an electronic document displayed on the touch screen display is translated in a first direction (**704**). In some embodiments, the electronic document is a web page, as illustrated in FIGS. **8**A-**8**D. In some embodiments, the electronic document is a digital image. In some embodiments, the electronic document is a word processing, spreadsheet, email, or presentation document. In some embodiments, the first direction is a vertical direction, a horizontal direction, or a diagonal direction. In some embodiments, the first direction corresponds to the direction of movement of the object detected on or near the display but is not necessarily identical to the direction of movement of the object.

In some embodiments, translating the electronic document in the first direction prior to reaching an edge of the electronic document has an associated speed of translation corresponding to a speed of movement of the object (**706**). In some embodiments, the electronic document is translated in accordance with a simulation of an equation of motion having friction (**708**).

If an edge of the electronic document is reached (e.g., upon reaching the edge of the document) while translating the electronic document in the first direction while the object is still detected on or near the touch screen display, an area beyond the edge of the electronic document is displayed (**710**-Yes, **714**). In some embodiments, the area beyond the edge of the electronic document is black, gray, a solid color, or white (**716**). In some embodiments, the area beyond the edge of the electronic document is visually distinct from the document (**718**). For example, the area **3930** beyond the edge of the web page **3912** in FIG. **8**C is black, in contrast to the white background of the web page **3912**. In some other embodiments, a wallpaper image such as a picture or pattern may be displayed in the area beyond the edge of the electronic document.

After the object is no longer detected on or near the touch screen display, the electronic document is translated in a second direction until the area beyond the edge of the electronic document is no longer displayed (**720**). For example, in FIG. **8**D the web page **3912** has been translated such that the area **3930** beyond its edge is no longer displayed. In some embodiments, the second direction is opposite the first direction. In some embodiments, the electronic document is translated in the second direction using a damped motion (**722**). In some embodiments, the change from translating the electronic document in the first direction to translating the electronic document in the second direction until the area beyond the edge of the electronic document is no longer displayed makes the edge of the electronic document appear to be elastically attached to an edge of the touch screen display or to an edge displayed on the touch screen display (**724**).

In some embodiments, translating in the first direction prior to reaching an edge of the electronic document has a first associated translating distance that corresponds to a distance of movement of the object prior to reaching the edge of the electronic document. For example, a distance of translation of the web page **3912** shown in FIGS. **8**A-**8**D prior to reaching the edge of the document may correspond to a distance traversed on the touch screen display by the swipe gesture **3925** before the edge is reached. In some embodiments, displaying an area beyond the edge of the electronic document includes translating the electronic document in the first direction for a second associated translating distance, wherein the second associated translating distance is less than a distance of movement of the object after reaching the edge of the electronic document. For example, in FIG. **8**C, after the edge is reached the web page **3912** is translated by a distance indicated by opposing arrows 3928-1 and 3928-2, which may be less than a distance traversed on the touch screen display by the swipe gesture **3925** after the terminus is reached.

In some embodiments, translating in the first direction prior to reaching an edge of the electronic document has a first associated translating speed that corresponds to a speed of movement of the object. For example, a speed of translation prior to reaching the edge of the web page **3912** shown in FIGS. **8**A-**8**D may correspond to a speed of movement of the swipe gesture **3925**. Displaying an area beyond the edge of the electronic document includes translating the electronic document in the first direction at a second associated translating speed. The second associated translating speed is slower than the first associated translating speed. For example, in FIG. **8**C, displaying the area **3930** beyond the edge of the web page **3912** may include translating the web page **3912** at a speed that is slower than the speed of translation before the edge is reached. In some embodiments, the second associated speed is a fraction (e.g., one-half or one-third) of the first associated speed. In some embodiments, the second associated speed is the square root of the first associated speed.

If an edge of the electronic document is not reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display, the process **700** is complete (**710**-No, **712**). The process **700** may be re-initiated upon subsequent detection of another movement of an object on or near the touch screen display (**702**).

FIGS. **8**A-**8**D illustrate the translation of an electronic document to an edge of the document, at which point an area beyond the edge is displayed and the document is then translated in a second direction until the area beyond the edge of the document is no longer displayed, in accordance with some embodiments. While FIGS. **8**A-**8**D illustrate this translation in the context of a portable multifunction device **100**, this translation is not limited to portable multifunction devices. In the example of FIGS. **8**A-**8**D, the document is a web page **3912**; FIGS. **8**A-**8**D illustrate an exemplary user interface for a browser in accordance with some embodiments. An analogous user interface may be used to display other types of electronic documents, such as word processing, spreadsheet, email, presentation documents, or digital images.

In some embodiments, user interface **3900**A of FIGS. **8**A-**8**D includes the following elements, or a subset or superset thereof:

- **402**, **404**, and **406**, as described above;
- Previous page icon **3902** that when activated (e.g., by a finger tap on the icon) initiates display of the previous web page;
- Web page name **3904**;
- Next page icon **3906** that when activated (e.g., by a finger tap on the icon) initiates display of the next web page;
- URL (Uniform Resource Locator) entry box **3908** for inputting URLs of web pages;
- Refresh icon **3910** that when activated (e.g., by a finger tap on the icon) initiates a refresh of the web page;
- Web page **3912** or other structured document, which is made of blocks **3914** of text content and other graphics (e.g., images);
- Settings icon **3916** that when activated (e.g., by a finger tap on the icon) initiates display of a settings menu for the browser;

Bookmarks icon **3918** that when activated (e.g., by a finger tap on the icon) initiates display of a bookmarks list or menu for the browser;

Add bookmark icon **3920** that when activated (e.g., by a finger tap on the icon) initiates display of a UI for adding bookmarks; and

New window icon **3922** that when activated (e.g., by a finger tap on the icon) initiates display of a UI for adding new windows to the browser.

In some embodiments, the device analyzes the render tree of the web page **3912** to determine the blocks **3914** in the web page. In some embodiments, a block **3914** corresponds to a render node that is: replaced; a block; an inline block; or an inline table.

In FIG. **8**A, the web page fills more than the allotted screen area: only the left sides of block **7** (**3914-7**) and block **8** (**3914-8**) are displayed and only the top left corner of block **9** (**3914-9**) is displayed. To view the partially displayed blocks, a user may translate the displayed document by gesturing on the touch screen in accordance with some embodiments.

In some embodiments, in response to a substantially vertical upward (or downward) swipe gesture by the user, the web page (or, more generally, other electronic documents) may translate one-dimensionally upward (or downward) in the vertical direction. In some embodiments, a gesture is considered substantially vertical if it is within a predetermined angle of being perfectly vertical. For example, in response to an upward swipe gesture by the user that is within a predetermined angle (e.g., 27°) of being perfectly vertical, the web page may scroll one-dimensionally upward in the vertical direction.

Conversely, in some embodiments, in response to a gesture that is not within a predetermined angle (e.g., 27°) of being perfectly vertical, the web page may translate two-dimensionally (i.e., with simultaneous movement in both the vertical and horizontal directions). For example, in response to an upward swipe gesture by the user that is not within a predetermined angle (e.g., 27°) of being perfectly vertical, the web page may translate two-dimensionally along the direction of the swipe.

In the example of FIG. **8**A, an upward swipe gesture **3925** is not within a predetermined angle of being perfectly vertical. Therefore, as a result of detecting the upward swipe gesture **3925**, the web page is translated in two dimensions. In this example, the translation is approximately diagonal. FIG. **8**B illustrates the result of this translation: blocks **8** (**3914-8**) and **9** (**3914-9**) are now fully displayed; blocks **1** (**3914-1**) and **2** (**3914-2**) are now only partially displayed, and block **3** (**3914-3**) is no longer displayed at all. This translation is an example of the translation described in operation **704** of process **700** (FIG. **7**).

In FIG. **8**B, block **9** (**3914-9**) is in the lower right-hand corner of the web page **3912**; both the bottom and right edges of the web page have been reached while translating the web page. Upon reaching these edges of the document, in response to continued detection of the upward gesture **3925**, an area **3930** (FIG. **8**C) beyond the bottom and right edges of the web page is displayed. In some embodiments, the area displayed beyond the edge(s) of an electronic document is visually distinct from the document, as described in operation **718** of process **700** (FIG. **7**). In FIG. **8**C, the area **3930** is black and thus is visually distinct from the white background of the web page **3912**.

Once the upward gesture **3925** is complete, such that a corresponding object is no longer detected on or near the touch screen display, the web page **3912** is translated (e.g., in a direction opposite to the original direction of translation) until the area **3930** is no longer displayed. FIG. **8**D illustrates the result of this translation, which corresponds to operation **720** of process **700** (FIG. **7**): block **9** (**3914-9**) is now displayed in the lower right-hand corner of the portion of the screen allotted to display the web page **3912** and the area **3930** not displayed. In some embodiments, the direction of translation is not necessarily opposite to the original direction but may be in any direction such that, upon completion of the translation, the area beyond the edge(s) of the electronic document is no longer displayed.

FIG. **9** is a flow diagram illustrating a process **900** of displaying an electronic document having a document length and a document width, in accordance with some embodiments. The process **900** provides a simple visual indicator to a user that an electronic document is being displayed at a minimum magnification (e.g., the electronic document cannot be zoomed out and/or demagnified further).

The process **900** is performed at a device with a touch screen display. In some embodiments, the device is a portable multifunction device. In some embodiments, the electronic document is a web page (e.g., web page **3912**, FIGS. **10**A-**10**C). In some embodiments, the electronic document is a digital image. In some embodiments, the electronic document is a word processing, spreadsheet, email or presentation document.

The electronic document is displayed (**902**) at a first magnification on the touch screen display. A gesture is detected (**904**) on or near the touch screen display corresponding to a command to zoom out by a user-specified amount. In some embodiments, the gesture is a pinching gesture (e.g., gesture **3951**/**3953**, FIG. **10**A).

In response to detecting the gesture, the electronic document is displayed (**906**) at a magnification less than the first magnification. For example, the web page **3912** is shown at a lesser magnification in FIG. **10**B than in FIG. **10**A.

If the document length or document width is not entirely displayed (**908**-No) while the gesture is still detected on or near the touch screen display, the process **900** is complete (**910**).

If, however, the document length (e.g., **3957**, FIG. **10**B) or document width (e.g., **3959**, FIG. **10**B) is entirely displayed (**908**-Yes) while the gesture (e.g., **3951**/**3953**) is still detected on or near the touch screen display, the electronic document is displayed (**912**) at a magnification wherein areas beyond opposite edges of the electronic document (e.g., areas **3955**, FIG. **10**B) are displayed.

In some embodiments, the areas beyond opposite edges of the electronic document include an area beyond a top edge of the document and an area beyond a bottom edge of the document. In some embodiments, the areas beyond opposite edges of the electronic document include an area beyond a right edge of the document and an area beyond a left edge of the document. In some embodiments, the areas beyond opposite edges of the electronic document include an area beyond a top edge of the document, an area beyond a bottom edge of the document, an area beyond a right edge of the document, and an area beyond a left edge of the document (e.g., FIG. **10**B).

In some embodiments, the areas beyond opposite edges of the electronic document are black, gray, a solid color, or white. In some embodiments, the areas beyond opposite edges of the electronic document are visually distinct from the document. For example, the areas **3955** (FIG. **10**B) are black and thus are visually distinct from the web page **3912**.

Upon detecting termination of the gesture, the electronic document is displayed (**914**) at a magnification wherein the

areas beyond opposite edges of the electronic document are no longer displayed. For example, the areas **3955** are not displayed in FIG. **10**C.

FIGS. **10**A-**10**C illustrate the display of an electronic document at multiple magnifications in accordance with some embodiments. While FIGS. **10**A-**10**C illustrate displaying these multiple magnifications in the context of a portable multifunction device **100**, displaying these multiple magnifications is not limited to portable multifunction devices. In the example of FIGS. **10**A-**10**C, the document is a web page **3912**; FIGS. **10**A-**10**C (like FIGS. **8**A-**8**D) illustrate an exemplary user interface for a browser in accordance with some embodiments. An analogous user interface may be used to display other types of electronic documents, such as digital images or word processing, spreadsheet, email, or presentation documents.

In FIG. **10**A, the web page **3912** is displayed at a first magnification. The web page **3912** fills more than the allotted screen area: only the left sides of block **7** (**3914-7**) and block **8** (**3914-8**) are displayed and only the top left corner of block **9** (**3914-9**) is displayed.

In response to detecting a pinching gesture **3951/3953** (FIG. **10**A), the web-page is displayed at a magnification less than the first magnification, as shown in FIG. **10**B. If a document length **3957** or a document width **3959** is entirely displayed while the gesture **3951/3953** is still detected, areas **3955** beyond opposite edges of the web page **3912** are displayed. Upon detecting termination of the gesture **3951/3953**, the web page **3912** is displayed at a magnification wherein the areas **3955** are no longer displayed, as shown in FIG. **10**C.

FIG. **11** is a flow diagram illustrating a process **1100** of displaying an electronic document at multiple magnifications in accordance with some embodiments. The process **1100** provides a simple visual indicator to a user that an electronic document is being displayed at a maximum magnification (e.g., the electronic document cannot be zoomed in and/or magnified further).

The process **1100** is performed at a device with a touch screen display. In some embodiments, the device is a portable multifunction device. In some embodiments, the electronic document is a web page (e.g., web page **3912**, FIGS. **12**A-**12**C). In some embodiments, the electronic document is a digital image (e.g., digital image **1302**, FIGS. **13**A-**13**C). In some embodiments, the electronic document is a word processing, spreadsheet, email or presentation document.

At least a first portion of the electronic document is displayed (**1102**) at a first magnification. A gesture is detected (**1104**) on or near the touch screen display corresponding to a command to zoom in by a user-specified amount. In some embodiments, the gesture is a de-pinching gesture (e.g., **3931/3933**, FIGS. **12**A and **13**A).

In response to detecting the gesture, decreasing portions of the electronic document are displayed (**1106**) at increasing magnifications. For example, in FIG. **12**B a decreased portion of the web page **3912** is displayed at a higher magnification than the portion in FIG. **12**A, and in FIG. **13**B a decreased portion of the digital image **1302** is displayed at a higher magnification than the portion in FIG. **13**A.

If, upon detecting termination of the gesture, the magnification does not exceed a predefined magnification (**1108**-No), the process **1100** is complete (**1110**).

If, however, upon detecting termination of the gesture, the magnification exceeds a predefined magnification (**1108**-Yes), a respective portion of the electronic document is displayed (**1112**) at the predefined magnification. In the examples of FIGS. **12**B and **13**B, the magnification exceeds a predefined magnification. Upon detecting termination of the gesture **3931/3933**, a portion of the web page **3912** is displayed at the predefined magnification, as illustrated in FIG. **12**C, and a portion of the digital image **1302** is displayed at the predefined magnification, as illustrated in FIG. **13**C.

In some embodiments, immediately prior to detecting termination of the gesture, a last decreased portion of the electronic document is displayed at a first resolution. Upon detecting termination of the gesture, the respective portion of the electronic document is displayed at a second resolution that is greater than the first resolution.

FIGS. **12**A-**12**C illustrate the display of an electronic document at multiple magnifications in accordance with some embodiments. While FIGS. **12**A-**12**C illustrate displaying these multiple magnifications in the context of a portable multifunction device **100**, displaying these multiple magnifications is not limited to portable multifunction devices. In the example of FIGS. **12**A-**12**C, the document is a web page **3912**; FIGS. **12**A-**12**C (like FIGS. **8**A-**8**D) illustrate an exemplary user interface for a browser in accordance with some embodiments. An analogous user interface may be used to display other types of electronic documents, such as digital images or word processing, spreadsheet, email, or presentation documents.

In FIG. **12**A, a first portion of the web page **3912** is displayed at a first magnification. The web page **3912** fills more than the allotted screen area: only the left sides of block **7** (**3914-7**) and block **8** (**3914-8**) are displayed and only the top left corner of block **9** (**3914-9**) is displayed.

In response to detecting a de-pinching gesture **3931/3933** (FIG. **12**A), decreasing portions of the web-page **3912** are displayed at increasing magnifications compared to the magnification shown in FIG. **12**A. For example, the portion of the web page **3912** shown in FIG. **12**B is smaller than and has a higher magnification than the portion of the web page **3912** shown in FIG. **12**A.

In the example of FIG. **12**B, the magnification exceeds a predefined magnification. Upon detecting termination of the gesture **3931/3933**, a portion of the web page **3912** is displayed at the predefined magnification, as illustrated in FIG. **12**C.

FIGS. **13**A-**13**C illustrate the display of an electronic document at multiple magnifications in accordance with some embodiments. While FIGS. **13**A-**13**C illustrate displaying these multiple magnifications in the context of a portable multifunction device **100**, displaying these multiple magnifications is not limited to portable multifunction devices. In the example of FIGS. **13**A-**13**C, the document is a digital image **1302** that includes an image of a person **1304**.

In FIG. **13**A, a digital image **1302** is displayed at a first magnification. In response to detecting a de-pinching gesture **3931/3933**, decreasing portions of the digital image **1302** are displayed at increasing magnifications compared to the magnification shown in FIG. **13**A. For example, the portion of the digital image **1302** shown in FIG. **13**B is smaller than and has a higher magnification than the portion of the digital image **1302** shown in FIG. **13**A.

In the example of FIG. **13**B, the magnification exceeds a predefined magnification. Upon detecting termination of the gesture **3931/3933**, a portion of the digital image **1302** is displayed at the predefined magnification, as illustrated in FIG. **13**C.

FIG. **14** is a flow diagram illustrating a process **1400** of executing a screen rotation command in accordance with some embodiments. The process **1400** provides a simple visual indicator to a user that the user has not provided a sufficient gesture to initiate a 90° screen rotation command.

The process **1400** is performed at a device with a touch screen display. In some embodiments, the device is a portable multifunction device.

A multifinger twisting gesture (e.g., **1506**, FIG. **15**A, or **1508**, FIG. **15**C) is detected (**1402**) on or near the touch screen display. The multifinger twisting gesture has a corresponding degree of rotation. In some embodiments, the multifinger twisting gesture includes gestures by two thumbs **1604**-L and **1604**-R (FIGS. **16**A and **16**D)

If the corresponding degree of rotation exceeds a predefined degree of rotation (**1404**-Yes), a 90° screen rotation command is executed (**1406**). For example, the digital image **1502** of FIGS. **15**A and **16**A is rotated from a portrait orientation to a landscape orientation, as shown respectively in FIGS. **15**B and **16**B.

If the corresponding degree of rotation does not exceed a predefined degree of rotation (**1404**-No), a screen rotation command with an acute angle of rotation (i.e., less than 90°) is executed (**1408**). For example, the digital image **1502** of FIGS. **15**C and **16**D is rotated by an acute angle, as shown respectively in FIGS. **15**D and **16**E. Upon ceasing to detect the multifinger twisting gesture, a screen rotation command is executed (**1410**) with an angle of rotation opposite to the acute angle (e.g., with the result shown in FIGS. **15**E and **16**F).

FIGS. **15**A-**15**E illustrate rotating the display of an electronic document or other digital object in accordance with some embodiments. While FIGS. **15**A-**15**E illustrate display rotation in the context of a portable multifunction device **100**, display rotation is not limited to portable multifunction devices. In the example of FIGS. **15**A-**15**E, the electronic document is a digital image **1502**.

In FIGS. **15**A and **15**C, the digital image **1502** is displayed in a portrait orientation. A multifinger twisting gesture **1506** (FIG. **15**A) or **1508** (FIG. **15**C) is detected on the touch screen display. The multifinger twisting gesture **1506** or **1508** has a corresponding degree of rotation. In some embodiments, the degree of rotation corresponds to a degree of rotation of an axis between the contact points on the touch screen display of the two fingers in the multifinger gesture (e.g., an axis between the center points or centroids of the contact regions of the two fingers).

In the example of FIG. **15**A, the multifinger twisting gesture **1506** has a corresponding degree of rotation that exceeds a predefined degree of rotation. Thus, a 90° screen rotation command is executed, with the result that the digital image is displayed in a landscape orientation, as shown in FIG. **15**B. In the example of FIG. **15**C, however, the multifinger twisting gesture **1508** has a corresponding degree of rotation that does not exceed a predefined degree of rotation. A screen rotation command with an acute angle of rotation is executed, with the result shown in FIG. **15**D. Upon ceasing to detect the multifinger twisting gesture **1508**, a screen rotation command with an angle opposite to the acute angle is executed, with the result that the portrait orientation of the digital image **1502** is restored, as shown in FIG. **15**E.

FIGS. **16**A-**16**F illustrate an exemplary screen rotation gesture in accordance with some embodiments. While FIGS. **16**A-**16**F illustrate this screen rotation gesture in the context of a portable multifunction device **100**, this screen rotation gesture is not limited to portable multifunction devices. In the example of FIGS. **16**A-**16**F, this screen rotation gesture is used to rotate the digital image **1502**.

In FIG. **16**A, the device **100** displays the digital image **1502** in a portrait orientation. Simultaneous rotation of two thumbs (e.g., **1604**-L and **1604**-R) in a first sense of rotation is detected on the touch screen display **112**. In some embodiments, the first sense of rotation is a clockwise rotation (e.g., FIG. **16**C). The simultaneous rotation of the two thumbs has a corresponding degree of rotation.

In some embodiments, the sense of rotation for each thumb is detected by monitoring the change in orientation of the contact area of the thumb with the touch screen display. For example, if the contact area of the thumb is elliptical, the change in the orientation of an axis of the ellipse may be detected (e.g., from contact ellipse **1606**-L in FIG. **16**A to contact ellipse **1608**-L in FIG. **16**B, as shown on an enlarged portion of touch screen **112** in FIG. **16**C). In some embodiments, the change in the orientation of the axis of the ellipse determines the corresponding degree of rotation. In some embodiments, at least some of a user's other fingers (i.e., fingers other than thumbs **1604**-L and **1604**-R) support the device **100** by contacting the backside of the device.

In some embodiments, the first sense of rotation is a counterclockwise rotation. For example, if thumb **1604**-L is initially on the lower left side of touch screen **112** (rather than the upper left side in FIG. **16**A), thumb **1604**-R is initially on the upper right side of touch screen **112** (rather than the lower right side in FIG. **16**A), and the thumbs are moved apart from each other, then the sense of rotation detected by the touch screen **112** will be counterclockwise for both thumbs.

If the corresponding degree of rotation exceeds a predefined degree of rotation, a 90° screen rotation command is executed. For example, display of the digital image **1502** is rotated from the portrait orientation of FIG. **16**A to a landscape orientation in FIG. **16**B.

If, however, the corresponding degree of rotation does not exceed a predefined degree of rotation, a screen rotation command with an acute angle of rotation is executed. For example, the digital image **1502** in FIG. **16**D is rotated by an acute angle, with the result shown in FIG. **16**E. Once detection of the two thumbs **1604**-L and **1604**-R ceases, a screen rotation command with an angle of rotation opposite to the acute angle is executed, thereby restoring the digital image **1502** to a portrait orientation, as shown in FIG. **16**F.

While FIGS. **6**A-**6**D, **8**A-**8**D, **10**A-**10**C, **12**A-**12**C, **13**A-**13**C, **15**A-**15**E, and **16**A-**16**F illustrate scrolling, translation, scaling, and rotation operations in the context of a portable multifunction device **100**, similar operations may be performed on any device with a touch-screen display, in accordance with some embodiments. The device, such as device **1700** below, may or may not be portable and the function or functions performed by the device may vary.

FIG. **17** is a block diagram illustrating a device **1700** with a touch-screen display in accordance with some embodiments. Device **1700** need not be portable. The device **1700** typically includes one or more processing units (CPU's) **1710**, one or more network or other communications interfaces **1760**, memory **1770**, and one or more communication buses **1720** for interconnecting these components. The communication buses **1720** may include circuitry (sometimes called a chipset) that interconnects and controls communications between system components. The device **1700** includes a user interface **1730** comprising a touch-screen display **1740**. The user interface **1730** also may include a keyboard and/or mouse (or other pointing device) **1750**. Memory **1770** includes high-speed random access memory, such as DRAM, SRAM, DDR RAM or other random access solid state memory devices; and may include non-volatile memory, such as one or more magnetic disk storage devices, optical disk storage devices, flash memory devices, or other non-volatile solid state storage devices. Memory **1770** may optionally include one or more storage devices remotely located from the CPU(s) **1710**. In some embodiments, memory **1770** stores

programs, modules, and data structures analogous to the programs, modules, and data structures stored in the memory **102** of portable multifunction device **100** (FIG. **1**), or a subset thereof. Furthermore, memory **1770** may store additional programs, modules, and data structures (not shown) not present in the memory **102** of portable multifunction device **100**.

Each of the above identified elements in FIG. **17** may be stored in one or more of the previously mentioned memory devices. Each of the above identified modules corresponds to a set of instructions for performing a function described above. The above identified modules or programs (i.e., sets of instructions) need not be implemented as separate software programs, procedures or modules, and thus various subsets of these modules may be combined or otherwise re-arranged in various embodiments. In some embodiments, memory **1770** may store a subset of the modules and data structures identified above. Furthermore, memory **1770** may store additional modules and data structures not described above.

The foregoing description, for purpose of explanation, has been described with reference to specific embodiments. However, the illustrative discussions above are not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain the principles of the invention and its practical applications, to thereby enable others skilled in the art to best utilize the invention and various embodiments with various modifications as are suited to the particular use contemplated.

What is claimed is:

1. A computer-implemented method, comprising:

at a device with a touch screen display:

displaying a first portion of an electronic document;

detecting a movement of an object on or near the touch screen display;

in response to detecting the movement, translating the electronic document displayed on the touch screen display in a first direction to display a second portion of the electronic document, wherein the second portion is different from the first portion;

in response to an edge of the electronic document being reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display:

displaying an area beyond the edge of the document, and

displaying a third portion of the electronic document, wherein the third portion is smaller than the first portion; and

in response to detecting that the object is no longer on or near the touch screen display, translating the electronic document in a second direction until the area beyond the edge of the electronic document is no longer displayed to display a fourth portion of the electronic document, wherein the fourth portion is different from the first portion.

2. The computer-implemented method of claim **1**, wherein the first portion of the electronic document, the second portion of the electronic document, the third portion of the electronic document, and the fourth portion of the electronic document are displayed at the same magnification.

3. The computer-implemented method of claim **1**, wherein the movement of the object is on the touch screen display.

4. The computer-implemented method of claim **1**, wherein the object is a finger.

5. The computer-implemented method of claim **1**, wherein the first direction is a vertical direction, a horizontal direction, or a diagonal direction.

6. The computer-implemented method of claim **1**, wherein the electronic document is a web page.

7. The computer-implemented method of claim **1**, wherein the electronic document is a digital image.

8. The computer-implemented method of claim **1**, wherein the electronic document is a word processing, spreadsheet, email or presentation document.

9. The computer-implemented method of claim **1**, wherein the electronic document includes a list of items.

10. The computer-implemented method of claim **1**, wherein the second direction is opposite the first direction.

11. The computer-implemented method of claim **1**, wherein translating in the first direction prior to reaching an edge of the document has an associated speed of translation that corresponds to a speed of movement of the object.

12. The computer-implemented method of claim **1**, wherein translating in the first direction is in accordance with a simulation of an equation of motion having friction.

13. The computer-implemented method of claim **1**, wherein the area beyond the edge of the document is black, gray, a solid color, or white.

14. The computer-implemented method of claim **1**, wherein the area beyond the edge of the document is visually distinct from the document.

15. The computer-implemented method of claim **1**, wherein translating the document in the second direction is a damped motion.

16. The computer-implemented method of claim **1**, wherein changing from translating in the first direction to translating in the second direction until the area beyond the edge of the document is no longer displayed makes the edge of the electronic document appear to be elastically attached to an edge of the touch screen display or to an edge displayed on the touch screen display.

17. The computer-implemented method of claim **1**, wherein translating in the first direction prior to reaching the edge of the electronic document has a first associated translating distance that corresponds to a distance of movement of the object prior to reaching the edge of the electronic document; and wherein displaying an area beyond the edge of the electronic document comprises translating the electronic document in the first direction for a second associated translating distance, wherein the second associated translating distance is less than a distance of movement of the object after reaching the edge of the electronic document.

18. The computer-implemented method of claim **1**, wherein translating in the first direction prior to reaching the edge of the electronic document has a first associated translating speed that corresponds to a speed of movement of the object, and wherein displaying an area beyond the edge of the electronic document comprises translating the electronic document in the first direction at a second associated translating speed, wherein the second associated translating speed is slower than the first associated translating speed.

19. A device, comprising:

a touch screen display;

one or more processors;

memory; and

one or more programs, wherein the one or more programs are stored in the memory and configured to be executed by the one or more processors, the programs including:

instructions for displaying a first portion of an electronic document;

instructions for detecting a movement of an object on or near the touch screen display;

instructions for translating the electronic document displayed on the touch screen display in a first direction to display a second portion of the electronic document, wherein the second portion is different from the first portion, in response to detecting the movement;

instructions for displaying an area beyond an edge of the electronic document and displaying a third portion of the electronic document, wherein the third portion is smaller than the first portion, in response to the edge of the electronic document being reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display; and

instructions for translating the electronic document in a second direction until the area beyond the edge of the electronic document is no longer displayed to display a fourth portion of the electronic document, wherein the fourth portion is different from the first portion, in response to detecting that the object is no longer on or near the touch screen display.

20. A computer readable storage medium having stored therein instructions, which when executed by a device with a touch screen display, cause the device to:

display a first portion of an electronic document;

detect a movement of an object on or near the touch screen display;

translate the electronic document displayed on the touch screen display in a first direction to display a second portion of the electronic document, wherein the second portion is different from the first portion, in response to detecting the movement

display an area beyond an edge of the electronic document and display a third portion of the electronic document, wherein the third portion is smaller than the first portion, if the edge of the electronic document is reached while translating the electronic document in the first direction while the object is still detected on or near the touch screen display; and

translate the electronic document in a second direction until the area beyond the edge of the electronic document is no longer displayed to display a fourth portion of the electronic document, wherein the fourth portion is different from the first portion, in response to detecting that the object is no longer on or near the touch screen display.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,469,381 B2
APPLICATION NO. : 11/956969
DATED : December 23, 2008
INVENTOR(S) : Ording

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 38, line 8, please insert -- ; -- after movement.

Signed and Sealed this

Seventeenth Day of February, 2009

John Doll

JOHN DOLL
Acting Director of the United States Patent and Trademark Office

Exhibit 5

US007844915B2

(12) **United States Patent**
Platzer et al.

(10) **Patent No.: US 7,844,915 B2**
(45) **Date of Patent: Nov. 30, 2010**

(54) **APPLICATION PROGRAMMING INTERFACES FOR SCROLLING OPERATIONS**

(75) Inventors: **Andrew Platzer**, Santa Clara, CA (US); **Scott Herz**, Santa Clara, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 583 days.

(21) Appl. No.: **11/620,717**

(22) Filed: **Jan. 7, 2007**

(65) **Prior Publication Data**

US 2008/0168384 A1 Jul. 10, 2008

(51) **Int. Cl.**

G06F 3/00	(2006.01)
G06F 3/033	(2006.01)
G06F 3/041	(2006.01)
G06F 3/048	(2006.01)

(52) **U.S. Cl.** **715/781**; 715/784; 715/800; 345/173

(58) **Field of Classification Search** 715/764, 715/765, 784, 786, 788, 800, 864, 866, 973, 715/974; 345/156, 157, 169, 173
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,534,893	A	7/1996	Hansen et al.	
5,903,902	A	5/1999	Orr et al.	
6,028,602	A	2/2000	Weidenfeller et al.	
6,486,896	B1	11/2002	Ubillos	
6,677,965	B1 *	1/2004	Ullmann et al.	715/786
6,741,996	B1	5/2004	Brechner et al.	
6,839,721	B2	1/2005	Schwols	
6,903,927	B2	6/2005	Anlauff	
6,957,392	B2	10/2005	Simister et al.	
6,958,749	B1 *	10/2005	Matsushita et al.	345/175
7,009,626	B2	3/2006	Anwar	
7,088,374	B2	8/2006	David et al.	
7,117,453	B2	10/2006	Drucker et al.	
7,173,623	B2	2/2007	Calkins et al.	
7,337,412	B2	2/2008	Guido et al.	
7,346,850	B2	3/2008	Swartz et al.	

(Continued)

FOREIGN PATENT DOCUMENTS

EP	1517228	3/2005

(Continued)

OTHER PUBLICATIONS

Toshiyuki Masui et al; "Elastic Graphical Interfaces for Precise Data Manipulation", 1995; ACM; pp. 143-144.*

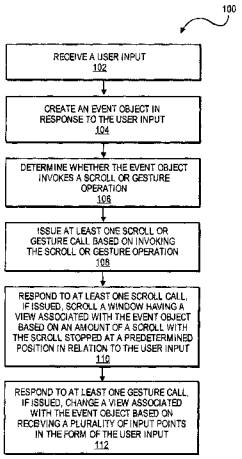
(Continued)

Primary Examiner—Xiomara L. Bautista
(74) *Attorney, Agent, or Firm*—Blakely, Sokoloff, Taylor & Zafman LLP

(57) **ABSTRACT**

At least certain embodiments of the present disclosure include an environment with user interface software interacting with a software application. A method for operating through an application programming interface (API) in this environment includes transferring a set bounce call. The method further includes setting at least one of maximum and minimum bounce values. The set bounce call causes a bounce of a scrolled region in an opposite direction of a scroll based on a region past an edge of the scrolled region being visible in a display region at the end of the scroll.

21 Claims, 37 Drawing Sheets



U.S. PATENT DOCUMENTS

7,561,159	B2	7/2009	Abel et al.	
7,576,732	B2 *	8/2009	Lii	345/173
2001/0045949	A1	11/2001	Chithambaram et al.	
2002/0194589	A1	12/2002	Cristofalo et al.	
2003/0095096	A1	5/2003	Robbin et al.	
2003/0122787	A1	7/2003	Zimmerman et al.	
2003/0132959	A1	7/2003	Simister et al.	
2003/0160832	A1	8/2003	Ridgley et al.	
2003/0174149	A1	9/2003	Fujisaki et al.	
2004/0021676	A1	2/2004	Chen et al.	
2004/0021698	A1	2/2004	Baldwin et al.	
2004/0100479	A1	5/2004	Nakano et al.	
2004/0215643	A1	10/2004	Brechner et al.	
2004/0222992	A1	11/2004	Calkins et al.	
2004/0224638	A1	11/2004	Fadell et al.	
2005/0057524	A1 *	3/2005	Hill et al.	345/173
2005/0088443	A1	4/2005	Blanco et al.	
2005/0193015	A1	9/2005	Logston et al.	
2006/0038796	A1 *	2/2006	Hinckley et al.	345/173
2006/0190833	A1	8/2006	SanGiovanni et al.	
2006/0236263	A1 *	10/2006	Bathiche et al.	715/786
2007/0055967	A1	3/2007	Poff et al.	
2007/0075965	A1	4/2007	Huppi et al.	
2007/0174257	A1	7/2007	Howard	
2007/0185876	A1	8/2007	Mendis et al.	
2007/0252821	A1 *	11/2007	Hollemans et al.	345/173
2007/0288856	A1	12/2007	Butlin et al.	
2008/0005703	A1 *	1/2008	Radivojevic et al.	715/863
2008/0016096	A1	1/2008	Wilding et al.	
2008/0034029	A1	2/2008	Fang et al.	
2008/0048978	A1 *	2/2008	Trent et al.	345/157
2008/0168395	A1 *	7/2008	Ording et al.	715/833
2008/0231610	A1 *	9/2008	Hotelling et al.	345/173
2009/0259969	A1 *	10/2009	Pallakoff	715/808

FOREIGN PATENT DOCUMENTS

GB	2 319 591 A		5/1998
GB	2319591 A	*	5/1998
WO	WO-2006/067711		6/2006
WO	WO 2008/085848 A1	*	7/2008
WO	WO 2008/085877 A1	*	7/2008
WO	WO-2008085848		7/2008
WO	WO-2008085877		7/2008

OTHER PUBLICATIONS

Office Action, U.S. Appl. No. 11/620,723, mailed Apr. 1, 2009, 8 pages.
Office Action, U.S. Appl. No. 11/620,709, mailed Apr. 1, 2009, 8 pages.
Office Action, U.S. Appl. No. 11/620,720, mailed Jun. 23, 2009, 17 pages.
Office Action, U.S. Appl. No. 11/620,720, mailed Dec. 23, 2008, 18 pages.
PCT International Search Report and Written Opinion for PCT International Appln. No. US2008/000058, mailed Jul. 31, 2008 (10 pages).
PCT International Search Report and Written Opinion for PCT International Appln. No. US2008/000089, mailed Apr. 6, 2008 (14 pages).
PCT International Search Report and Written Opinion for PCT International Appln. No. PCT/US2008/000103, mailed Jun. 3, 2008 (15 pages).
PCT International Search Report and Written Opinion for PCT International Appln. No. PCT/US2008/000069, mailed May 2, 2008 (16 pages).
PCT International Search Report and Written Opinion for PCT International Appin. No. PCT/US2008/000060, mailed Apr. 22, 2008 (12 pages).
Office Action, U.S. Appl. No. 11/620,723, mailed Jun. 8, 2010, 7 pages.
Office Action, U.S. Appl. No. 11/620,709, mailed Jun. 9, 2010, 7 pages.
Final Office Action, U.S. Appl. No. 11/620,709 mailed Nov. 13, 2009, 8 pages.
Final Office Action, U.S. Appl. No. 11/620,723, mailed Nov. 17, 2009., 10 pages.
Office Action, U.S. Appl. No. 11/620,720 mailed Nov. 18, 2009, 17 pages.

* cited by examiner

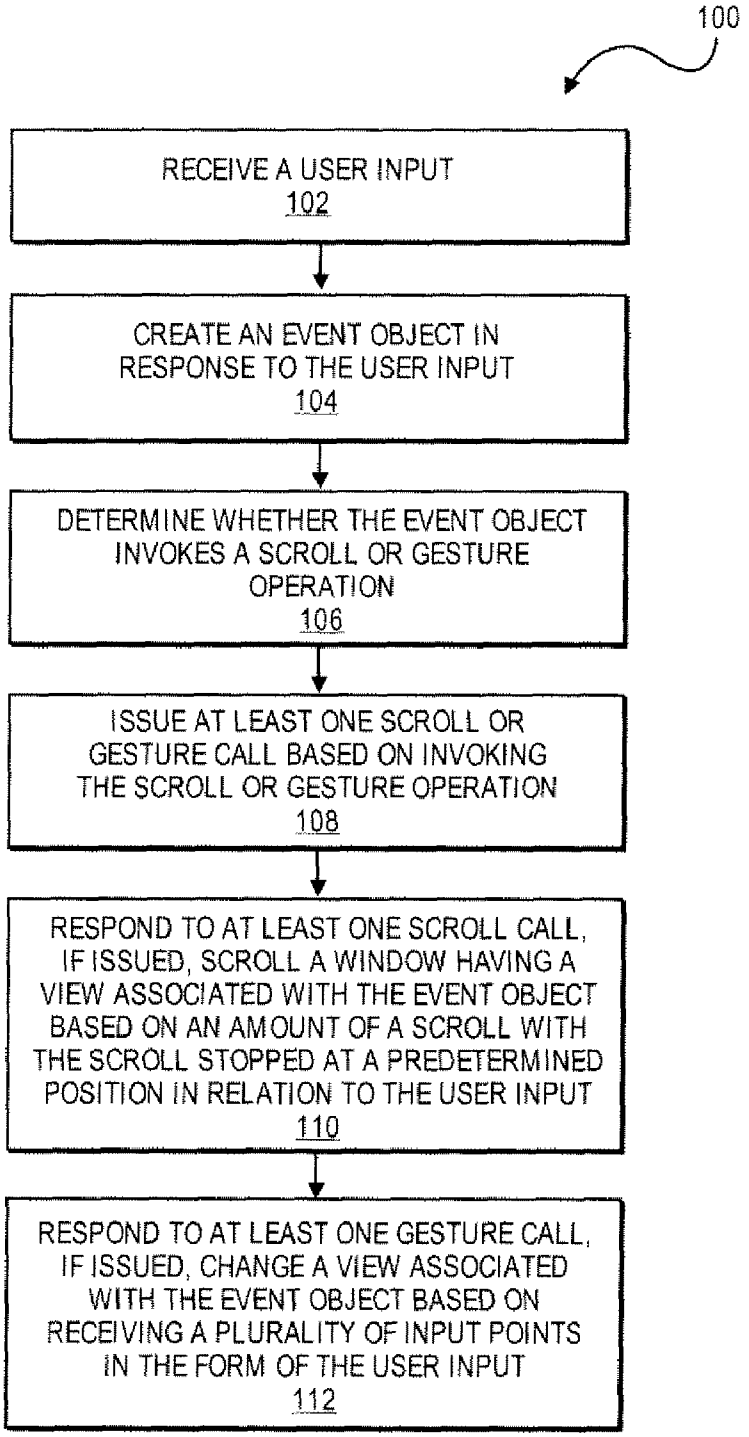

100
RECEIVE A USER INPUT
102
CREATE AN EVENT OBJECT IN RESPONSE TO THE USER INPUT
104
DETERMINE WHETHER THE EVENT OBJECT INVOKES A SCROLL OR GESTURE OPERATION
106
ISSUE AT LEAST ONE SCROLL OR GESTURE CALL BASED ON INVOKING THE SCROLL OR GESTURE OPERATION
108
RESPOND TO AT LEAST ONE SCROLL CALL, IF ISSUED, SCROLL A WINDOW HAVING A VIEW ASSOCIATED WITH THE EVENT OBJECT BASED ON AN AMOUNT OF A SCROLL WITH THE SCROLL STOPPED AT A PREDETERMINED POSITION IN RELATION TO THE USER INPUT
110
RESPOND TO AT LEAST ONE GESTURE CALL, IF ISSUED, CHANGE A VIEW ASSOCIATED WITH THE EVENT OBJECT BASED ON RECEIVING A PLURALITY OF INPUT POINTS IN THE FORM OF THE USER INPUT
112

FIG. 1

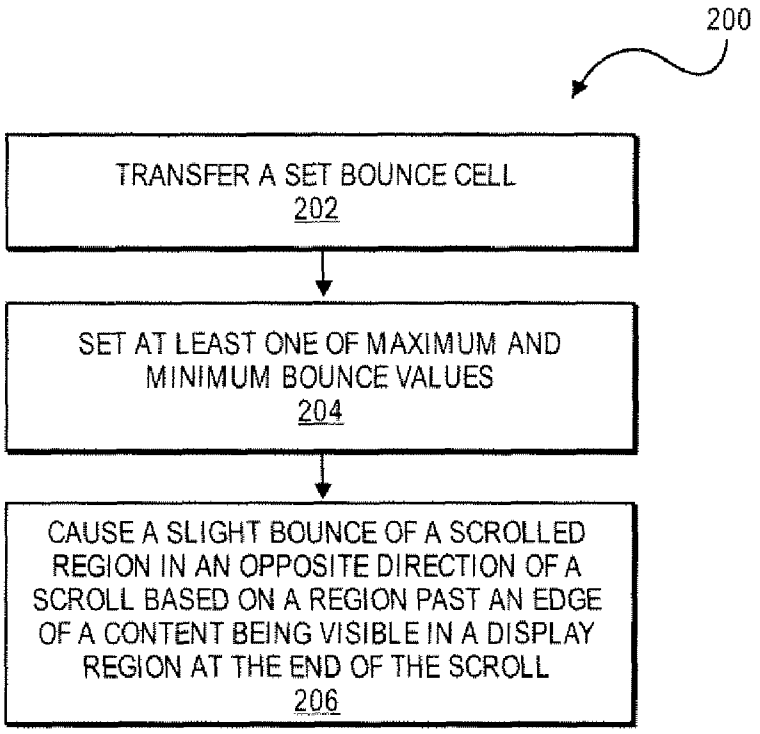

200
TRANSFER A SET BOUNCE CELL
202
SET AT LEAST ONE OF MAXIMUM AND MINIMUM BOUNCE VALUES
204
CAUSE A SLIGHT BOUNCE OF A SCROLLED REGION IN AN OPPOSITE DIRECTION OF A SCROLL BASED ON A REGION PAST AN EDGE OF A CONTENT BEING VISIBLE IN A DISPLAY REGION AT THE END OF THE SCROLL
206

FIG. 2

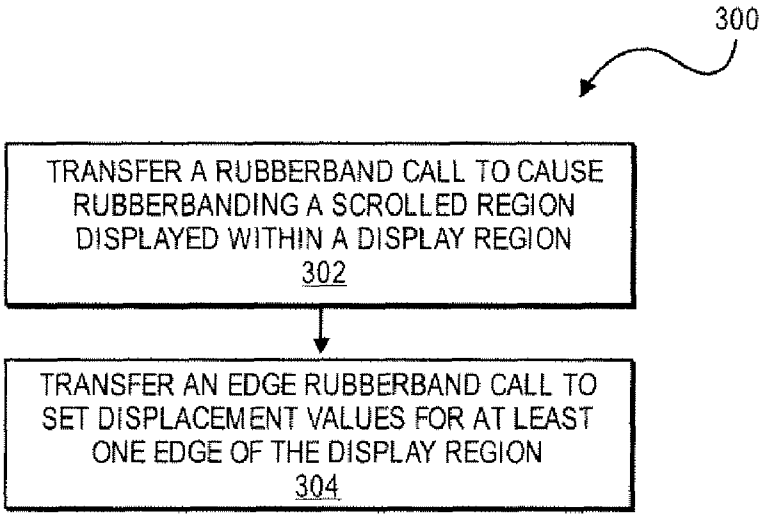

300
TRANSFER A RUBBERBAND CALL TO CAUSE RUBBERBANDING A SCROLLED REGION DISPLAYED WITHIN A DISPLAY REGION
302
TRANSFER AN EDGE RUBBERBAND CALL TO SET DISPLACEMENT VALUES FOR AT LEAST ONE EDGE OF THE DISPLAY REGION
304

FIG. 3

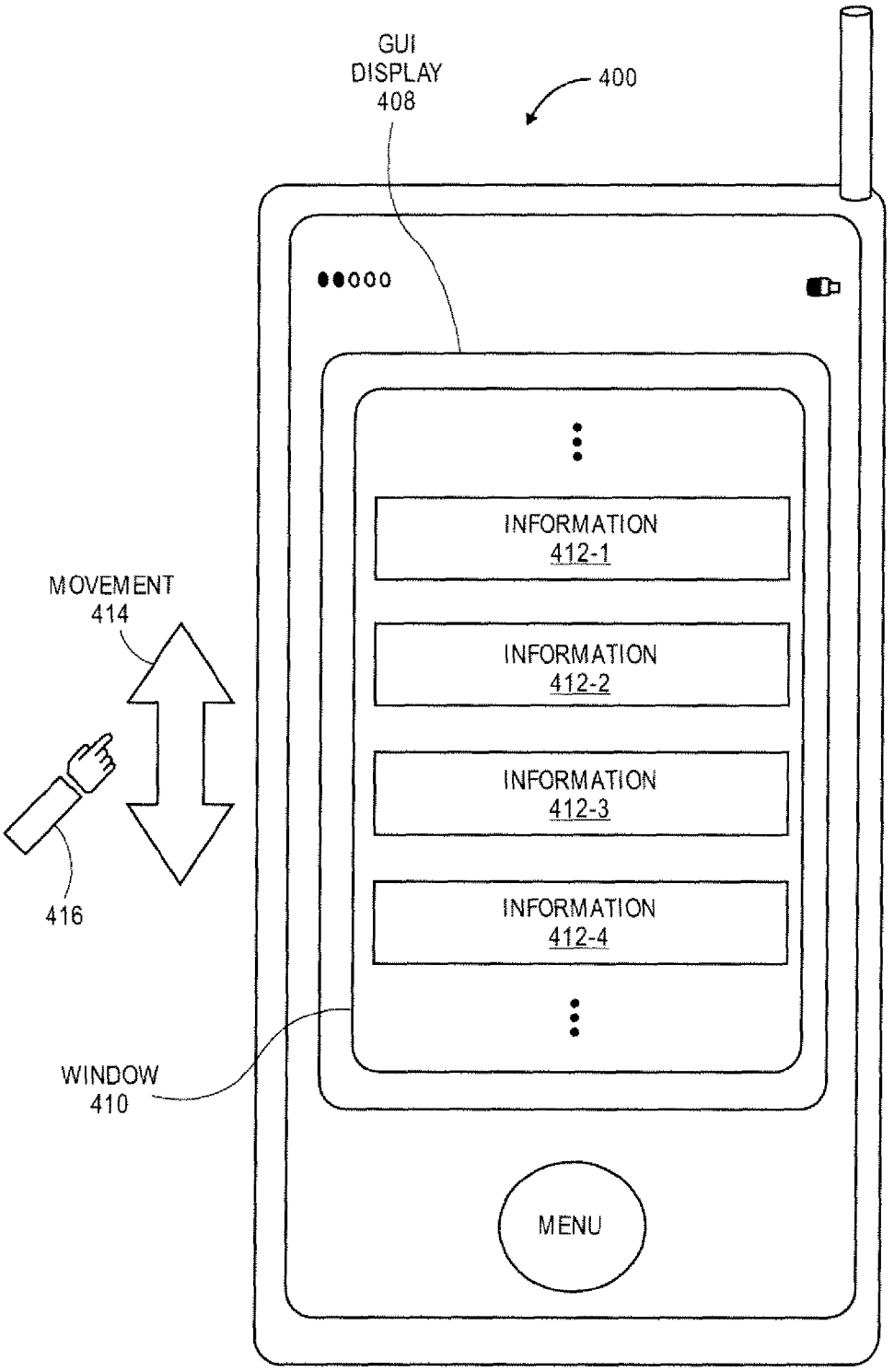

GUI
DISPLAY
408
400
INFORMATION
412-1
INFORMATION
412-2
INFORMATION
412-3
INFORMATION
412-4
MOVEMENT
414
416
WINDOW
410
MENU

FIG. 4

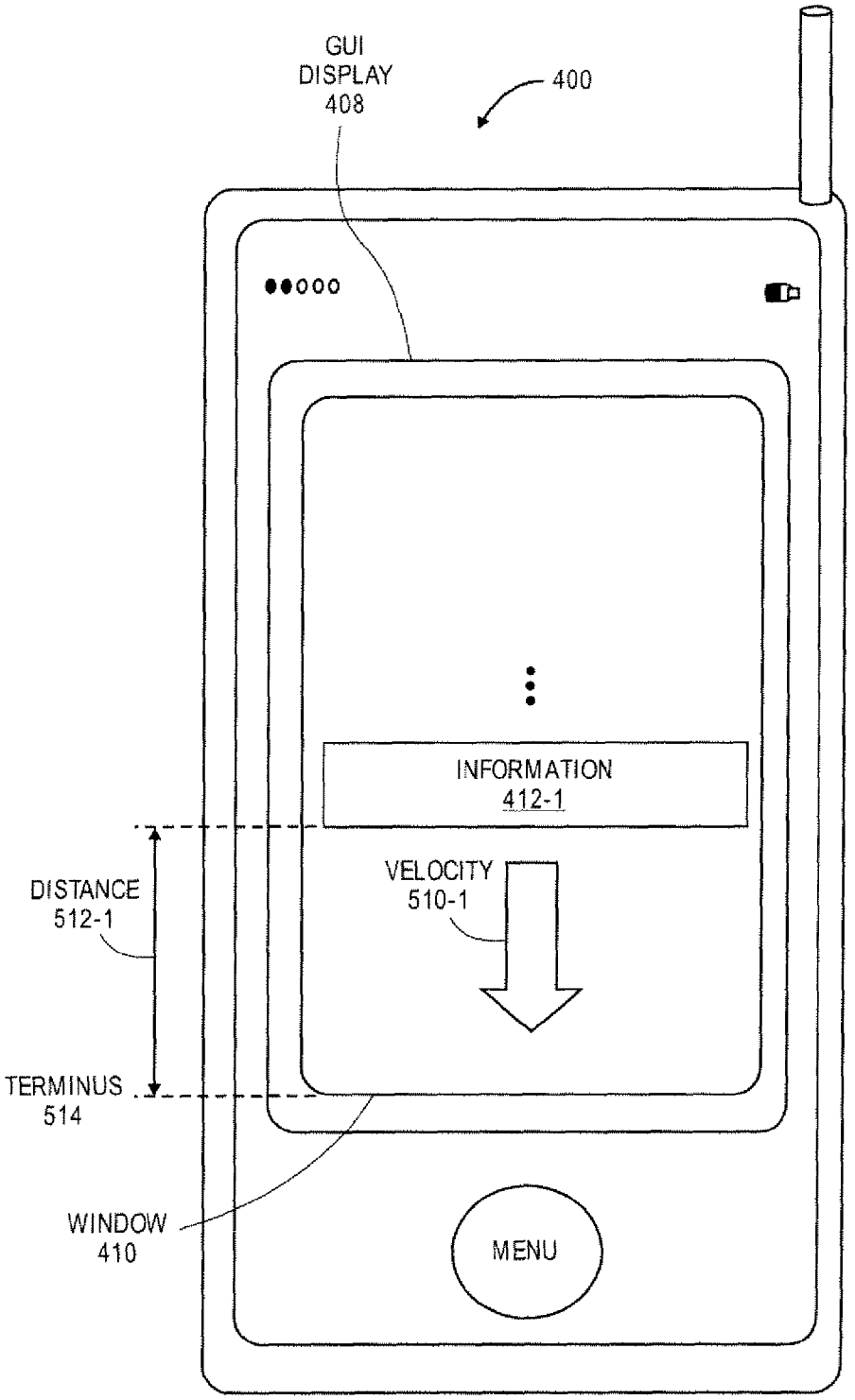

GUI
DISPLAY
408
400
INFORMATION
412-1
VELOCITY
510-1
DISTANCE
512-1
TERMINUS
514
WINDOW
410
MENU

FIG. 5A

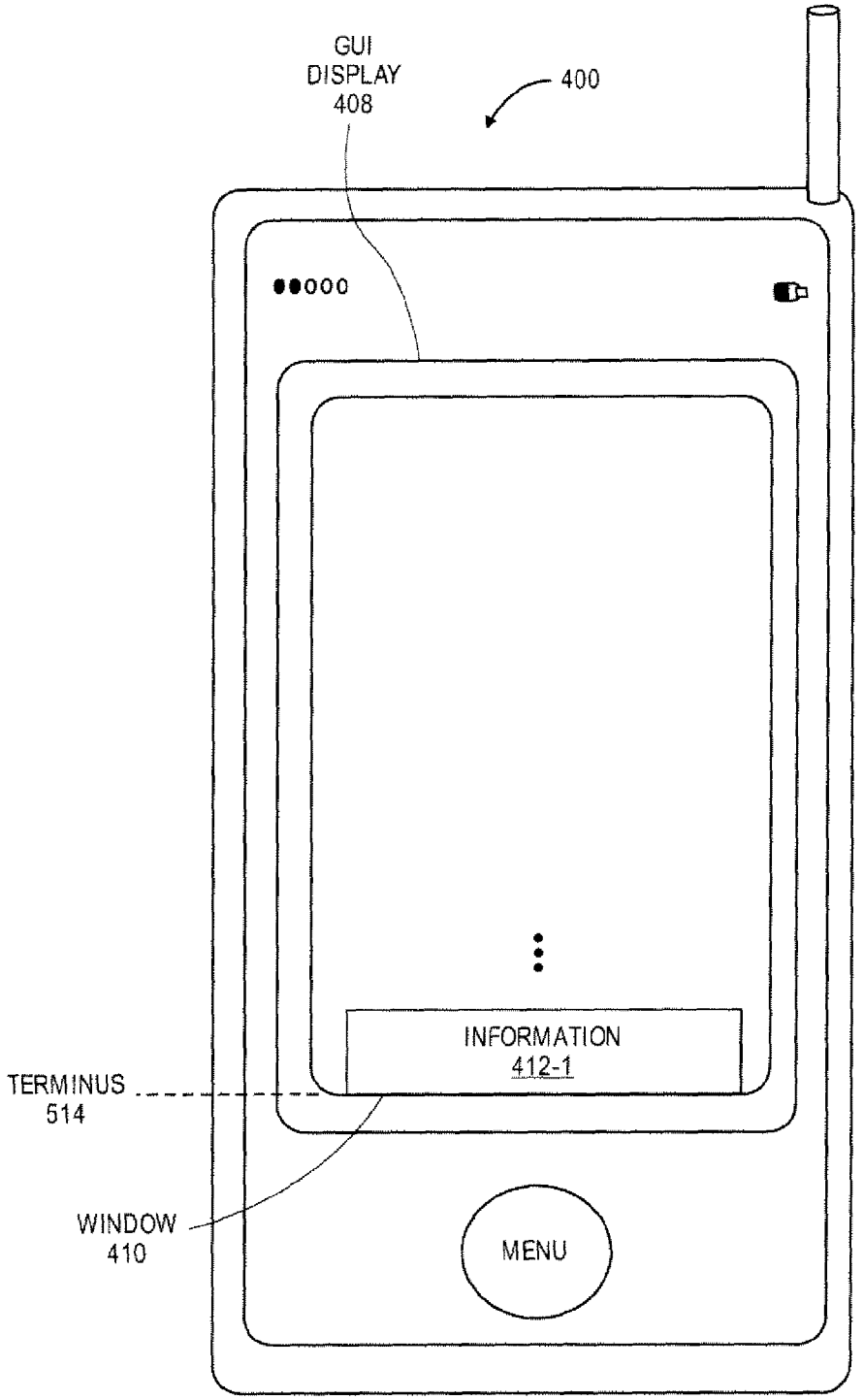

GUI
DISPLAY
408
400
INFORMATION
412-1
TERMINUS
514
WINDOW
410
MENU

FIG. 5B

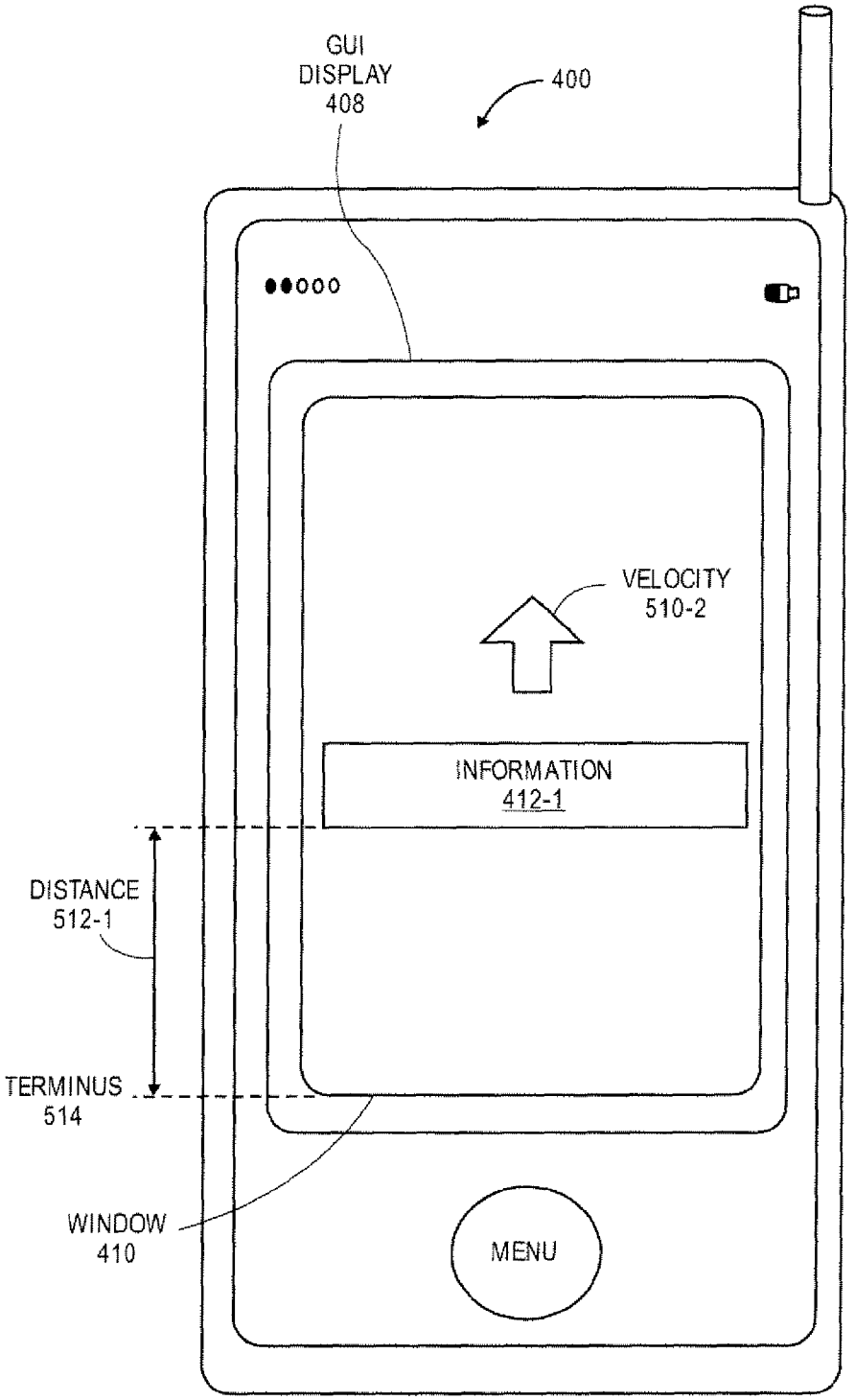

GUI
DISPLAY
408
400
VELOCITY
510-2
INFORMATION
412-1
DISTANCE
512-1
TERMINUS
514
WINDOW
410
MENU

FIG. 5C

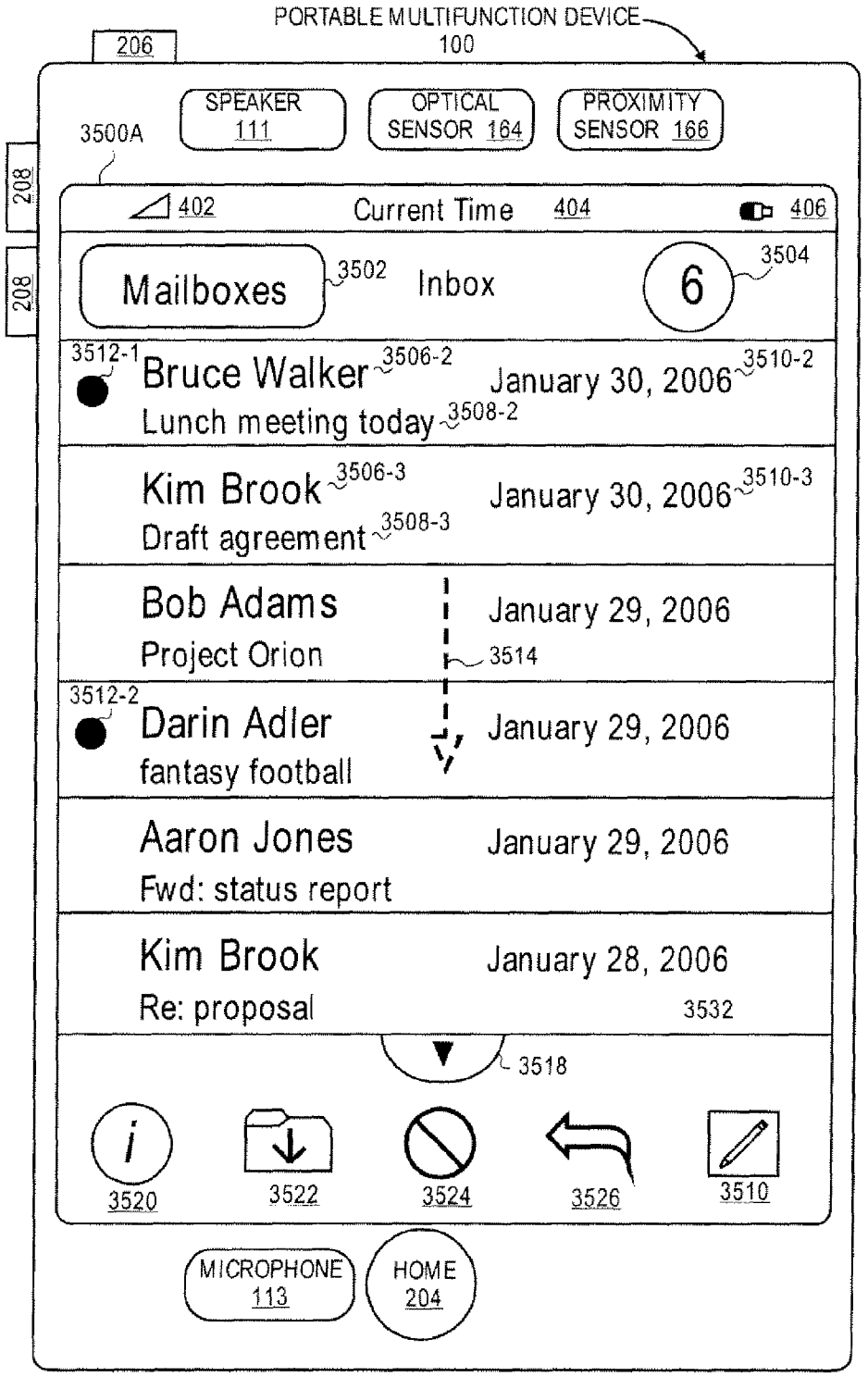

PORTABLE MULTIFUNCTION DEVICE
100
206
SPEAKER
111
OPTICAL
SENSOR 164
PROXIMITY
SENSOR 166
3500A
208
208
402
Current Time
404
406
Mailboxes
3502
Inbox
6
3504
3512-1
Bruce Walker
3506-2
January 30, 2006
3510-2
Lunch meeting today
3508-2
Kim Brook
3506-3
January 30, 2006
3510-3
Draft agreement
3508-3
Bob Adams
January 29, 2006
Project Orion
3514
3512-2
Darin Adler
January 29, 2006
fantasy football
Aaron Jones
January 29, 2006
Fwd: status report
Kim Brook
January 28, 2006
Re: proposal
3532
3518
3520
3522
3524
3526
3510
MICROPHONE
113
HOME
204

FIG. 6A

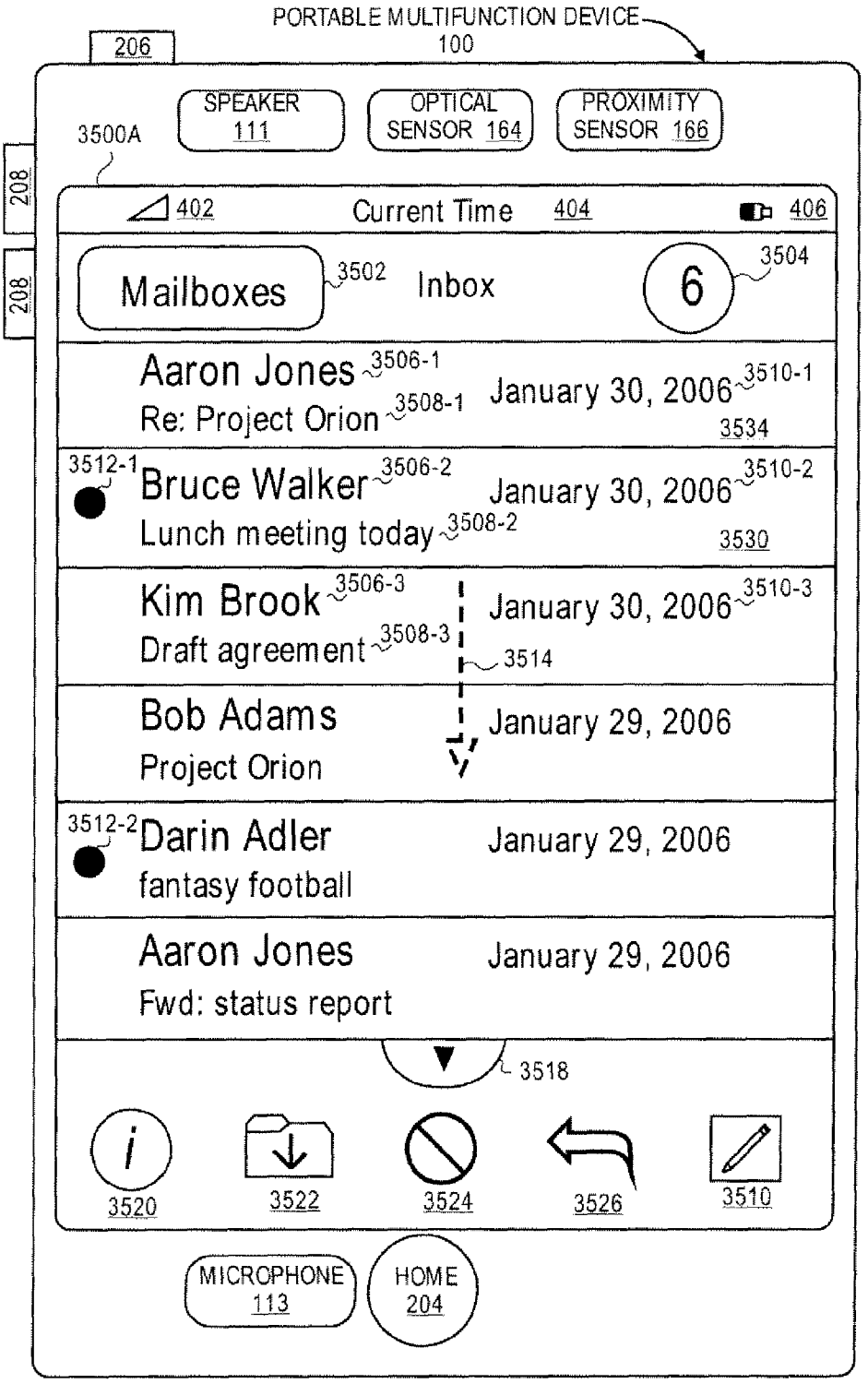

PORTABLE MULTIFUNCTION DEVICE
100
206
SPEAKER
111
OPTICAL
SENSOR 164
PROXIMITY
SENSOR 166
3500A
208
208
402
Current Time
404
406
Mailboxes
3502
Inbox
6
3504
Aaron Jones
3506-1
Re: Project Orion
3508-1
January 30, 2006
3510-1
3534
3512-1
Bruce Walker
3506-2
Lunch meeting today
3508-2
January 30, 2006
3510-2
3530
Kim Brook
3506-3
Draft agreement
3508-3
January 30, 2006
3510-3
3514
Bob Adams
Project Orion
January 29, 2006
3512-2
Darin Adler
fantasy football
January 29, 2006
Aaron Jones
Fwd: status report
January 29, 2006
3518
3520
3522
3524
3526
3510
MICROPHONE
113
HOME
204

FIG. 6B

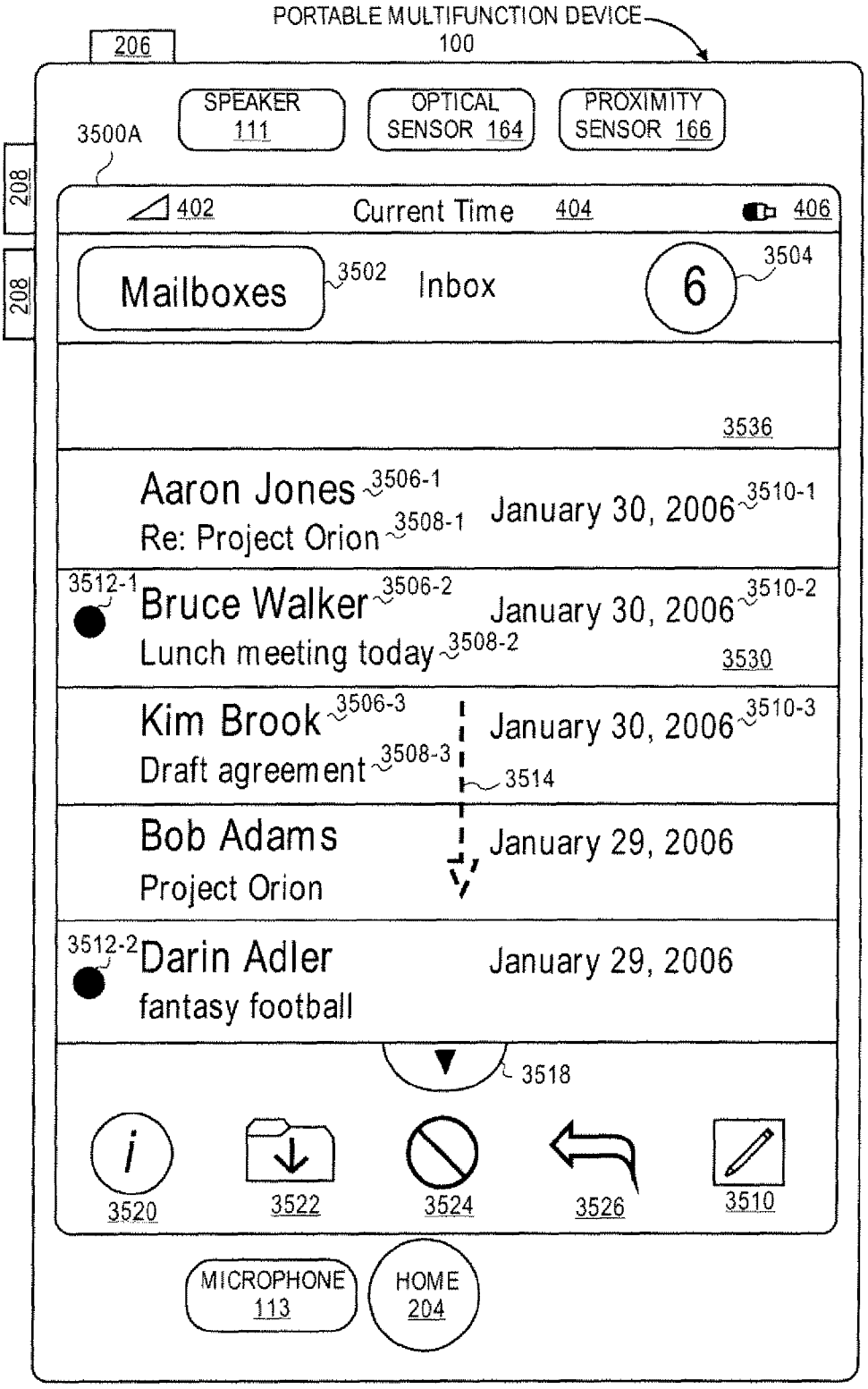

PORTABLE MULTIFUNCTION DEVICE
100
206
SPEAKER
111
OPTICAL
SENSOR 164
PROXIMITY
SENSOR 166
3500A
208
208
402
Current Time
404
406
Mailboxes
3502
Inbox
6
3504
3536
Aaron Jones
3506-1
Re: Project Orion
3508-1
January 30, 2006
3510-1
3512-1
Bruce Walker
3506-2
Lunch meeting today
3508-2
January 30, 2006
3510-2
3530
Kim Brook
3506-3
Draft agreement
3508-3
3514
January 30, 2006
3510-3
Bob Adams
Project Orion
January 29, 2006
3512-2
Darin Adler
fantasy football
January 29, 2006
3518
3520
3522
3524
3526
3510
MICROPHONE
113
HOME
204

FIG. 6C

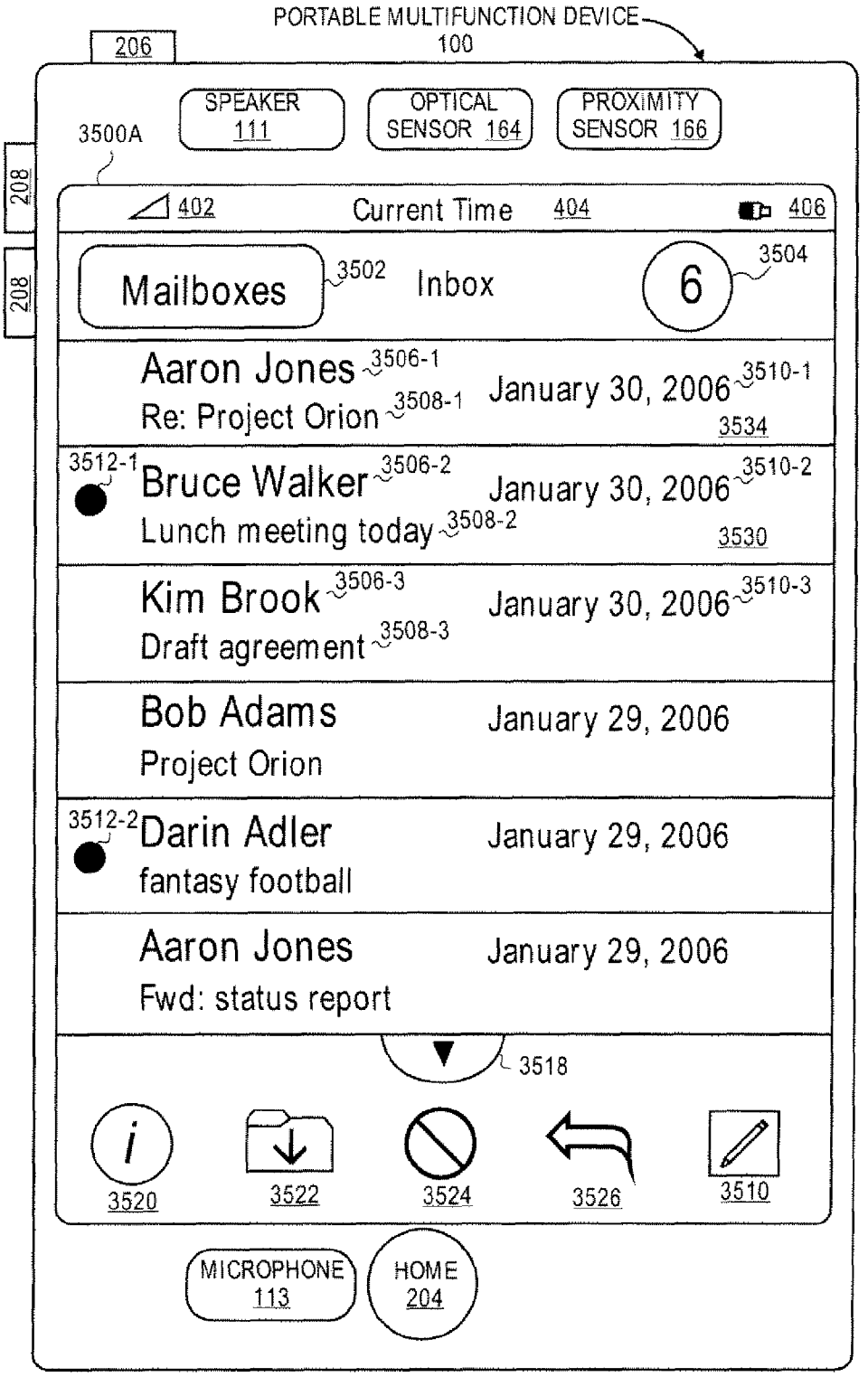

PORTABLE MULTIFUNCTION DEVICE
100
206
SPEAKER
111
OPTICAL
SENSOR 164
PROXIMITY
SENSOR 166
3500A
208
208
402
Current Time
404
406
Mailboxes
3502
Inbox
6
3504
Aaron Jones
3506-1
January 30, 2006
3510-1
Re: Project Orion
3508-1
3534
3512-1
Bruce Walker
3506-2
January 30, 2006
3510-2
Lunch meeting today
3508-2
3530
Kim Brook
3506-3
January 30, 2006
3510-3
Draft agreement
3508-3
Bob Adams
January 29, 2006
Project Orion
3512-2
Darin Adler
January 29, 2006
fantasy football
Aaron Jones
January 29, 2006
Fwd: status report
3518
3520
3522
3524
3526
3510
MICROPHONE
113
HOME
204

FIG. 6D

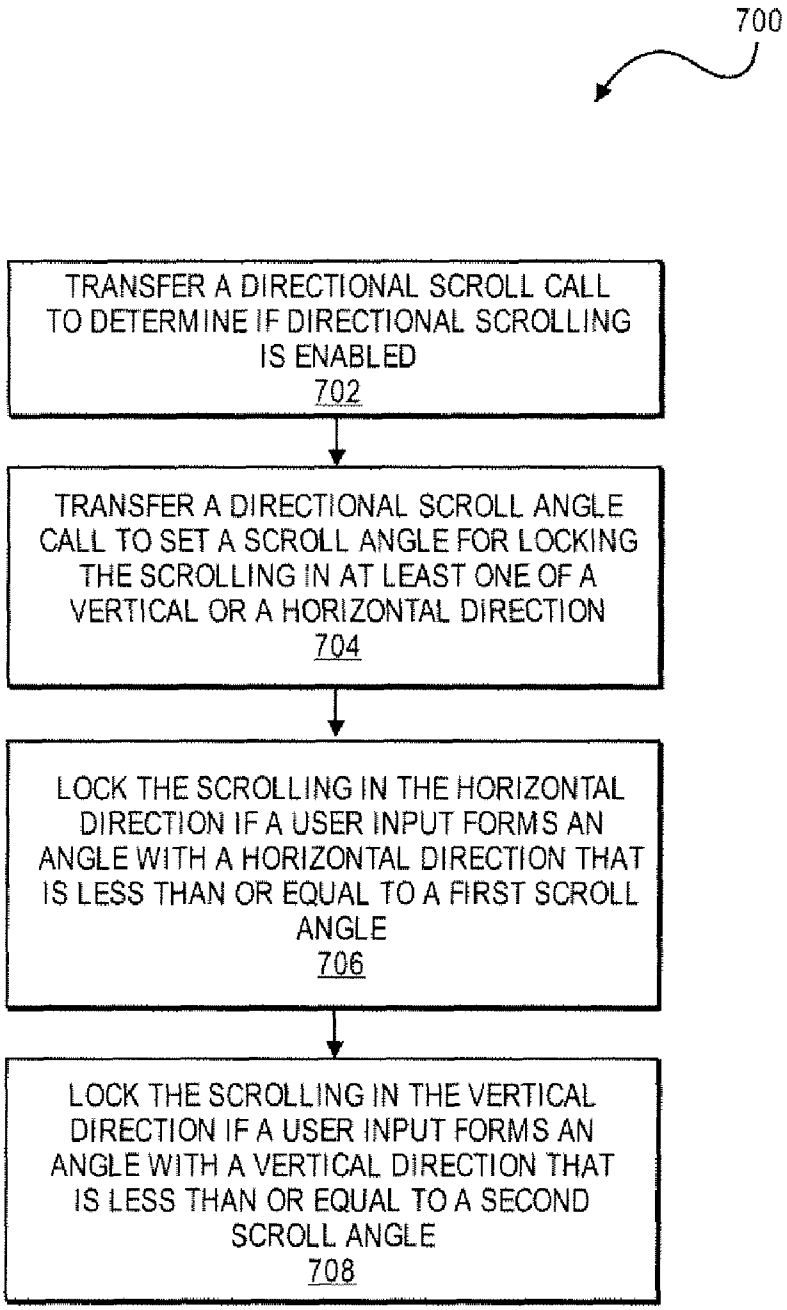

700
TRANSFER A DIRECTIONAL SCROLL CALL TO DETERMINE IF DIRECTIONAL SCROLLING IS ENABLED
702
TRANSFER A DIRECTIONAL SCROLL ANGLE CALL TO SET A SCROLL ANGLE FOR LOCKING THE SCROLLING IN AT LEAST ONE OF A VERTICAL OR A HORIZONTAL DIRECTION
704
LOCK THE SCROLLING IN THE HORIZONTAL DIRECTION IF A USER INPUT FORMS AN ANGLE WITH A HORIZONTAL DIRECTION THAT IS LESS THAN OR EQUAL TO A FIRST SCROLL ANGLE
706
LOCK THE SCROLLING IN THE VERTICAL DIRECTION IF A USER INPUT FORMS AN ANGLE WITH A VERTICAL DIRECTION THAT IS LESS THAN OR EQUAL TO A SECOND SCROLL ANGLE
708

FIG. 7

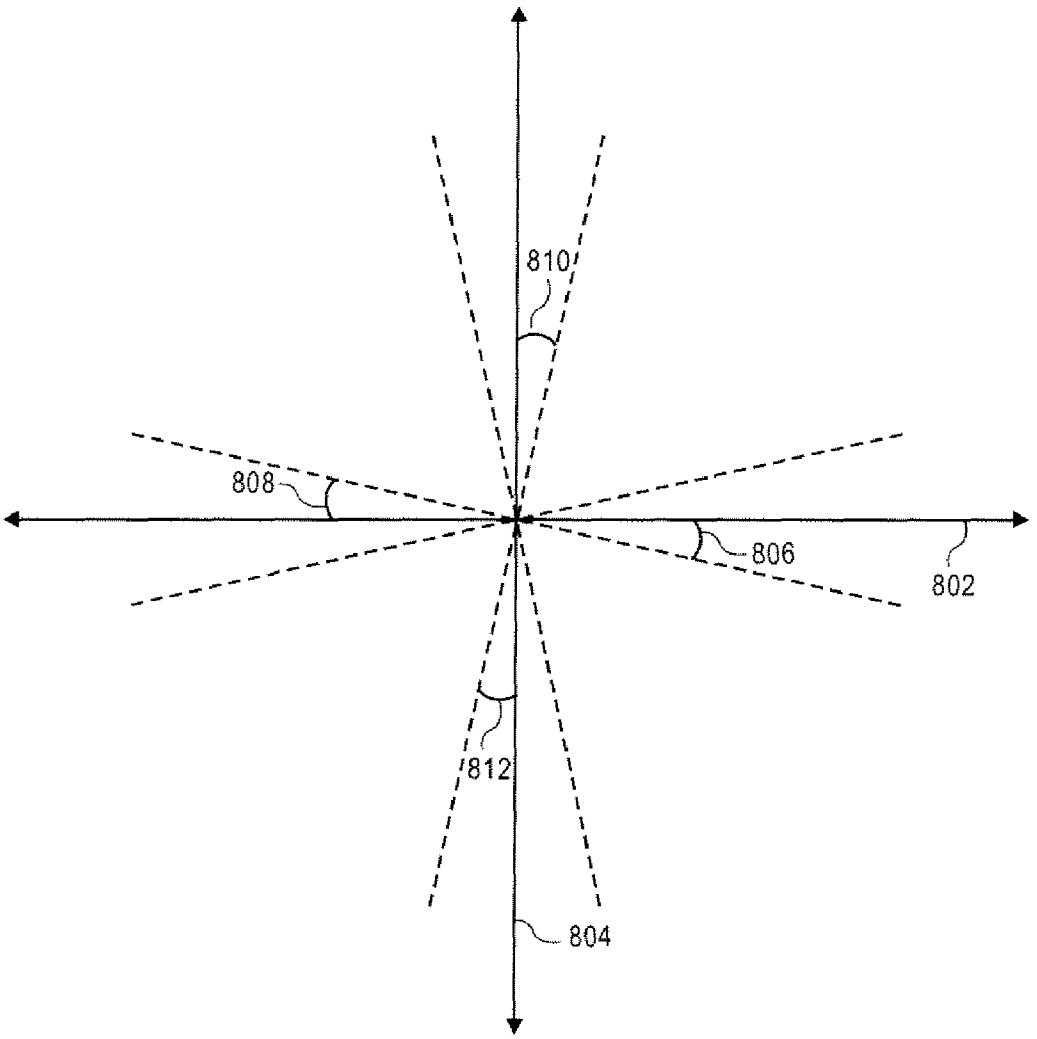


FIG. 8

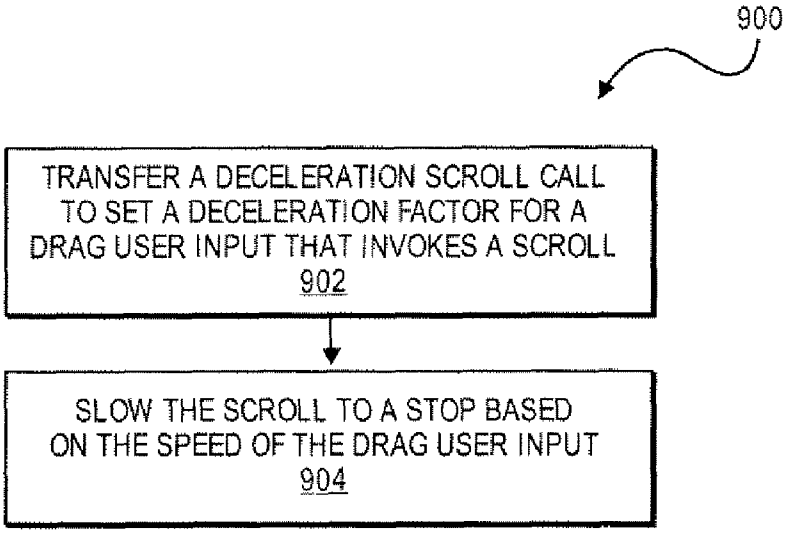

900
TRANSFER A DECELERATION SCROLL CALL TO SET A DECELERATION FACTOR FOR A DRAG USER INPUT THAT INVOKES A SCROLL
902
SLOW THE SCROLL TO A STOP BASED ON THE SPEED OF THE DRAG USER INPUT
904

FIG. 9

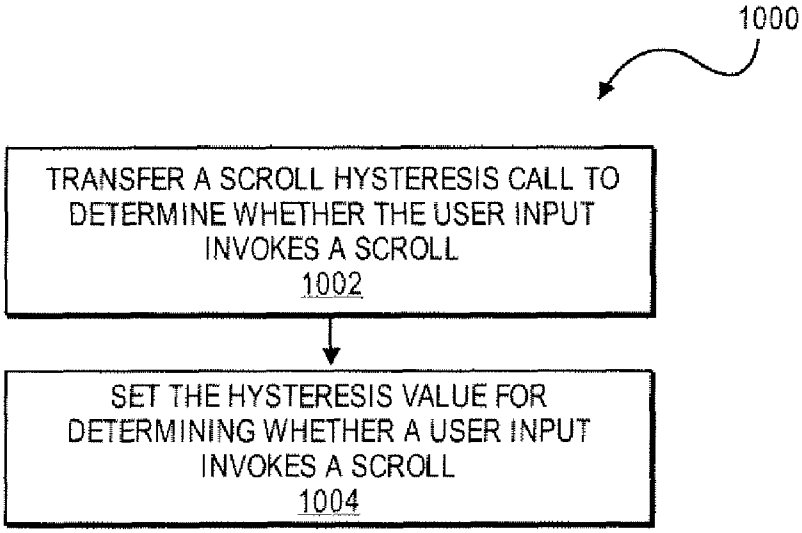

1000
TRANSFER A SCROLL HYSTERESIS CALL TO DETERMINE WHETHER THE USER INPUT INVOKES A SCROLL
1002
SET THE HYSTERESIS VALUE FOR DETERMINING WHETHER A USER INPUT INVOKES A SCROLL
1004

FIG. 10

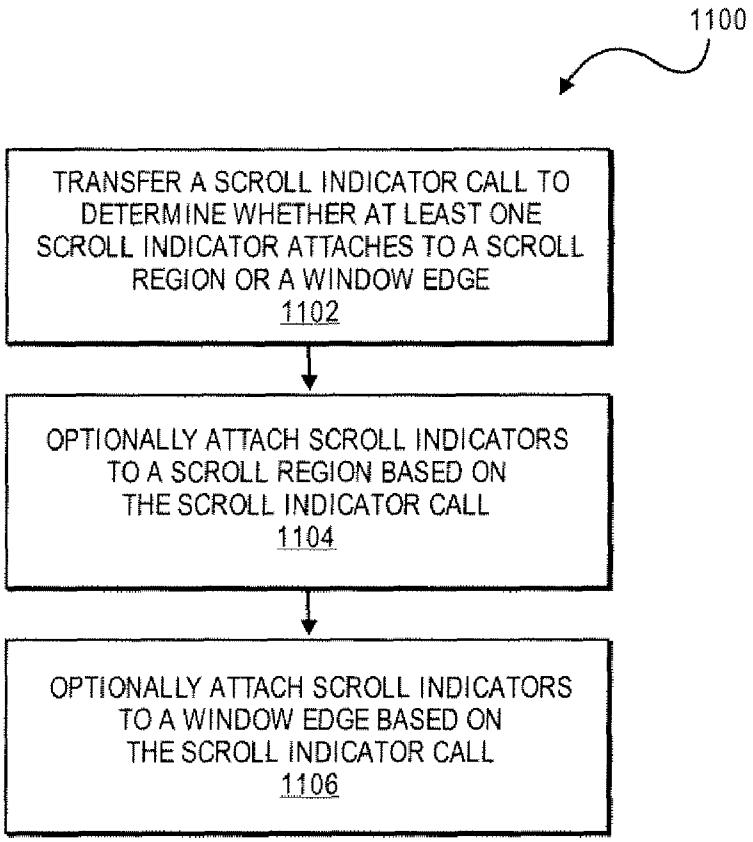

1100
TRANSFER A SCROLL INDICATOR CALL TO DETERMINE WHETHER AT LEAST ONE SCROLL INDICATOR ATTACHES TO A SCROLL REGION OR A WINDOW EDGE
1102
OPTIONALLY ATTACH SCROLL INDICATORS TO A SCROLL REGION BASED ON THE SCROLL INDICATOR CALL
1104
OPTIONALLY ATTACH SCROLL INDICATORS TO A WINDOW EDGE BASED ON THE SCROLL INDICATOR CALL
1106

FIG. 11

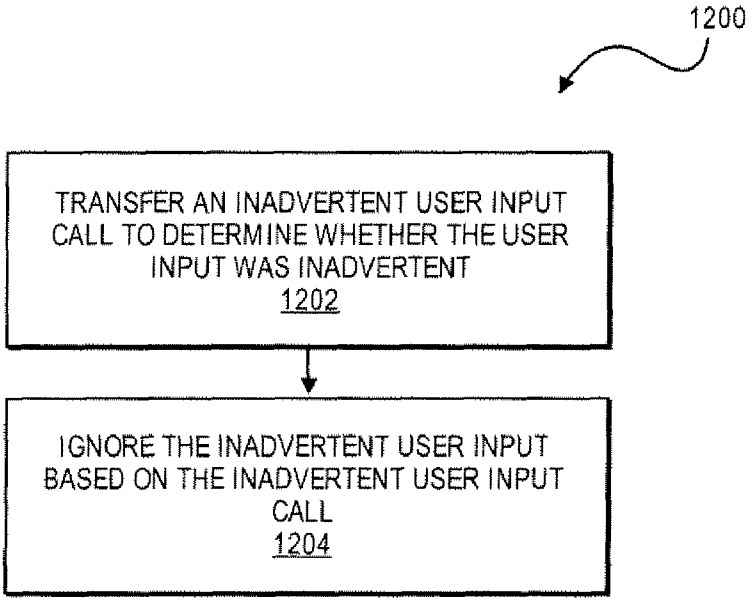

1200
TRANSFER AN INADVERTENT USER INPUT CALL TO DETERMINE WHETHER THE USER INPUT WAS INADVERTENT
1202
IGNORE THE INADVERTENT USER INPUT BASED ON THE INADVERTENT USER INPUT CALL
1204

FIG. 12

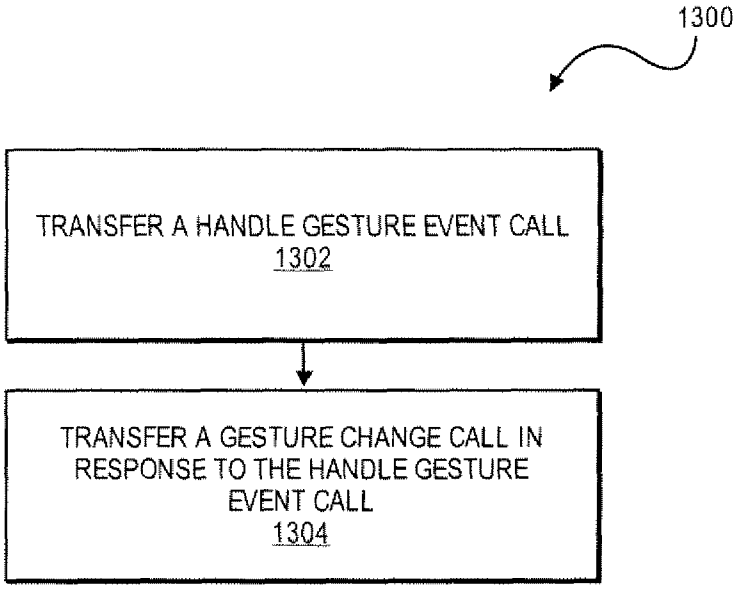

1300
TRANSFER A HANDLE GESTURE EVENT CALL
1302
TRANSFER A GESTURE CHANGE CALL IN RESPONSE TO THE HANDLE GESTURE EVENT CALL
1304

FIG. 13

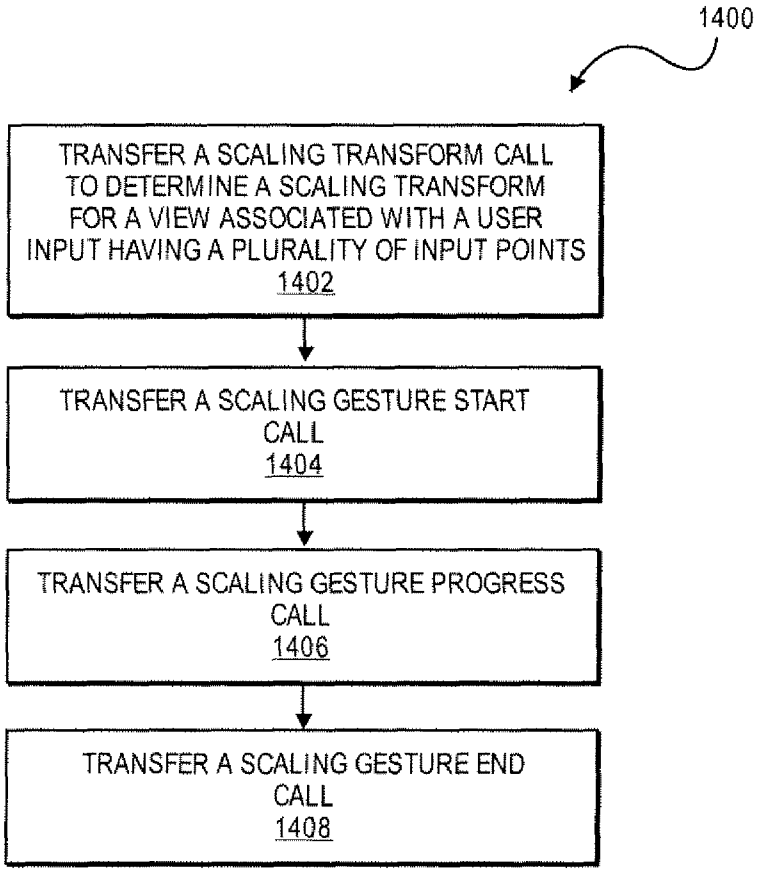

1400
TRANSFER A SCALING TRANSFORM CALL TO DETERMINE A SCALING TRANSFORM FOR A VIEW ASSOCIATED WITH A USER INPUT HAVING A PLURALITY OF INPUT POINTS
1402
TRANSFER A SCALING GESTURE START CALL
1404
TRANSFER A SCALING GESTURE PROGRESS CALL
1406
TRANSFER A SCALING GESTURE END CALL
1408

FIG. 14

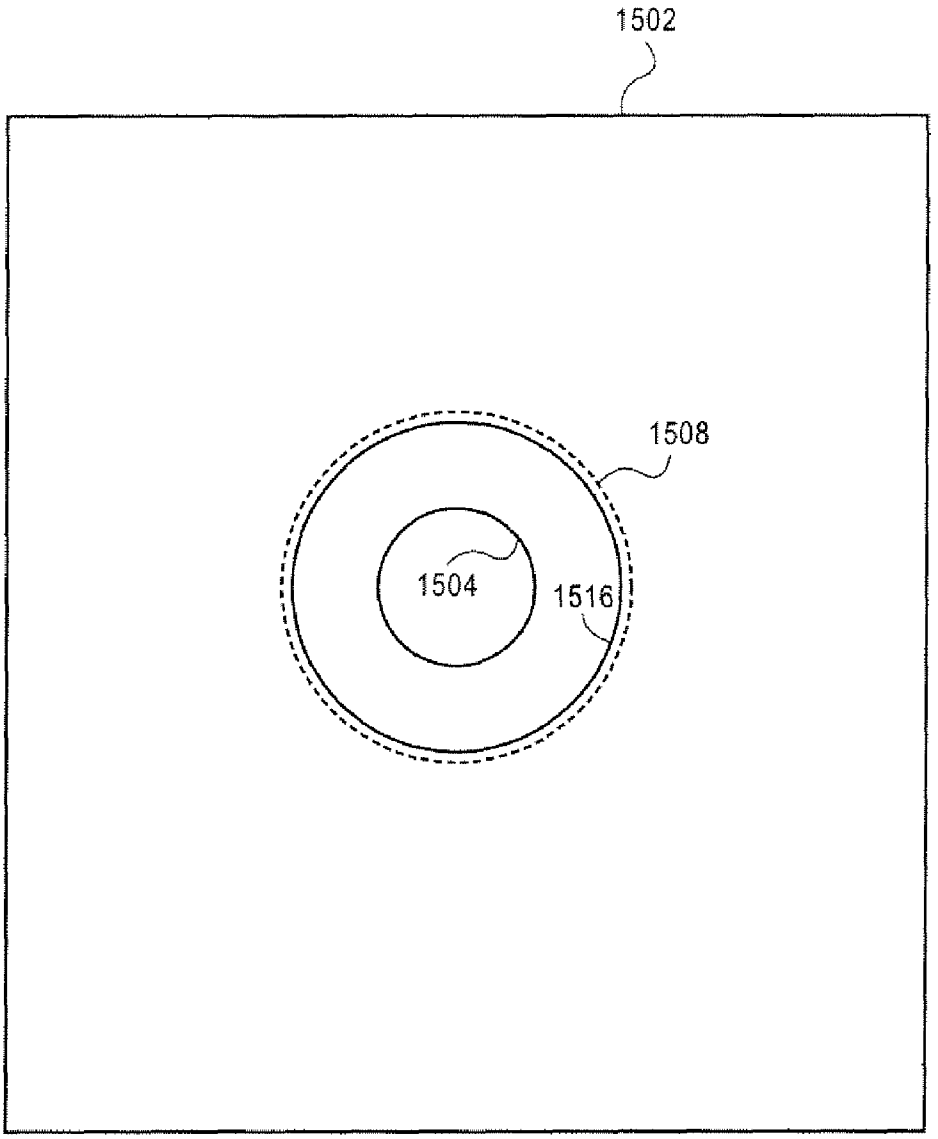


FIG. 15

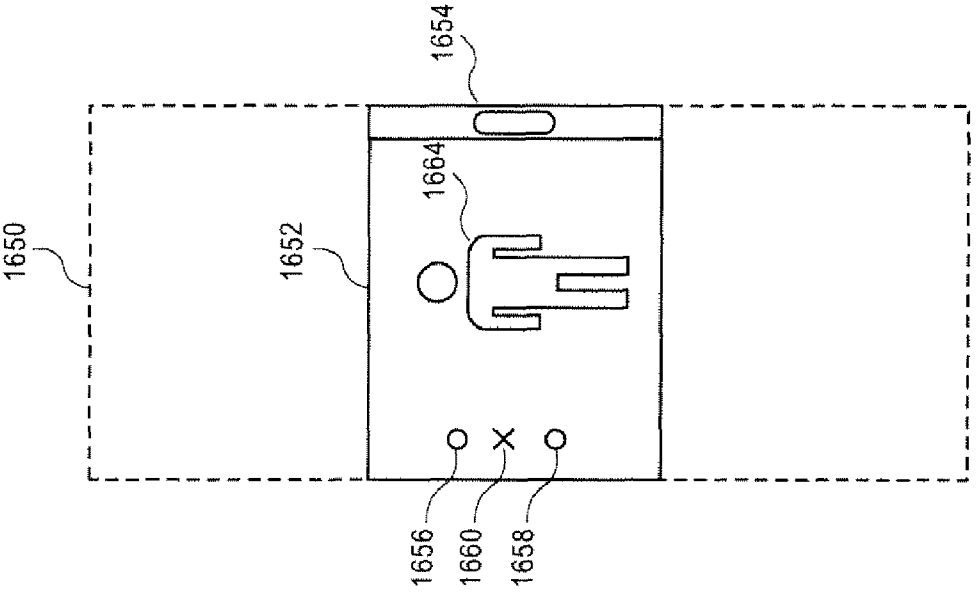


FIG. 16B

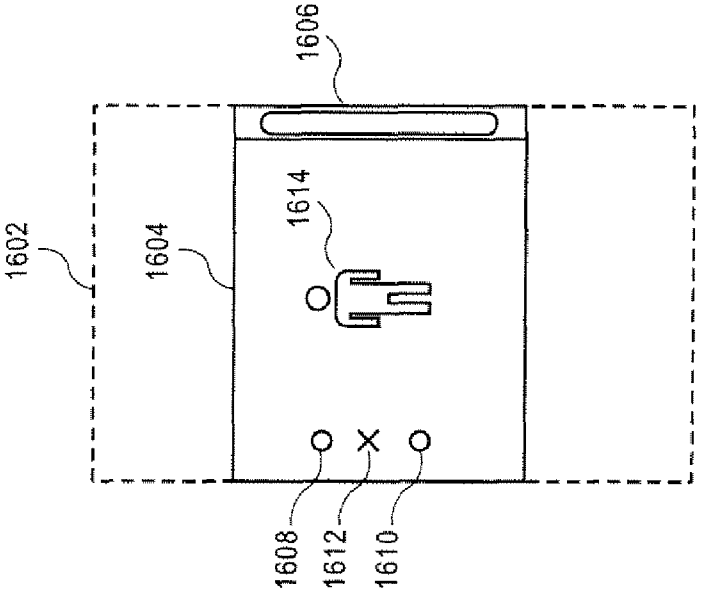


FIG. 16A

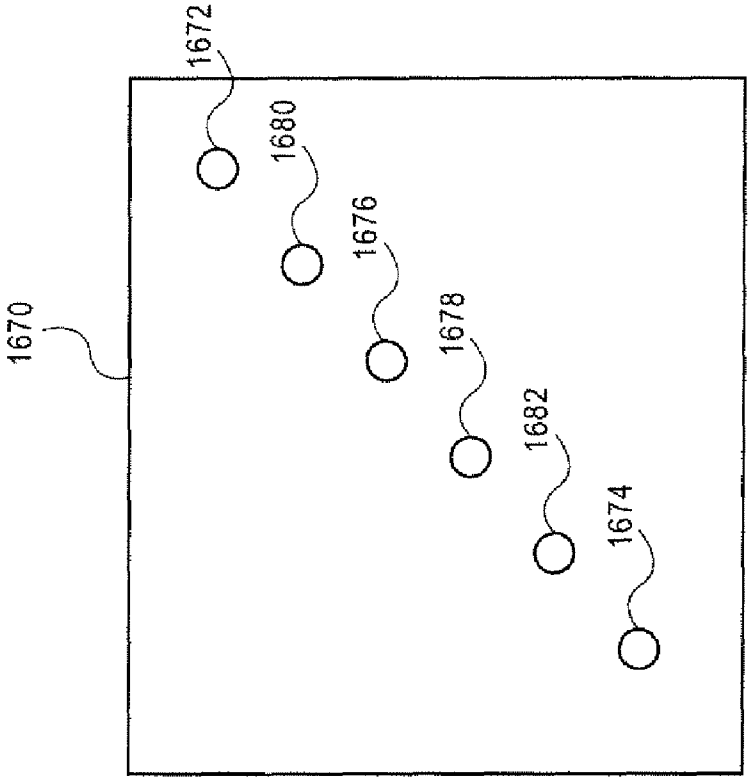


FIG. 16C

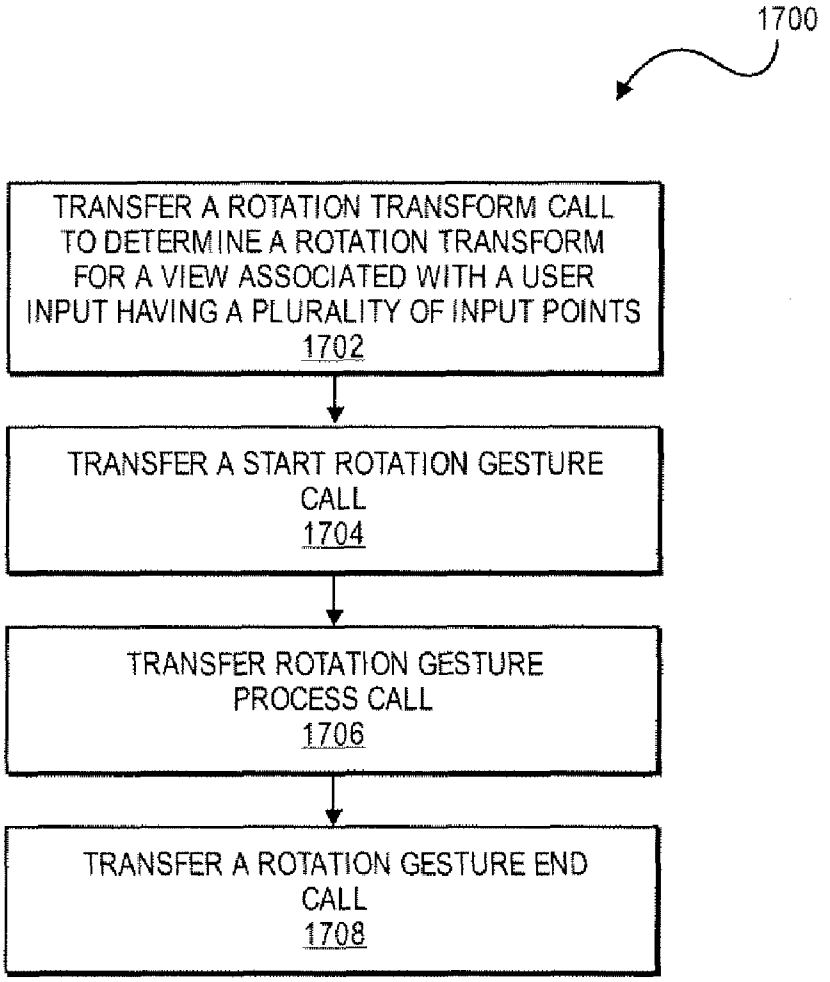

1700
TRANSFER A ROTATION TRANSFORM CALL TO DETERMINE A ROTATION TRANSFORM FOR A VIEW ASSOCIATED WITH A USER INPUT HAVING A PLURALITY OF INPUT POINTS
1702
TRANSFER A START ROTATION GESTURE CALL
1704
TRANSFER ROTATION GESTURE PROCESS CALL
1706
TRANSFER A ROTATION GESTURE END CALL
1708

FIG. 17

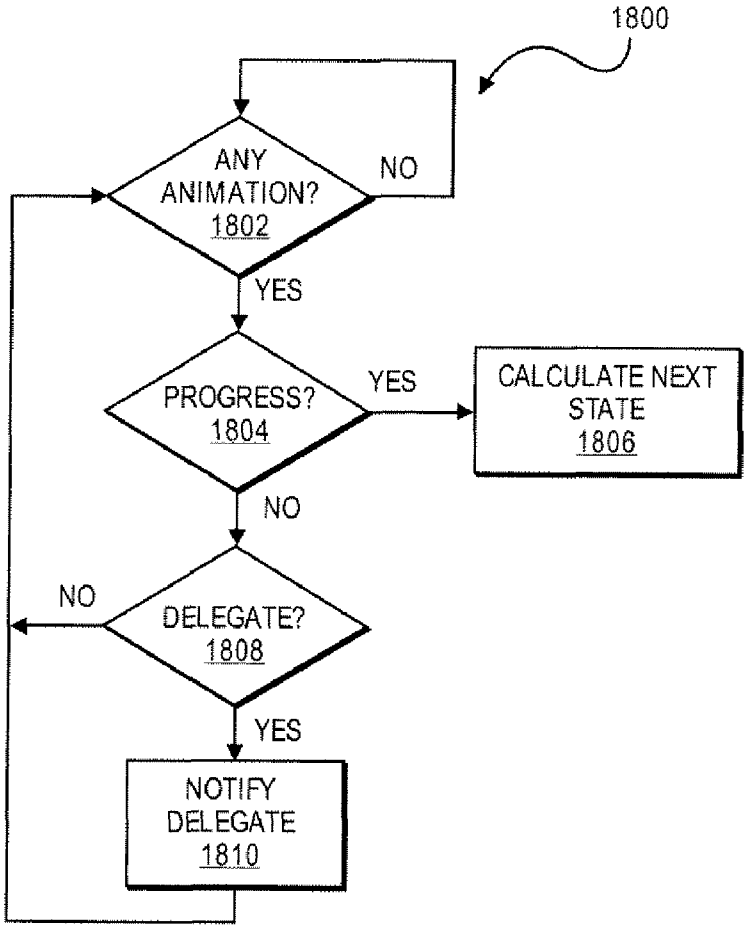

1800
ANY ANIMATION? 1802
NO
YES
PROGRESS? 1804
YES
CALCULATE NEXT STATE 1806
NO
DELEGATE? 1808
NO
YES
NOTIFY DELEGATE 1810

FIG. 18

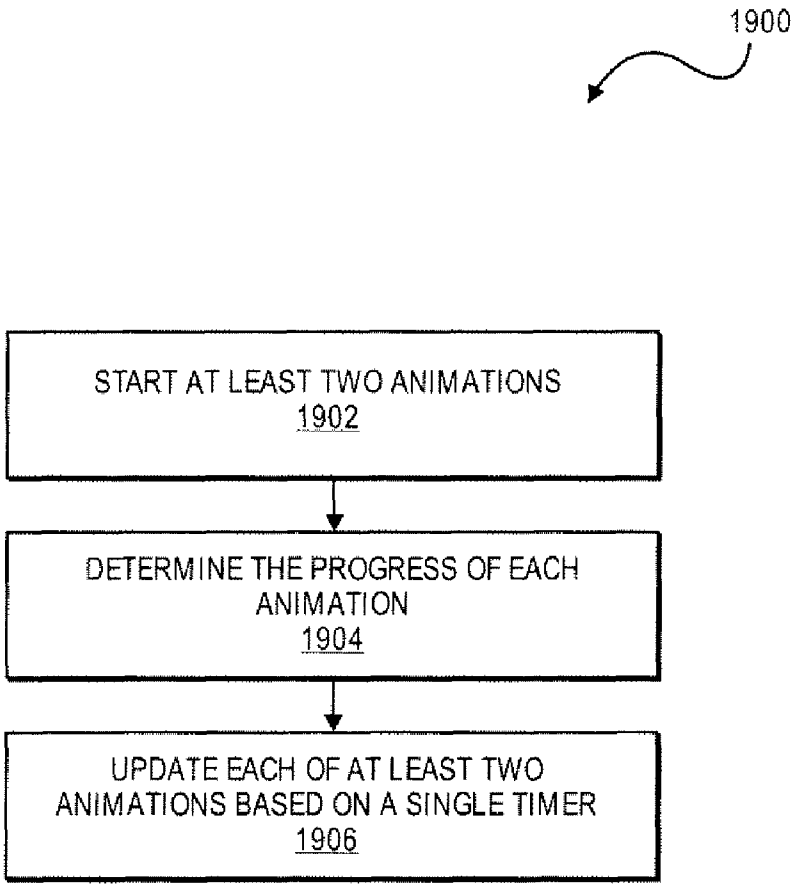

1900
START AT LEAST TWO ANIMATIONS
1902
DETERMINE THE PROGRESS OF EACH ANIMATION
1904
UPDATE EACH OF AT LEAST TWO ANIMATIONS BASED ON A SINGLE TIMER
1906

FIG. 19

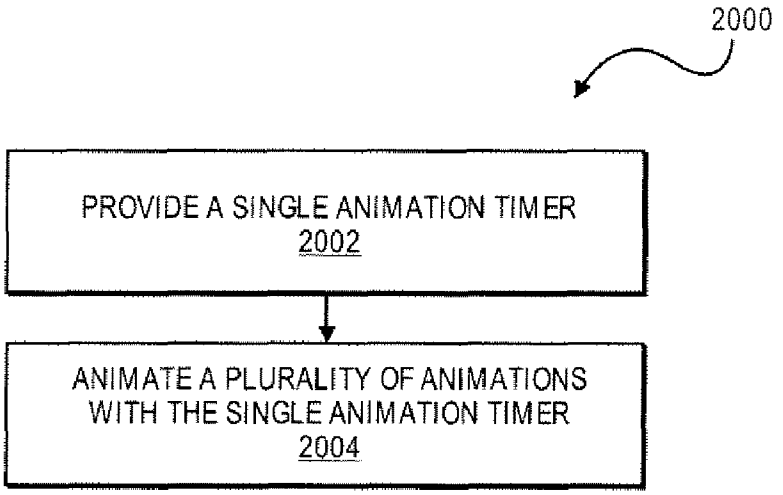

2000
PROVIDE A SINGLE ANIMATION TIMER
2002
ANIMATE A PLURALITY OF ANIMATIONS
WITH THE SINGLE ANIMATION TIMER
2004

FIG. 20

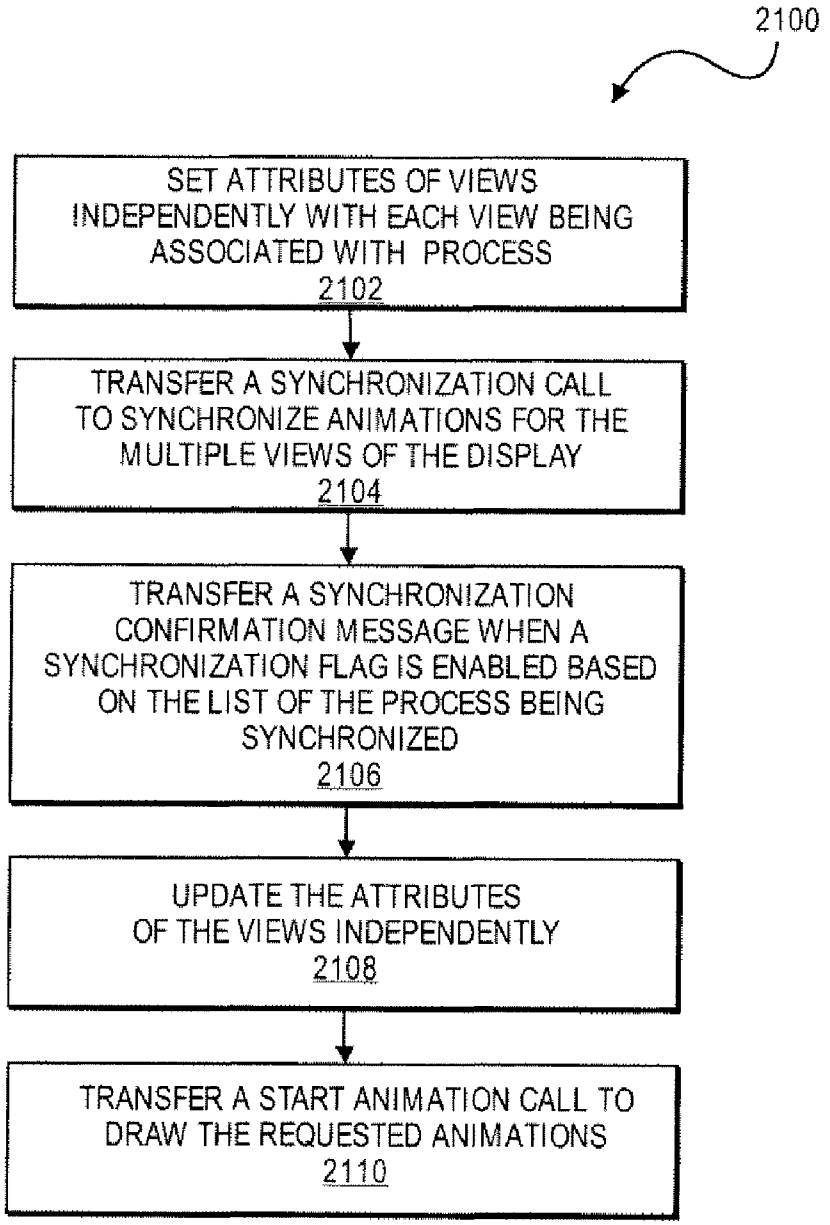

2100
SET ATTRIBUTES OF VIEWS INDEPENDENTLY WITH EACH VIEW BEING ASSOCIATED WITH PROCESS
2102
TRANSFER A SYNCHRONIZATION CALL TO SYNCHRONIZE ANIMATIONS FOR THE MULTIPLE VIEWS OF THE DISPLAY
2104
TRANSFER A SYNCHRONIZATION CONFIRMATION MESSAGE WHEN A SYNCHRONIZATION FLAG IS ENABLED BASED ON THE LIST OF THE PROCESS BEING SYNCHRONIZED
2106
UPDATE THE ATTRIBUTES OF THE VIEWS INDEPENDENTLY
2108
TRANSFER A START ANIMATION CALL TO DRAW THE REQUESTED ANIMATIONS
2110

FIG. 21

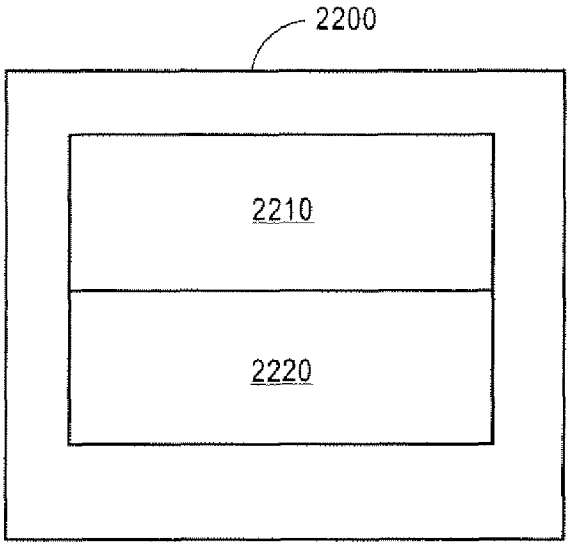


FIG. 22A

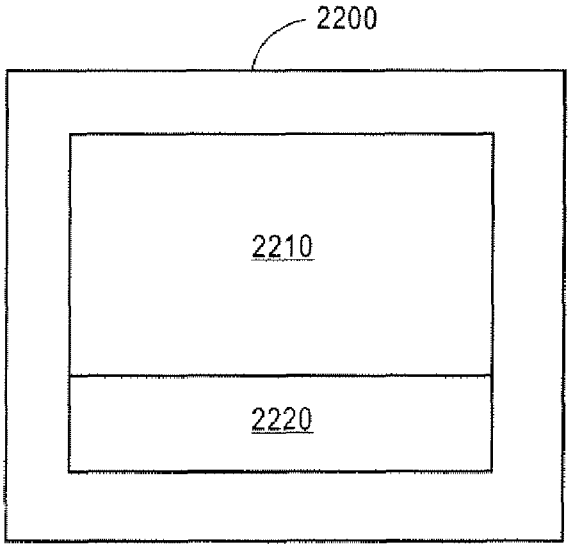


FIG. 22B

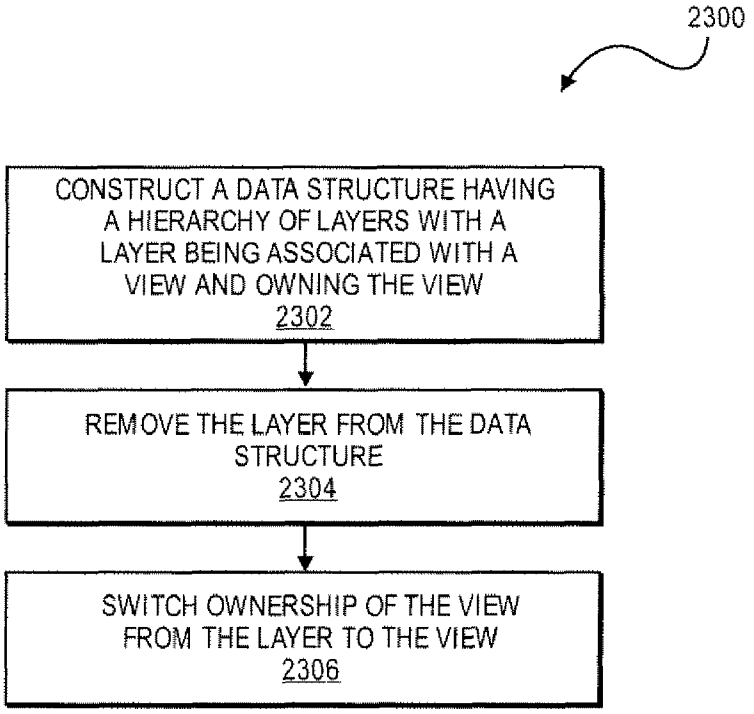

2300
CONSTRUCT A DATA STRUCTURE HAVING A HIERARCHY OF LAYERS WITH A LAYER BEING ASSOCIATED WITH A VIEW AND OWNING THE VIEW
2302
REMOVE THE LAYER FROM THE DATA STRUCTURE
2304
SWITCH OWNERSHIP OF THE VIEW FROM THE LAYER TO THE VIEW
2306

FIG. 23

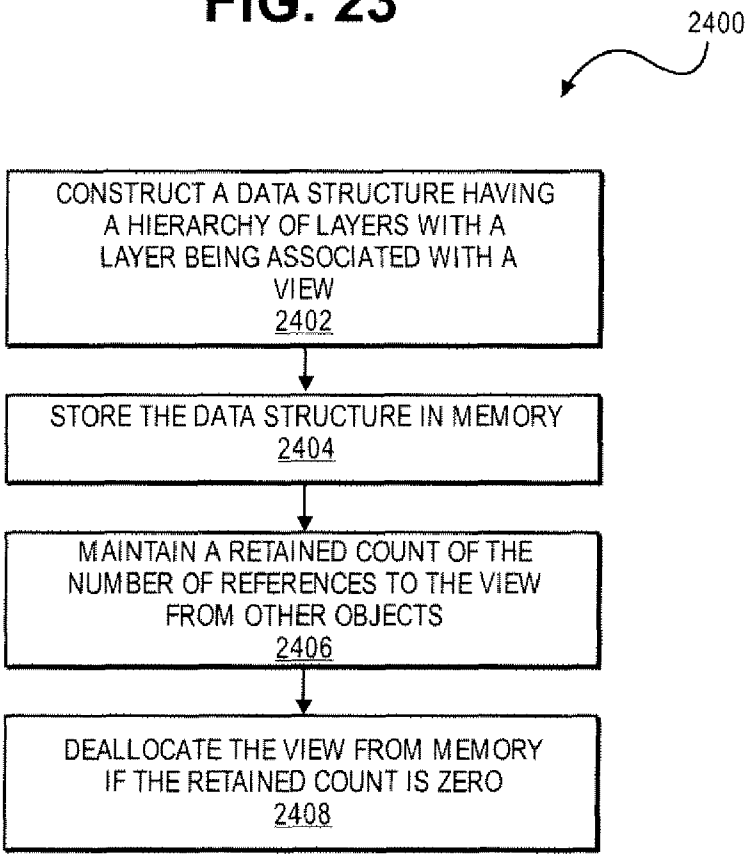

2400
CONSTRUCT A DATA STRUCTURE HAVING A HIERARCHY OF LAYERS WITH A LAYER BEING ASSOCIATED WITH A VIEW
2402
STORE THE DATA STRUCTURE IN MEMORY
2404
MAINTAIN A RETAINED COUNT OF THE NUMBER OF REFERENCES TO THE VIEW FROM OTHER OBJECTS
2406
DEALLOCATE THE VIEW FROM MEMORY IF THE RETAINED COUNT IS ZERO
2408

FIG. 24

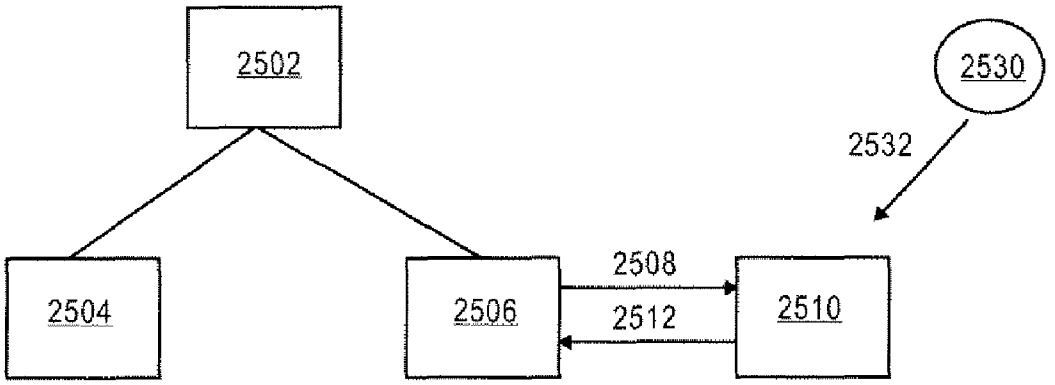


FIG. 25A

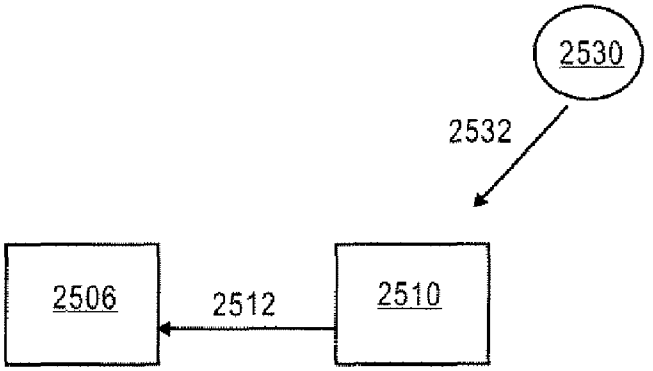


FIG. 25B

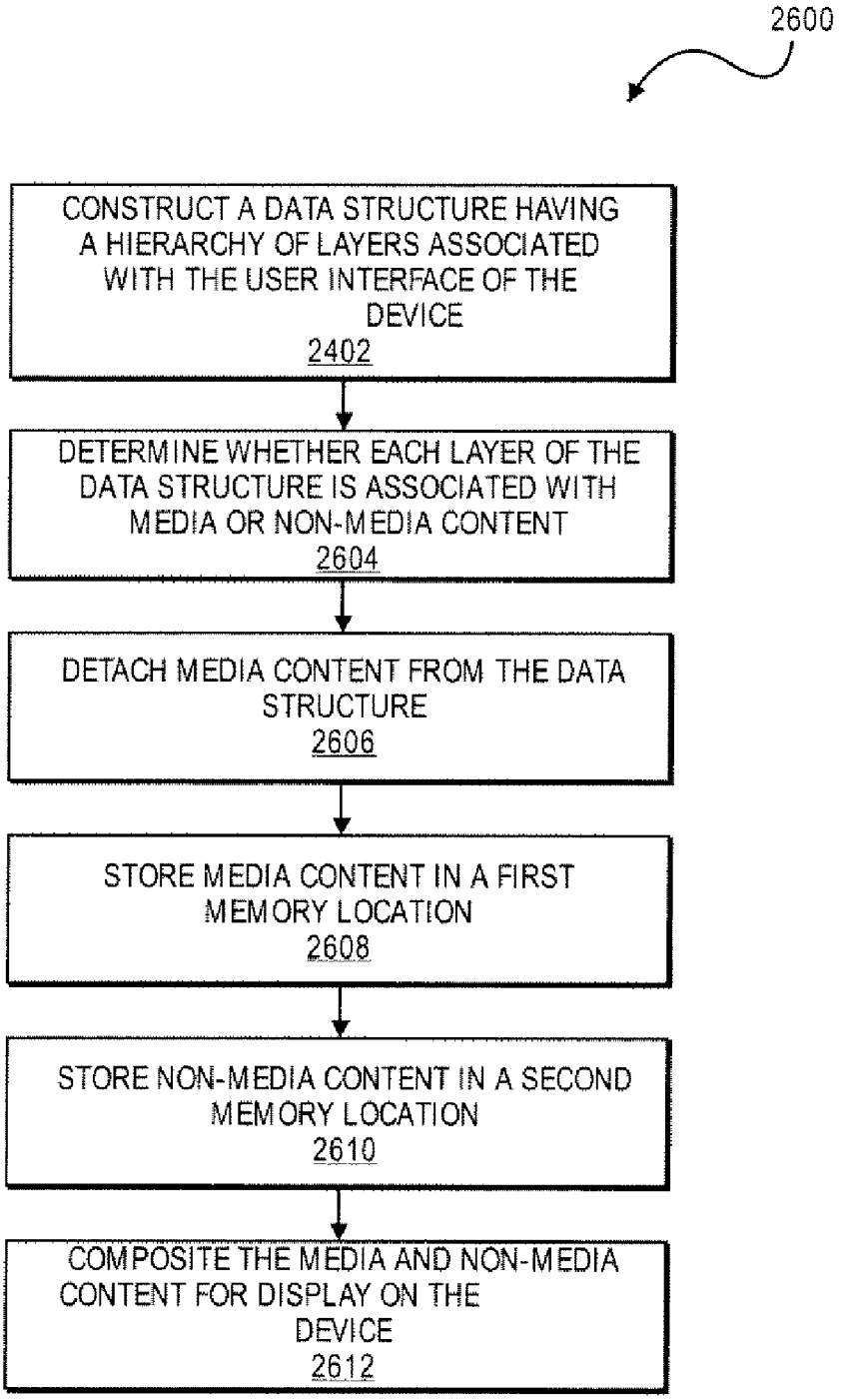

2600
CONSTRUCT A DATA STRUCTURE HAVING A HIERARCHY OF LAYERS ASSOCIATED WITH THE USER INTERFACE OF THE DEVICE
2402
DETERMINE WHETHER EACH LAYER OF THE DATA STRUCTURE IS ASSOCIATED WITH MEDIA OR NON-MEDIA CONTENT
2604
DETACH MEDIA CONTENT FROM THE DATA STRUCTURE
2606
STORE MEDIA CONTENT IN A FIRST MEMORY LOCATION
2608
STORE NON-MEDIA CONTENT IN A SECOND MEMORY LOCATION
2610
COMPOSITE THE MEDIA AND NON-MEDIA CONTENT FOR DISPLAY ON THE DEVICE
2612

FIG. 26

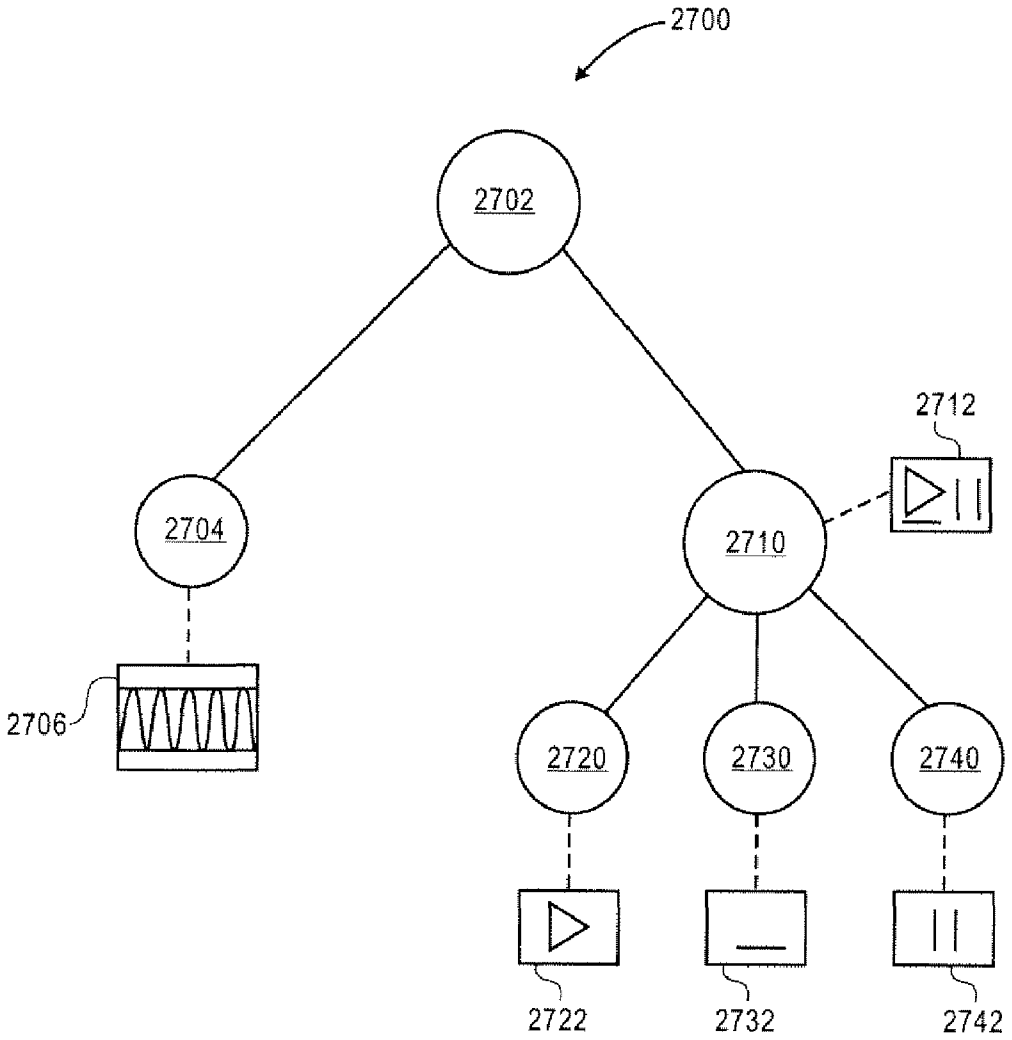


FIG. 27

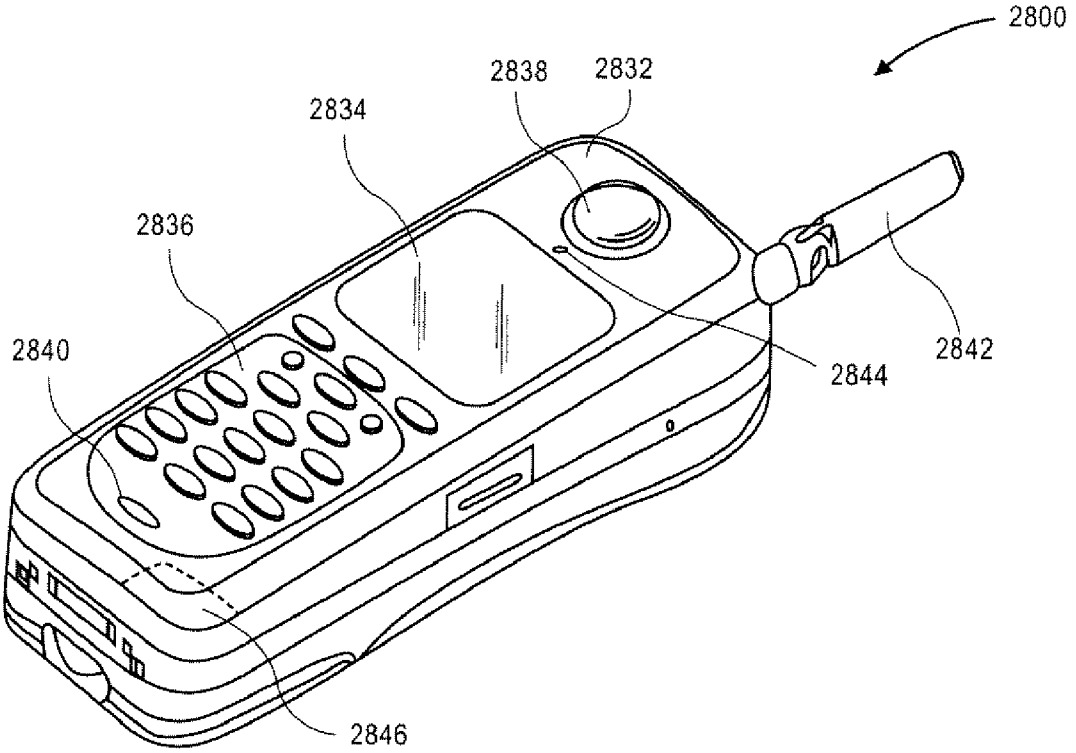


FIG. 28

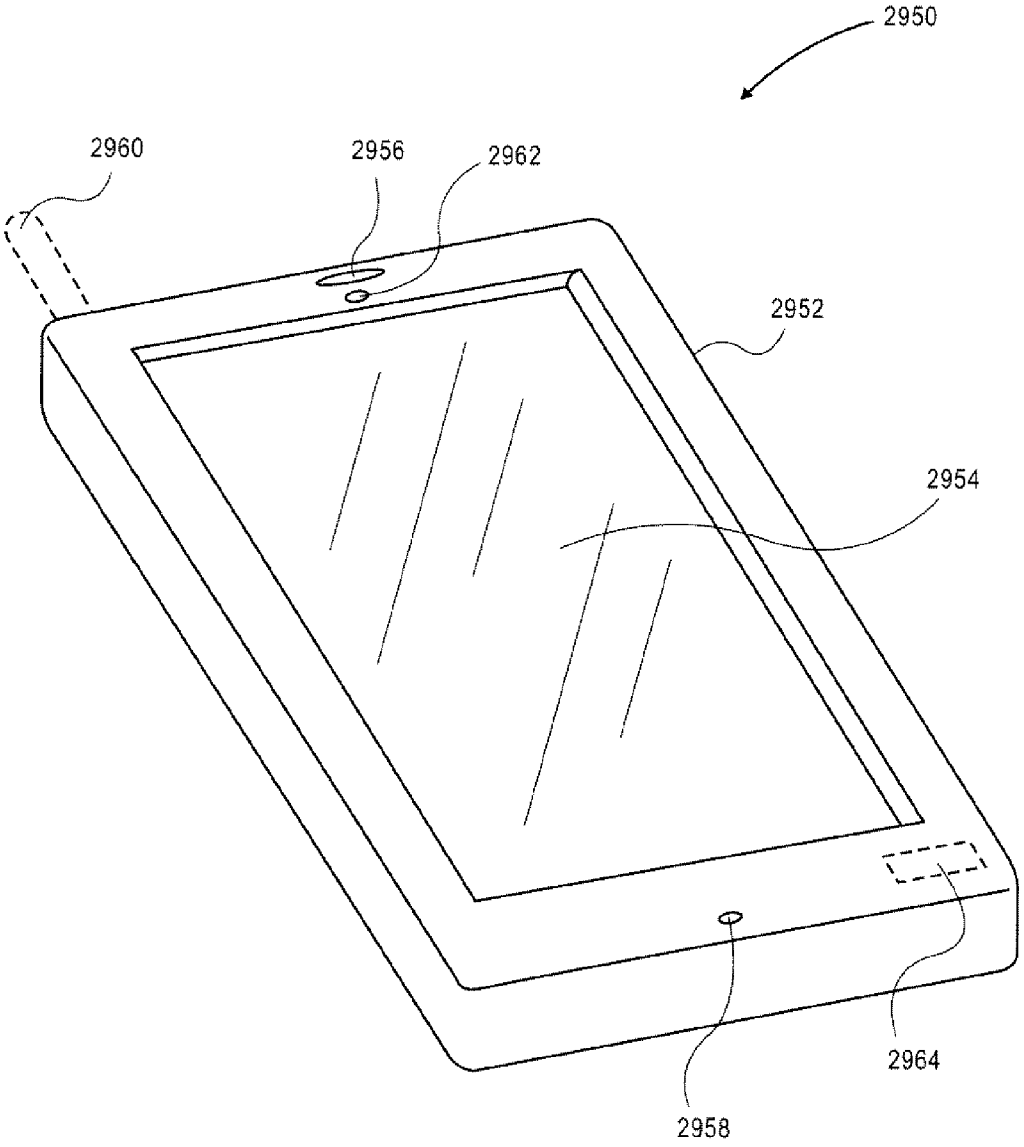


FIG. 29

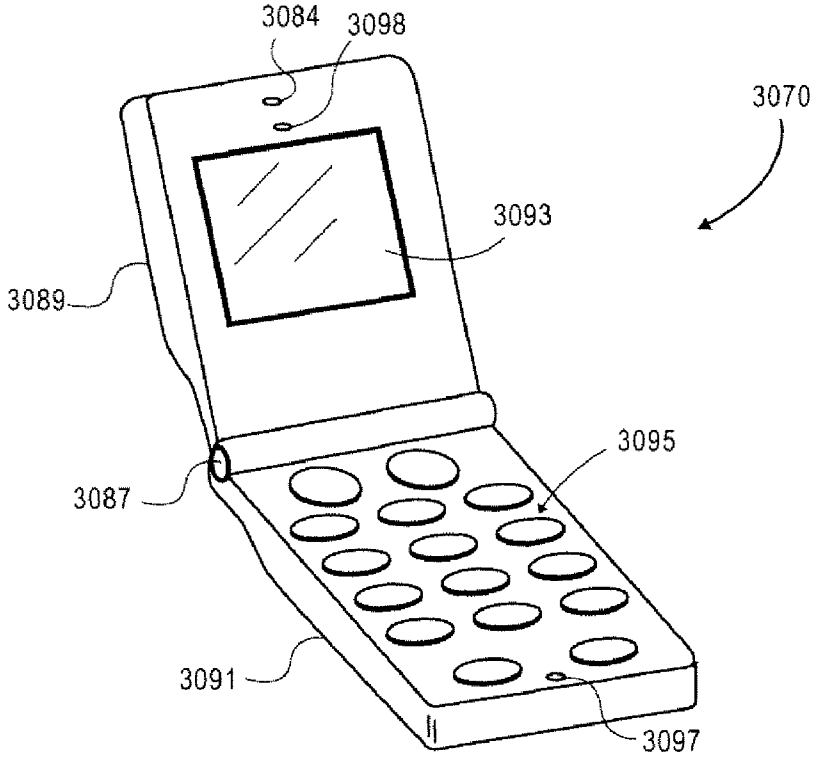


FIG. 30A

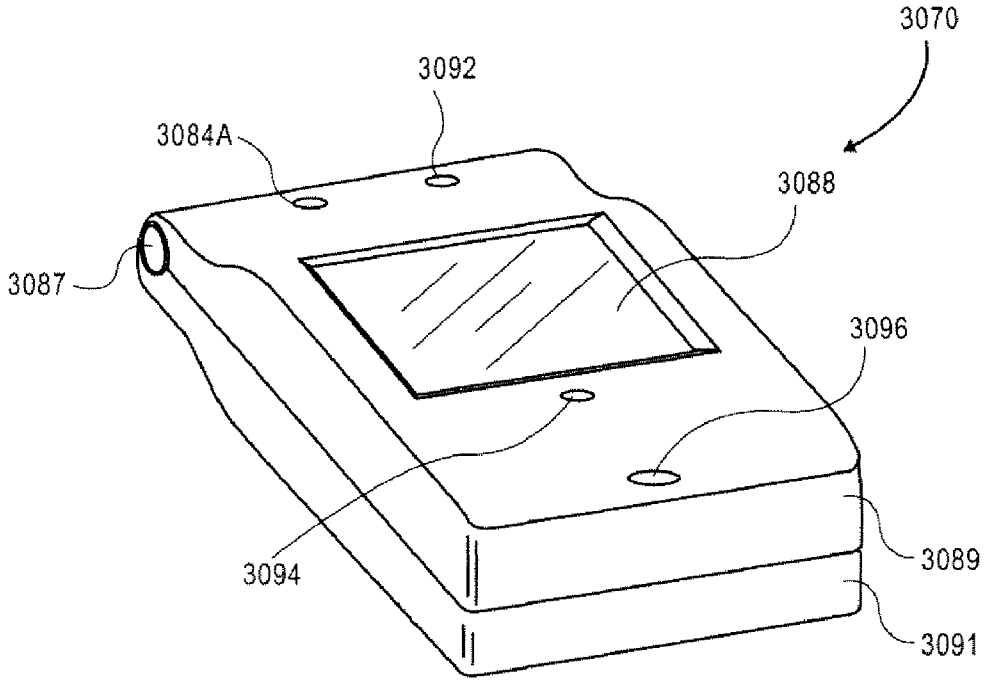


FIG. 30B

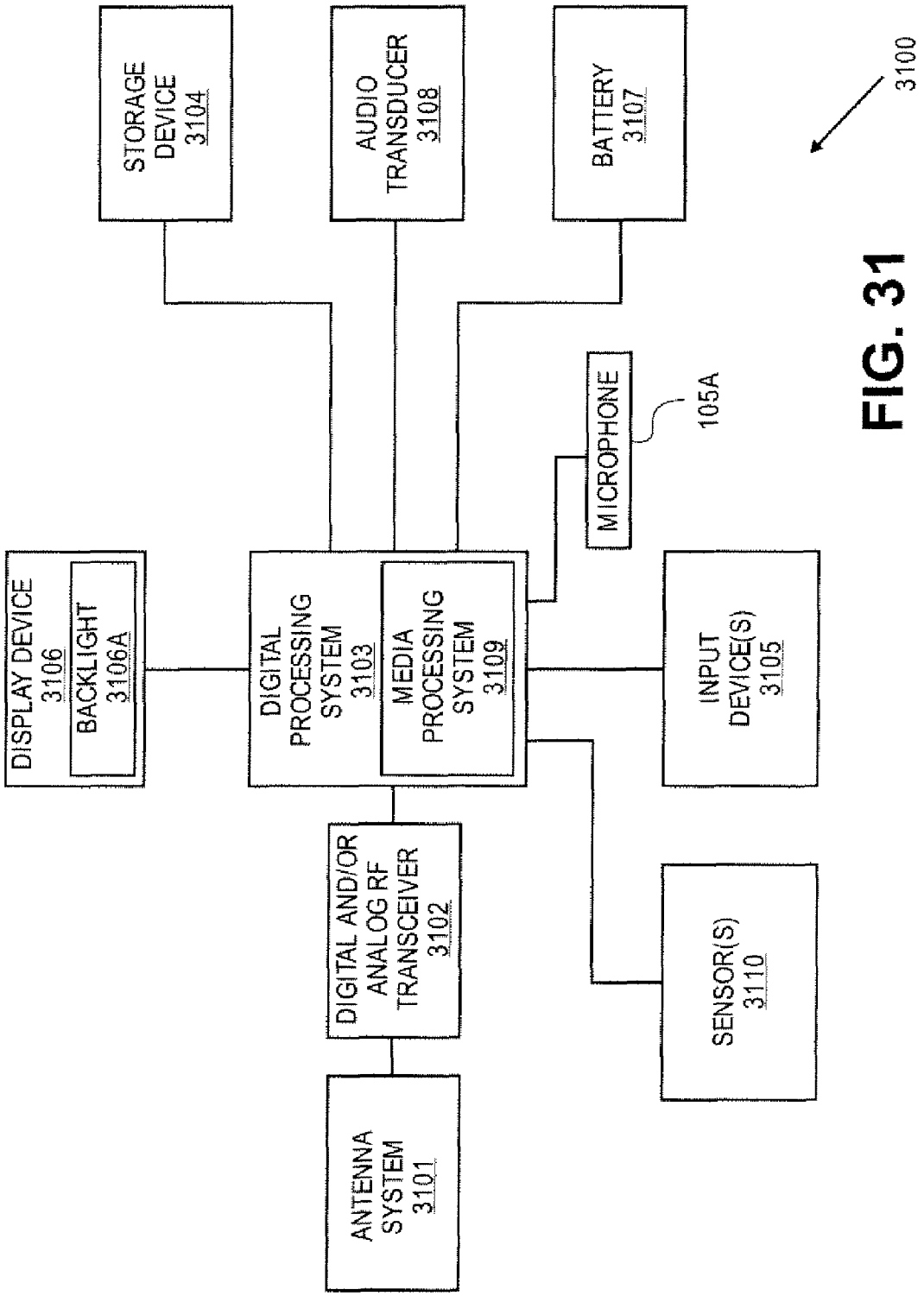

STORAGE DEVICE 3104
AUDIO TRANSDUCER 3108
BATTERY 3107
3100
MICROPHONE
105A
DISPLAY DEVICE 3106
BACKLIGHT 3106A
DIGITAL PROCESSING SYSTEM 3103
MEDIA PROCESSING SYSTEM 3109
INPUT DEVICE(S) 3105
DIGITAL AND/OR ANALOG RF TRANSCEIVER 3102
SENSOR(S) 3110
ANTENNA SYSTEM 3101

FIG. 31

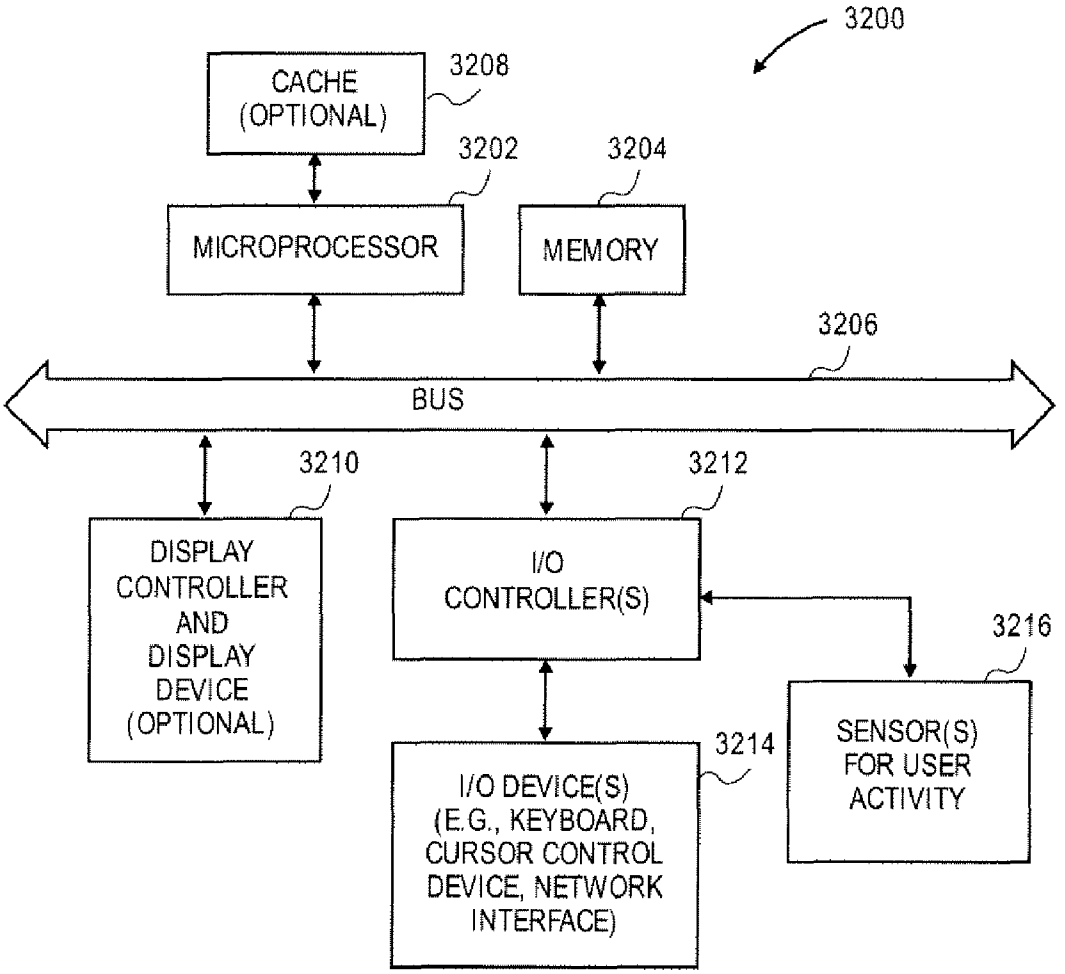

3200
CACHE (OPTIONAL)
3208
3202
3204
MICROPROCESSOR
MEMORY
3206
BUS
3210
3212
DISPLAY CONTROLLER AND DISPLAY DEVICE (OPTIONAL)
I/O CONTROLLER(S)
3216
SENSOR(S) FOR USER ACTIVITY
3214
I/O DEVICE(S) (E.G., KEYBOARD, CURSOR CONTROL DEVICE, NETWORK INTERFACE)

FIG. 32

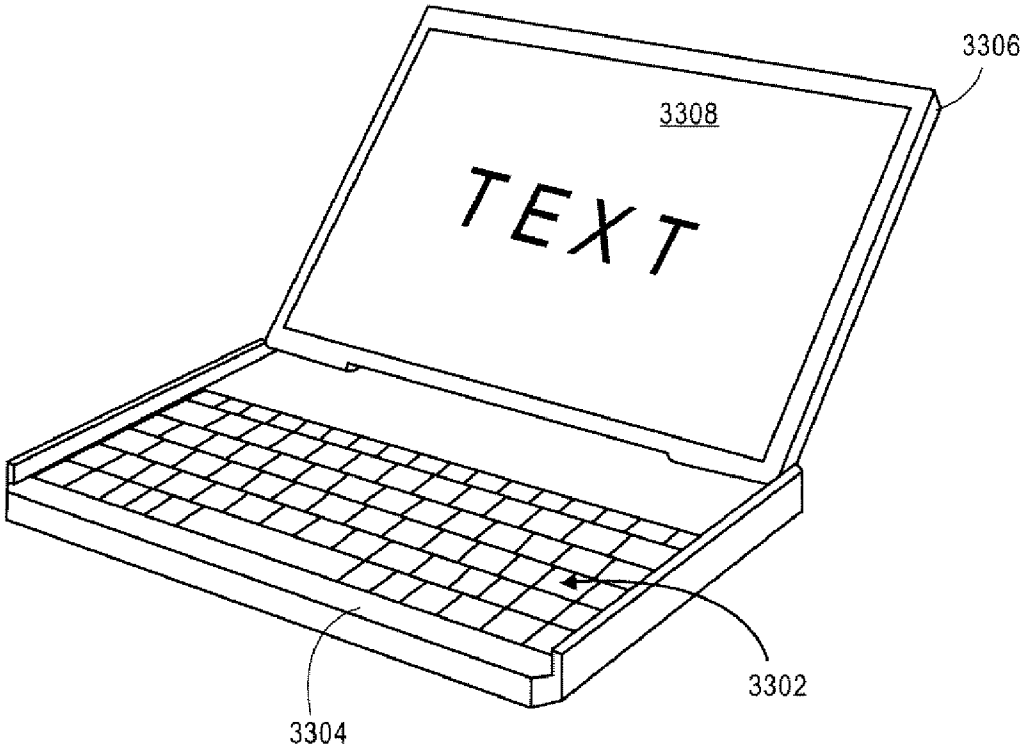


FIG. 33A

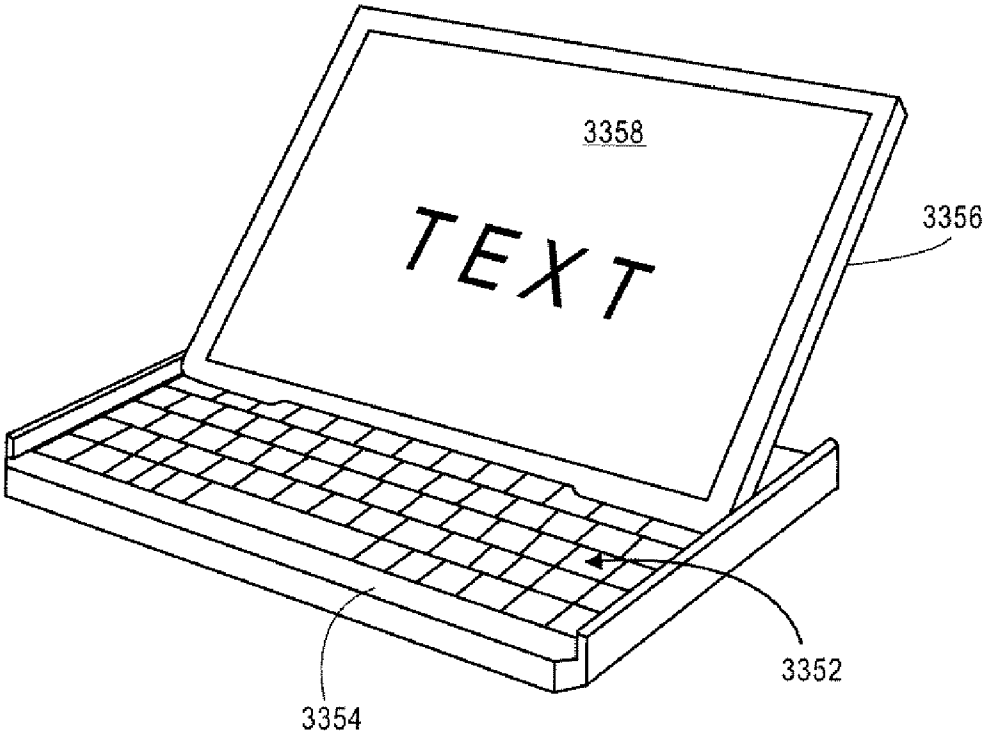


FIG. 33B

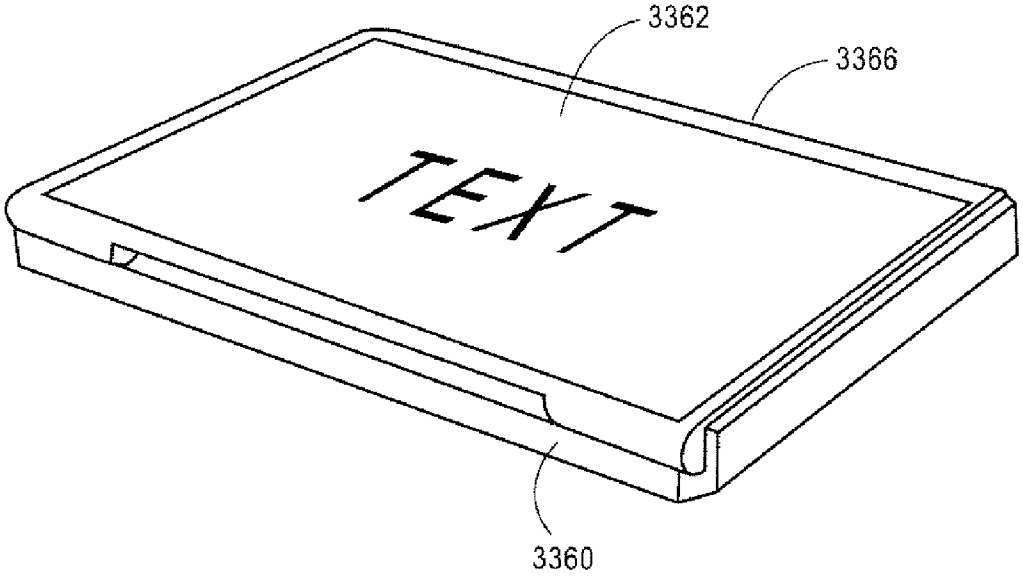


FIG. 33C

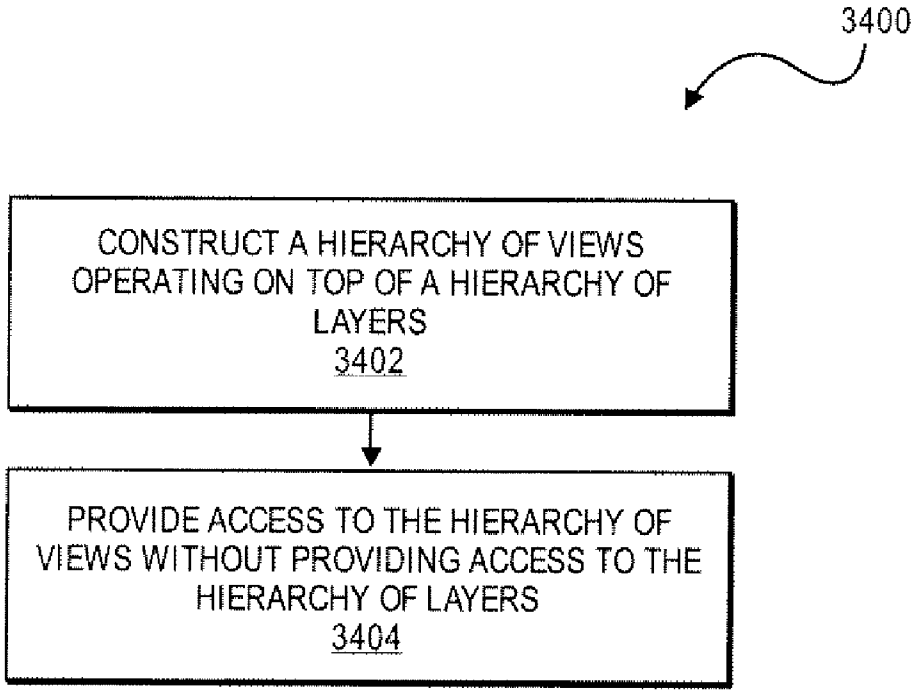

3400
CONSTRUCT A HIERARCHY OF VIEWS OPERATING ON TOP OF A HIERARCHY OF LAYERS
3402
PROVIDE ACCESS TO THE HIERARCHY OF VIEWS WITHOUT PROVIDING ACCESS TO THE HIERARCHY OF LAYERS
3404

FIG. 34

APPLICATION PROGRAMMING INTERFACES FOR SCROLLING OPERATIONS

FIELD OF THE DISCLOSURE

This disclosure relates to application programming interfaces that provide scrolling operations.

COMPUTER PROGRAM LISTING

Applicant has submitted herewith Computer Program Listings which are included as Appendix A, attached.

BACKGROUND OF THE DISCLOSURE

An API is a source code interface that a computer system or program library provides in order to support requests for services from a software application. An API is specified in terms of a programming language that can be interpretative or compiled when an application is built, rather than an explicit low level description of how data is laid out in memory. The software that provides the functionality described by an API is said to be an implementation of the API.

Various devices such as electronic devices, computing systems, portable devices, and handheld devices have software applications. The API interfaces between the software applications and user interface software to provide a user of the device with certain features and operations. A user may desire certain operations such as scrolling, selecting, gesturing, and animating operations for a display of the device.

Scrolling is the act of sliding a directional (e.g., horizontal or vertical) presentation of content, such as text, drawings, or images, across a screen or display window. In a typical graphical user interface, scrolling is done with the help of a scrollbar or using keyboard shortcuts often the arrow keys. Gesturing is a type of user input with two or more input points. Animating operations include changing content within a given time period.

The various types of devices may have a limited display size, user interface, software, API interface and/or processing capability which limits the ease of use of the devices. User interfaces of devices implement APIs in order to provide requested functionality and features. These user interfaces can have difficulty interpreting the various types of user inputs and providing the intended functionality associated with the user inputs.

SUMMARY OF THE DESCRIPTION

At least certain embodiments of the present disclosure include one or more application programming interfaces in an environment with user interface software interacting with a software application. Various function calls or messages are transferred via the application programming interfaces between the user interface software and software applications. Example application programming interfaces transfer function calls to implement scrolling, gesturing, and animating operations for a device.

At least certain embodiments of the present disclosure include an environment with user interface software interacting with a software application. A method for operating through an application programming interface (API) in this environment includes transferring a set bounce call. The method further includes setting at least one of maximum and minimum bounce values. The set bounce call causes a bounce of a scrolled region in an opposite direction of a scroll based on a region past an edge of the scrolled region being visible in a display region at the end of the scroll.

At least certain embodiments of the present disclosure include an environment with user interface software interacting with a software application. A method for operating through an application programming interface (API) in this environment includes transferring a rubberband call. Rubberbanding a scrolled region within a display region occurs by a predetermined maximum displacement when the scrolled region exceeds a display edge. The method further includes transferring an edge rubberband call to set displacement values for at least one edge of the display (e.g., top and bottom edges, left and right edges).

At least certain embodiments of the present disclosure include gesture operations for a display of a device. The gesture operations include performing a scaling transform such as a zoom in or zoom out in response to a user input having two or more input points. The gesture operations also include performing a rotation transform to rotate an image or view in response to a user input having two or more input points.

At least certain embodiments of the present disclosure include a method for performing animations for a display of a device. The method includes starting at least one animation. The method further includes determining the progress of each animation. The method further includes completing each animation based on a single timer. The single timer can be based on a redraw interval of the display hardware.

Various devices which perform one or more of the foregoing methods and machine readable media which, when executed by a processing system, cause the processing system to perform these methods, are also described.

Other methods, devices and machine readable media are also described.

BRIEF DESCRIPTION OF THE DRAWINGS

The disclosure is described by way of example with reference to the accompanying drawings, wherein:

FIG. **1** is flow chart of a method for responding to a user input of a data processing device;

FIG. **2** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **3** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **4** is a schematic diagram illustrating an embodiment of user interface of a portable electronic device **400** having a touch-sensitive display **408**;

FIG. **5**A-**5**C illustrate at least some embodiments of user interface of a portable electronic device **400** having a touch-sensitive display;

FIG. **6**A-**6**D illustrate the scrolling of a list of items to a terminus of the list, at which point an area beyond the terminus is displayed and the list is then scrolled in an opposite direction until the area beyond the terminus is no longer displayed, in accordance with some embodiments;

FIG. **7** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **8** illustrates first and second scroll angles for locking a scroll of a display of a device in a horizontal or vertical direction according to certain teachings of the present disclosure;

FIG. **9** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **10** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **11** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **12** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **13** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **14** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **15** illustrates a display of a device having a scaling transform of a view:

FIGS. **16**A and **16**B illustrate a display of a device with a view having a first and a second scaling factor;

FIG. **16**C illustrates changing a view from a scale factor of 2× to a scale factor of 1× in at least some embodiments of the present disclosure;

FIG. **17** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **18** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIG. **19** is flow chart of a method for animating views displayed on a display of a device;

FIG. **20** is flow chart of a method for animating views displayed on a display of a device;

FIG. **21** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure;

FIGS. **22**A and **22**B illustrate synchronizing the resizing of windows of a display of a device;

FIG. **23** illustrates a method for switching ownership of a view of an application displayed on a display of a data processing device;

FIG. **24** illustrates a method for memory management of a view of an application displayed on a display of a device;

FIGS. **25**A and **258** illustrate a data structure having a hierarchy of layers with a layer being associated with a view;

FIG. **26** illustrates a method for compositing media and non-media content of user interface for display on a device;

FIG. **27** illustrates a data structure or layer tree having a hierarchy of layers;

FIG. **28** is a perspective view of a device in accordance with one embodiment of the present disclosure;

FIG. **29** is a perspective view of a device in accordance with one embodiment of the present disclosure;

FIGS. **30**A and **30**B illustrate a device **3070** according to one embodiment of the disclosure;

FIG. **31** is a block diagram of a system in which embodiments of the present disclosure can be implemented;

FIG. **32** shows another example of a device in accordance with one embodiment of the present disclosure;

FIG. **33**A is a perspective view of a device in a first configuration (e.g. in a laptop configuration) in accordance with one embodiment of the present disclosure;

FIG. **33**B is a perspective view of the device of FIG. **33**A in a second configuration (e.g. a transition configuration) in accordance with one embodiment of the present disclosure;

FIG. **33**C is a perspective view of the device of FIG. **33**A in a third configuration (e.g., a tablet configuration) in accordance with one embodiment of the present disclosure; and

FIG. **34** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure.

DETAILED DESCRIPTION

Various embodiments and aspects of the disclosure will be described with reference to details discussed below, and the accompanying drawings will illustrate the various embodiments. The following description and drawings are illustrative of the disclosure and are not to be construed as limiting the disclosure. Numerous specific details are described to provide a through understanding of various embodiments of the present disclosure. However, in certain instances, well-known or conventional details are not described in order to provide a concise discussion of embodiments of the present disclosure.

Some portions of the detailed descriptions which follow are presented in terms of algorithms which include operations on data stored within a computer memory. An algorithm is generally a self-consistent sequence of operations leading to a desired result. The operations typically require or involve physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like.

It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the following discussion, it is appreciated that throughout the description, discussions utilizing terms such as "processing" or "computing" or "calculating" or "determining" or "displaying" or the like, can refer to the action and processes of a data processing system, or similar electronic device, that manipulates and transforms data represented as physical (electronic) quantities within the system's registers and memories into other data similarly represented as physical quantities within the system's memories or registers or other such information storage, transmission or display devices.

The present disclosure can relate to an apparatus for performing one or more of the operations described herein. This apparatus may be specially constructed for the required purposes, or it may comprise a general purpose computer selectively activated or reconfigured by a computer program stored in the computer. Such a computer program may be stored in a machine (e.g. computer) readable storage medium, such as, but is not limited to, any type of disk including floppy disks, optical disks, CD-ROMs, and magnetic-optical disks, read-only memories (ROMs), random access memories (RAMs), erasable programmable ROMs (EPROMs), electrically erasable programmable ROMs (EEPROMs), flash memory, mag-

netic or optical cards, or any type of media suitable for storing electronic instructions, and each coupled to a bus.

A machine-readable medium includes any mechanism for storing information in a form readable by a machine (e.g., a computer). For example, a machine-readable medium includes read only memory ("ROM"); random access memory ("RAM"); magnetic disk storage media; optical storage media; flash memory devices; etc.

At least certain embodiments of the present disclosure include one or application programming interfaces in an environment with user interface software interacting with a software application. Various function calls or messages are transferred via the application programming interfaces between the user interface software and software applications. Transferring the function calls or messages may include issuing, initiating, invoking or receiving the function calls or messages. Example application programming interfaces transfer function calls to implement scrolling, gesturing, and animating operations for a device having a display region. An API may also implement functions having parameters, variables, or pointers. An API may receive parameters as disclosed or other combinations of parameters. In addition to the APIs disclosed, other APIs individually or in combination can perform similar functionality as the disclosed APIs.

The display region is a form of a window. A window is a display region which may not have a border and may be the entire display region or area of a display. In some embodiments, a display region may have at least one window and/or at least one view (e.g., web, text, or image content). A window may have at least one view. The methods, systems, and apparatuses disclosed can be implemented with display regions, windows, and/or views.

At least certain embodiments of the present disclosure include scrolling operations for scrolling a display of a device. The scrolling operations include bouncing a scrolled region in an opposite direction of a scroll when a scroll completes, rubberbanding a scrolled region by a predetermined maximum displacement when the scrolled region exceeds a display edge, and setting a scrolling angle that locks the scroll in a horizontal or vertical direction.

At least certain embodiments of the present disclosure include gesture operations for a display of a device. The gesture operations include performing a scaling transform such as a zoom in or zoom out in response to a user input having two or more input points. The gesture operations also include performing a rotation transform to rotate an image or view in response to a user input having two or more input points.

At least certain embodiments of the present disclosure include a method for performing animations for a display of a device. The method includes starting at least one animation. The method further includes determining the progress of each animation. The method further includes completing each animation based on a single timer. The single timer can be based on a redraw interval of the display hardware.

At least certain embodiments of the disclosure may be part of a digital media player, such as a portable music and/or video media player, which may include a media processing system to present the media, a storage device to store the media and may further include a radio frequency (RF) transceiver (e.g., an RF transceiver for a cellular telephone) coupled with an antenna system and the media processing system. In certain embodiments, media stored on a remote storage device may be transmitted to the media player through the RF transceiver. The media may be, for example, one or more of music or other audio, still pictures, or motion pictures.

The portable media player may include a media selection device, such as a click wheel input device on an iPod® or iPod Nano® media player from Apple Computer, Inc. of Cupertino, Calif., a touch screen input device, pushbutton device, movable pointing input device or other input device. The media selection device may be used to select the media stored on the storage device and/or the remote storage device. The portable media player may, in at least certain embodiments, include a display device which is coupled to the media processing system to display titles or other indicators of media being selected through the input device and being presented, either through a speaker or earphone(s), or on the display device, or on both display device and a speaker or earphone(s). In some embodiments, the display device and input device are integrated while in other embodiments the display device and input device are separate devices. Examples of a portable media player are described in published U.S. patent application Nos. 2003/0095096 and 2004/0224638, both of which are incorporated by reference.

Embodiments of the disclosure described herein may be part of other types of data processing systems, such as, for example, entertainment systems or personal digital assistants (PDAs), or general purpose computer systems, or special purpose computer systems, or an embedded device within another device, or cellular telephones which do not include media players, or multi touch tablet devices, or other multi touch devices, or devices which combine aspects or functions of these devices (e.g., a media player, such as an iPod®, combined with a PDA, an entertainment system, and a cellular telephone in one device). In this disclosure, electronic devices and consumer devices are types of devices.

FIG. **1** is flow chart of a method for responding to a user input of a device. The method **100** includes receiving a user input at block **102**. The user input may be in the form of an input key, button, wheel, touch, or other means for interacting with the device. The method **100** further includes creating an event object in response to the user input at block **104**. The method **100** further includes determining whether the event object invokes a scroll or gesture operation at block **106**. For example, a single touch that drags a distance across a display of the device may be interpreted as a scroll operation. In one embodiment, a two or more finger touch of the display may be interpreted as a gesture operation. In certain embodiments, determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period. The method **100** further includes issuing at least one scroll or gesture call based on invoking the scroll or gesture operation at block **108**. The method **100** further includes responding to at least one scroll call, if issued, by scrolling a window having a view (e.g., web, text, or image content) associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input at block **110**. For example, an input may end at a certain position on a display of the device. The scrolling may continue until reaching a predetermined position in relation to the last input received from the user. The method **100** further includes responding to at least one gesture call, if issued, by changing a view associated with the event object based on receiving a plurality of input points in the form of the user input at block **112**.

In certain embodiments of the present disclosure scroll operations include attaching scroll indicators to a content edge of a display. Alternatively, the scroll indicators can be attached to the display edge. In some embodiments, user input in the form of a mouse/finger down causes the scroll indicators to be displayed on the display edge, content edge, or window edge of the scrolled region. If a mouse/finger up is

then detected, the scroll indicators are faded out from the display region, content edge, or window edge of the scrolled region.

In certain embodiments of the present disclosure, gesture operations include responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input. Gesture operations may also include scaling a view associated with the event object by zooming in or zooming out based on receiving the user input.

In some embodiments, a device includes a display region having multiple views or windows. Each window may have a multiple views including superviews and subviews. It is necessary to determine which window, view, superview, or subview is contacted by a user input in the form of a mouse up, mouse down, or drag, etc. An API can set various modes for making this determination. In one embodiment, a pass mode sends mouse down, mouse up, and drag inputs to the nearest subview. In another embodiment, an intercept on drag mode sends a drag input to the superview while mouse up and down inputs are sent to the subview. In another embodiment, an intercept mode sends all drag, mouse up and down inputs to the superview. The superview may be scroller software operating as a subclass of a view software. The subview may be view software operating as a subclass of the user interface software.

FIG. **2** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a bounce operation. The method **200** for providing a bounce operation includes transferring a set bounce call at block **202**. The method **200** further includes setting at least one of maximum and minimum bounce values at block **204**. The minimum and maximum bounce values may be associated with at least one edge of a window that has received a user input. The method **200** further includes causing a bounce of a scrolled region in an opposite direction of a scroll based on a region past the scrolled region being visible in a display region at the end of the scroll at block **206**. The scrolled region may be a content region.

In certain embodiments of the present disclosure, transferring the set bounce call is either one of issuing, initiating, invoking or receiving the set bounce call.

FIG. **3** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a rubberband operation. The method **300** for providing a rubberband operation includes transferring a rubberband call to cause rubberbanding a scrolled region displayed within a display at block **302**. The method **300** further includes transferring an edge rubberband call to set displacement values for at least one edge of the display at block **304**. In some embodiments, the displacement values are set for top and bottom edges, left and right edges, or all edges.

Rubberbanding a scrolled region according to the method **300** occurs by a predetermined maximum displacement value when the scrolled region exceeds a display edge of a display of a device based on the scroll. If a user scrolls content of the display making a region past the edge of the content visible in the display, then the displacement value limits the maximum amount for the region outside the content. At the end of the scroll, the content slides back making the region outside of the content no longer visible on the display.

In certain embodiments of the present disclosure, transferring the rubberband call is either one of issuing, initiating, invoking or receiving the rubberband call.

FIG. **4** is a schematic diagram illustrating an embodiment of user interface of a portable electronic device **400** having a touch-sensitive display **408**. The display **408** may include a window **410**. The window **410** may include one or more displayed objects, such as information objects **412-1** to **412-4**. In an exemplary embodiment, the information objects **412** may correspond to contact information for one or more individuals in a list of items. The displayed objects may be moved in response to detecting or determining movement **414** of a point of contact with the display, such as that associated with one or more digits **416** of a user (which are not drawn to Scale in FIG. **4**). In some embodiments, movement of the displayed objects may be accelerated in response to detecting or determining accelerated movement of the point of contact. While embodiment **400** includes one window **410**, in other embodiments there may be two or more display windows. In addition, while embodiment **400** illustrates movement **414** in a particular direction, in other embodiments movement of the displayed objects may be in response to movement **414** in one or more other directions, or in response to a scalar (i.e., a determined or detected movement independent of the direction).

FIGS. **5**A-**5**C illustrate the scrolling of a list of items on a device to a terminus of the list, at which point one or more displayed items at the end of the list smoothly bounce off the end of the display, reverse direction, and then optionally come to a stop. FIG. **5**A is a schematic diagram illustrating an embodiment of user interface of a portable electronic device **400** having a touch-sensitive display. One or more displayed objects, such as information object **412-1** may be a distance **512-1** from a terminus **514** of the list of items which is an edge of a scrolled region and may be moving with a velocity **510-1** while the list is being scrolled. Note that the terminus **514** is a virtual boundary associated with the displayed objects, as opposed to a physical boundary associated with the window **410** and/or the display **408**. As illustrated in FIG. **5**B, when the one or more displayed objects, such as the information object **412-1**, reach or intersect with the terminus **514**, the movement corresponding to the scrolling may stop, i.e., the scrolling velocity may be zero at an instant in time. As illustrated in FIG. **5**C, the one or more displayed objects, such as the information **412-1**, may subsequently reverse direction. At a time after the intersection with the terminus **514**, the information object **412-1** may have velocity **510-2** and may be a distance **512-2** from the terminus **514**. In some embodiments, the magnitude of velocity **510-2** may be less than the magnitude of velocity **510-1** when the distance **512-2** equals the distance **512-1**, i.e., the motion of the one or more displayed objects is damped after the scrolling list reaches and "bounces" at its terminus.

In at least some embodiments of the present disclosure, the method **200** performs the bounce operations described in FIGS. **5**A-**5**C. The bounce call transferred at block **202** determines whether a bounce operation is enabled. The maximum and minimum bounces values determine the amount of bouncing of the scrolled region in an opposite direction of the scroll.

FIGS. **6**A-**6**D illustrate the scrolling of a list of items to a terminus of the list, at which point an area beyond the terminus is displayed and the list is then scrolled in an opposite direction until the area beyond the terminus is no longer displayed, in accordance with some embodiments. The rubberband operation of method **300** is illustrated in the example of FIGS. **6**A-**6**D with the listed items being email messages.

FIGS. **6**A-**6**D illustrate an exemplary user interface **3500**A for managing an inbox in accordance with some embodiments. An analogous user interface may be used to display and manage other mailboxes (e.g., drafts, sent, trash, personal, etc.). In addition, other types of lists are possible, including but not limited to lists of instant message conversations, favorite phone numbers, contact information, labels, email folders, email addresses, physical addresses, ringtones, or album names.

If the list of emails fills more than the allotted screen area, the user may scroll through the emails using vertically upward and/or vertically downward swipe gestures on the touch screen. In the example of FIG. **6**A, a portion of a list of emails is displayed in the screen area, including a top displayed email **3530** from Bruce Walker and a bottom displayed email **3532** from Kim Brook. A user performs a vertically downward swipe gesture **3514** to scroll toward the top of the list. The vertically downward gesture **3514** need not be exactly vertical; a substantially vertical gesture is sufficient. In some embodiments, a gesture within a predetermined angle of being perfectly vertical results in vertical scrolling.

As a result of detecting the vertically downward gesture **3514**, in FIG. **6**B the displayed emails have shifted down, such that the previous bottom displayed email **3532** from Kim Brook is no longer displayed, the previous top displayed email **3530** from Bruce Walker is now second from the top, and the email **3534** from Aaron Jones, which was not displayed in FIG. **6**A, is now displayed at the top of the list.

In this example, the email **3534** from Aaron Jones is the first email in the list and thus is the terminus of the list. Upon reaching this email **3534**, in response to continued detection of the vertically downward gesture **3514**, an area **3536** (FIG. **6**C) above the first email **3534** (i.e., beyond the terminus of the list) is displayed. In some embodiments, the area displayed beyond the terminus of the list is visually indistinct from the background of the list. In FIG. **6**C, both the area **3536** and the background of the emails (e.g., emails **3534** and **3530**) are white and thus are visually indistinct.

Once vertically downward gesture **3514** is complete, such that a corresponding object is no longer detected on or near the touch screen display, the list is scrolled in an opposite direction until the area **3536** is no longer displayed. FIG. **6**D illustrates the result of this scrolling in the opposite direction, the email **3534** from Aaron Jones is now displayed at the top of the screen area allotted to the list and the area **3536** is not displayed.

In the example of FIGS. **6**A-**6**D, a vertically downward gesture resulted in display of an area beyond the first item in the list. As described in FIG. **3**, the values for the predetermined maximum displacement (e.g., display of an area beyond the first item in the list) are set at block **304** for top and bottom edges or at block **306** for all edges of the window.

Similarly, a vertically upward gesture may result in display of an area beyond the last item of the list, if the vertically upward gesture continues once the list has been scrolled to the last item. The last item may be considered a terminus of the list, similar to the first item. As discussed above, the gesture need not be exactly vertical to result in vertical scrolling; a gesture within a predefined range of angles from perfectly vertical is sufficient.

FIG. **7** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a directional scrolling operation. The method **700** for operating through an application programming interface (API) includes transferring a directional scroll angle call to determine if directional scrolling is enabled at block **702**. The method **700** further includes transferring a directional scroll angle call to set a scroll angle for locking the scrolling in at least one of a vertical or a horizontal direction at block **704**. The method **700** further includes locking the scrolling in the horizontal direction if a user input forms an angle with a horizontal direction that is less than or equal to a first scroll angle at block **706**. The method **700** further includes locking the scrolling in the vertical direction if a user input forms an angle with the vertical direction that is less than or equal to a second scroll angle at block **708**.

In certain embodiments, a user input in the form of a drag forms an angle with the horizontal direction that is less than the first scroll angle. In this case, the user presumably intends to scroll in the horizontal direction. The scrolling will be locked in the horizontal direction until the user input exceeds the first scroll angle. A second scroll angle may be used for locking the user input in the vertical direction. The second scroll angle may be set equal to the first scroll angle.

FIG. **8** illustrates first and second scroll angles for locking a scroll of a display of a device in a horizontal or vertical direction. The horizontal direction **802** and vertical direction **804** are in reference to a window or a display of a device. As discussed in the method **700**, a user input such as a drag movement forming an angle with the horizontal direction **802** less than or equal to the first scrolling angle **806** or **808** will lock the user input in the horizontal direction. In a similar manner, a user input forming an angle with the vertical direction **810** less than or equal to the second scrolling angle **810** or **812** will lock the user input in the vertical direction. The first and second scrolling angles may be set at the same angle or at different angles as well. For example, the first and second scrolling angles may be set at 25 degrees. A user input less than or equal to 25 degrees with respect to the horizontal or vertical direction will lock the scrolling in the appropriate direction.

In some embodiments, the horizontal and vertical locking angles can be determined in part by the aspect of the content. For example, content in the form of a tall page may receive a larger vertical locking angle compared to the horizontal locking angle.

FIG. **9** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a deceleration scroll operation. The method **900** for providing the deceleration scroll operation includes transferring a deceleration scroll call to set a deceleration factor for a drag user input at block **902**. The method **900** further includes slowing the scroll to a stop based on the speed of the drag user input and the deceleration factor at block **904**.

In certain embodiments, a user input in the form of a drag invokes a scroll operation for a certain time period. The user input has a certain speed. The scroll of the scrolled region of a window or a display region of a display of a device will be stopped after the user input stops by applying a deceleration factor to the speed of the user input during the drag movement.

FIG. **10** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a scroll hysteresis operation. The method **1000** for providing the scroll hysteresis operation includes transferring a scroll

hysteresis call to determine whether a user input invokes a scroll at block **1002**. The method **1000** further includes setting a hysteresis value for determining whether a user input invokes a scroll at block **1004**.

In certain embodiments, a user input in the form of a drag over a certain distance across a display or window within a display of a device invokes a scroll operation. The hysteresis value determines the certain distance which the user input must drag across the display or window prior to invoking a scroll operation. A user input that does not drag the certain predetermined distance will not invoke a scroll operation and may be considered a mouse up or down input or other type of input.

FIG. **11** illustrates details of all application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to attach a scroll indicator to a scroll region edge or a window edge of a device. In some embodiments, the scroll region edge is associated with a content edge. The window or display edge may be associated with the edge of a display region. The method **1100** for providing the scroll indicator includes transferring a scroll indicator call to determine whether at least one scroll indicator attaches to an edge of a scroll region or a window edge at block **1102**. A scroll indicator may be displayed on any display edge, window edge or scroll region edge. The method **1100** further includes optionally attaching at least one scroll indicator to the edge of the scroll region based on the scroll indicator call at block **1104**. Alternatively, the method **1100** further includes optionally attaching at least one scroll indicator to the window edge of the view based on the scroll indicator call at block **1106**.

In some embodiments, the operations of method **1100** can be altered, modified, combined, or deleted. For example, block **1104** can be deleted. Likewise, block **1106** can be deleted from the method **1100**. Alternatively, the order of block **1104** and block **1106** can be switched. Other methods having various operations that have been disclosed within the present disclosure can also be altered, modified, rearranged, collapsed, combined, or deleted.

In certain embodiments of the present disclosure, transferring the scroll indicator call is either one of issuing, initiating, invoking or receiving the scroll indicator call. For example, the user interface software (e.g., software kit or library) may receive the scroll indicator call from the software application.

FIG. **12** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to determine if an inadvertent user input contacts a view of a display of a device. The method **1200** includes transferring an inadvertent user input call to determine whether the user input was inadvertent at block **1202**. The method **1200** further includes ignoring the inadvertent user input based on the determination of the inadvertent user input call at block **1204**. In one embodiment, the inadvertent user input call comprises a thumb detection call to determine whether the user input was an inadvertent thumb.

In certain embodiments of the present disclosure, transferring the inadvertent user input call is either one of issuing, initiating, invoking or receiving the inadvertent user input call.

A gesture API provides an interface between an application and user software in order to handle gesturing. Gesturing may include scaling, rotating, or other changes to a view, window, or display. A mask may merely permit certain changes while limiting or not permitting other changes. Events of all kinds come into the application via a graphics framework. They are enqueued, collaleced if necessary and dispatched. If the events are system level events (e.g., application should suspend, device orientation has chanted, etc) they are routed to the application having an instance of a class of the user interface software. If the events are hand events based on a user input, the events are routed to the window they occurred over. The window then routes these events to the appropriate control by calling the instance's mouse and gesture methods. The control that receives a mouse down or mouse entered function will continue to get all future calls until the hand is lifted. If a second finger is detected, the gesture methods or functions are invoked. These functions may include start, change, and end gesture calls. The control that receives start gesture call will be sent all future change gesture calls until the gesture ends.

FIG. **13** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a gesture operation. The method **1300** for providing the gesture operation includes transferring a handle gesture event call at block **1302**. The method **1300** further includes transferring a gesture change call in response to the handle gesture event call at block **1304**.

In certain embodiments, a user input in the form of two or more points is received by a display of a device. A multi-touch driver of the device receives the user input and packages the event into an event object. A window server receives the event object and determines whether the event object is a gesture event object. If the window server determines that a gesture event object has been received, then user interface software issues or transfers the handle gesture call at block **1302** to a software application associated with the view. The software application confirms that a gesture event has been received and passes the handle gesture call to a library of the user interface software. The window server also associates the gesture event object with the view that received the user input. The library responds by transferring a gesture change call in response to the handle gesture event call at block **1304**.

In one embodiment, a window or view associated with the user input receives the change call in order to perform the gesture event. The user software that provides the view receives a gesture start event call, a gesture changed event call, a zoom to scale setting for the view, and a gesture end call. The gesture calls receive an input of a gesture event which may be base event having a type such as a hand event, keyboard event, system event, etc. A delegate associated with the application receives a start gesture call, gesture did change call, and gesture did finish call. The user software is dynamically linking into the application during the run time of the gesture process.

In some embodiments, the gesture changed function call contains the following information about the gesture:

the number of fingers currently down;

the number of fingers initially down;

the rotation of the hand;

the scale of the hand;

the translation of the hand;

the position of the inner and outermost fingers; and

the pressure of the first finger.

In other embodiments, more information about each finger down may be included as follows.

the stage of the finger Oust touch down, fully pressed, lifting off, etc);

the position of the finger;

the proximity of the finger (how hard you're touching);

the orientation of the finger (what angle the ovoid is at);

the length of the major and minor axis,

the velocity of the finger; and

the eccentricity of the finger's ovoid.

A gesture event object may be a chord event object having a chord count (e.g., number of fingers contacted the view or display), a chord start event, a chord change event, and a chord end event. A chord change event may include a scaling or rotation transform.

FIG. **14** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a scaling transform of a display region, window, or view of a display of a device. The method **1400** for providing the scaling transform includes transferring a scaling transform call to determine a scaling transform for a view associated with a user input having a plurality of input points at block **1402**. The method **1400** further includes transferring a scaling gesture start call at block **1404**. The method **1400** further includes transferring a scaling gesture progress call at block **1406**. The method **1200** further includes transferring a scaling gesture end call at block **1408**.

In certain embodiments, a user input in the form of two or more input points (e.g., fingers) moves together or apart to invoke a gesture event that performs a scaling transform on the view associated with the user input. A scale transform includes a minimum and maximum scale factor. FIG. **15** illustrates a display **1502** of a device having a scaling transform of a view. The view **1504** (e.g., web, text, or image content) has a first scale factor. A user input (e.g., two fingers moving apart) associated with the view **1504** is interpreted as a gesture event to zoom out from view **1504** to view **1508** having a second scale factor that exceeds the maximum scale factor of the view **1516**. A snapback flag determines whether the zoom out can proceed past the maximum scale factor to view **1508** prior to snapping back to the maximum scale factor associated with view **1516**.

FIG. **16**A illustrates a display **1604** of a device having a first scaling factor of a view **1616**. A user input (e.g., two fingers **1608** and **1610** moving together) associated with the view **1614** is interpreted as a gesture event to zoom in from view **1614** to view **1664** having a second scale factor as illustrated in FIG. **16**B. The dashed regions **1602** and **1650** represent the total area of the content with the only content being displayed in the display area **1604** and **1652**. In performing the scaling transform from FIG. **16**A to FIG. **16**B, the center of the gesture event, center **1612** for FIG. **16**A and center **1660** for FIG. **16**B, remains in the same position with respect to the display **1604**. The scroll indicator **1606** shrinks to become scroll indicator **1654** during the transform to indicate that a smaller portion of the total content **1650** is being displayed on display **1604** as a result of the zoom in operation. The dashed region **1650** is larger than the dashed region **1602** to represent that a larger portion of content is not being displayed on display **1652** in FIG. **16**B as a result of the zoom in operation.

In at least some embodiments of the present disclosure, a user desires to change a view **1670** from a scale factor of 2× to a scale factor of 1× as illustrated in FIG. **16**C. A first set of user inputs **1672** and **1674** that move to the second set of user inputs **1676** and **1678** will decrease the scale factor from 2× to 1×. It may be desirable for the user to scale from 2× to 1× without having to move the user inputs a large distance across the view **1670**. In an environment with user interface software interacting with a software application, a gesture scaling transform flag may be set in order to determine a scaling transform for a view associated with a user input having a plurality of input points. The scaling transform flag scales either from a current scale factor to a minimum scale factor or from the current scale factor to a maximum scale factor. For example, a flag may be set at the position associated with a 1.5× scale factor and a third set of user inputs **1680** and **1682**. A user desiring to change the scale factor from 2× to 1× would only have to move his fingers, the user inputs, from the first set **1672** and **1674** to the third set **1680** and **1682** if the gesture scaling transform flag has been set at a scale factor of 1.5×.

FIG. **17** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to provide a rotation transform of a view, window, or display region of a display of a device. The method **1700** for providing the rotation transform includes transferring a rotation transform call to determine a rotation transform for a view associated with a user input having a plurality of input points at block **1702**. The method **1700** further includes transferring a rotation gesture start call at block **1704**. The method **1700** further includes transferring a scaling gesture progress call at block **1706**. The method **1700** further includes transferring a scaling gesture end call at block **1708**.

In certain embodiments, a user input in the form of two or more input points rotates to invoke a gesture event that performs a rotation transform on the view associated with the user input. The rotation transform includes a minimum and maximum degree of rotation for associated minimum and maximum rotation views. The user input may temporarily rotate a view past a maximum degree of rotation prior to the view snapping back to the maximum degree of rotation.

FIG. **18** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application in order to notify a delegate of at least one animation associated with a display region, window, or view of a display of a device. A delay in the animation may be specified by the API. Also, multiple animations may be assigned priority by the API. The method **1800** for notifying the delegate includes determining whether any animation occurs at block **1802**. The method **1800** further includes checking the progress of an animation at block **1804**. If progress has occurred, then the next state (e.g., position, opacity, or transform) of the animation can be calculated at block **1806**. If progress has completed at block **1806**, then at block **1808** it is determined whether the view associated with the completed animation is associated with a delegate. If so, a delegate call is transferred to notify the delegate of the animation for the view at block **1810**. The delegate operating

under the control of the software application can change other views in response to the view being modified by the animation.

In certain embodiments, software invokes an animation that performs a scaling transform on the view associated with the user input. A display may include numerous views. The view being increased in size by the scaling transform may obstruct other views in which case the other views may need to be reduced in size. Alternatively, the view being decreased in size by the scaling transform may create additional area for other views to increase in size.

FIG. **19** is flow chart of a method for animating a display region, windows, or views displayed on a display of a device. The method **1900** includes starting at least two animations at block **1902**. The method **1900** further includes determining the progress of each animation at block **1904**. The method **1900** further includes completing each animation based on a single timer at block **1906**.

In certain embodiments of the present disclosure, the single timer includes a timer based on a redraw interval which is a time period between the display of a current frame and a next frame of the display of the device. In this case, changes in animation are updated to the display during the redraw interval in order to display the chances during the next frame of the display. The progress of each animation may be calculated periodically or based upon a progress call.

The method **1900** may further include determining whether each animation is associated with a delegate. The delegate is then notified of the animation. Other views not associated with an animation may be changed depending on the software application controlling the delegate.

FIG. **20** is flow chart of a method for animating a display region, windows, or views displayed on a display of a device. The method **2000** includes providing a single animation timer at block **2002**. The method **2000** further includes animating a plurality of animations with the single animation timer at block **2004**. For example, a single timer may control all animations which occur simultaneously. The animations may include a transform, a frame, and an opacity animation. A animation transform may include a scaling or rotation transform. A frame animation may include resizing of a frame. An opacity animation changes the opacity from opaque to transparent or vice versa.

FIG. **21** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with multiple software applications or processes in order to synchronize animations associated with multiple views or windows of a display of a device. The method **2100** for synchronizing the animations includes setting attributes of views independently with each view being associated with a process at block **2102**. For example, an attribute or property of a view may include a position, size, opacity, etc. An animation alters one or more attributes from a first state to a second state. The method **2100** further includes transferring a synchronization call to synchronize animations for the multiple views of the display at block **2104**. The synchronization call may include input parameters or arguments such as an identification of the synchronization of the processes and a list of the processes that are requesting animation of the multiple views. In one embodiment, the synchronization call includes the identification and the number of processes that are requesting animation. In one embodiment, each application or process sends a synchronization call at different times. The method **2100** further includes transferring a synchronization confirmation message when a synchronization flag is enabled at block **2106**. The synchronization flag can be enabled when the processes to be synchronized have each sent messages to a window server operating the user interface software. The method **2100** further includes updating the attributes of the views from a first state to a second state independently at block **2108**. In one embodiment, the window server receives the updated attributes from each process at different times. The method **2100** further includes transferring a start animation call to draw the requested animations when both processes have updated attributes associated with the second state at block **2110**.

In some embodiments, a first data structure or layer tree represents a hierarchy of layers that correspond to the views or windows of the processes. A second data structure or render tree represents a similar copy of the layer tree. However, the render tree is not updated until the independent processes have completed their separate animations. At this time, the render tree updates and redraws the screen with the new animations.

FIGS. **22**A and **22**B illustrate synchronizing the resizing of views or windows of a display of a device. For example, a window **2210** associated with a first process with a size attribute may increase in size by changing from a first state, window **2210** in FIG. **22**A, to a second state window **2210** in FIG. **22**B. At approximately the same time, a second window **2220** may decrease in size in proportion to the increase in size of the first window **2210**. The method **2100** provides synchronization of the resizing of the windows **2210** and **2220** illustrated in FIGS. **22**A and **22**B. The animations in changing from the first state to the second state may occur incrementally and occur with the synchronization of method **2100**.

FIG. **23** illustrates a method for switching ownership of a view of an application displayed on a display of a data processing device. The method **2300** includes constructing a data structure having a hierarchy of layers with a layer being associated with a view and owning the view at block **2302**. The layers may be content, windows, video, images, text, media, or any other type of object for user interface of the application. The method **2300** further includes removing the layer from the data structure at block **2304**. The method **2300** further includes switching ownership of the view from the layer to the view at block **2306**.

In some embodiments, each layer from the data structure is associated with a view. The layer associated with the view sends a delegate function call to the view in order to generate content provided by the view. A first pointer reference points from the layer to the view. A second pointer reference points from the view to the layer. The number of references pointing to an object such as the view is defined as the retained count of the object. The view may receive notification that the layer will be removed from the data structure. Removing the layer from the data structure may occur based on the view associated with the layer being removed from the display of the device. When the layer is removed from the data structure or layer tree the pointer from the layer to the view will be removed. The view will have a retained count of zero and be deallocated or removed from memory if the ownership of the view is not reversed. The view will have a retained count of at least one if ownership is reversed.

FIG. **24** illustrates a method for memory management of a view of an application displayed on a display of a device. The method **2400** includes constructing a data structure having a hierarchy of layers with at least one layer being associated with the view at block **2402**. The method **2400** further includes storing the data structure in memory at block **2404**. The method **2400** further includes maintaining a retained count of the number of references to the view from other

objects at block **2406**. The method **2400** further includes deallocating the view from memory if the retained count is zero at block **2408**. As discussed above, the retained count of the view will be decremented if the layer is removed from the data structure. Removing the layer from the data structure may occur based on the view associated with the layer being removed from the display of the device.

FIGS. **25**A and **25**B illustrate a data structure having a hierarchy of layers with a layer being associated with a view. The data structure includes layers **2502**, **2504**, and **2506**. Layer **2506** is associated with the view **2510**. The layer **2506** associated with the **2510** view sends a delegate call to the view in order to generate content provided by the view. A first pointer reference **2508** points from the layer **2506** to the view **2510**. A second pointer reference **2512** points from the view **2510** to the layer **2506**. A third pointer reference **2532** may point from user interface (UI) controller **2530** to the view **2510**. The UI controller **2530** may control operations associated with the view **2510** such as scrolling the view **2510** in response to a user input. The view **2510** in FIG. **25**A has a retained count of two based on the pointer references **2508** and **2532**.

If the layer **2506** is removed from the data structure as illustrated in FIG. **25**B, then the pointer **2508** is removed. View **2510** will have a lower retained count as illustrated in FIG. **25**B. If view **2510** has a retained count of zero, then the memory storing the view **2510** will be deallocated.

FIG. **26** illustrates a method for compositing media and non-media content of user interface for display on a device. The method **2600** includes constructing a data structure having a hierarchy of layers associated with the user interface of the device at block **2602**. The method **2600** further includes determining whether each layer of the data structure is associated with media or non-media content at block **2604**. The data structure or layer tree is traversed in order to determine whether each of the layers of the data structure is associated with media or non-media content. The method **2600** further includes detaching a layer associated with media content from the data structure at block **2606**. The method **2600** further includes storing media content in a first memory location at block **2606**. The method **2600** further includes storing non-media content in a second memory location at block **2608**. The method **2600** further includes compositing the media and non-media content for display on the device at block **2610**.

In some embodiments, compositing the media and non-media content includes retrieving the media content from the first memory location, retrieving the non-media content from the second memory location, and scanning the media and non-media content directly to the display. The memory location can be any type of memory located in cache, main memory, a graphics processing unit, or other location within a device. The media content may include video, video plug-in, audio, image, or other time varying media. The media content may be in the form of a YUV model with the Y representing a luminance component (the brightness) and U and V representing chrominance (color) components. The media content may be scanned to the display at a rate of substantially twenty to forty frames per second. The media content may be scaled prior to being scanned to the display of the device.

The non-media content may include content, views, and images that do not require frequent updating. The non-media content may be in the form of a RGB model which is an additive model in which red, green, and blue (often used in additive light models) are combined in various ways to reproduce other colors. The non-media content may be scanned to the display at a slower rate compared to the media content.

FIG. **27** illustrates a data structure or layer tree having a hierarchy of layers. The layers can be associated with media and non-media content. For example, layer **2704** is associated with media content **2706** such as a video. Layer **2710** is associated with non-media content **2712** which may be user interface view for the video. Layers **2720**, **2730**, and **2740** are associated with non-media content **2722**, **2732**, and **2742**, respectively, that forms the components of the non-media content **2712**. The method **2600** will determine whether each layer of the data structure is associated with media or non-content. Any layers associated with media content such as layer **2704** will be removed from the data structure and processed in a separate memory location.

In some embodiments, the methods, systems, and apparatuses of the present disclosure can be implemented in various devices including electronic devices, consumer devices, data processing devices, desktop computers, portable computers, wireless devices, cellular devices, tablet devices, handheld devices, multi touch devices, multi touch data processing devices, any combination of these devices, or other like devices. FIGS. **4-6** and **28-33** illustrate examples of a few of these devices.

FIG. **28** illustrates a device **2800** according to one embodiment of the disclosure. FIG. **28** shows a wireless device in a telephone configuration having a "candy-bar" style. In FIG. **28**, the wireless device **2800** may include a housing **2832**, a display device **2834**, an input device **2836** which may be an alphanumeric keypad, a speaker **2838**, a microphone **2840** and an antenna **2842**. The wireless device **2800** also may include a proximity sensor **2844** and an accelerometer **2846**. It will be appreciated that the embodiment of FIG. **28** may use more or fewer sensors and may have a different form factor from the form factor shown in FIG. **28**.

The display device **2834** is shown positioned at an upper portion of the housing **2832**, and the input device **2836** is shown positioned at a lower portion of the housing **2832**. The antenna **2842** is shown extending from the housing **2832** at an upper portion of the housing **2832**. The speaker **2838** is also shown at an upper portion of the housing **2832** above the display device **2834**. The microphone **2840** is shown at a lower portion of the housing **2832**, below the input device **3286**. It will be appreciated that the speaker **2838** and microphone **2840** can be positioned at any location on the housing, but are typically positioned in accordance with a user's ear and mouth, respectively.

The display device **2834** may be, for example, a liquid crystal display (LCD) which does not include the ability to accept inputs or a touch input screen which also includes an LCD. The input device **2836** may include, for example, buttons, switches, dials, sliders, keys or keypad, navigation pad, touch pad, touch screen, and the like. Any well-known speaker, microphone and antenna can be used for speaker **2838**, microphone **2840** and antenna **2842**, respectively.

The data acquired from the proximity sensor **2844** and the accelerometer **2846** can be combined together, or used alone, to gather information about the user's activities. The data from the proximity sensor **2844**, the accelerometer **2846** or both can be used, for example, to activate/deactivate a display backlight, initiate commands, make selections, control scrolling, gesturing, animating or other movement in a display, control input device settings, or to make other changes to one or more settings of the device. In certain embodiments of the present disclosure, the device **2800** can be used to implement at least some of the methods discussed in the present disclosure.

FIG. **29** shows a device **2950** in accordance with one embodiment of the disclosure. The device **2950** may include a housing **2952**, a display/input device **2954**, a speaker **2956**, a microphone **2958** and an optional antenna **2960** (which may be visible on the exterior of the housing or may be concealed within the housing). The device **2950** also may include a proximity sensor **2962** and an accelerometer **2964**. The device **2950** may be a cellular telephone or a device which is an integrated PDA and a cellular telephone or a device which is an integrated media player and a cellular telephone or a device which is both an entertainment system (e.g. for playing games) and a cellular telephone, or the device **2950** may be other types of devices described herein. In one particular embodiment, the device **2950** may include a cellular telephone and a media player and a PDA, all contained within the housing **2952**. The device **2950** may have a form factor which is small enough that it fits within the hand of a normal adult and is light enough that it can be carried in one hand by an adult. It will be appreciated that the term "portable" means the device can be easily held in an adult user's hands (one or both); for example, a laptop computer and an iPod are portable devices.

In one embodiment, the display/input device **2954** may include a multi-point touch input screen in addition to being a display, such as an LCD. In one embodiment, the multi-point touch screen is a capacitive sensing medium configured to detect multiple touches (e.g., blobs on the display from a user's face or multiple fingers concurrently touching or nearly touching the display) or near touches (e.g., blobs on the display) that occur at the same time and at distinct locations in the plane of the touch panel and to produce distinct signals representative of the location of the touches on the plane of the touch panel for each of the multiple touches.

In certain embodiments of the present disclosure, the device **2800** can be used to implement at least some of the methods discussed in the present disclosure.

FIGS. **30**A and **30**B illustrate a device **3070** according to one embodiment of the disclosure. The device **3070** may be a cellular telephone which includes a hinge **3087** that couples a display housing **3089** to a keypad housing **3091**. The hinge **3087** allows a user to open and close the cellular telephone so that it can be placed in at least one of two different configurations shown in FIGS. **30**A and **30**B. In one particular embodiment, the hinge **3087** may rotatably couple the display housing to the keypad housing. In particular, a user can open the cellular telephone to place it in the open configuration shown in FIG. **30**A and can close the cellular telephone to place it in the closed configuration shown in FIG. **30**B. The keypad housing **3091** may include a keypad **3095** which receives inputs (e.g. telephone number inputs or other alphanumeric inputs) from a user and a microphone **3097** which receives voice input from the user. The display housing **3089** may include, on its interior surface, a display **3093** (e.g. an LCD) and a speaker **3098** and a proximity sensor **3084**; on its exterior surface, the display housing **3089** may include a speaker **3096**, a temperature sensor **3094**, a display **3088** (e.g. another LCD), an ambient light sensor **3092**, and a proximity sensor **3084**A. Hence, in this embodiment, the display housing **3089** may include a first proximity sensor on its interior surface and a second proximity sensor on its exterior surface.

In at least certain embodiments, the device **3070** may contain components which provide one or more of the functions of a wireless communication device such as a cellular telephone, a media player, an entertainment system, a PDA, or other types of devices described herein. In one implementation of an embodiment, the device **3070** may be a cellular telephone integrated with a media player which plays MP3 files, such as MP3 music files.

Each of the devices shown in FIGS. **4**, **5**A, **5**B, **5**C, **6**A, **6**B, **6**C, **6**D, **28**, **29**, **30**A and **30**B may be a wireless communication device, such as a cellular telephone, and may include a plurality of components which provide a capability for wireless communication. FIG. **31** shows an embodiment of a wireless device **3070** which includes the capability for wireless communication. The wireless device **3070** may be included in any one of the devices shown in FIGS. **4**, **5**A, **5**B, **5**C, **6**A, **6**B, **6**C, **6**D, **28**, **29**, **30**A and **30**B, although alternative embodiments of those devices of FIGS. **4**, **5**A, **5**B, **5**C, **6**A, **6**B, **6**C, **6**D, **28**, **29**, **30**A and **308** may include more or fewer components than the Wireless device **3070**.

Wireless device **3070** may include an antenna system **3101**. Wireless device **3070** may also include a digital and/or analog radio frequency (RF) transceiver **3102**, coupled to the antenna system **3101**, to transmit and/or receive voice, digital data and/or media signals through antenna system **3101**.

Wireless device **3070** may also include a digital processing system **3103** to control the digital RF transceiver and to manage the voice, digital data and/or media signals. Digital processing system **3103** may be a general purpose processing device, such as a microprocessor or controller for example. Digital processing system **3103** may also be a special purpose processing device, such as an ASIC (application specific integrated circuit), FPGA (field-programmable gate array) or DSP (digital signal processor). Digital processing system **3103** may also include other devices, as are known in the art, to interface with other components of wireless device **3070**. For example, digital processing system **3103** may include analog-to-digital and digital-to-analog converters to interface with other components of wireless device **3070**. Digital processing system **3103** may include a media processing system **3109**, which may also include a general purpose or special purpose processing device to manage media, such as files of audio data.

Wireless device **3070** may also include a storage device **3104**, coupled to the digital processing system, to store data and/or operating programs for the Wireless device **3070**. Storage device **3104** may be, for example, any type of solid-state or magnetic memory device.

Wireless device **3070** may also include one or more input devices **3105**, coupled to the digital processing system **3103**, to accept user inputs (e.g., telephone numbers, names, addresses, media selections, etc.) Input device **3105** may be, for example, one or more of a keypad, a touchpad, a touch screen, a pointing device in combination with a display device or similar input device.

Wireless device **3070** may also include at least one display device **33106**, coupled to the digital processing system **3103**, to display information such as messages, telephone call information, contact information, pictures, movies and/or titles or other indicators of media being selected via the input device **3105**. Display device **3106** may be, for example, an LCD display device. In one embodiment, display device **3106** and input device **3105** may be integrated together in the same device (e.g., a touch screen LCD such as a multi-touch input panel which is integrated with a display device, such as an LCD display device). The display device **3106** may include a backlight **3106**A to illuminate the display device **3106** under certain circumstances. It will be appreciated that the Wireless device **3070** may include multiple displays.

Wireless device **3070** may also include a battery **3107** to supply operating power to components of the system including digital RF transceiver **3102**, digital processing system **3103**, storage device **3104**, input device **3105**, microphone

310A, audio transducer **3108**, media processing system **3109**, sensor(s) **3110**, and display device **3106**. Battery **3107** may be, for example, a rechargeable or non-rechargeable lithium or nickel metal hydride battery. Wireless device **3070** may also include audio transducers **3108**, which may include one or more speakers, and at least one microphone **3105**A. In certain embodiments of the present disclosure, the wireless device **3070** can be used to implement at least some of the methods discussed in the present disclosure.

FIG. **32** shows another example of a device according to an embodiment of the disclosure. This device **3200** may include a processor, such as microprocessor **3202**, and a memory **3204**, which are coupled to each other through a bus **3206**. The device **3200** may optionally include a cache **3208** which is coupled to the microprocessor **3202**. This device may also optionally include a display controller and display device **3210** which is coupled to the other components through the bus **3206**. One or more input/output controllers **3212** are also coupled to the bus **3206** to provide an interface for input/output devices **3214** and to provide an interface for one or more sensors **3216** which are for sensing user activity. The bus **3206** may include one or more buses connected to each other through various bridges, controllers, and/or adapters as is well known in the art. The input/output devices **3214** may include a keypad or keyboard or a cursor control device such as a touch input panel. Furthermore, the input/output devices **3214** may include a network interface which is either for a wired network or a wireless network (e.g. an RF transceiver). The sensors **3216** may be any one of the sensors described herein including, for example, a proximity sensor or an ambient light sensor. In at least certain implementations of the device **3200**, the microprocessor **3202** may receive data from one or more sensors **3216** and may perform the analysis of that data in the manner described herein. For example, the data may be analyzed through an artificial intelligence process or in the other ways described herein. As a result of that analysis, the microprocessor **3202** may then automatically cause an adjustment in one or more settings of the device.

In certain embodiments of the present disclosure, the device **3200** can be used to implement at least some of the methods discussed in the present disclosure.

FIGS. **33**A-C show another example of a device according to at least certain embodiments of the disclosure. FIG. **33**A illustrates a laptop device **3300** with a keyboard **3302**, a body **3304**, a display frame **3306**, and a display **3308**. The laptop device **3300** can be converted into a tablet device as illustrated in FIG. **33**B and FIG. **33**C. FIG. **33**B illustrates the conversion of the laptop device into a tablet device. An edge of a display frame **3356** containing a display **3358** is slide within the body **3354** across the top of a keyboard **3352** until forming a tablet device as illustrated in FIG. **33**C. The tablet device with a display **2362** and a display frame **3366** rests on top of a body **3360**.

In certain embodiments of the present disclosure, the laptop device **3300** can be used to implement at least some of the methods discussed in the present disclosure.

FIG. **34** illustrates details of an application programming interface in flow chart form according to certain teachings of the present disclosure. The application programming interface operates in an environment with user interface software interacting with a software application. In some embodiments, a hierarchy of views operates on top of a hierarchy of layers within the user interface software. The API operates as illustrated in method **3400** that includes constructing a hierarchy of views operating on top of a hierarchy of layers at block **3402**. The method **3400** further includes providing access to the hierarchy of views without providing access to the hierarchy of layers at block **3404**. An application may interact with the hierarchy of views via the API without accessing the hierarchy of layers operating below the hierarchy of views.

In some embodiments, a platform provides various scrolling, gesturing, and animating operations. The platform includes hardware components and an operating system. The hardware components may include a processing unit coupled to an input panel and a memory coupled to the processor. The operating system includes one or more programs that are stored in the memory and configured to be executed by the processing unit. One or more programs include various instructions for transferring function calls or messages through an application programming interface in order to perform various scrolling, gesturing, and animating operations.

In an embodiment, the one or more programs include instructions for transferring a bounce call through an API to cause a bounce of a scrolled region in an opposite direction of a scroll based on a region past an edge of the scrolled region being visible in a display region at the end of the scroll. In an embodiment, the one or more programs include instructions for transferring a rubberband call through an API to cause a rubberband effect on a scrolled region by a predetermined maximum displacement when the scrolled region exceeds a display edge based on a scroll. In an embodiment, the one or more programs include instructions for transferring a directional scroll call through an API to set a scroll angle for locking the scrolling in at least one of a vertical or a horizontal direction.

In an embodiment, the one or more programs include instructions for transferring a scroll hysteresis call through an API to determine whether a user input invokes a scroll. In an embodiment, the one or more programs include instructions for transferring a deceleration scroll call through an API to set a deceleration factor for a user input based on the user input invoking a scroll. In an embodiment, the one or more programs include instructions for transferring a scroll indicator call through an API to determine whether at least one scroll indicator attaches to a content edge or a display edge of a display region.

In some embodiments, the platform includes a framework containing a library of software code. The framework interacts with the programs of the platform to provide application programming interfaces for performing various scrolling, gesturing, and animating operations. The framework also includes associated resources (e.g., images, text, etc.) that are stored in a single directory.

In an embodiment, the library of the framework provides an API for specifying a bounce operation to cause a bounce of a scrolled region in an opposite direction of a scroll based on a region past an edge of the scrolled region being visible in a display region at the end of the scroll. In an embodiment, the library of the framework provides an API for specifying a rubberband operation that has a rubberband effect on a scrolled region by a predetermined maximum displacement when the scrolled region exceeds a display edge based on a scroll. In an embodiment, the library of the framework provides an API for specifying a directional scroll operation to set a scroll angle for locking the scrolling in at least one of a vertical or a horizontal direction.

In an embodiment, the library of the framework provides an API for specifying a scroll hysteresis operation to determine whether a user input invokes a scroll. In an embodiment, the library of the framework provides an API for specifying a deceleration scroll operation to set a deceleration factor for a user input based on the user input invoking a scroll. In an

embodiment, the library of the framework provides an API for specifying a scroll indicator operation to determine whether at least one scroll indicator attaches to a content edge or a display edge of a display region.

In the foregoing specification, the disclosure has been described with reference to specific exemplary embodiments thereof. It will be evident that various modifications may be made thereto without departing from the broader spirit and scope of the disclosure as set forth in the following claims. The specification and drawings are, accordingly, to be regarded in an illustrative sense rather than a restrictive sense.

What is claimed is:

1. A machine implemented method for scrolling on a touch-sensitive display of a device comprising:

receiving a user input, the user input is one or more input points applied to the touch-sensitive display that is integrated with the device;

creating an event object in response to the user input;

determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation;

issuing at least one scroll or gesture call based on invoking the scroll or gesture operation;

responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object based on an amount of a scroll with the scroll stopped at a predetermined position in relation to the user input; and

responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input.

2. The method as in claim **1**, further comprising:

rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll.

3. The method as in claim **1**, further comprising:

attaching scroll indicators to a content edge of the window.

4. The method as in claim **1**, further comprising:

attaching scroll indicators to the window edge.

5. The method as in claim **1**, wherein determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period.

6. The method as in claim **1**, further comprising:

responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input.

7. The method as in claim **1**, wherein the device is one of: a data processing device, a portable device, a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone.

8. A machine readable storage medium storing executable program instructions which when executed cause a data processing system to perform a method comprising:

receiving a user input, the user input is one or more input points applied to a touch-sensitive display that is integrated with the data processing system;

creating an event object in response to the user input;

determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation;

issuing at least one scroll or gesture call based on invoking the scroll or gesture operation;

responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object; and

responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input.

9. The medium as in claim **8**, further comprising:

rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolled region exceeds a window edge based on the scroll.

10. The medium as in claim **8**, further comprising:

attaching scroll indicators to a content edge of the view.

11. The medium as in claim **8**, further comprising:

attaching scroll indicators to a window edge of the view.

12. The medium as in claim **8**, wherein determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period.

13. The medium as in claim **8**, further comprising:

responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input.

14. The medium as in claim **8**, wherein the data processing system is one of: a data processing device, a portable device, a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone.

15. An apparatus, comprising:

means for receiving, through a hardware device, a user input on a touch-sensitive display of the apparatus, the user input is one or more input points applied to the touch-sensitive display that is integrated with the apparatus;

means for creating an event object in response to the user input;

means for determining whether the event object invokes a scroll or gesture operation by distinguishing between a single input point applied to the touch-sensitive display that is interpreted as the scroll operation and two or more input points applied to the touch-sensitive display that are interpreted as the gesture operation;

means for issuing at least one scroll or gesture call based on invoking the scroll or gesture operation;

means for responding to at least one scroll call, if issued, by scrolling a window having a view associated with the event object; and

means for responding to at least one gesture call, if issued, by scaling the view associated with the event object based on receiving the two or more input points in the form of the user input.

16. The apparatus as in claim **15**, further comprising:

means for rubberbanding a scrolling region displayed within the window by a predetermined maximum displacement when the scrolling region exceeds a window edge based on the scroll.

17. The apparatus as in claim **15**, further comprising:

means for attaching scroll indicators to a content edge of the window.

18. The apparatus as in claim **15**, further comprising:

means for attaching scroll indicators to the window edge.

19. The apparatus as in claim **15**, wherein determining whether the event object invokes a scroll or gesture operation is based on receiving a drag user input for a certain time period.

20. The apparatus as in claim **15**, further comprising:

means for responding to at least one gesture call, if issued, by rotating a view associated with the event object based on receiving a plurality of input points in the form of the user input.

21. The apparatus as in claim **15**, wherein the apparatus is one of: a data processing device, a portable device, a portable data processing device, a multi touch device, a multi touch portable device, a wireless device, and a cell phone.

* * * * *

Exhibit 6

US007853891B2

(12) **United States Patent**
Chaudhri et al.

(10) **Patent No.: US 7,853,891 B2**
(45) **Date of Patent: Dec. 14, 2010**

(54) **METHOD AND APPARATUS FOR DISPLAYING A WINDOW FOR A USER INTERFACE**

(75) Inventors: **Imran Chaudhri**, San Francisco, CA (US); **Bas Ording**, San Francisco, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 247 days.

(21) Appl. No.: **12/012,384**

(22) Filed: **Feb. 1, 2008**

(65) **Prior Publication Data**

US 2008/0222554 A1 Sep. 11, 2008

Related U.S. Application Data

(63) Continuation of application No. 11/635,847, filed on Dec. 8, 2006, which is a continuation of application No. 10/193,573, filed on Jul. 10, 2002, now Pat. No. 7,343,566.

(51) **Int. Cl.**
G06F 17/00 (2006.01)

(52) **U.S. Cl.** **715/781**; 715/768; 715/788; 715/795; 719/318; 345/629

(58) **Field of Classification Search** 715/768, 715/781, 788, 795; 719/318; 345/629
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,191,620 A 3/1993 Lee
5,333,272 A 7/1994 Capek et al.
5,831,615 A 11/1998 Drews et al.
5,838,318 A 11/1998 Porter et al.
5,892,511 A 4/1999 Gelsinger et al.
5,929,854 A 7/1999 Ross
5,940,517 A 8/1999 Shinanda et al.
5,999,191 A 12/1999 Frank et al.
6,008,809 A 12/1999 Brooks
6,072,489 A 6/2000 Gough et al.
6,246,407 B1 6/2001 Wilks et al.
6,249,826 B1 * 6/2001 Parry et al. 710/19
6,252,595 B1 6/2001 Birmingham et al.
6,307,545 B1 10/2001 Conrad et al.
6,336,131 B1 * 1/2002 Wolfe 709/203
6,409,603 B1 6/2002 Nishino et al.
6,429,883 B1 8/2002 Plow et al.
6,600,500 B1 * 7/2003 Yamamoto 715/795

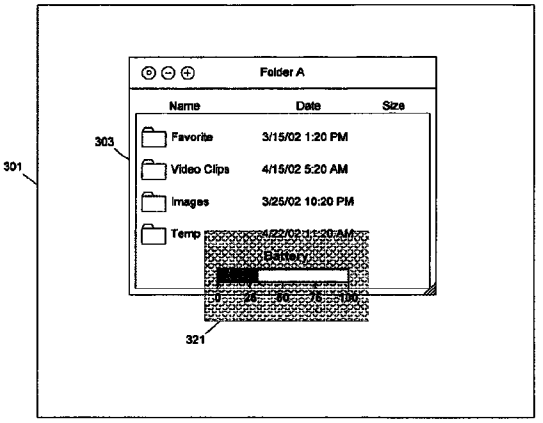
(Continued)

Primary Examiner—Tadeese Hailu
(74) *Attorney, Agent, or Firm*—Brian K. McKnight; Novak Druce + Quigg LLP

(57) **ABSTRACT**

Methods and apparatuses to display windows. In more than one embodiments of the invention, a window is closed automatically (e.g., after a timer expires, or when a condition or criterion is met, or a system input is received) without user input. In some examples, the window is translucent so that the portion of another window, when present, is visible under the window. In some examples, the image of the window is faded out before the window is closed and destroyed. In some examples, the window does not close in response to any input from a user input device. In some examples, the window is repositioned (or hidden) automatically when another translucent window is displayed. The degree of translucency, the speed for fading out, the discrete levels of translucency for fading out, the time to expire, and/or other parameters for controlling the display of the window may be set by the user or adjusted by the system (or application software programs) automatically according to system conditions or other criteria.

75 Claims, 21 Drawing Sheets



U.S. PATENT DOCUMENTS

6,654,036	B1	11/2003	Jones	
6,670,970	B1	12/2003	Bonura et al.	
6,720,982	B1	4/2004	Sakaguchi	
6,828,989	B2 *	12/2004	Cortright	715/769
6,907,447	B1 *	6/2005	Cooperman et al.	709/203
7,046,254	B2	5/2006	Brown et al.	
7,068,266	B1	6/2006	Ruelle	
7,155,729	B1	12/2006	Andrew et al.	
7,342,594	B1	3/2008	Ort et al.	
2002/0191028	A1	12/2002	Senechalle et al.	
2003/0001899	A1	1/2003	Partanen et al.	
2003/0016253	A1	1/2003	Aoki et al.	
2003/0043197	A1	3/2003	Kremer et al.	
2003/0051228	A1	3/2003	Martinez et al.	
2003/0145060	A1	7/2003	Martin	

* cited by examiner

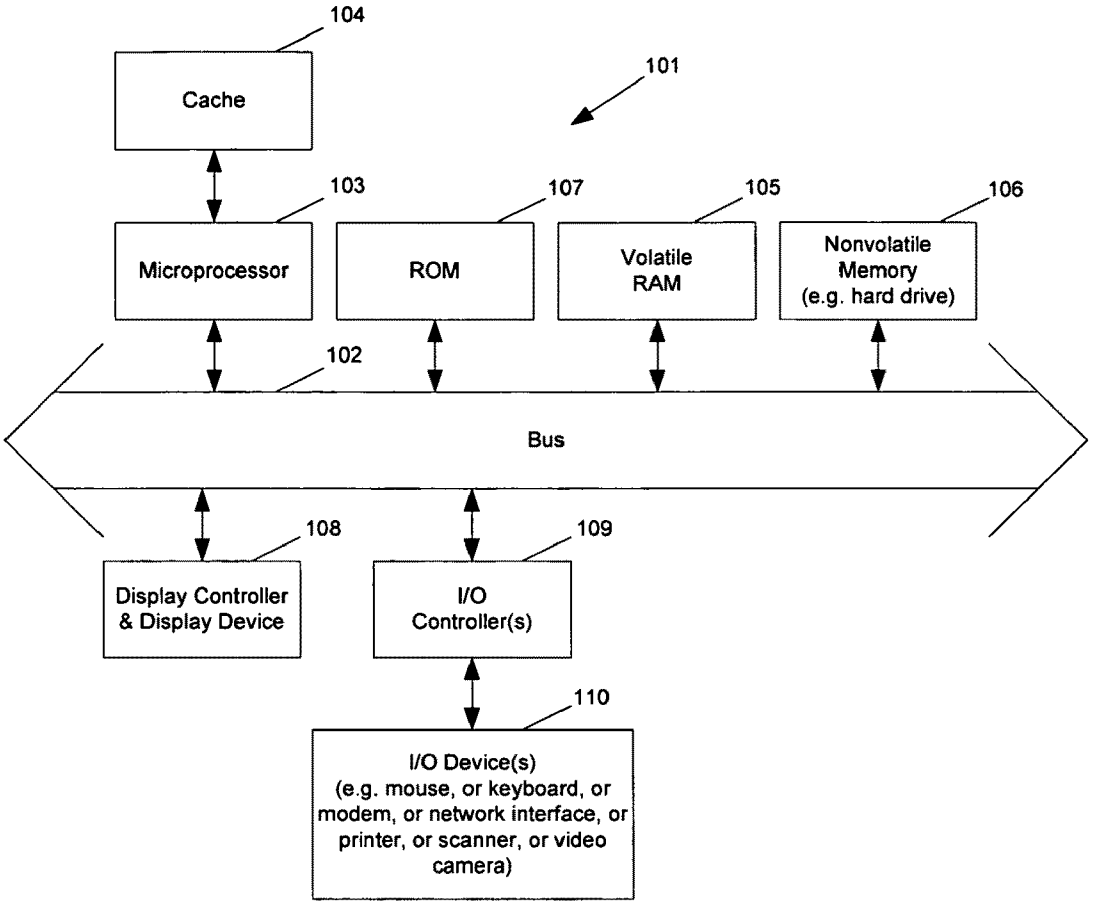

104
Cache
101
103
107
105
106
Microprocessor
ROM
Volatile RAM
Nonvolatile Memory (e.g. hard drive)
102
Bus
108
109
Display Controller & Display Device
I/O Controller(s)
110
I/O Device(s) (e.g. mouse, or keyboard, or modem, or network interface, or printer, or scanner, or video camera)

Fig. 1

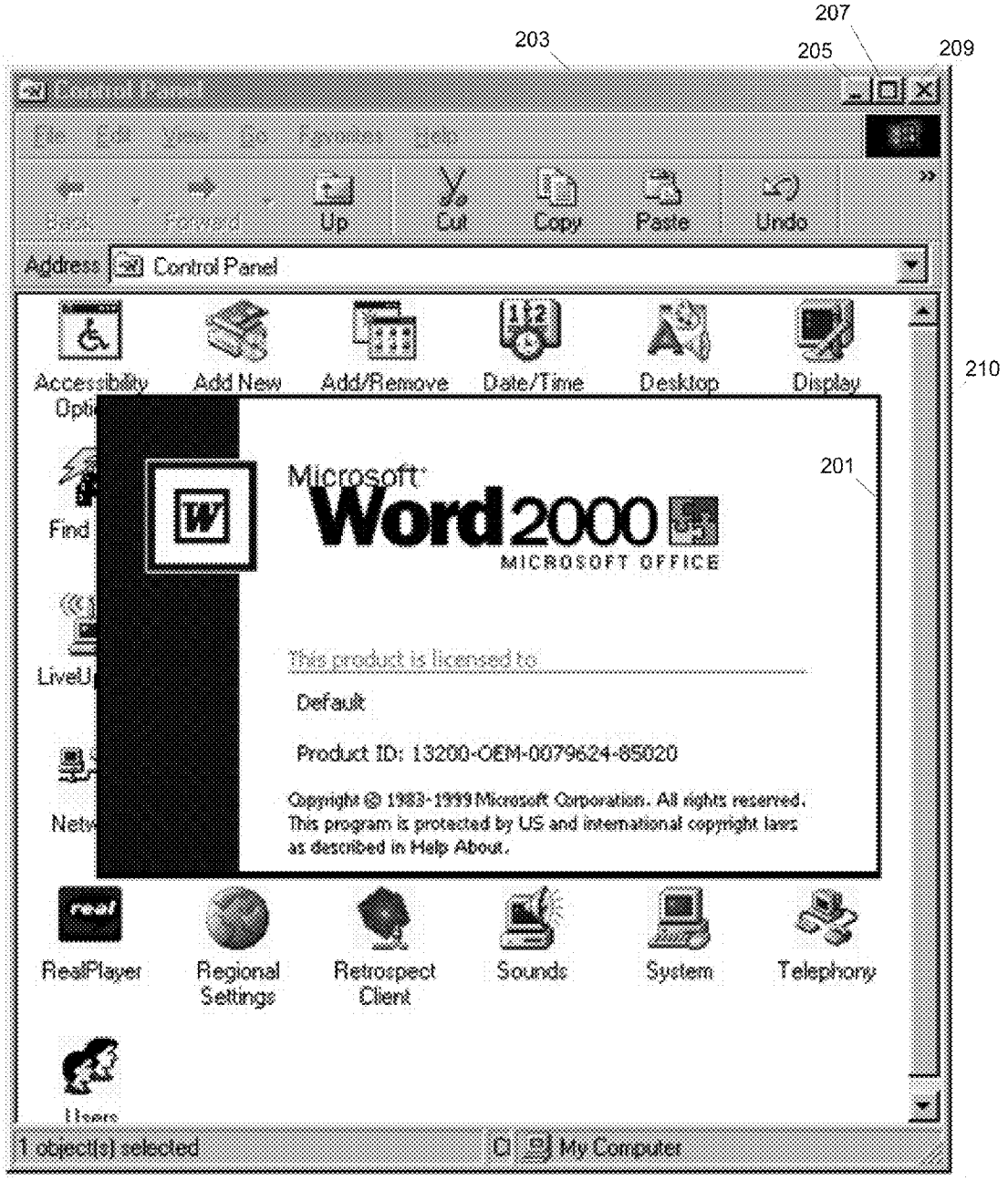


Fig. 2

Prior Art

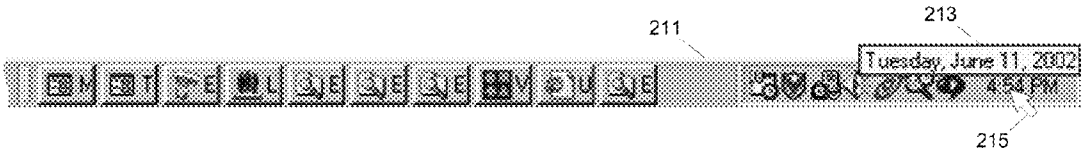


Fig. 3

Prior Art

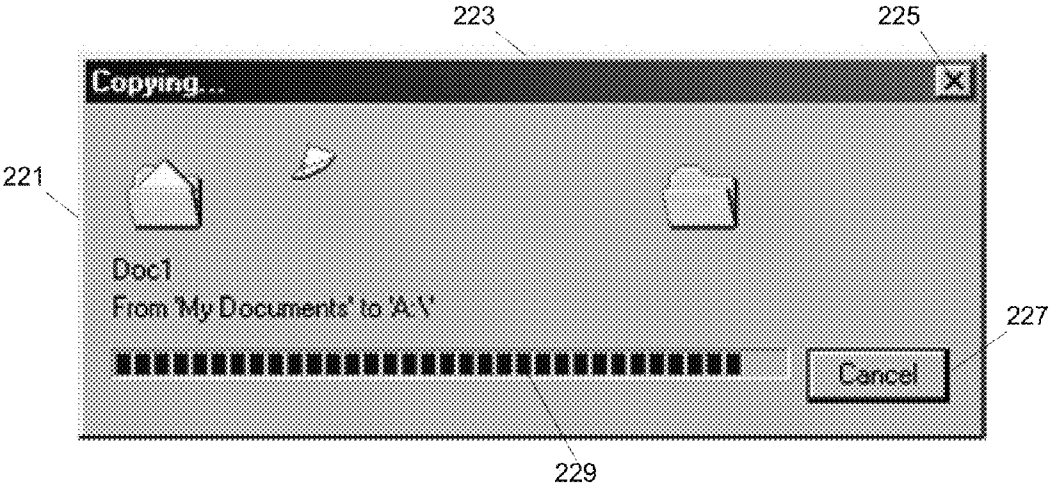


Fig. 4

Prior Art

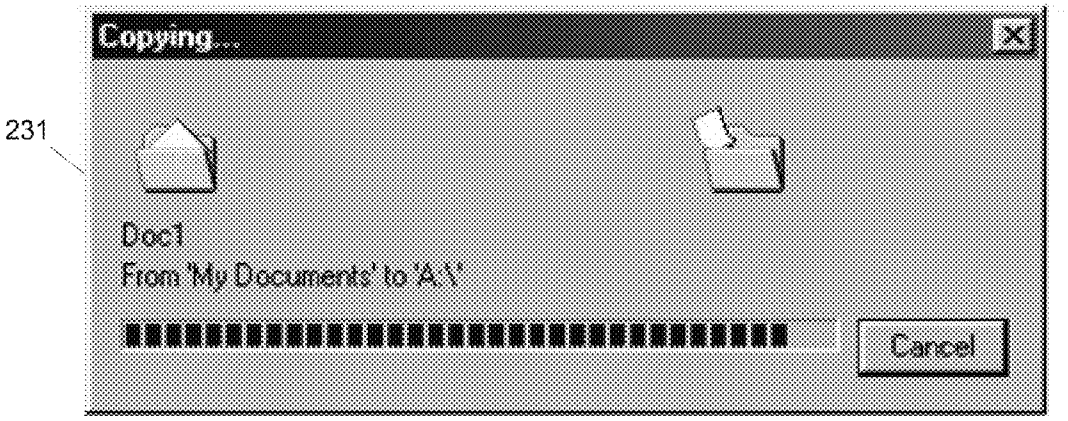


Fig. 5

Prior Art

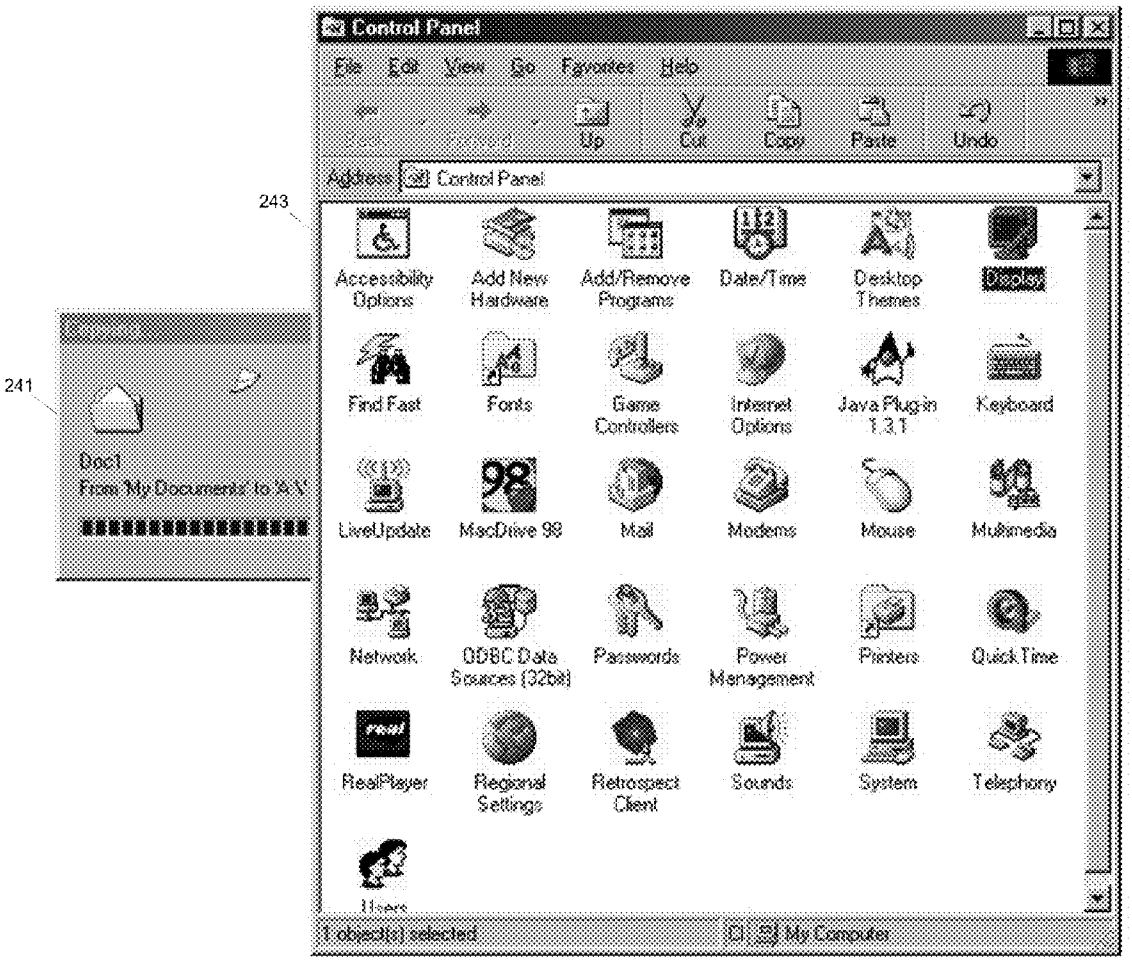


Fig. 6

Prior Art

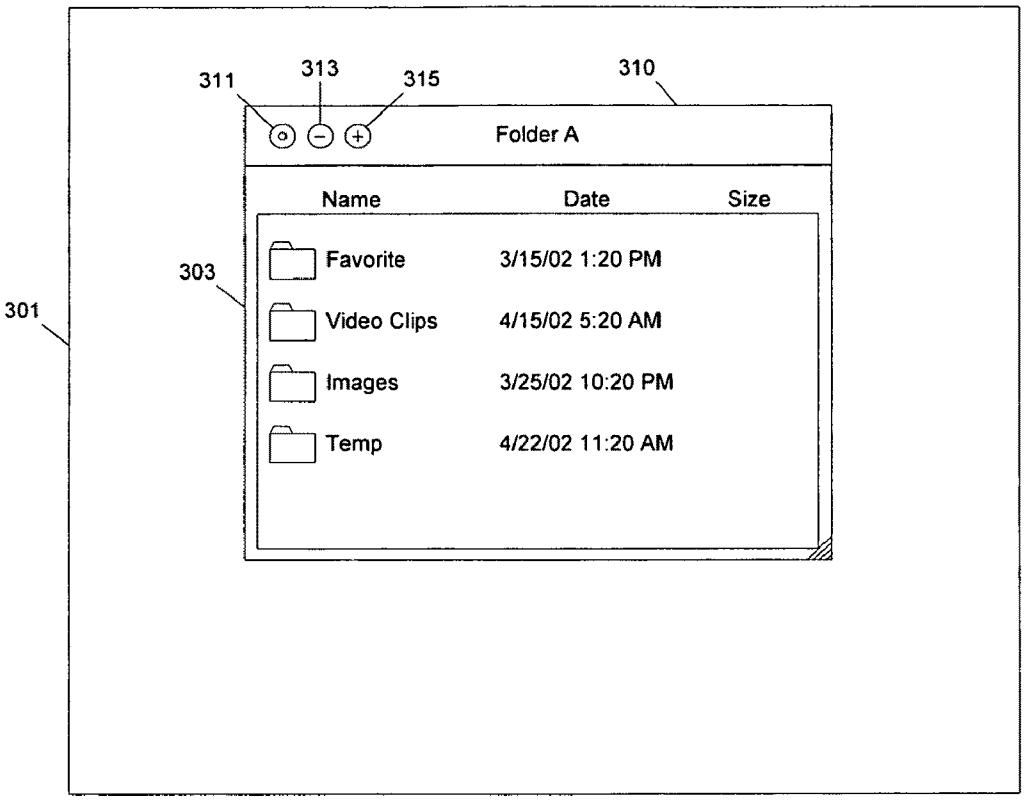

311
313
315
310
Folder A
Name
Date
Size
303
301
Favorite
3/15/02 1:20 PM
Video Clips
4/15/02 5:20 AM
Images
3/25/02 10:20 PM
Temp
4/22/02 11:20 AM

Fig. 7

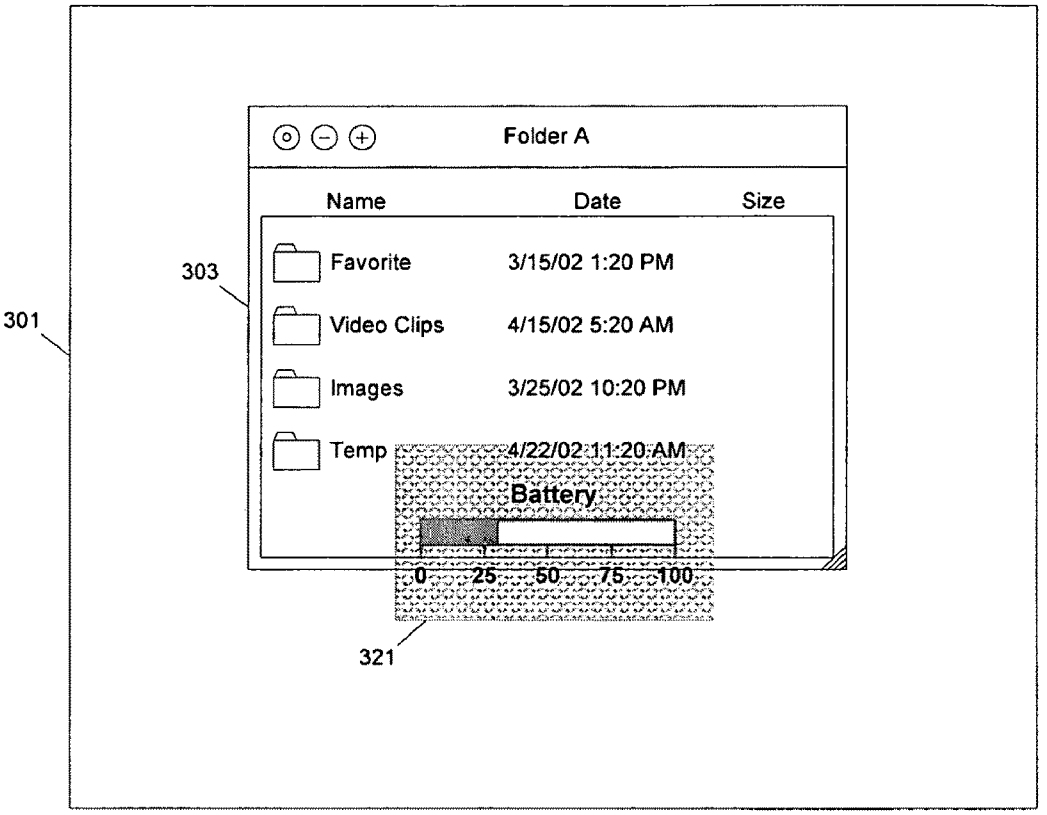

Folder A
Name
Date
Size
Favorite
3/15/02 1:20 PM
Video Clips
4/15/02 5:20 AM
Images
3/25/02 10:20 PM
Temp
4/22/02 11:20 AM
Battery
0
25
50
75
100
301
303
321

Fig. 8

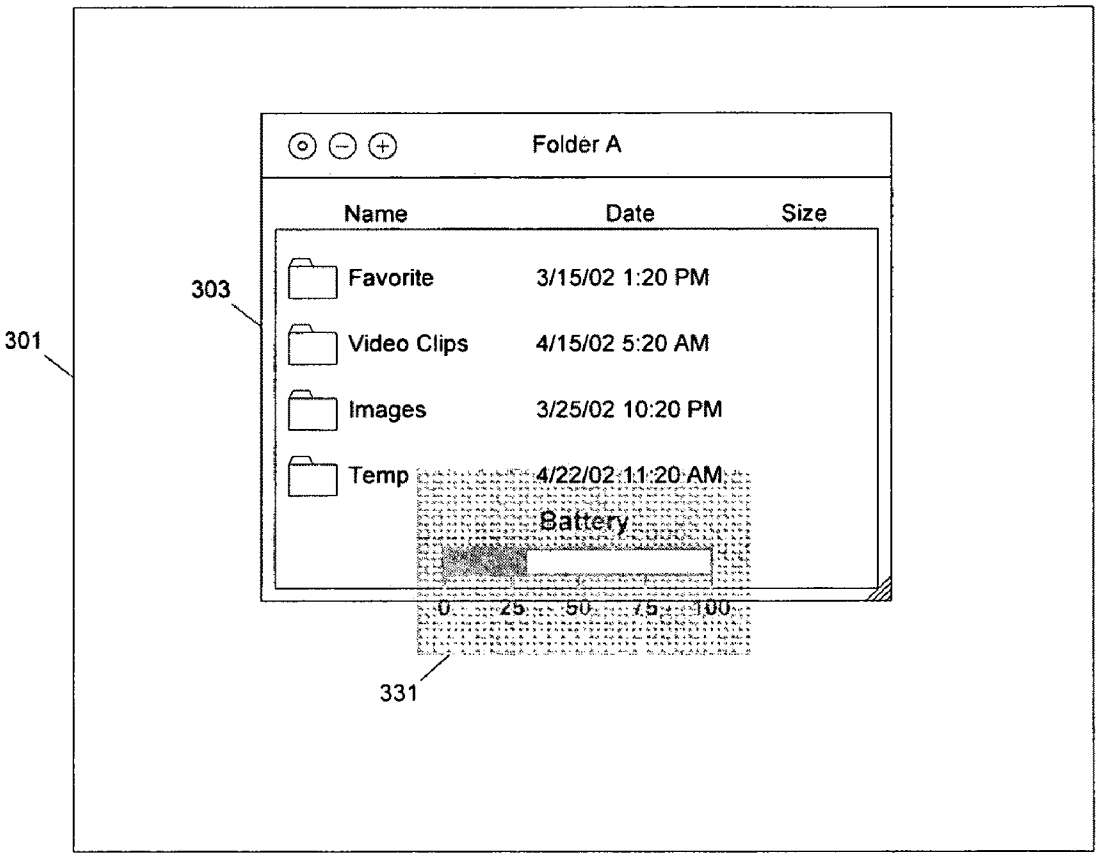

Folder A
Name
Date
Size
Favorite
3/15/02 1:20 PM
Video Clips
4/15/02 5:20 AM
Images
3/25/02 10:20 PM
Temp
4/22/02 11:20 AM
Battery
0
25
50
75
100
301
303
331

Fig. 9

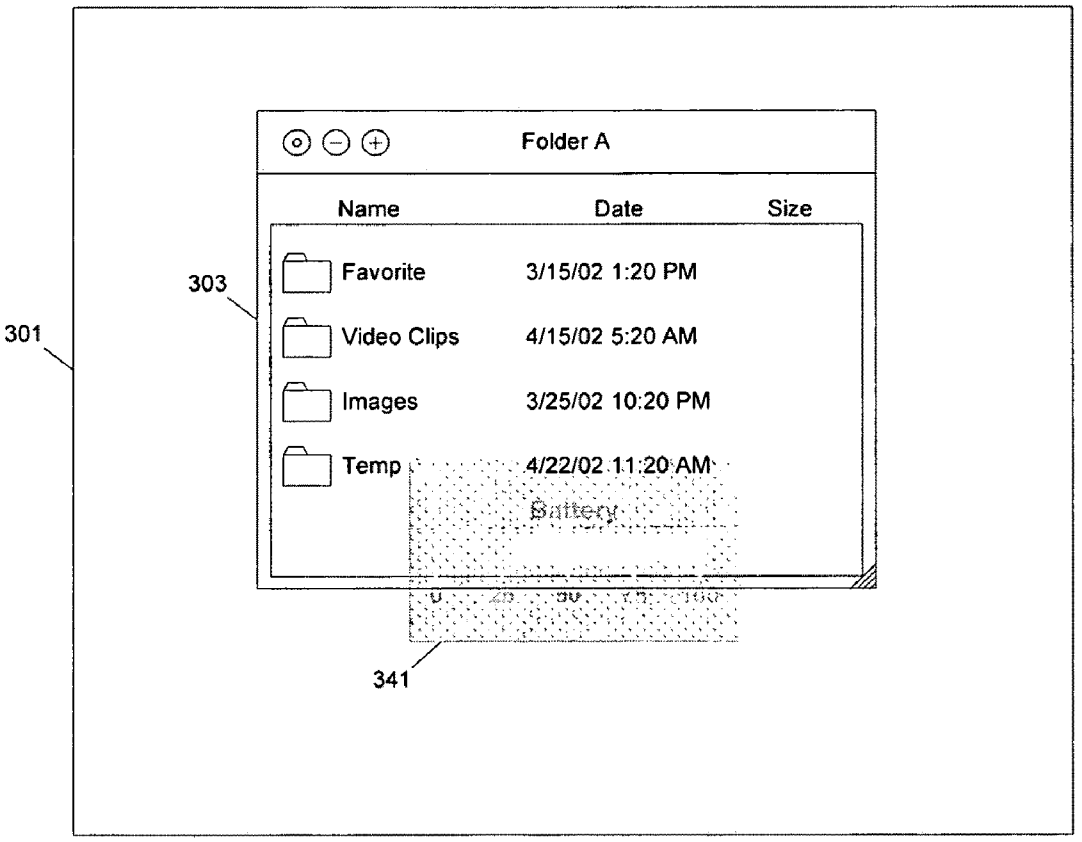

Folder A
Name
Date
Size
Favorite
3/15/02 1:20 PM
Video Clips
4/15/02 5:20 AM
Images
3/25/02 10:20 PM
Temp
4/22/02 11:20 AM
Battery
301
303
341

Fig. 10

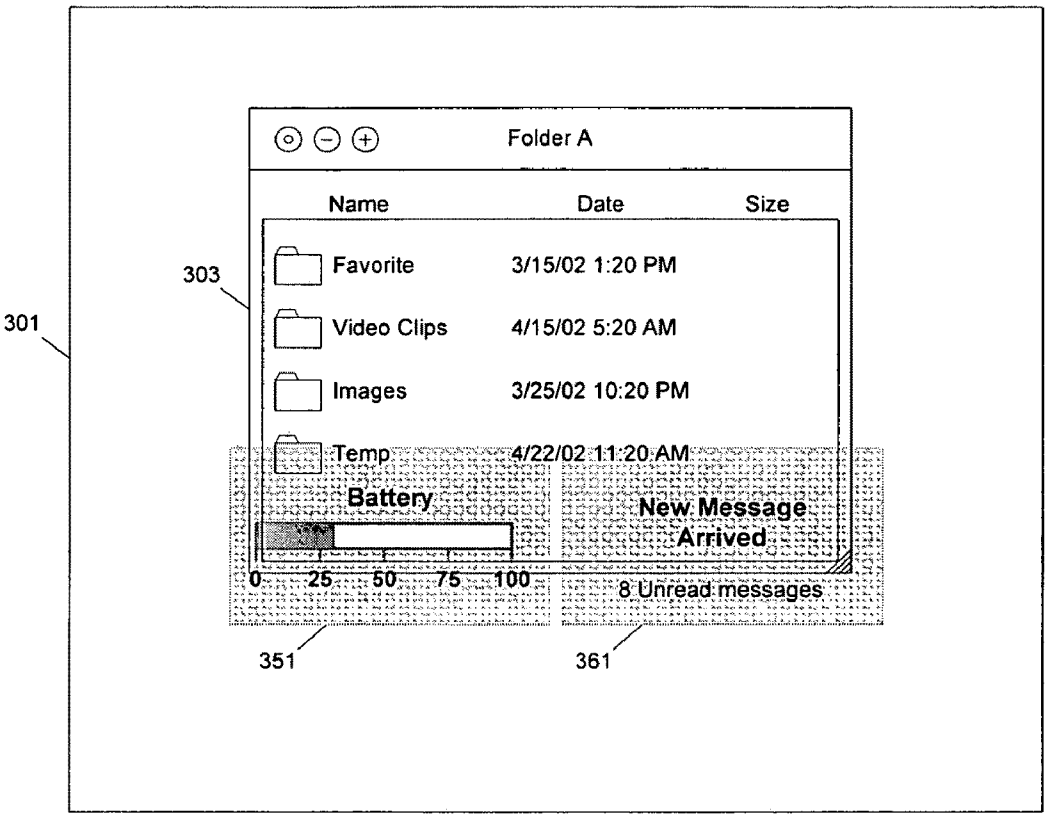

Folder A
Name
Date
Size
Favorite
3/15/02 1:20 PM
Video Clips
4/15/02 5:20 AM
Images
3/25/02 10:20 PM
Temp
4/22/02 11:20 AM
Battery
0
25
50
75
100
New Message
Arrived
8 Unread messages
301
303
351
361

Fig. 11

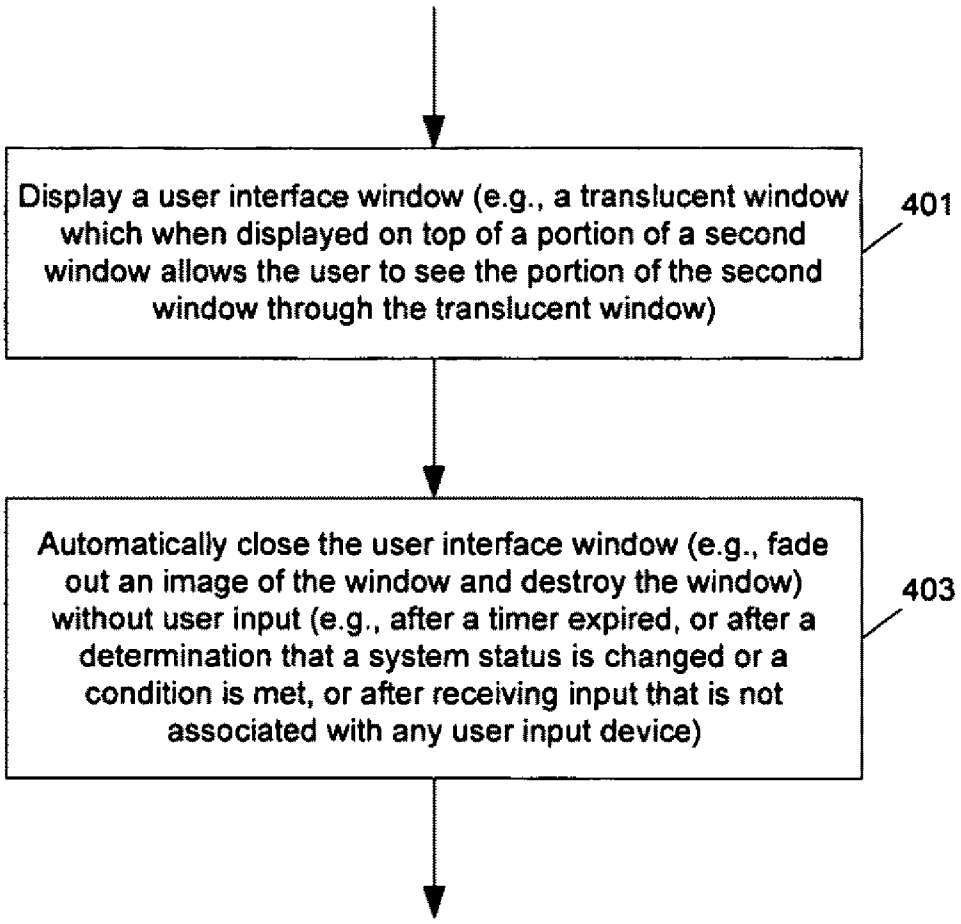

Display a user interface window (e.g., a translucent window which when displayed on top of a portion of a second window allows the user to see the portion of the second window through the translucent window)
401
Automatically close the user interface window (e.g., fade out an image of the window and destroy the window) without user input (e.g., after a timer expired, or after a determination that a system status is changed or a condition is met, or after receiving input that is not associated with any user input device)
403

Fig. 12

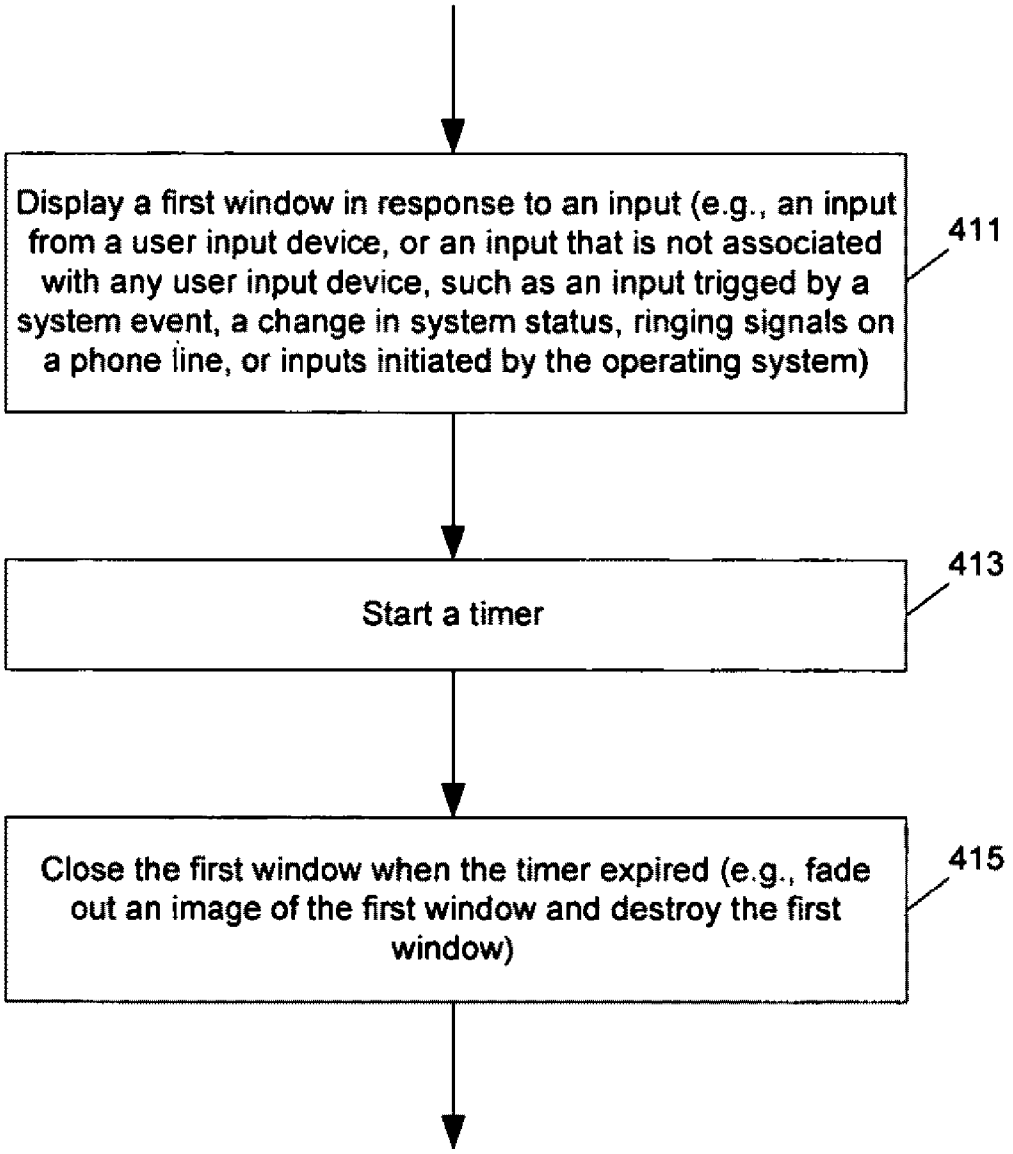

Display a first window in response to an input (e.g., an input from a user input device, or an input that is not associated with any user input device, such as an input trigged by a system event, a change in system status, ringing signals on a phone line, or inputs initiated by the operating system)
411
Start a timer
413
Close the first window when the timer expired (e.g., fade out an image of the first window and destroy the first window)
415

Fig. 13

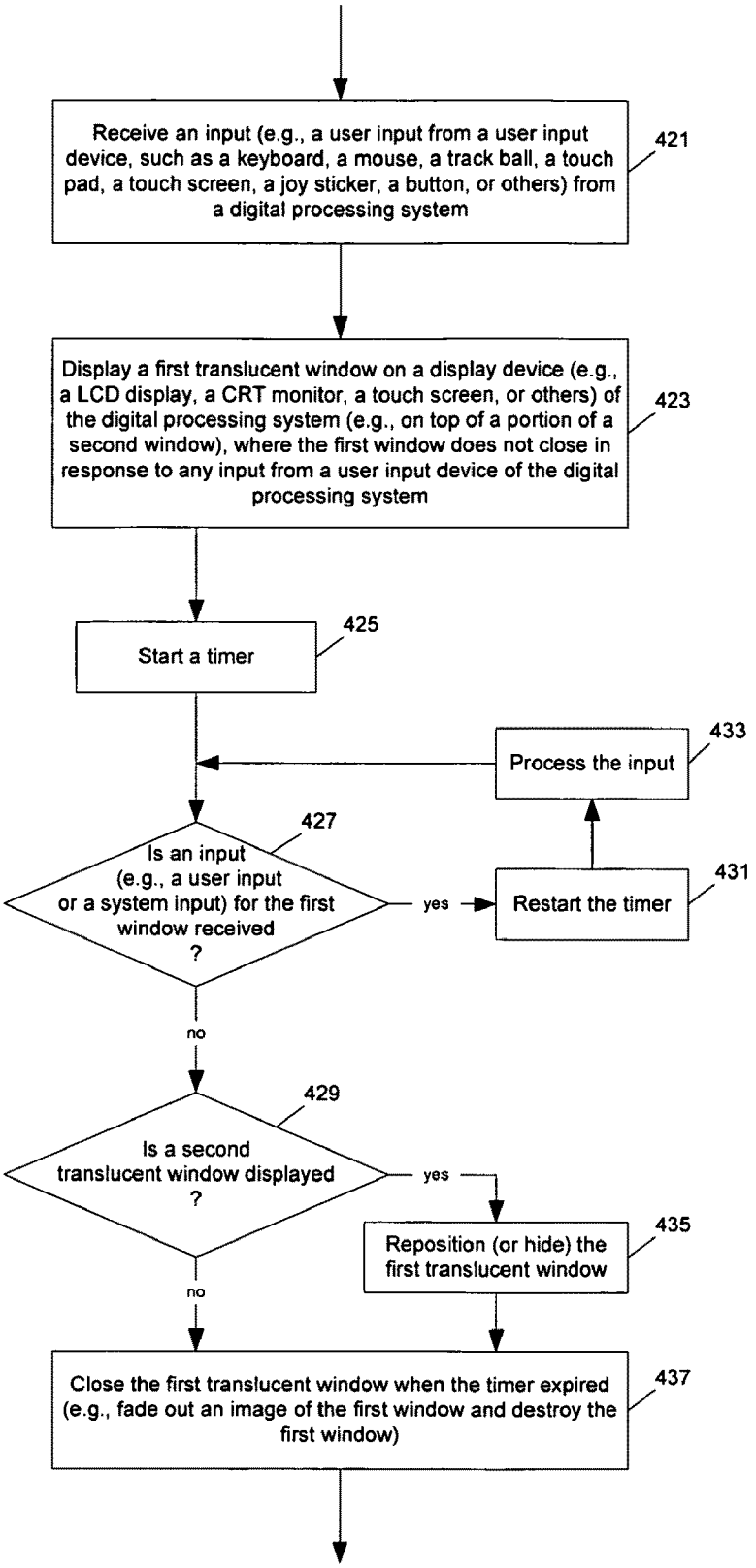

Receive an input (e.g., a user input from a user input device, such as a keyboard, a mouse, a track ball, a touch pad, a touch screen, a joy sticker, a button, or others) from a digital processing system
421
Display a first translucent window on a display device (e.g., a LCD display, a CRT monitor, a touch screen, or others) of the digital processing system (e.g., on top of a portion of a second window), where the first window does not close in response to any input from a user input device of the digital processing system
423
Start a timer
425
Process the input
433
Is an input (e.g., a user input or a system input) for the first window received ?
427
yes
Restart the timer
431
no
Is a second translucent window displayed ?
429
yes
Reposition (or hide) the first translucent window
435
no
Close the first translucent window when the timer expired (e.g., fade out an image of the first window and destroy the first window)
437

Fig. 14

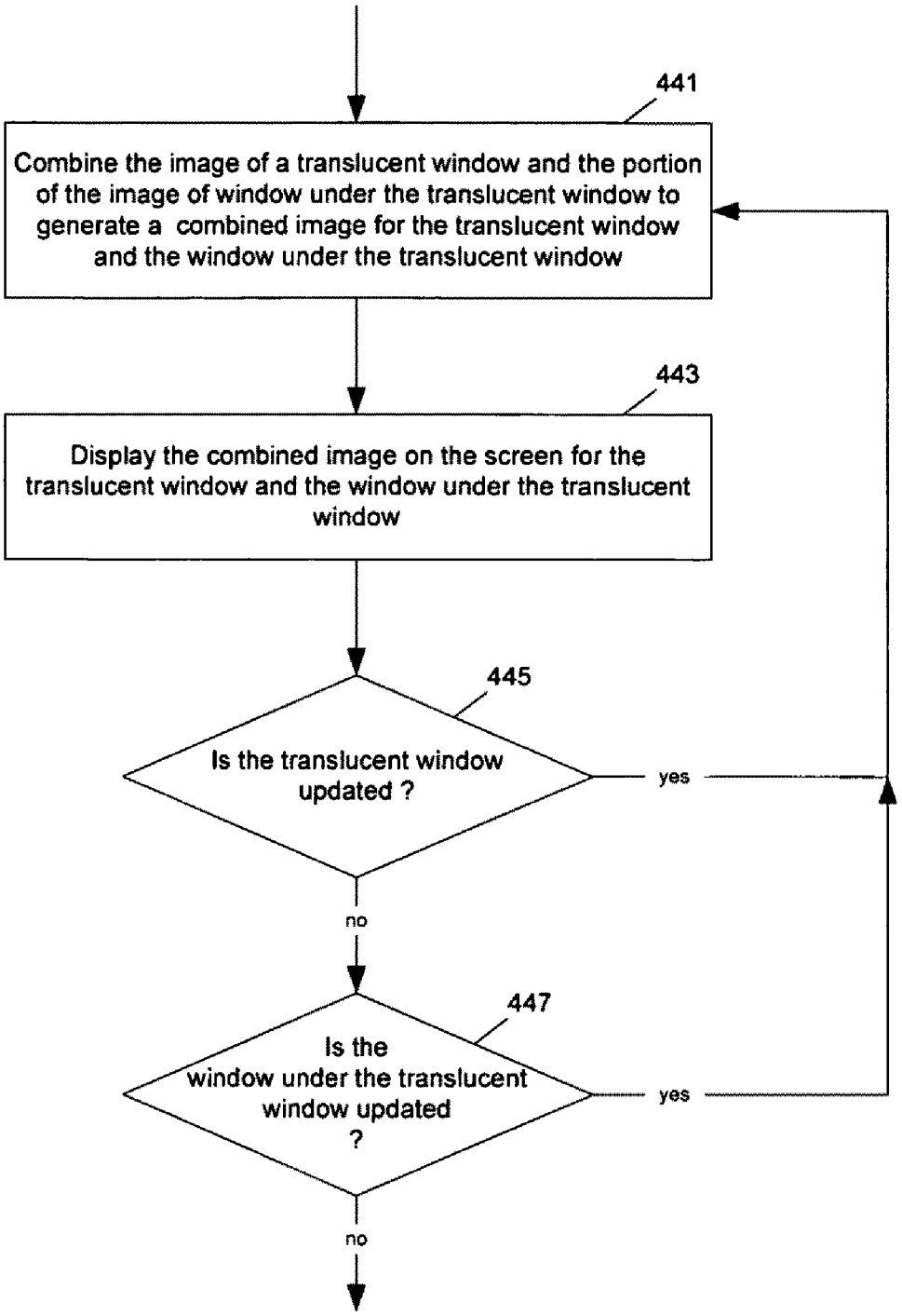

441
Combine the image of a translucent window and the portion of the image of window under the translucent window to generate a combined image for the translucent window and the window under the translucent window
443
Display the combined image on the screen for the translucent window and the window under the translucent window
445
Is the translucent window updated ?
yes
no
447
Is the window under the translucent window updated ?
yes
no

Fig. 15

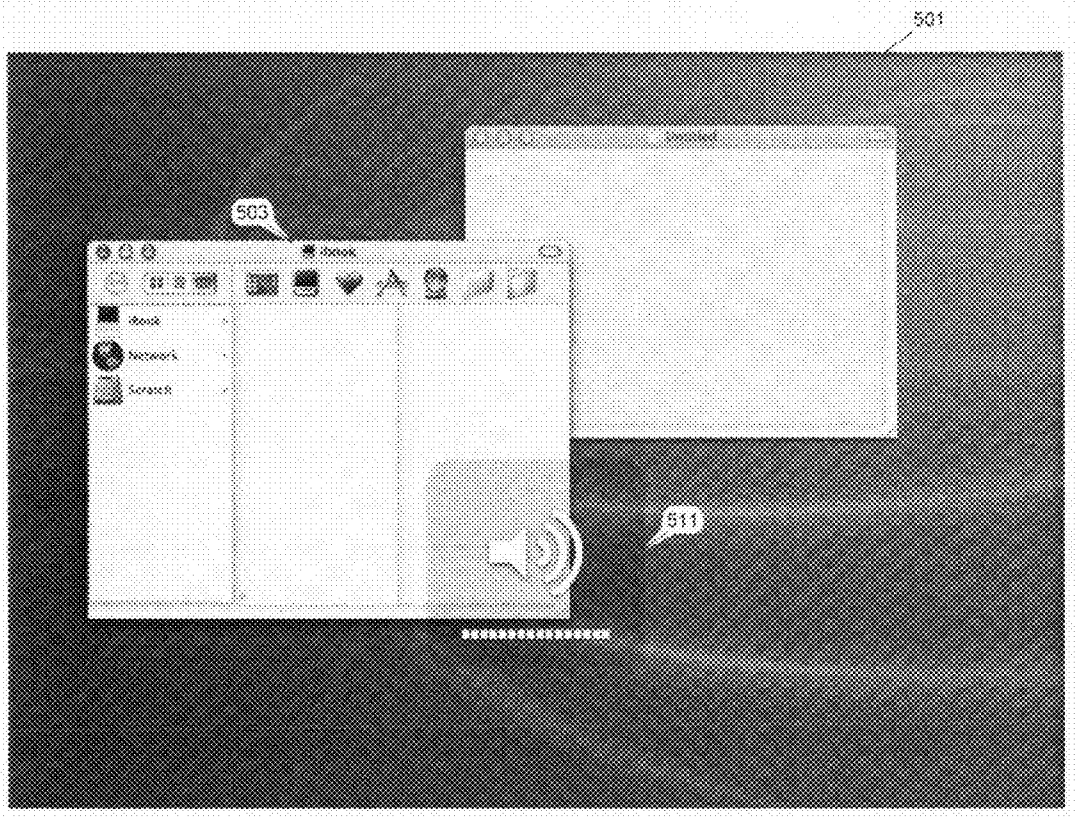


Fig. 16

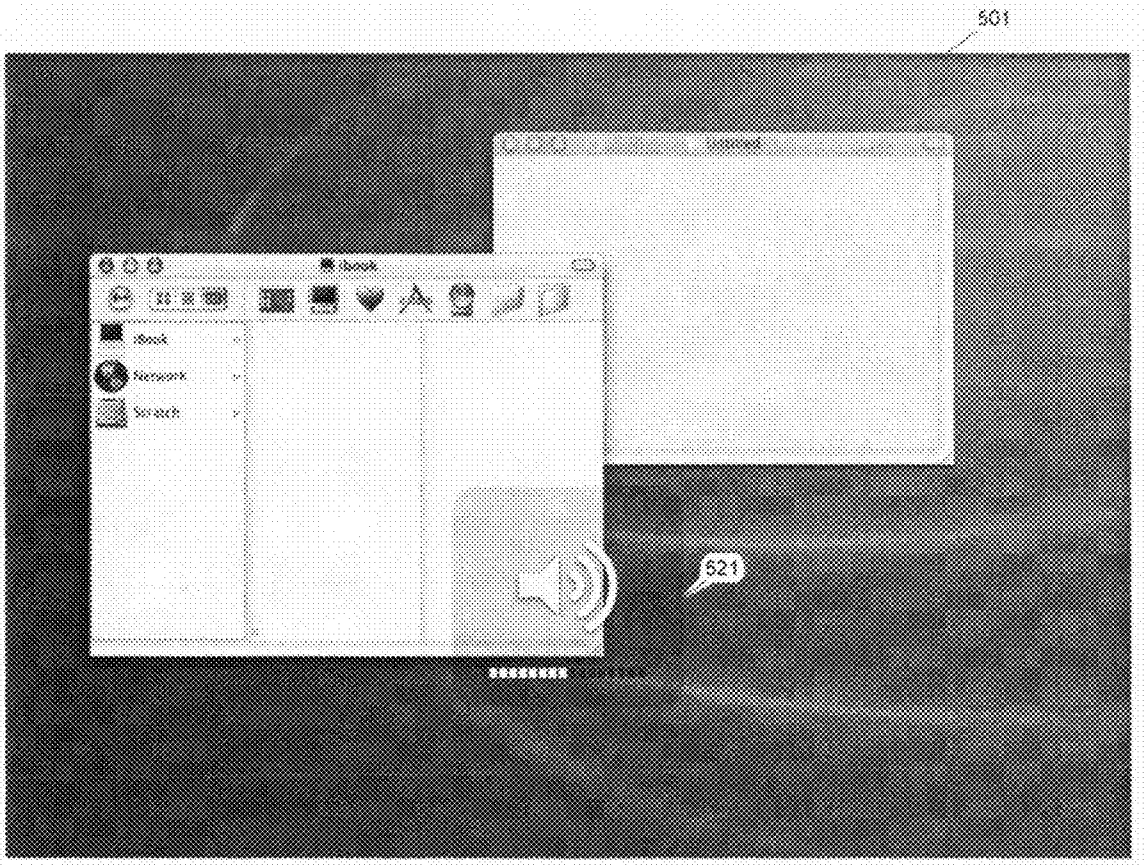


Fig. 17

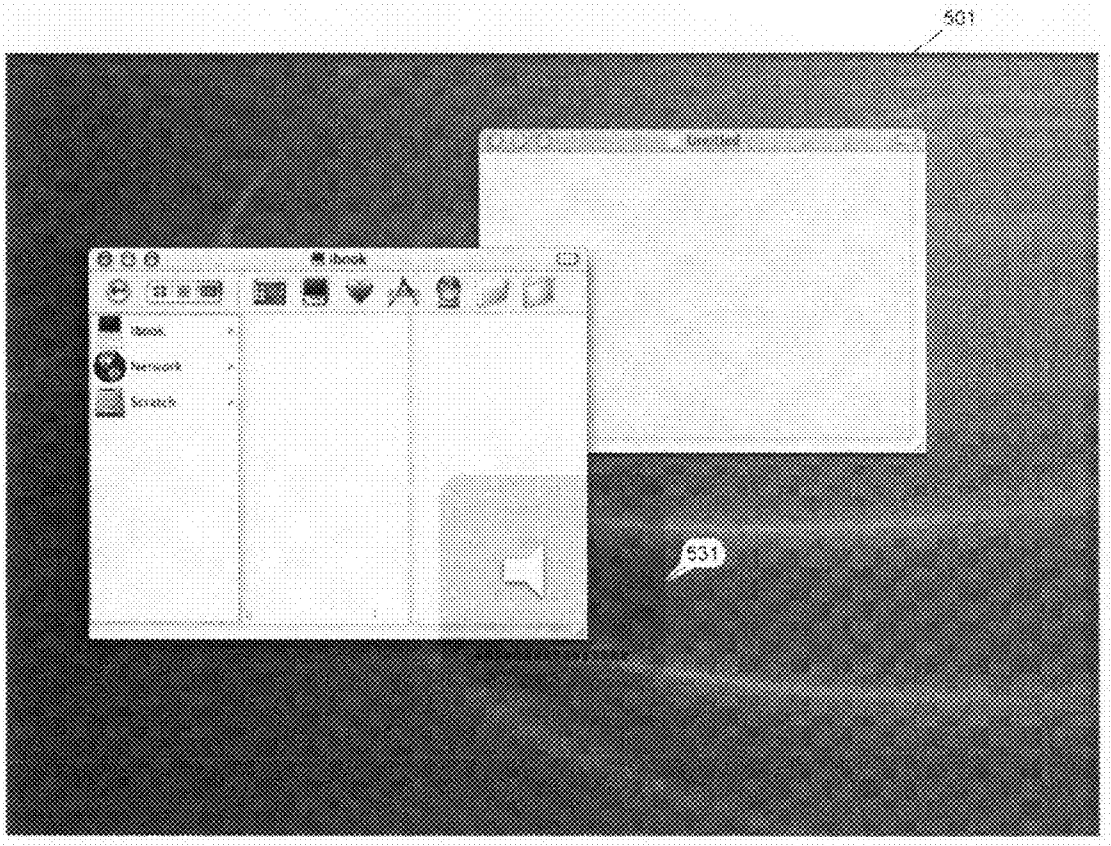


Fig. 18

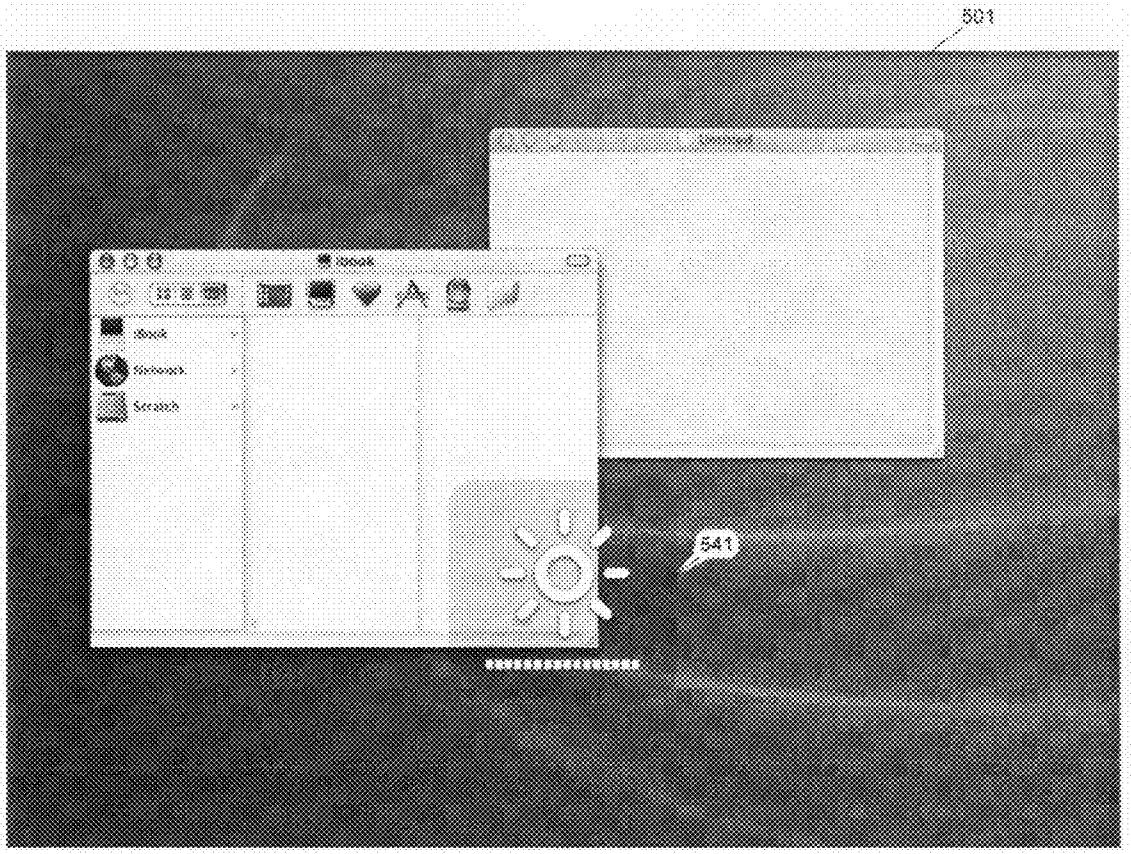


Fig. 19

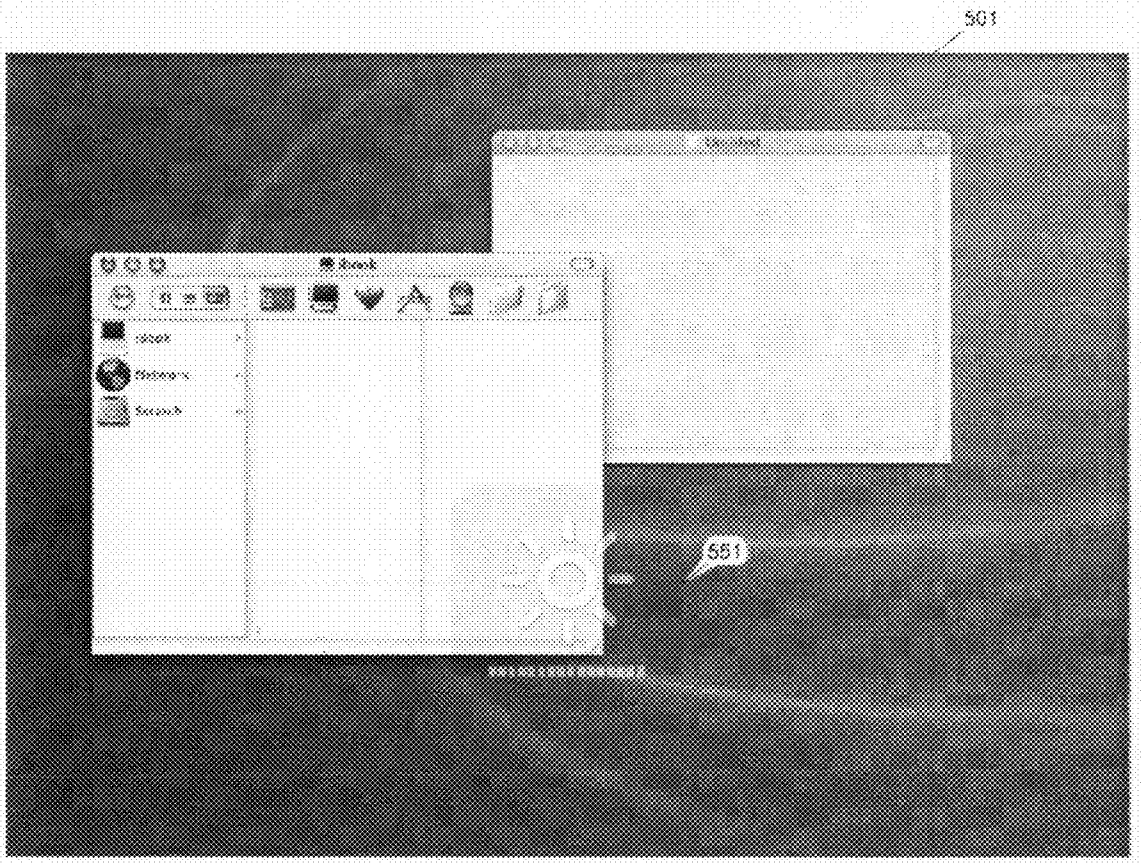


Fig. 20

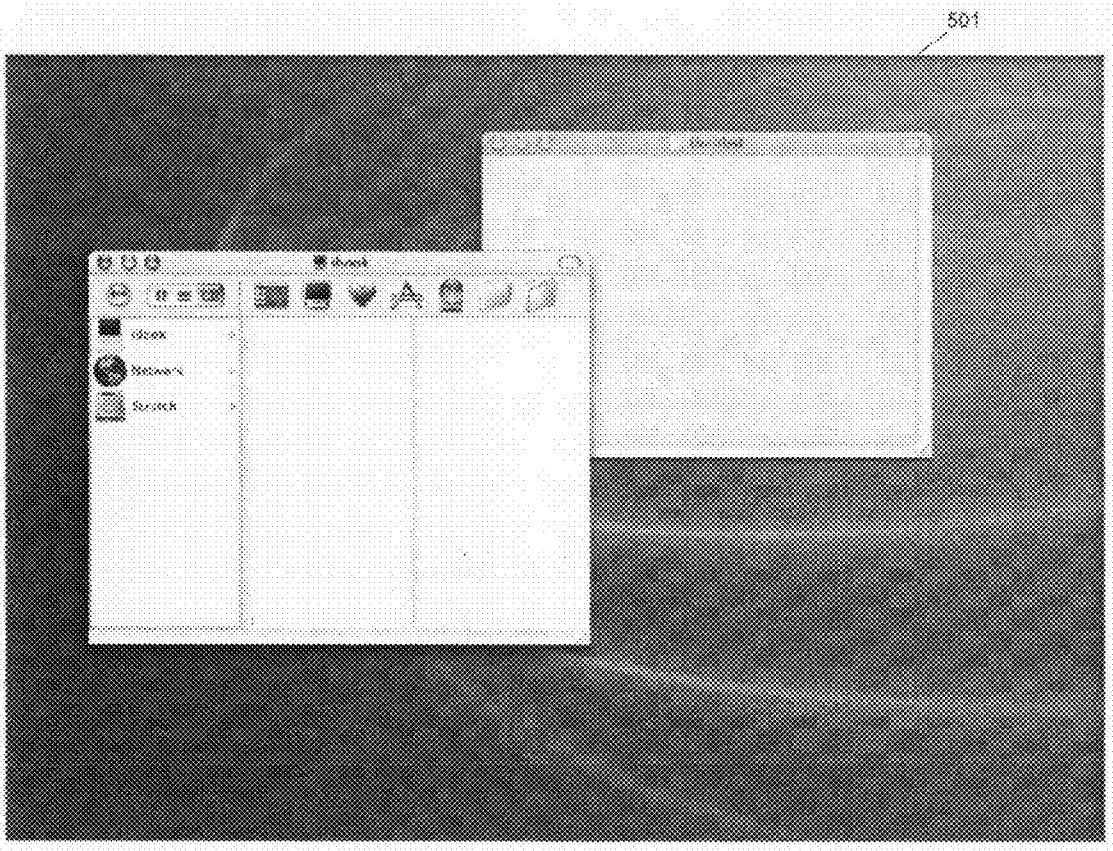


Fig. 21

METHOD AND APPARATUS FOR DISPLAYING A WINDOW FOR A USER INTERFACE

The present application is a continuation of co-pending U.S. application Ser. No. 11/635,847, filed Dec. 8, 2006, which is a continuation of U.S. application Ser. No. 10/193,573, filed Jul. 10, 2002 now U.S. Pat. No. 7,343,566.

FIELD OF THE INVENTION

The invention relates to graphical user interfaces, and more particularly to such interfaces with windows.

BACKGROUND OF THE INVENTION

Many digital processing systems use window-based graphical user interfaces for users to interact with software programs. Images of windows are displayed on a display device to show the user the states of the software programs; and user input devices (e.g., a keyboard and a mouse) are used to accept user inputs. In addition to user input devices, a digital processing system may have other devices (e.g., sensors) for accepting system inputs, such as phone line status, power supply status, storage disk usage status, communication connection status, execution status of software programs, and others that are not directly related to user inputs (signals associated with user input devices).

FIGS. **2**-**6** illustrate examples of traditional windows. FIG. **2** shows a typical window **210** that has title bar **203** and buttons **205**, **207** and **209** for minimizing, maximizing, and closing the window. The title bar and the buttons on the title bar can be used to manipulating the position and size of the window. For example, title bar **203** may be clicked (e.g., pressing a button of a mouse while the cursor is on the title bar) to bring the window to the top level of the window displaying hierarchy so that if there are any other windows displayed at the same location of window **210**, these windows will be hidden under window **210**.

When a user starts an application program, a window (e.g., window **201**) may be displayed to show the license/copyright information while the components of the software program are being loaded. After the components of the software program are fully loaded, the license/copyright window is closed automatically so that the user can start to use the software program without interference from the license/copyright window.

FIG. **3** shows task bar **211** with flash help window **213**. When the user pauses cursor **215** at a location of the task bar for a short period of time, flash help window **213** appears. If the user does not move the cursor for another short period of time while window **213** is displayed, flash window **213** disappears. If the user moves cursor **215** slightly (e.g., using a mouse, a track ball, or a touch pad) and pauses the cursor **215** again, flash help window may appear again.

FIGS. **4**-**6** show a window that displays the progress of copying a file. When a file is copied from one location to another location, window **221** is displayed to indicate the progress. Button **227** is provided for canceling the copy operation; and button **225** is for closing the progress window.

The progress of the operation is indicated by progress bar **229** and an animation showing that the document is going from one folder to another. Windows **221** and **231** in FIGS. **4** and **5** show two snap shots of the animation. A user may drag title bar **223** (e.g., pressing down and holding a button of a mouse and moving the mouse while holding down the button) to drag the window from one location on a screen to another; and the user can click on the title bar to bring the window to the top level when the window is partially covered by another window (e.g., when window **241** is partially covered by window **243**, as shown in FIG. **6**). When the copy operation completes, the progress window closes automatically.

Traditional windows typically provide strong user interactions, which may cause distractions. For example, a user waits for window **201** to disappear to view window **210** in FIG. **2**; the user manipulates a cursor control device (e.g., a mouse, a track ball, or a touch pad) to view or dismiss flash help window **213** in FIG. **3**; and, the user interaction is provided to relocate the progress window or change the window displaying hierarchy to see the progress of window **241** in FIG. **6**.

SUMMARY OF THE INVENTION

Methods and apparatuses to display windows are described here. There are many different embodiments which are described here. Some of these embodiments are summarized in this section.

In more than one embodiment of the invention, a window is closed automatically (e.g., after a timer expires, or when a condition or criterion is met, or system input is received) without user input. In some examples, the window is translucent so that the portion of another window, when present, is visible under the window. In some examples, the image of the window is faded out before the window is closed and destroyed. In a further example, the level of translucency, the speed for fading out, the discrete levels of translucency for fading out, the time to expire, and/or other parameters for controlling the display of the window may be set by the user or adjusted by the system (or application software programs) automatically according to system conditions or other criteria.

In one embodiment of the invention, a method to display a user interface window for a digital processing system includes: displaying a first window in response to receiving a first input from a user input device (e.g., a keyboard, mouse, track ball, touch pad, touch screen, joy stick, button, or others) of the digital processing system which is capable of displaying at least a portion of a second window under the first window; starting a timer; and closing the first window in response to a determination that the timer expired. The first window does not close in response to any input from a user input device of the digital processing system. In one example according to this embodiment, the first window is translucent; the portion of the second window is visible while under the first window; and the first window is at a top level in a window displaying hierarchy. In one example, an image of the first window is faded out on the screen before the first window is destroyed to close the first window. In one example, the second window, if displayed, closes in response to an input from a user input device of the digital processing system; and the first window does not respond to any input from a user input device of the digital processing system. In one example, the first window is repositioned in response to a third window (e.g., an alert window or a translucent window) being displayed; in another example, the first window is hidden in response to a third window being displayed at a location where the first window is displayed. In one example, the first

window is repositioned on a display in response to a second input for the first window (e.g., an input indicating that a third window is displayed, or an input from a user input device of the digital processing system to reposition the window, such as dragging and dropping the window); and a position of the first window in a window displaying hierarchy can be adjusted in response to a third input (e.g., bringing another window in front of the first window). In one example, the first window is displayed at a position on a display of the digital processing system that is independent of a position of a cursor on the display (e.g., a position centered horizontally on the display); and the timer is restarted in response to receiving a second input for the first window (e.g., from a user input device of the digital processing system).

In another embodiment of the invention, a method to display a user interface window for a digital processing system includes: displaying a first translucent window such that if a portion of a second window is displayed on the digital processing system under the first window, the portion of the second window is visible under the first window; and closing the first window without user input. In one example according to this embodiment, a timer is started so that when the timer expires the first window is closed (e.g., fading out an image of the first window and destroy the first window). In another example, the first window is closed in response to an input that is not associated with a user input device of the digital processing system. In a further example, the first window is closed in response to a determination that a system condition is met (e.g., a system status is changed, or other criteria).

In a further embodiment of the invention, a method to display a user interface window for a digital processing system includes: displaying a first window in response to receiving a first input that is not associated with a user input device of the digital processing system; starting a timer; and closing the first window in response to a determination that the timer expired (e.g., fading out an image of the first window; and destroying the first window). In one example, the first window does not close in response to any input from a user input device of the digital processing system (e.g., the first window does not respond to any input from a user input device of the digital processing system); and the first window is translucent such that a portion of a second window is visible when displayed under the first window. In one example, the first window is repositioned on a display without user input (e.g., in response to a third window being displayed). In another example, the timer is restarted in response to receiving a second input for the first window; and the second input is received from a user input device of the digital processing system.

The present invention includes apparatuses which perform these methods, including data processing systems which perform these methods and computer readable media which when executed on data processing systems cause the systems to perform these methods.

Other features of the present invention will be apparent from the accompanying drawings and from the detailed description which follow.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings in which like references indicate similar elements.

FIG. **1** shows a block diagram example of a data processing system which may be used with the present invention.

FIGS. **2-6** illustrate examples of traditional windows.

FIGS. **7-11** illustrate example scenarios of displaying a window according to one embodiment of the present invention.

FIG. **12** shows a flow diagram of a method to display a window according to one embodiment of the present invention.

FIG. **13** shows a flow diagram of a method to close a window according to one embodiment of the present invention.

FIG. **14** shows a detailed flow diagram of a method to control a translucent window according to one embodiment of the present invention.

FIG. **15** shows a method to display a translucent window according to one embodiment of the present invention.

FIGS. **16-21** show example screen images of windows displayed according to one embodiment of the present invention.

DETAILED DESCRIPTION

The following description and drawings are illustrative of the invention and are not to be construed as limiting the invention. Numerous specific details are described to provide a thorough understanding of the present invention. However, in certain instances, well known or conventional details are not described in order to avoid obscuring the description of the present invention.

Many of the methods of the present invention may be performed with a digital processing system, such as a conventional, general purpose computer system. Special purpose computers which are designed or programmed to perform only one function may also be used.

FIG. **1** shows one example of a typical computer system which may be used with the present invention. Note that while FIG. **1** illustrates various components of a computer system, it is not intended to represent any particular architecture or manner of interconnecting the components as such details are not germane to the present invention. It will also be appreciated that network computers and other data processing systems which have fewer components or perhaps more components may also be used with the present invention. The computer system of FIG. **1** may, for example, be an Apple Macintosh computer.

As shown in FIG. **1**, the computer system **101**, which is a form of a data processing system, includes a bus **102** which is coupled to a microprocessor **103** and a ROM **107** and volatile RAM **105** and a non-volatile memory **106**. The microprocessor **103**, which may be, for example, a G3 or G4 microprocessor from Motorola, Inc. or IBM is coupled to cache memory **104** as shown in the example of FIG. **1**. The bus **102** interconnects these various components together and also interconnects these components **103**, **107**, **105**, and **106** to a display controller and display device **108** and to peripheral devices such as input/output (I/O) devices which may be mice, keyboards, modems, network interfaces, printers, scanners, video cameras and other devices which are well known in the art. Typically, the input/output devices **110** are coupled to the system through input/output controllers **109**. The volatile RAM **105** is typically implemented as dynamic RAM (DRAM) which requires power continually in order to refresh or maintain the data in the memory. The non-volatile memory **106** is typically a magnetic hard drive or a magnetic optical drive or an optical drive or a DVD RAM or other type of memory systems which maintain data even after power is removed from the system. Typically, the non-volatile memory will also be a random access memory although this is not required. While FIG. **1** shows that the non-volatile memory is

a local device coupled directly to the rest of the components in the data processing system, it will be appreciated that the present invention may utilize a non-volatile memory which is remote from the system, such as a network storage device which is coupled to the data processing system through a network interface such as a modem or Ethernet interface. The bus **102** may include one or more buses connected to each other through various bridges, controllers and/or adapters as is well known in the art. In one embodiment the I/O controller **109** includes a USB (Universal Serial Bus) adapter for controlling USB peripherals, and/or an IEEE-1394 bus adapter for controlling IEEE-1394 peripherals.

It will be apparent from this description that aspects of the present invention may be embodied, at least in part, in software. That is, the techniques may be carried out in a computer system or other data processing system in response to its processor, such as a microprocessor, executing sequences of instructions contained in a memory, such as ROM **107**, volatile RAM **105**, non-volatile memory **106**, cache **104** or a remote storage device. In various embodiments, hardwired circuitry may be used in combination with software instructions to implement the present invention. Thus, the techniques are not limited to any specific combination of hardware circuitry and software nor to any particular source for the instructions executed by the data processing system. In addition, throughout this description, various functions and operations are described as being performed by or caused by software code to simplify description. However, those skilled in the art will recognize what is meant by such expressions is that the functions result from execution of the code by a processor, such as the microprocessor **103**.

A machine readable media can be used to store software and data which when executed by a data processing system causes the system to perform various methods of the present invention. This executable software and data may be stored in various places including for example ROM **107**, volatile RAM **105**, non-volatile memory **106** and/or cache **104** as shown in FIG. **1**. Portions of this software and/or data may be stored in any one of these storage devices.

Thus, a machine readable media includes any mechanism that provides (i.e., stores and/or transmits) information in a form accessible by a machine (e.g., a computer, network device, personal digital assistant, manufacturing tool, any device with a set of one or more processors, etc.). For example, a machine readable media includes recordable/non-recordable media (e.g., read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; etc.), as well as electrical, optical, acoustical or other forms of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.); etc.

At least one embodiment of the present invention seeks to display a window with reduced distractions so that a user can focus on more important windows.

FIGS. **7-11** illustrate example scenarios of displaying a window according to one embodiment of the present invention. Traditional window **303** is shown in FIG. **7**. Window **303** contains control buttons **311**, **313** and **315** for closing, minimizing and maximizing the window. Window **303** also has title bar **310**, which may be used to relocate the window on screen **301**. Consider a scenario where the battery power of the system is lower than a threshold. After the system detects such a system status change, window **321** may be displayed near the center of screen **301**, as shown in FIG. **8**. Window **321** is translucent so that regular window **303** is still visible under window **321**. Once window **321** is displayed on the screen, a timer is started to control the closing of the window. When the timer expires, window **321** is automatically closed without any user input. Thus, window **321** displays the message of low battery power to the user without forcing the user to provide inputs to dismiss the message window. Since window **321** is translucent and transient, the portion of window **303** that is under window **321** is still visible. Thus, the user can continue working with window **303** (or other window) without having to provide additional input to get message window **321** out of the way.

In one embodiment of the present invention, translucent window **321** is always displayed at the top level of the window displaying hierarchy so that the translucent window is always visible when displayed. This eliminates the need for the user to change the window displaying hierarchy to bring up the translucent window when another traditional window is brought up to the top of the window displaying hierarchy (e.g., by creating a new window or accidentally changing the hierarchy). In another embodiment of the present invention, the user can change the position of the translucent window in the hierarchy so that if the user desires the translucent window may be sent to a background position.

In one embodiment of the present invention, the image of window **321** is faded out when the timer expires, which is illustrated by the images of windows **321**, **331**, and **341** in FIGS. **8**, **9** and **10**. After the image of window **321** is faded out, window **321** is destroyed.

In another embodiment of the present invention, a translucent message window starts to fade out when a status change is detected. For example, a message window is displayed when the system detects the ringing signal on a phone line. When the system detects that the ringing signal is no longer present on the phone line, the image of the message window is faded out; and the message window is destroyed. Similarly, a translucent progress window for showing the progress of copying a file can be faded out and destroyed after the copy process ends. In one example, message window **361** as shown in FIG. **11** is displayed when a new message arrives. When the user starts to open an application to view the new message, message window **361** is closed automatically so that the user does not have to provide input to dismiss the message window or wait for the message window to fade out.

In one embodiment of the present invention, the image of window **321** gradually sets in when the window is first displayed. In another embodiment of the present invention, window **321** in FIG. **8** is automatically relocated or moved (e.g., in an animation fashion) to a different location so that the image of window **321** does not obscure the display of any particular portion of windows **303** for an extended period of timed. For example, window **321** may be automatically moved across the screen horizontally from the left hand side of screen **301** to the right hand side of screen **301** (or near the center of screen **321** in a circular motion).

In one embodiment of the present invention, the system detects (or manages) all the translucent windows so that when a second translucent window is displayed before the first translucent window is closed, the first translucent window is repositioned so that the second translucent window can be easily seen on the screen without interference with each other. For example, after battery low window **321** is displayed as in FIG. **8**, the system may detect a new message arrived for the user. Thus, translucent window **361** is displayed as in FIG. **11** to inform the user about the new message. At the same time, window **351** is automatically moved to a position as seen in FIG. **11** so that both translucent windows **351** and **361** can be easily seen on the screen. Alternatively, the first translucent window (e.g., window **351**) is hidden so that only the second window (e.g., window **361**) is displayed. The timer of the first window is stopped while being hidden until the second win-

dow is automatically closed. In a further embodiment of the present invention, the window system tracks the translucent windows to automatically schedule the sequence and the screen positions of the display of the translucent windows according to the importance of the windows, the time to close (or estimated time to close), and/or other conditions.

FIGS. **8-11** show an example of translucent windows that are initiated by a system without any input associated with user input devices (e.g., a keyboard, mouse, track ball, touch pad, touch screen, joy stick, button, or other criteria). In one embodiment of the present invention, these translucent windows do not consume any user input; and no user input can be provided to these windows to close these windows, which close automatically when certain conditions are met (e.g., the expiration of a timer, the change in system status, and others). In one alternative embodiment of the present invention, these windows accept predetermined inputs (e.g., special function keys, such as the escape key "ESC" for closing) so that a user has the option to directly control the display of these translucent windows.

A user may initiate a translucent window through an input associated with a user input device. For example, a user may use a special function key to adjust volume (or contrast, or brightness). In response to the special function key, a translucent window is displayed to show the current volume level (or contrast, or brightness). In one embodiment of the present invention, the window system (or an application program) automatically determines a location for displaying the translucent volume window (e.g., independent from the location of a cursor on the screen). When the volume window receives an input from the function key for adjust volume, the timer for the translucent volume window is restarted. After the user stops adjusting the volume for a predetermined amount of time, the timer expires; and the volume control window is faded out and closed automatically. In one embodiment of the present invention, the volume window is not translucent. In one embodiment of the present invention, the translucent window initiated by an input associated with a user input device does not close in response to any input from a user input device (e.g., the window does not have a button for closing the window, nor takes a short cut key for closing the window); the window closes only automatically. When the window does not close in response to any input from a user input device, the window may still respond to system inputs, such as a request from the operating system to close the window (e.g., when the user starts to shut down a computer system). In one embodiment of the present invention, a message window initiated by a user only displays a message to the user without consuming any input from user input devices.

In one embodiment of the present invention, when a translucent window accepts user input, the translucent window consumes only predetermined inputs for user input devices; other inputs are forwarded to normal windows as if the translucent window does not exist. For example, if a cursor related event (e.g., a click) is not accepted by the translucent window, the input is considered for the window that is just under the translucent window so that the user can interact with the window under the translucent window as if the translucent window does not exist. If the translucent window does not consume a keyboard input, the keyboard input is forwarded to the window that has the keyboard focus (which is typically indicated by a highlighted title bar). Thus, the presence of the translucent window has minimum distractions for the user working on regular windows.

FIGS. **7-11** illustrate one embodiment of the present invention with translucent windows. It is apparent to one skilled in the art from this description that some methods of the present invention can be implemented for windows that are not translucent.

FIG. **12** shows a flow diagram of a method to display a window according to one embodiment of the present invention. Operation **401** displays a user interface window (e.g., a translucent window which when displayed on top of a portion of a second window allows the user to see the portion of the second window through the translucent window); and operation **403** automatically closes the user interface window (e.g., fade out an image of the window and destroy the window) without user input (e.g., after a timer expired, or after a determination that a system status is changed or a condition is met, or after receiving input that is not associated with any user input device).

FIG. **13** shows a flow diagram of a method to close a window according to one embodiment of the present invention. Operation **411** displays a first window in response to an input (e.g., an input from a user input device, or an input that is not associated with any user input device, such as an input trigged by a system event, a change in system status, ringing signals on a phone line, or inputs initiated by the operating system). Operation **413** starts a timer. Operation **415** closes the first window when the timer expires (e.g., fade out an image of the first window and destroy the first window).

FIG. **14** shows a detailed flow diagram of a method to control a translucent window according to one embodiment of the present invention. After operation **421** receives an input (e.g., a user input from a user input device, such as a keyboard, a mouse, a track ball, a touch pad, a touch screen, a joy sticker, a button, or others) from a digital processing system, operation **423** displays a first translucent window on a display device (e.g., a LCD display, a CRT monitor, a touch screen, or others) of the digital processing system (e.g., on top of a portion of a second window), where the first window does not close in response to any input from a user input device of the digital processing system. Operation **425** starts a timer. When operation **427** determines that an input (e.g., a user input or a system input) for the first window is received, operation **431** restarts the timer; and operation **433** processes the input (alternatively, the timer may be stopped and restarted after the input is processed). When operation **429** determines that a second translucent window is displayed, operation **435** repositions (or hides) the first translucent window. When one of a number of translucent windows is closed, the remaining translucent window(s) may be repositioned (or displayed if hidden). Operation **437** closes the first translucent window when the timer expires (e.g., by fading out an image of the first window and destroying the first window).

FIG. **15** shows a method to display a translucent window according to one embodiment of the present invention. Operation **441** combines the image of a translucent window and the portion of the image of window under the translucent window to generate a combined image for the translucent window and the window under the translucent window. Operation **443** displays the combined image on the screen for the translucent window and the window under the translucent window. If operation **445** determines that the translucent window is updated or operation **447** determines that the window under the translucent window is updated, operation **441** is performed to update the corresponding portion of the screen image. In a buffered window system, the images of the translucent window and the window under the translucent window are generated separately; and the window system combines the images of the windows to display the translucent window and the window under it. In a non-buffered window system, the translucent window may generate the image of the trans-

lucent window on top of the other window using the image of the window under it. For example, the translucent window obtains the image of the window under it after the window under it draws on the frame buffer; then, the translucent window generates a combined image to update the corresponding portion of the screen.

FIGS. **16-21** show example screen images of windows displayed according to one embodiment of the present invention. When a user starts to adjust the volume level (e.g., pressing on a function key for increasing or decreasing volume, or selecting an item from a system control menu with a cursor control device, such as a mouse or a touch pad), translucent volume window **511** appears on screen **501**. Since window **511** is translucent, the portion of window **503** under window **511** is still visible. In one embodiment of the present invention, when window **511** is initially loaded, the background of volume window **511** has a high degree of transparency; and the content of window **511** has a low degree of transparency (or no transparency). Therefore, the user can easily see the content of window **511** when the user is supposed to focus on window **511**. As the user provides input to adjust the volume level, window **511** remains in a state with a high degree of transparency for the background and a low degree of transparency for the content. For example, when the user decreases the volume level (e.g., pressing a function key, or an array key), the volume level is decreased as indicated by window **521** in FIG. **17**. When the user further decreases the volume level to mute the speakers, window **531** changes an icon to indicate that the speakers are muted, as shown in FIG. **18**. When the user starts to adjust the brightness of the monitor, translucent brightness window **541** appears, as shown in FIG. **19**, while the volume window is hidden (or destroyed, or converted into the brightness window by redrawing the icon and the level bar). If the user stops providing input for the brightness window for an amount of time (e.g., a predetermined amount of time, a randomly selected amount of time, a time period determined according to a system condition or other criteria, a time period calculated on the fly, or a time period specified by a user) window **541** starts to fade away and be destroyed, as shown in FIGS. **20** and **21**. In one embodiment of the present invention, when a translucent window starts to fade away, the degree of transparency of the content in the translucent window is increased to allow the user to see better the window under the translucent window, as illustrated by window **551** in FIG. **20**. Thus, the degree of transparency of the window can be adjusted during the life cycle of the window to lead the focus point of the user. Further, a user may specify the degree of transparency of the window (e.g., as a preference parameter). The image of the window may fade out smoothly in an animation; or the image of the window may fade out in a number of discrete steps. The degree of translucency, the speed for fading out, the discrete levels of translucency for fading out, the time to expire, and/or other parameters for controlling the display of the window may be set by the user or adjusted by the system (or application software programs) automatically according to system conditions or other criteria. For example, the system (or application programs) may adjust the time to expire according to the number of translucent windows displayed concurrently on the screen; or the system (or an application program) may adjust the initial degree of translucency according to the color pattern at the location where the translucent window is displayed.

In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will be evident that various modifications may be made thereto without departing from the broader spirit and scope of the invention as set forth in the following claims. The specification and drawings are, accordingly, to be regarded in an illustrative sense rather than a restrictive sense.

What is claimed is:

1. A method to display a user interface window for a digital processing system, the method comprising:

displaying a first window in response to receiving a first input from a user input device of the digital processing system which is capable of displaying at least a portion of a second window concurrently with the first window on a screen;

starting a timer; and

closing the first window in response to a determination that the timer expired;

wherein the first window does not close in response to any input from a user input device of the digital processing system, wherein the first window has been displayed independently from a position of a cursor on the screen.

2. A method as in claim **1** wherein the first window is translucent; and the portion of the second window is visible while under the first window.

3. A method as in claim **2** wherein the first window is at a top level in a window displaying hierarchy.

4. A method as in claim **2** wherein a degree of translucency of the first window is adjustable.

5. A method as in claim **1** wherein said closing the first window comprises:

fading out an image of the first window.

6. A method as in claim **1** wherein the second window, if displayed, does close in response to an input from a user input device of the digital processing system.

7. A method as in claim **6** wherein the first window does not respond to any input from a user input device of the digital processing system.

8. A method as in claim **1** further comprising:

repositioning the first window in response to a third window being displayed.

9. A method as in claim **1** further comprising:

hiding the first window in response to a third window being displayed at a location where the first window is displayed.

10. A method as in claim **1** further comprising:

repositioning the first window on a display in response to a second input for the first window.

11. A method as in claim **10** wherein the second input indicates that a third window is displayed.

12. A method as in claim **10** wherein the second input is received from a user input device of the digital processing system.

13. A method as in claim **10** further comprising:

adjusting a position of the first window in a window displaying hierarchy in response to a third input.

14. A method as in claim **1** further comprising:

determining a position on a display of the digital processing system independent of a position of a cursor on the display;

wherein the first window is displayed at the position.

15. A method as in claim **14** wherein the position is centered horizontally on the display.

16. A method as in claim **1** further comprising:

restarting the timer in response to receiving a second input for the first window.

17. A method as in claim **16** wherein the second input is received from a user input device of the digital processing system.

18. A method as in claim **16** wherein the first window is created by a first application and the second window is cre-

ated by a second application, wherein the first application is different from the second application.

19. A method as in claim **1** wherein the user input device is one of:

a) a keyboard;
b) a mouse;
c) a track ball;
d) a touch pad;
e) a touch screen;
f) a joy stick; and
g) a button.

20. A method to display a user interface window for a digital processing system, the method comprising:

displaying a first window, the first window being translucent, at least a portion of a second window being capable of being displayed on the digital processing system under the first window, the portion of the second window, when present, being visible under the first window on a screen; and

closing the first window without user input, wherein the first window has been displayed independent from a position of a cursor on the screen.

21. A method as in claim **20** further comprising:

starting a timer;

wherein said closing the first window is in response to expiration of the timer.

22. A method as in claim **20** further comprising:

receiving an input, the input not associated with a user input device of the digital processing system;

wherein said closing the first window is in response to the input.

23. A method as in claim **20** further comprising:

determining whether or not a condition is met;

wherein said closing the first window is in response to a determination that the condition is met.

24. A method as in claim **20** wherein said closing the first window comprises:

fading out an image of the first window.

25. A method as in claim **20** wherein a degree of translucency of the first window is adjustable.

26. A machine readable media containing executable computer program instructions which when executed by a digital processing system cause said system to perform a method to display a user interface window, the method comprising:

displaying a first window in response to receiving a first input from a user input device of the digital processing system which is capable of displaying at least a portion of a second window concurrently with the first window on a screen;

starting a timer; and

closing the first window in response to a determination that the timer expired;

wherein the first window does not close in response to any input from a user input device of the digital processing system, wherein the first window has been displayed independently from a position of a cursor on the screen.

27. A media as in claim **26** wherein the first window is translucent; and the portion of the second window is visible while under the first window.

28. A media as in claim **27** wherein the first window is at a top level in a window displaying hierarchy.

29. A media as in claim **27** wherein a degree of translucency of the first window is adjustable.

30. A media as in claim **26** wherein said closing the first window comprises:

fading out an image of the first window.

31. A media as in claim **26** wherein the second window, if displayed, does close in response to an input from a user input device of the digital processing system.

32. A media as in claim **31** wherein the first window does not respond to any input from a user input device of the digital processing system.

33. A media as in claim **26** wherein the method further comprises:

repositioning the first window in response to a third window being displayed.

34. A media as in claim **26** wherein the method further comprises:

hiding the first window in response to a third window being displayed at a location where the first window is displayed.

35. A media as in claim **26** wherein the method further comprises:

repositioning the first window on a display in response to a second input for the first window.

36. A media as in claim **35** wherein the second input indicates that a third window is displayed.

37. A media as in claim **35** wherein the second input is received from a user input device of the digital processing system.

38. A media as in claim **35** wherein the method further comprises:

adjusting a position of the first window in a window displaying hierarchy in response to a third input.

39. A media as in claim **26** wherein the method further comprises:

determining a position on a display of the digital processing system independent of a position of a cursor on the display;

wherein the first window is displayed at the position.

40. A media as in claim **39** wherein the position is centered horizontally on the display.

41. A media as in claim **26** wherein the method further comprises:

restarting the timer in response to receiving a second input for the first window.

42. A media as in claim **41** wherein the second input is received from a user input device of the digital processing system.

43. A machine readable media as in claim **41** wherein the first window is created by a first application and the second window is created by a second application, wherein the first application is different from the second application.

44. A media as in claim **26** wherein the user input device is one of:

a) a keyboard;
b) a mouse;
c) a track ball;
d) a touch pad;
e) a touch screen;
f) a joy stick; and
g) a button.

45. A machine readable media containing executable computer program instructions which when executed by a digital processing system cause said system to perform a method to display a user interface window, the method comprising:

displaying a first window, the first window being translucent, at least a portion of a second window being capable of being displayed on the digital processing system under the first window, the portion of the second window, when present, being visible under the first window on a screen; and

closing the first window without user input, wherein the first window has been displayed independent from a position of a cursor on the screen.

46. A media as in claim **45** wherein the method further comprises:

starting a timer;

wherein said closing the first window is in response to expiration of the timer.

47. A media as in claim **45** wherein the method further comprises:

receiving an input, the input not associated with a user input device of the digital processing system;

wherein said closing the first window is in response to the input.

48. A media as in claim **45** wherein the method further comprises:

determining whether or not a condition is met;

wherein said closing the first window is in response to a determination that the condition is met.

49. A media as in claim **45** wherein said closing the first window comprises:

fading out an image of the first window.

50. A media as in claim **45** wherein a degree of translucency of the first window is adjustable.

51. A digital processing system to display a user interface window, the system comprising:

means for displaying a first window in response to receiving a first input from a user input device of the digital processing system which is capable of displaying at least a portion of a second window concurrently with the first window on a screen;

means for starting a timer; and

means for closing the first window in response to a determination that the timer expired;

wherein the first window does not close in response to any input from a user input device of the digital processing system, wherein the first window has been displayed independently from a position of a cursor on the screen.

52. A digital processing system as in claim **51** wherein the first window is translucent; and the portion of the second window is visible while under the first window.

53. A digital processing system as in claim **52** wherein the first window is at a top level in a window displaying hierarchy.

54. A digital processing system as in claim **52** wherein a degree of translucency of the first window is adjustable.

55. A digital processing system as in claim **51** wherein said means for closing the first window comprises:

means for fading out an image of the first window.

56. A digital processing system as in claim **51** wherein the second window, if displayed, does close in response to an input from a user input device of the digital processing system.

57. A digital processing system as in claim **56** wherein the first window does not respond to any input from a user input device of the digital processing system.

58. A digital processing system as in claim **51** further comprising:

means for repositioning the first window in response to a third window being displayed.

59. A digital processing system as in claim **51** further comprising:

means for hiding the first window in response to a third window being displayed at a location where the first window is displayed.

60. A digital processing system as in claim **51** further comprising:

means for repositioning the first window on a display in response to a second input for the first window.

61. A digital processing system as in claim **60** wherein the second input indicates that a third window is displayed.

62. A digital processing system as in claim **60** wherein the second input is received from a user input device of the digital processing system.

63. A digital processing system as in claim **60** further comprising:

means for adjusting a position of the first window in a window displaying hierarchy in response to a third input.

64. A digital processing system as in claim **51** further comprising:

means for determining a position on a display of the digital processing system independent of a position of a cursor on the display;

wherein the first window is displayed at the position.

65. A digital processing system as in claim **64** wherein the position is centered horizontally on the display.

66. A digital processing system as in claim **51** further comprising:

means for restarting the timer in response to receiving a second input for the first window.

67. A digital processing system as in claim **66** wherein the second input is received from a user input device of the digital processing system.

68. A digital processing system as in claim **66** wherein the first window is created by a first application and the second window is created by a second application, wherein the first application is different from the second application.

69. A digital processing system as in claim **51** wherein the user input device is one of:

a) a keyboard;

b) a mouse;

c) a track ball;

d) a touch pad;

e) a touch screen;

f) a joy stick; and

g) a button.

70. A digital processing system to display a user interface window, the system comprising:

means for displaying a first window, the first window being translucent, at least a portion of a second window being capable of being displayed on the digital processing system under the first window, the portion of the second window, when present, being visible under the first window on a screen; and

means for closing the first window without user input, wherein the first window has been displayed independent from a position of a cursor on the screen.

71. A digital processing system as in claim **70** further comprising:

means for starting a timer;

wherein the first window is closed in response to expiration of the timer.

72. A digital processing system as in claim **70** further comprising:

means for receiving an input, the input not associated with a user input device of the digital processing system;

wherein the first window is closed in response to the input.

73. A digital processing system as in claim **70** further comprising:

means for determining whether or not a condition is met;

wherein the first window is closed in response to a determination that the condition is met.

74. A digital processing system as in claim **70** wherein said means for closing the first window comprises:

means for fading out an image of the first window.

75. A digital processing system as in claim **70** wherein a degree of translucency of the first window is adjustable.

* * * * *

Exhibit 7

US007863533B2

(12) **United States Patent**
Hamel et al.

(10) **Patent No.: US 7,863,533 B2**
(45) **Date of Patent: Jan. 4, 2011**

(54) **CANTILEVERED PUSH BUTTON HAVING MULTIPLE CONTACTS AND FULCRUMS**

(75) Inventors: **Bradley J. Hamel**, Sunnyvale, CA (US); **Tang Yew Tan**, San Francisco, CA (US); **Erik Wang**, Redwood City, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 188 days.

(21) Appl. No.: **12/239,102**

(22) Filed: **Sep. 26, 2008**

(65) **Prior Publication Data**

US 2009/0301857 A1 Dec. 10, 2009

Related U.S. Application Data

(60) Provisional application No. 61/059,753, filed on Jun. 7, 2008.

(51) **Int. Cl.**
H01H 23/00 (2006.01)

(52) **U.S. Cl.** ..... **200/339**; 200/5 R; 200/553

(58) **Field of Classification Search** 200/512, 200/516, 517, 553, 557, 558, 339; 341/20–22; 345/156, 157, 161, 168, 169, 184
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,329,278 A * 7/1994 Dombroski 341/20
5,508,479 A * 4/1996 Schooley ..... 200/5 R
5,584,380 A * 12/1996 Naitou 200/315
6,239,392 B1 * 5/2001 Serizawa et al. ..... 200/5 R
6,246,019 B1 * 6/2001 Nakamura et al. ..... 200/6 A
6,274,826 B1 * 8/2001 Serizawa et al. ..... 200/5 R
6,961,052 B1 * 11/2005 Vaziri 345/184
6,967,300 B1 * 11/2005 Mizuno et al. 200/512
6,995,326 B2 * 2/2006 Sugimura et al. ..... 200/5 R
7,259,339 B1 * 8/2007 Babella et al. ..... 200/5 A

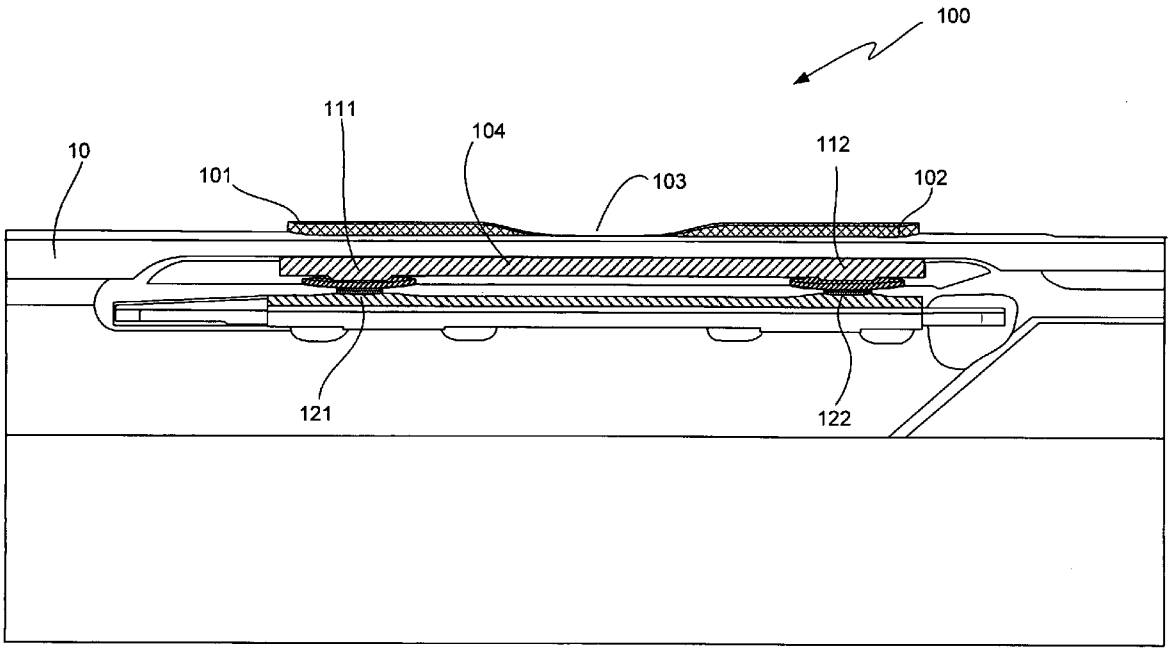
* cited by examiner

Primary Examiner—Michael A Friedhofer
(74) *Attorney, Agent, or Firm*—Beyer Law Group LLP

(57) **ABSTRACT**

A cantilevered push button adapted for accepting an input on an electrical or electronic device is disclosed. The button can include an elongated button top component disposed about an exterior surface of an electrical or electronic device such that it is accessible to a user, and having two opposing distal ends associated with separate user inputs. A first fulcrum is located between the first distal end and the midpoint of the elongated button top component, while a second fulcrum is located between the second distal end and the midpoint. A first electrical contact is associated with the first distal end, such that when a user presses on the first distal end, the elongated button top component pivots about the second fulcrum and the first electrical contact is actuated. A second electrical contact is similarly associated with the second distal end and first fulcrum.

20 Claims, 6 Drawing Sheets



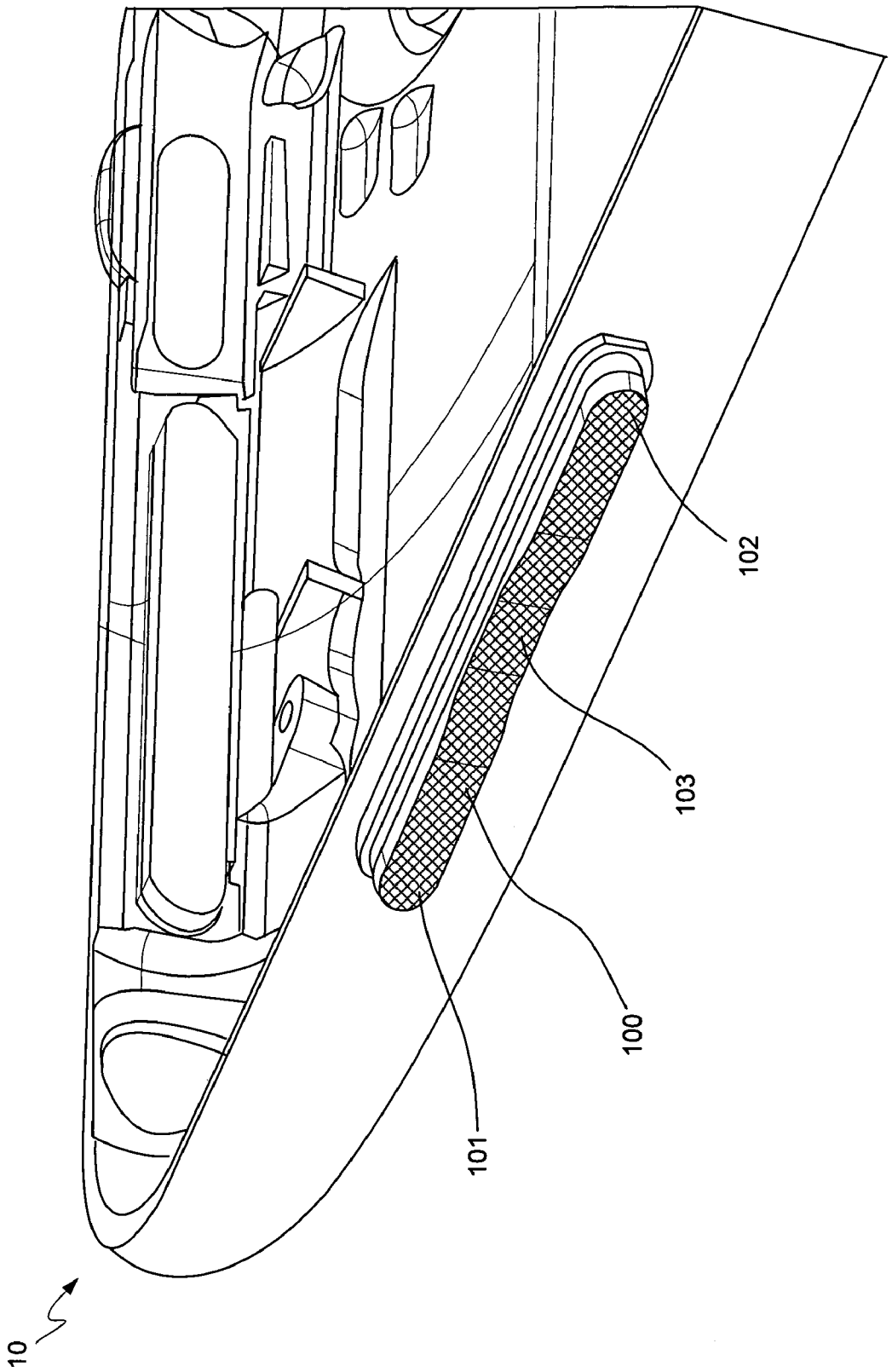


FIG. 1

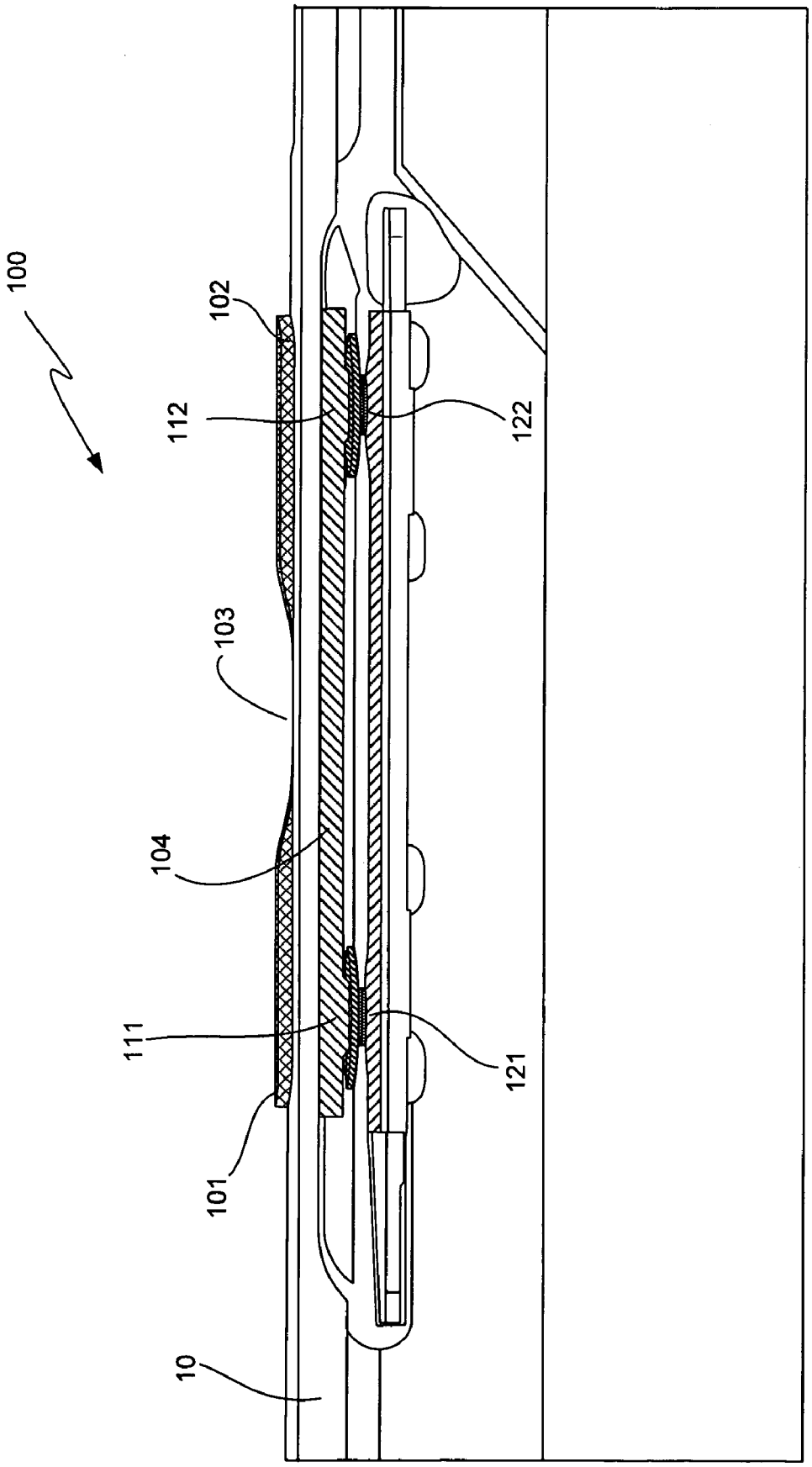


FIG. 2A

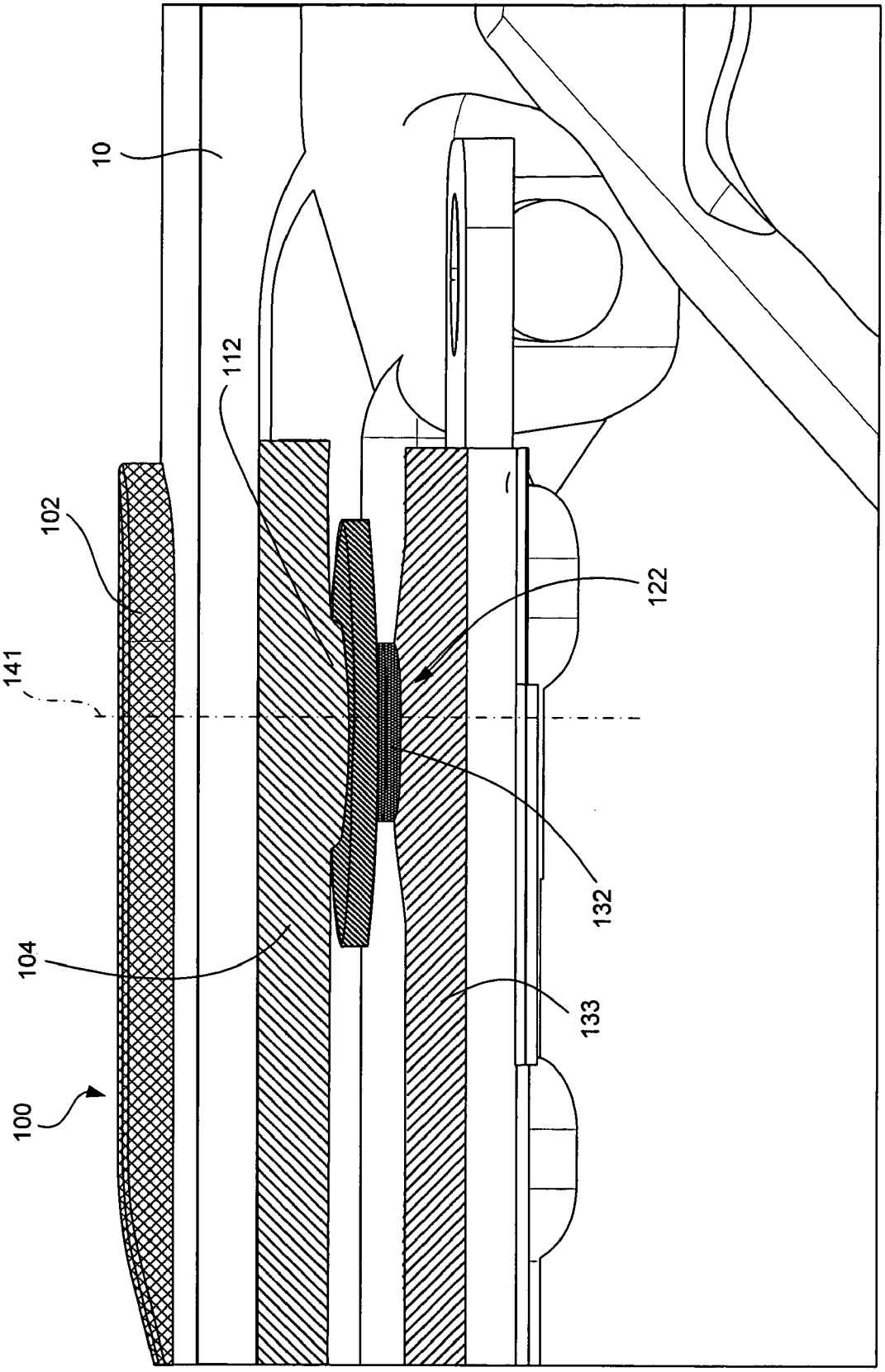


FIG. 2B

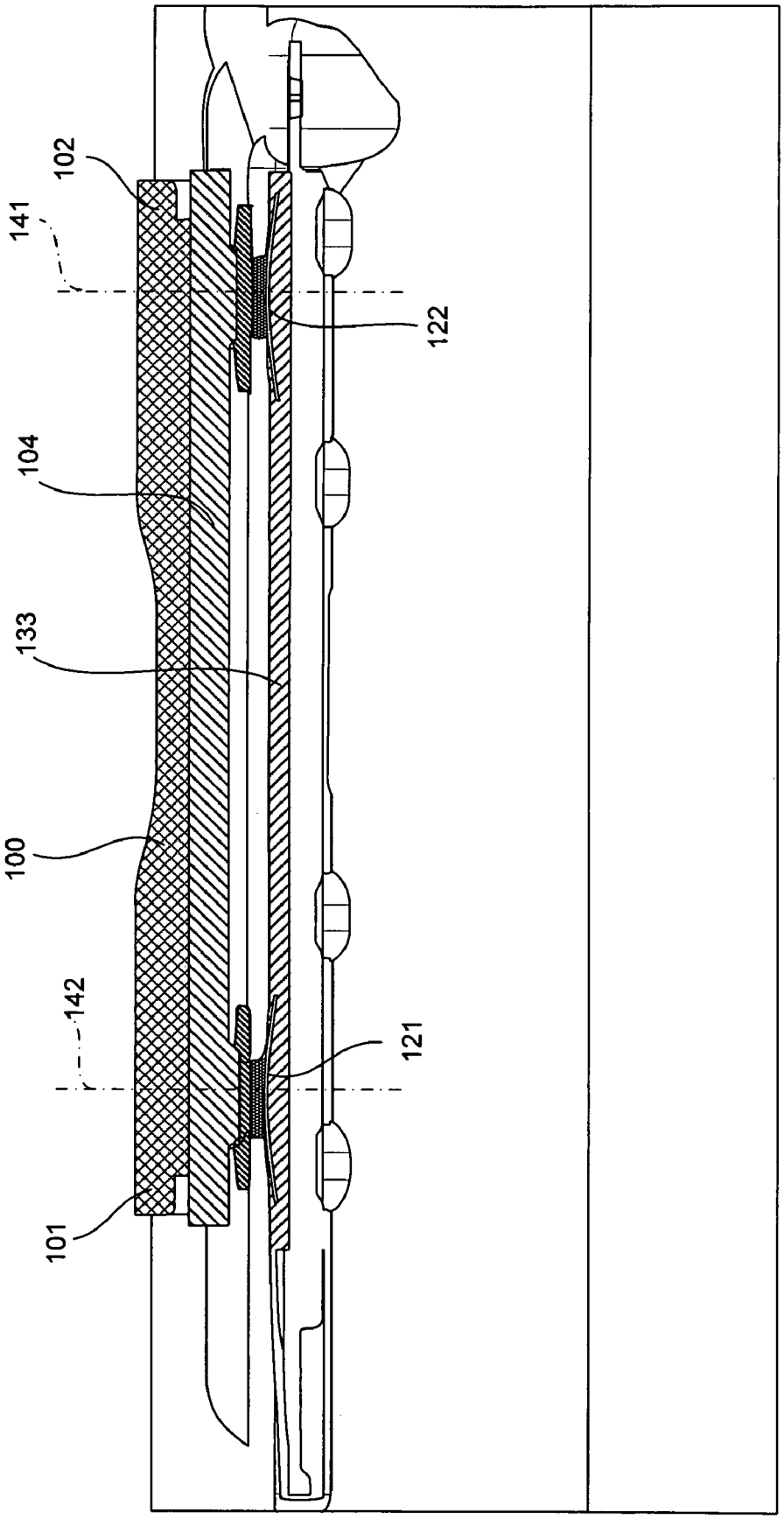


FIG. 2C

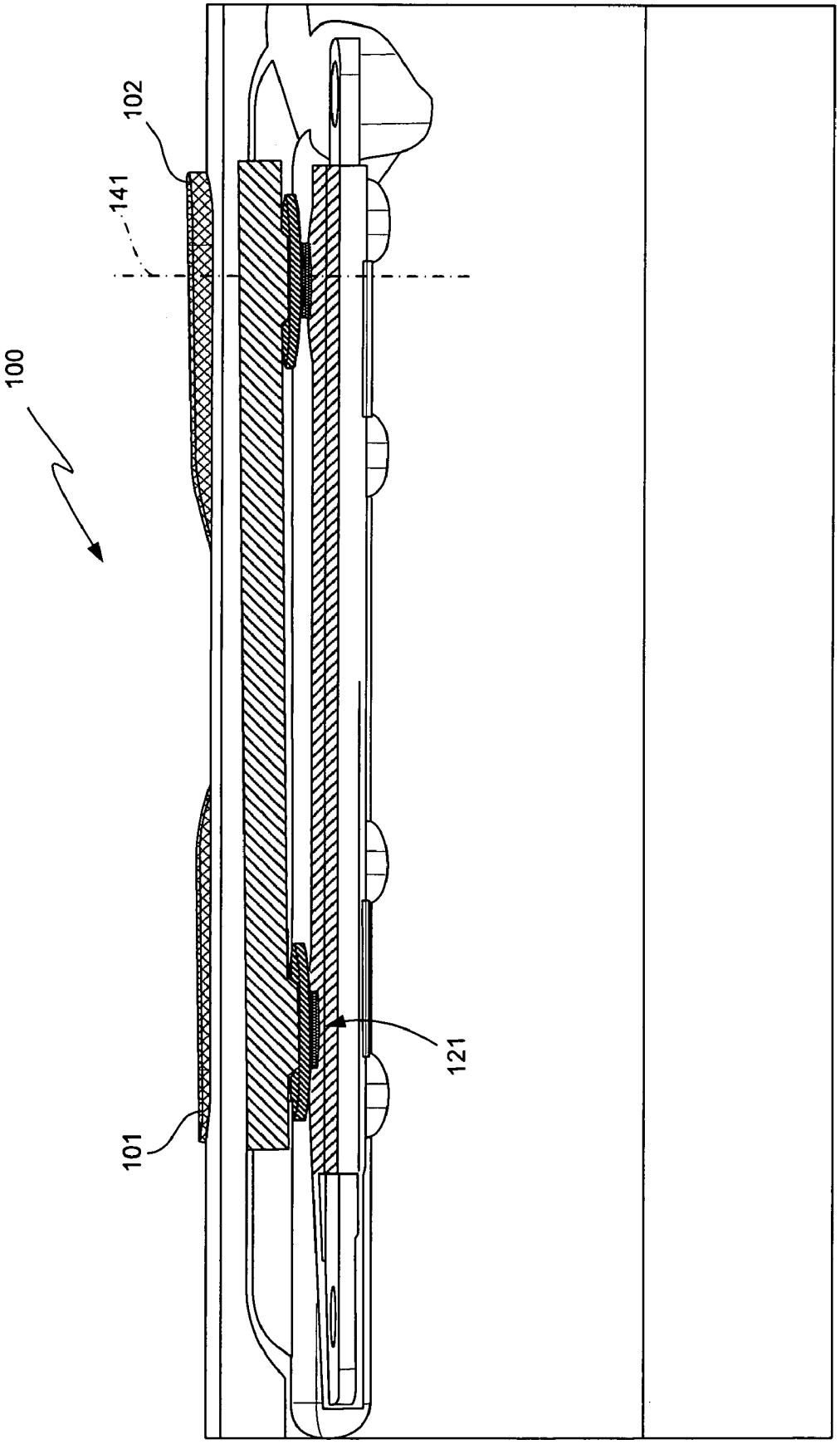


FIG. 3A

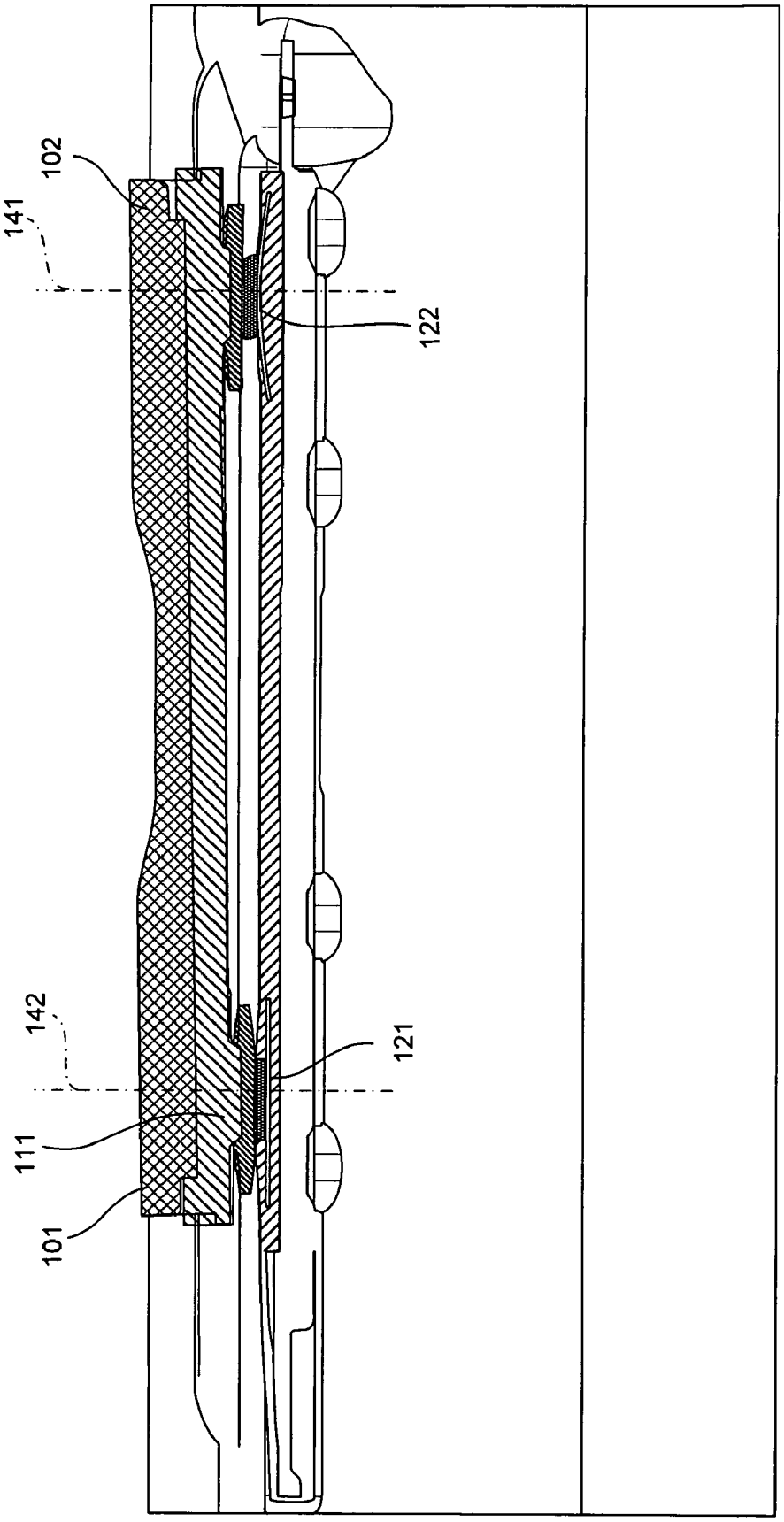


FIG. 3B

CANTILEVERED PUSH BUTTON HAVING MULTIPLE CONTACTS AND FULCRUMS

CROSS REFERENCE TO RELATED APPLICATIONS

This application claims priority to U.S. Provisional Patent Application No. 61/059,753, filed Jun. 7, 2008, and entitled "CANTILEVERED PUSH BUTTON HAVING MULTIPLE CONTACTS AND FULCRUMS," which is incorporated herein by reference in its entirety and for all purposes.

TECHNICAL FIELD

The present invention relates generally to push buttons, and more particularly to the use of cantilevered push buttons on electrical or electronic devices.

BACKGROUND

Media players, cellular telephones and numerous other electrical or electronic personal devices or appliances are now ubiquitous. Such devices often require ways of inputting information or commands by a user, with resulting features including touch screens, dials, knobs and push buttons. Such buttons can be part of a keyboard, a 10-key pad, or can be used in isolation, as desired by a designer. One drawback to using push buttons as a means for permitting input can be a cluttered or complex appearance for the overall device where many such buttons are used.

Cantilevered push buttons, also known as seesaw buttons, are one way of providing the functional effect of two buttons while providing a more streamlined and aesthetically pleasing overall appearance. In such an arrangement, a single elongated outer member effectively serves as two different input buttons. When one end is pushed, a first inner electrical contact is actuated, and when the other end is pushed, a second and separate inner electrical contact is made. It is usually not possible to actuate both buttons at the same time with many true cantilevered push buttons.

Such cantilevered or seesaw push buttons typically pivot about a central fulcrum, such that when one end is pushed in by a user to actuate its respective button contact, the other end actually rises upward or away from the rest of the device. Although the actual distances traveled by such cantilevered buttons when actuated can be quite small, it can still be perceived as somewhat less aesthetically appealing to have opposing ends of such cantilevered buttons move away from the device when a particular end is pushed.

In order to compensate for this issue with button ends or other portions moving away from their respective devices, some cantilevered buttons are formed from rubber, soft plastic, or other pliable materials, such that some or all of the cantilevered push button flexes to permit the non-actuated end to be held in place by the device housing or another mechanical stop while the actuated end is pushed inward. Problems with this approach can arise, however, when a metal or other less flexible material is used to construct the actual cantilever portion of the button itself.

While many designs for cantilevered or seesaw push buttons have generally worked well in the past, there is always a desire to provide other cantilever button designs or techniques that can achieve the same or similar objectives in a

reliable fashion while allowing for a greater variety of possible materials that may also be more aesthetically pleasing.

SUMMARY

It is an advantage of the present invention to provide metallic or otherwise rigid cantilevered push buttons having multiple electrical contacts that also do not require the non-actuated ends of the buttons to rise when the actuated ends are pushed inward. This can be accomplished at least in part through the use of multiple fulcrums about which the cantilevered push button can pivot, with the fulcrum being determined by which end of the button is pushed.

In various embodiments of the present invention, a cantilevered push button adapted for accepting an input on an electrical or electronic device is provided. This cantilevered push button can include an elongated button top component, a first fulcrum, a second fulcrum, a first electrical contact and a second electrical contact. The elongated button top can have a first distal end associated with a first user input and a second distal end opposite the first distal end and associated with a second user input separate from the first user input. The elongated button top component is preferably disposed about an exterior surface of an electrical or electronic device and is accessible to a user. The first fulcrum can be located at the first distal end or between the first distal end and the midpoint of the elongated button top component, while the second fulcrum can be located at the second distal end or between the second distal end and the midpoint of the elongated button top component. The first electrical contact can be associated with the first distal end of the elongated button top component, while the second electrical contact can be associated with the second distal end of the elongated button top component.

When a user presses on the first distal end of the elongated button top component, then the elongated button top component can pivot about the second fulcrum and the first electrical contact is actuated. Similarly, when a user presses on the second distal end of the elongated button top component, then the elongated button top component pivots about the first fulcrum and the second electrical contact is actuated. In various embodiments, both of the first and second distal ends can be actuated simultaneously by a user during ordinary use of the device, and such simultaneous actuation can result in the entire cantilevered push button being pressed inward toward the device.

In some embodiments of this cantilevered push button, the first fulcrum can be located at the second electrical contact, and/or the second fulcrum can be located at the first electrical contact. Alternatively, the first fulcrum can be located at one distal end and the second fulcrum can be located at another distal end of the cantilevered push button. Although an optimal location for a fulcrum is at a distal end, any location that is significantly far away from the midpoint of the button is preferable. Further, although the fulcrums can be located an equal distance from the midpoint of the button, this is not necessary. Additional fulcrums may also be used, as may be desired for a given design. In particular, one fulcrum per button input is thought to be best. For example, a cantilevered button with four inputs can have four different fulcrums, with one fulcrum for each separate input.

In various embodiments, the elongated button top may be comprised of a substantially rigid material, such as a metal, hard plastic, or other inflexible material. In addition, the first and second electrical contacts can comprise dome button type electrical contacts, which may be preloaded with a positive upward force. The various foregoing features can all be

included or only partially included in any desired combination for a given cantilevered push button.

In further embodiments of the present invention, an electrical or electronic device can include a housing adapted to contain one or more internal electrical device components therein, one or more user interface regions, and a cantilevered push button located at one of the user interface regions and adapted for accepting one or more inputs on said electrical device, with the cantilevered push button being any of the cantilevered push buttons described above. In the event of an electronic device, a processor located within the housing may be included, and the user interface region or regions can be in communication with the processor. A plurality of cantilevered push buttons may also be used on such a device, and may be placed at one or multiple user interface regions.

Other apparatuses, methods, features and advantages of the invention will be or will become apparent to one with skill in the art upon examination of the following figures and detailed description. It is intended that all such additional systems, methods, features and advantages be included within this description, be within the scope of the invention, and be protected by the accompanying claims.

BRIEF DESCRIPTION OF THE DRAWINGS

The included drawings are for illustrative purposes and serve only to provide examples of possible structures and arrangements for the disclosed inventive apparatuses and methods for providing cantilevered buttons having multiple contacts and fulcrums. These drawings in no way limit any changes in form and detail that may be made to the invention by one skilled in the art without departing from the spirit and scope of the invention.

FIG. **1** illustrates in side perspective view an exemplary cantilevered button along the side of an outer housing for an associated cellular telephone.

FIG. **2**A illustrates in side elevation and partially cutaway view an exemplary cantilevered button according to one embodiment of the present invention.

FIG. **2**B illustrates in side elevation and partially cutaway view a close-up of one end of the exemplary cantilevered button of FIG. **2**A.

FIG. **2**C illustrates in side cross-sectional view the exemplary cantilevered button of FIG. **2**A.

FIG. **3**A illustrates in side elevation and partially cutaway view the exemplary cantilevered button of FIG. **2**A with the left end actuated according to one embodiment of the present invention.

FIG. **3**B illustrates in side cross-sectional view the exemplary cantilevered button of FIG. **3**A with the left end actuated.

DETAILED DESCRIPTION

Exemplary applications of apparatuses and methods according to the present invention are described in this section. These examples are being provided solely to add context and aid in the understanding of the invention. It will thus be apparent to one skilled in the art that the present invention may be practiced without some or all of these specific details. In other instances, well known process steps have not been described in detail in order to avoid unnecessarily obscuring the present invention. Other applications are possible, such that the following examples should not be taken as limiting.

In the following detailed description, references are made to the accompanying drawings, which form a part of the description and in which are shown, by way of illustration, specific embodiments of the present invention. Although these embodiments are described in sufficient detail to enable one skilled in the art to practice the invention, it is understood that these examples are not limiting; such that other embodiments may be used, and changes may be made without departing from the spirit and scope of the invention.

The invention relates in various embodiments to the implementation and use of cantilevered push buttons as input mechanisms on electrical devices. Such a device can be, for example, an electronic device, such as an iPod® media player or iPhone® cellular telephone made by Apple Inc., as well as a PDA, handheld game or video game controller, radio, miniature television, keyboard, or any other electrical or electronic device that uses push buttons for user input.

In various embodiments of the present invention, a cantilevered push button having multiple contacts and multiple fulcrums is provided. The cantilevered push button can be adapted such that different fulcrums are used to pivot the button depending upon which end or portion of the button is pressed. In this manner, the overall motion and reactive displacement of the button can be controlled, even where metal or other rigid materials are used to construct the user portion of the button. The overall effect, look and feel can generally be more aesthetically pleasing to the user.

Referring first to FIG. **1**, an exemplary cantilevered button along the side of an outer housing for an associated cellular telephone is shown in side perspective view. Cellular telephone housing **10** can include various protrusions, ports and features, as will be readily appreciated. For example, a hole or recess along the side of housing **10** can be made for cantilevered push button **100**. Such a button **100** can have a first distal end **101** and a second distal end **102**, as well as a midpoint or center **103**. Each of distal ends **101**, **102** can be adapted to actuate an associated button or electrical contact within the device when they are depressed or otherwise used. Although shown on the side of a cellular telephone, there are many suitable locations for such cantilevered buttons, and numerous other types of devices can feature such a button or buttons, as will be readily appreciated by those skilled in the art.

Moving next to FIG. **2**A, an exemplary cantilevered button according to one embodiment of the present invention is illustrates in side elevation and partially cutaway view. As noted above, cantilevered push button **100** can be embedded within a hole or other recess in housing **10**. An elongated button top component **100** can include opposing distal ends **101**, **102**, and a midpoint or center **103**. First distal end **101** can be associated with a first internal button or electrical contact **121** inside the device, while second distal end **102** can be associated with a second internal button or electrical contact **122** inside the device. The elongated button top component **100** is generally outside or about the device and exposed to a user, while an internal elongated button base **104** can be attached or otherwise coupled to the elongated button top component. One or more protrusions **111** in the elongated button base component **104** can be adapted to help actuate the electrical contacts **121**, as will be readily appreciated.

FIG. **2**B illustrates in side elevation and partially cutaway view a close-up of the right end of the exemplary cantilevered button of FIG. **2**A. As shown, distal end **102** is disposed directly above internal button or electrical contact **122**. Internal button or electrical contact can be, for example, a preloaded dome type button or other suitable electrical contact device, as will be readily appreciated by those skilled in the art. When a user presses downward on or about end **102**, the button top component **100** generally depresses downward at distal end **102** and in the direction of electrical contact **122**.

This forces the portion of button base **104** directly beneath end **102** downward as well, such that protrusion **112** presses on the top **132** of button or electrical contact **122**, and this second "button" or electrical contact is thereby actuated. For this process of actuating second button or contact **122** by depressing second distal end **102**, the actual fulcrum is at the other end of the cantilevered button and is not shown in FIG. **2**B. In the event that the other end of the cantilevered button is depressed to activate the other electrical contact (both not shown in FIG. **2**B), then the cantilever pivots about contact **122** (i.e., axis **141**). This process is made possible due to the gap between button base **104** and internal structure **133**.

FIG. **2**C illustrates in side cross-sectional view the exemplary cantilevered button of FIG. **2**A. Again, pressing at end **101** actuates contact **121**, while pressing at end **102** actuates contact **122**. However, the fulcrum is different for each of these different actuations. In the event that end **101** is pressed, then the fulcrum is along contact **122** (i.e., axis **141**), while if end **102** is pressed, then the fulcrum is at contact **121** (i.e., axis **142**). Since button base **104** is effectively separated from internal structure **133**, the effective mechanical contact points between the elongated button top and base and the internal structure are at the internal buttons or electrical contacts **121** and **122**. Thus, when a mechanical downward force is applied at one end above one electrical contact, the effective fulcrum is at the other electrical contact at the other end. The result is that multiple fulcrums are used with the overall cantilevered button, such that the opposite end does not perceptively extend away from the device when one end is pushed inward to activate its button.

While not necessary in order for the cantilevered push button to effectively have multiple fulcrums, the spring loaded internal button at each end aids in providing more rigidity and support to the overall device. That is, when end **101** (and thus button or contact **121**) is pushed downward, the spring loaded force on contact or button **122** generally disposes end **102** to remain up and steady while button **122** is the fulcrum for the entire elongated button top component **100**. Although it is thought that the top of the dome of a spring loaded dome type button makes an excellent location for a fulcrum, other locations are also possible, as will be readily appreciated. In fact, any location at a distal end or between one distal end of the cantilevered push button and its center or midpoint could be suitable for a fulcrum when the other distal end is being pressed or actuated.

As will be readily appreciated, the farther a fulcrum is away from the midpoint or center of the button, the better the effect will be as far as minimal movement of the opposite end when a particular end is pressed. Thus, an optimal location for a fulcrum can actually be at the opposite end of the button from the end that is being pressed or actuated. Conversely, a fulcrum location that is at or close to the midpoint of the button can be less desirable, although not impossible to implement in a particular design. As shown in the exemplary illustrations provided, the fulcrums used are near the distal ends of the cantilevered push button, with the results of such locations being favorable.

Also, it should be noted that the exemplary design illustrated and described herein utilized "soft" fulcrums, in that the fulcrums do provide resistance and are effective in ordinary use, but can be defeated if desired in a particular way. That is, each dome loaded electrical contact provides sufficient resistance to function as a fulcrum when the opposite end is pressed or actuated, but will not provide enough resistance to be a rigid "hard" fulcrum if force is also placed generally above it at the same time. Because a "hard" or permanent central or midpoint fulcrum is not physically present on cantilevered button **100**, it is possible to depress both ends **101**, **102** to actuate both internal buttons or electrical contacts **121**, **122** simultaneously. In such instances, such simultaneous actuation results in the entire cantilevered push button being pressed inward toward its respective electrical or electronic device. This feature provides an additional advantage over traditional or customary cantilever buttons, in that most such devices do not permit both buttons at both ends to be actuated simultaneously. Accordingly, the respective electrical or electronic device can be designed or programmed to accept and act upon simultaneous inputs from multiple inputs on a single cantilevered button.

Continuing now to FIGS. **3**A and **3**B, the exemplary cantilevered button of FIG. **2**A is shown with the left end actuated according to one embodiment of the present invention. FIG. **3**A depicts a side elevation and partially cutaway view, while FIG. **3**B depicts a side cross-sectional view. As illustrated, distal end **101** has been pressed downward such that protrusion **111** has activated button or electrical contact **121** directly below distal end **101**. The fulcrum for the entire cantilevered button **100** for this actuation is effectively button or electrical contact **122** at the opposite end, with rotation generally being about a point atop the dome of contact **122** (i.e., along axis **141**). Of course, the overall device can be substantially symmetrical in nature, such that the same relationships and results can be had for depressing distal end **102**.

Again, because the fulcrum is moved away from the midpoint or center of the cantilevered button, any resulting motion rise of the opposite end is eliminated or substantially reduced when either end is pressed. This is true even where the upper portion of the cantilevered button is a rigid material, such as a metal, hard plastic or other inflexible material. As will be appreciated, such an ability provides designers with added flexibility in their choices for materials and appearances in the design and presentation of cantilevered push buttons that are fully functional and aesthetically pleasing in appearance and use.

It should be noted that although the fulcrums in the examples provided above are generally located an equal distance from the midpoint of the button, this is not absolutely necessary. For example, a particular design might favor one fulcrum being located at a distal end, and another fulcrum being located halfway between the other distal end and the midpoint of the button, or even closer to the midpoint.

Furthermore, although the foregoing examples all use a relatively simple cantilevered button with only two inputs on opposing ends, more complex buttons with additional fulcrums may also be used, as may be desired for a given design. For example, a four-way cantilevered button having four inputs and four fulcrums may be used. In such a design, the button top component may be shaped like a cross and may have a fulcrum that corresponds to each of the four different inputs at each distal end of the cross. Alternatively, the button top could be shaped like a circle, with inputs similarly at the "north," "west," "south" and "east" coordinates of the circle. The use of a simple single user input on such a button (e.g., "north") could result in a pivoting about an opposing fulcrum (e.g., "south"), similar to the two input elongated cantilevered push button in the foregoing examples. In such an embodiment, the simultaneous actuation of two adjacent inputs (e.g., north and east), could result in the simultaneous pivoting about the two opposing fulcrums for those inputs (e.g., south and west). Similar to the foregoing two input button embodiments, a cantilevered push button having four or more inputs could also have "soft" fulcrums, such that the entire button and all inputs could be actuated at once. As will be readily appreciated, further designs and additional inputs for even

more complex cantilevered buttons having multiple fulcrums could also be used within the spirit of the present invention.

Although the foregoing invention has been described in detail by way of illustration and example for purposes of clarity and understanding, it will be recognized that the above described invention may be embodied in numerous other specific variations and embodiments without departing from the spirit or essential characteristics of the invention. For example, although many illustrations have pointed to the use of metal as the material for the cantilevered push button, a rigid and inflexible plastic or other material may alternatively be used. Multiple materials may also be used to form the button. Other changes and modifications may be practiced, and it is understood that the invention is not to be limited by the foregoing details, but rather is to be defined by the scope of the appended claims.

What is claimed is:

1. A cantilevered push button adapted for accepting one or more inputs on an electrical or electronic device, comprising:

a button top component having a first distal end associated with a first user input and a second distal end opposite said first distal end and associated with a second user input separate from said first user input, wherein said button top component is disposed about an exterior surface of an electrical or electronic device and is accessible to a user, and wherein both of said first and second distal ends can be actuated simultaneously by a user during ordinary use of said electrical or electronic device;

a first fulcrum located at said first distal end or between said first distal end and the midpoint of said button top component;

a second fulcrum located at said second distal end or between said second distal end and said midpoint of said button top component;

a first electrical contact associated with said first distal end of said button top component, wherein said button top component pivots about said second fulcrum and said first electrical contact is actuated when a user presses on said first distal end; and

a second electrical contact associated with said second distal end of said button top component, wherein said button top component pivots about said first fulcrum and said second electrical contact is actuated when a user presses on said second distal end.

2. The cantilevered push button of claim **1**, wherein said first fulcrum is located at said second electrical contact.

3. The cantilevered push button of claim **2**, wherein said second fulcrum is located at said first electrical contact.

4. The cantilevered push button of claim **1**, wherein said button top component is comprised of a substantially rigid material.

5. The cantilevered push button of claim **4**, wherein said button top component is metal.

6. The cantilevered push button of claim **1**, wherein said first fulcrum is located at said first distal end and said second fulcrum is located at said second distal end.

7. The cantilevered push button of claim **1**, wherein said first and second electrical contacts comprise dome button type electrical contacts.

8. The cantilevered push button of claim **7**, wherein said dome button type electrical contacts are preloaded with a positive upward force.

9. The cantilevered push button of claim **1**, wherein said simultaneous actuation results in the entire cantilevered push button being pressed inward toward said electrical or electronic device.

10. The cantilevered push button of claim **1**, wherein said button top component is adapted to be disposed through an opening of and not contact a housing of said electrical or electronic device.

11. An electrical device, comprising:

a housing adapted to contain one or more internal electrical device components therein;

one or more user interface regions; and

a cantilevered push button located at one of said one or more user interface regions and adapted for accepting one or more inputs on said electrical device, wherein said cantilevered push button includes

an elongated button top component having a first distal end associated with a first user input and a second distal end opposite said first distal end and associated with a second user input separate from said first user input, wherein said elongated button top component is disposed about an exterior surface of an electrical or electronic device and is accessible to a user, and wherein both of said first and second distal ends can be actuated simultaneously by a user during ordinary use of said electrical device,

a first fulcrum located at said first distal end or between said first distal end and the midpoint of said elongated button top component,

a second fulcrum located at said second distal end or between said second distal end and said midpoint of said elongated button top component;

a first electrical contact associated with said first distal end of said elongated button top component, wherein said elongated button top component pivots about said second fulcrum and said first electrical contact is actuated when a user presses on said first distal end, and

a second electrical contact associated with said second distal end of said elongated button top component, wherein said elongated button top component pivots about said first fulcrum and said second electrical contact is actuated when a user presses on said second distal end.

12. The electrical device of claim **11**, wherein said electrical device comprises an electronic device.

13. The electrical device of claim **11**, wherein said first fulcrum is located at said second electrical contact.

14. The electrical device of claim **13**, wherein said second fulcrum is located at said first electrical contact.

15. The electrical device of claim **11**, wherein said elongated button top component is comprised of a substantially rigid material.

16. The electrical device of claim **11**, wherein said first and second electrical contacts comprise dome button type electrical contacts.

17. The electrical device of claim **11**, wherein said simultaneous actuation results in the entire cantilevered push button being pressed inward toward said electrical device.

18. The electrical device of claim **11**, wherein said elongated button top component is disposed through an opening in said housing and does not contact said housing.

19. An electronic device, comprising:

a housing adapted to contain one or more internal electrical device components therein;

a processor located within said housing;

one or more user interface regions having one or more user interface components in communication with said processor; and

one or more cantilevered push buttons located at at least one of said one or more user interface regions and

adapted for accepting a plurality of inputs on said electronic device, wherein each of said one or more cantilevered push buttons includes

an elongated button top component having a first distal end associated with a first user input and a second distal end opposite said first distal end and associated with a second user input separate from said first user input, wherein said elongated button top component is disposed through an opening in said housing, does not contact said housing, and is accessible to a user,

a first fulcrum and a second fulcrum about which said elongated button top pivots when actuated by a user,

a first electrical contact associated with said first distal end of said elongated button top component, wherein said elongated button top component pivots about said second fulcrum and said first electrical contact is actuated when a user presses on said first distal end, and

a second electrical contact associated with said second distal end of said elongated button top component, wherein said elongated button top component pivots about said first fulcrum and said second electrical contact is actuated when a user presses on said second distal end.

20. The electronic device of claim **19**, wherein both of said first and second distal ends can be actuated simultaneously by a user during ordinary use of said electronic device.

* * * * *

Exhibit 8

US00D627790S

(12) United States Design Patent
Chaudhri

(10) Patent No.: US D627,790 S
(45) Date of Patent: ** Nov. 23, 2010

(54) **GRAPHICAL USER INTERFACE FOR A DISPLAY SCREEN OR PORTION THEREOF**

(75) Inventor: **Imran Chaudhri**, San Francisco, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(**) Term: **14 Years**

(21) Appl. No.: **29/283,656**

(22) Filed: **Aug. 20, 2007**

Related U.S. Application Data

(63) Continuation of application No. 29/281,695, filed on Jun. 28, 2007, which is a continuation-in-part of application No. 29/281,507, filed on Jun. 25, 2007, now Pat. No. Des. 608,366, which is a continuation-in-part of application No. 29/281,460, filed on Jun. 23, 2007, now Pat. No. Des. 604,305.

(51) **LOC (9) Cl.** **14-04**
(52) **U.S. Cl.** **D14/486**
(58) **Field of Classification Search** D14/485–95; D18/24–33; D19/6, 52, 9, 10; D20/11; D21/324–33; 715/700–867, 973–77
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,736,199	A *	4/1988	Chadwick et al.	715/781
D295,877	S	5/1988	Wells-Papanek et al.	
5,214,756	A	5/1993	Franklin et al.	
5,345,552	A *	9/1994	Brown	715/784
5,555,364	A	9/1996	Goldstein	
5,642,490	A *	6/1997	Morgan et al.	715/762
5,712,995	A *	1/1998	Cohn	715/792
D390,548	S	2/1998	Maekawa et al.	
D392,266	S	3/1998	Snyder et al.	
5,995,103	A	11/1999	Ashe	
D418,826	S	1/2000	Pavely et al.	
D420,993	S	2/2000	Decker	
D438,213	S	2/2001	Herget et al.	
D441,763	S	5/2001	Kahn et al.	
6,229,532	B1 *	5/2001	Fujii	715/733
D445,428	S	7/2001	Pattenden	
D462,076	S	8/2002	Robbin et al.	

(Continued)

FOREIGN PATENT DOCUMENTS

JP	D1235127	4/2005

OTHER PUBLICATIONS

"PDAs with a built-in wireless LAN, using the Nursing Support System", *Casio News Release,* http://www.casio.co.jp/release/2003/nns.html, dated Jan. 28, 2003.

(Continued)

Primary Examiner—Melanie H Tung
(74) *Attorney, Agent, or Firm*—Sterne, Kessler, Goldstein & Fox PLLC

(57) **CLAIM**

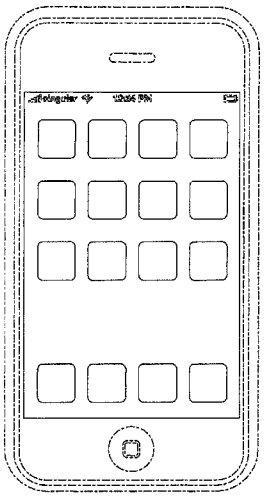
The ornamental design for a graphical user interface for a display screen or portion thereof, as shown and described.

DESCRIPTION

The FIGURE is a front view of a graphical user interface for a display screen or portion thereof showing my new design.

The broken lines of the display screen or portion thereof and other elements form no part of the claimed design.

1 Claim, 1 Drawing Sheet



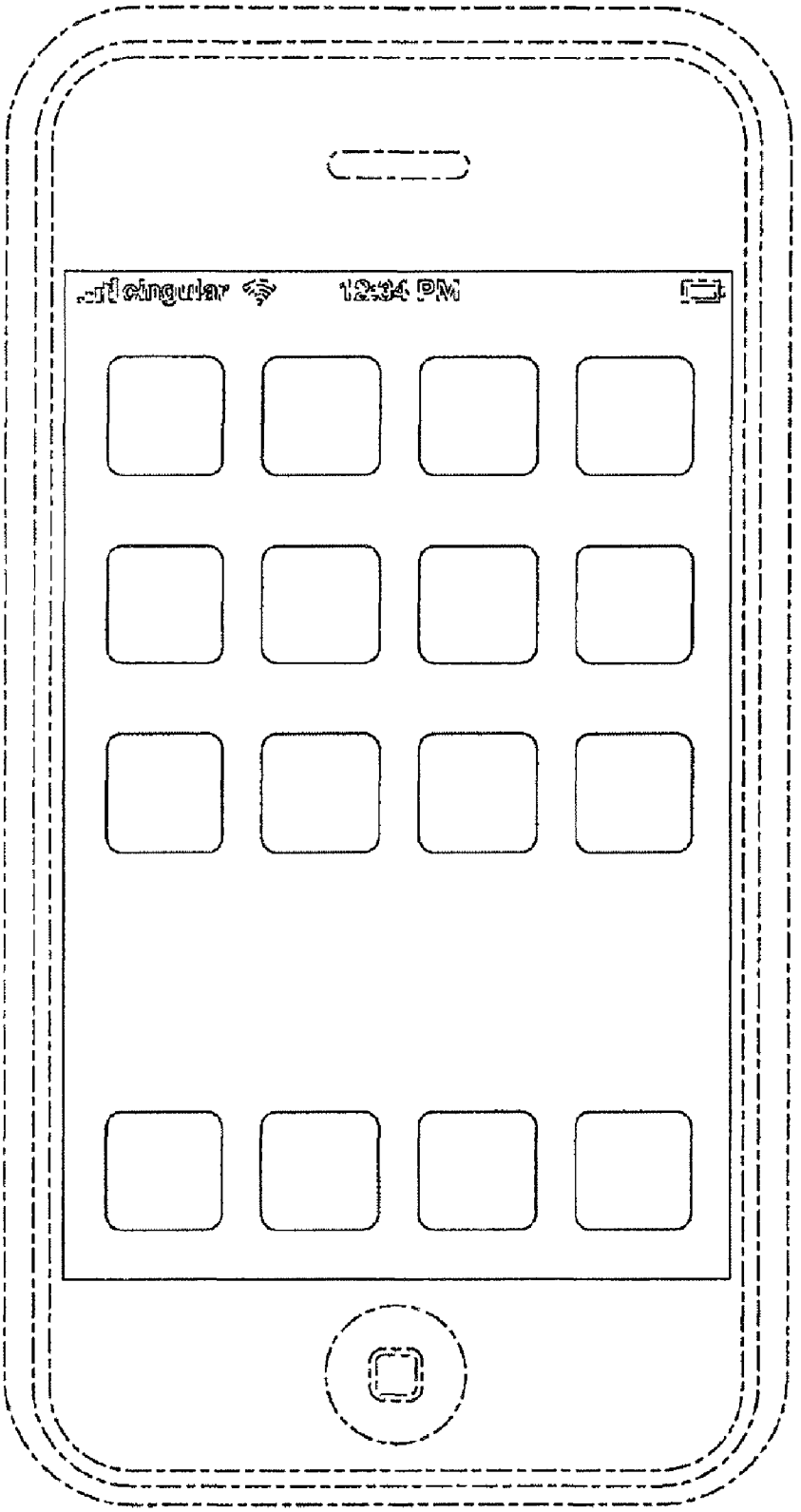
U.S. PATENT DOCUMENTS

6,452,608	B1 *	9/2002	Goken	715/716
6,498,612	B1 *	12/2002	Brown et al.	715/762
D472,244	S	3/2003	Wasko	
6,538,670	B1 *	3/2003	Kido	715/767
6,731,316	B2	5/2004	Herigstad et al.	
6,798,630	B1 *	9/2004	Del Vecchio et al.	361/97
6,807,557	B1 *	10/2004	Novaes et al.	709/201
D507,577	S	7/2005	Totten et al.	
6,983,424	B1	1/2006	Dutta	
7,020,848	B2 *	3/2006	Rosenzweig et al.	715/855
7,058,902	B2 *	6/2006	Iwema et al.	715/810
7,075,512	B1 *	7/2006	Fabre et al.	345/156
7,107,549	B2	9/2006	Deaton et al.	
7,185,289	B1 *	2/2007	Taima	715/810
D540,340	S	4/2007	Cummins	
D544,875	S	6/2007	Wang et al.	
D548,239	S	8/2007	Rimas-Ribikauskas et al.	
D552,620	S	10/2007	Sato et al.	
D555,660	S	11/2007	Noviello et al.	
D559,261	S	1/2008	Jung et al.	
D564,530	S	3/2008	Kim et al.	
D565,586	S	4/2008	Shin et al.	
D565,588	S	4/2008	Sherry	
D568,900	S	5/2008	Seo et al.	
D569,383	S	5/2008	Jung et al.	
D570,358	S	6/2008	Anastasopoulos et al.	
D574,009	S	7/2008	DelPonte	
D574,388	S	8/2008	Armendariz et al.	
D576,171	S	9/2008	Armendariz et al.	
D579,456	S	10/2008	Chen et al.	
D579,946	S	11/2008	Lee et al.	
D580,949	S	11/2008	Durarte	
D581,424	S	11/2008	Hong	
D585,075	S	1/2009	Flynt et al.	
7,487,467	B1 *	2/2009	Kawahara et al.	715/810
D587,720	S	3/2009	Noviello et al.	
7,552,401	B2 *	6/2009	Guido et al.	715/838
7,574,666	B2 *	8/2009	Miyamoto	715/769
7,587,680	B2 *	9/2009	Wada	715/804
7,600,192	B1 *	10/2009	Hashimoto et al.	715/802
7,643,036	B1 *	1/2010	Rees	345/594
7,653,882	B2 *	1/2010	Horentrup et al.	715/821

OTHER PUBLICATIONS

U.S. Appl. No. 29/281,460, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jun. 23, 2007.
U.S. Appl. No. 29/281,507, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jun. 25, 2007.
U.S. Appl. No. 29/281,695, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jun. 28, 2007.
U.S. Appl. No. 29/281,747, Anzures at al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Aug. 10, 2007.
U.S. Appl. No. 29/282,835, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,837, Anzures at al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,838, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,839, Anzures at al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,840, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,841, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,842, Anzures at al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,843, Anzures et al., Icons, Graphical User Interface, and Animated Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 30, 2007.
U.S. Appl. No. 29/284,264, Lemay at al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Sep. 4, 2007.
U.S. Appl. No. 29/302,019, Chaudhri et al., Animated Image for A Portion of Display Screen, filed Jan. 8, 2008.
U.S. Appl. No. 29/302,063, Anzures et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jan. 9, 2008.
U.S. Appl. No. 29/302,217, Anzures et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jan. 11, 2008.
U.S. Appl. No. 29/302,429, Chaudhri at al., Animated Image for A Portion of Display Screen, filed Jan. 16, 2008.
U.S. Appl. No. 29/304,584, Anzures et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Mar. 4, 2008.
U.S. Appl. No. 29/317,755, Anzures et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed May 6, 2008.
U.S. Appl. No. 29/317,758, Christie et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed May 6, 2008.
U.S. Appl. No. 29/318,446 Chaudhri et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed May 20, 2008.
U.S. Appl. No. 29/319,389, Anzures et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jun. 8, 2008.
U.S. Appl. No. 29/319,397, Anzures et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jun. 9, 2008.
U.S. Appl. No. 29/319,434, Anzures et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jun. 9, 2008.
U.S. Appl. No. 29/321,374, Chaudhri et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Jul. 15, 2008.
U.S. Appl. No. 29/324,194, Evans et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Sep. 8, 2008.
U.S. Appl. No. 29/324,195, Chaudhri et al., Graphical User Interfaces for a Display Screen or Portion Thereof, filed Sep. 8, 2008.
U.S. Appl. No. 29/324,196, Chaudhri et al., Graphical User Interfaces and Icons for a Display Screen or Portion Thereof, filed Sep. 8, 2008.
U.S. Appl. No. 29/324,935, Anzures et al., Graphical User Interfaces and Icons for a Display Screen or Portion Thereof, filed Sep. 23, 2008.
U.S. Appl. No. 29/326,396, Christie et al., Icon for a Display Screen or Portion Thereof, filed Oct. 16, 2008.
U.S. Appl. No. 29/333,233 Graphical User Interfaces and Icons for a Display Screen or Portion Thereof, filed Mar. 4, 2009.
U.S. Appl. No. 29/333,559, Animated Image, Graphical User Interfaces and Icons for a Display Screen or Portion Thereof, filed Mar. 11, 2009.
U.S. Appl. No. 29/333,867, Animated Image, Graphical User Interfaces and Icon for a Display Screen or Portion Thereof, filed Mar. 16, 2009.
U.S. Appl. No. 29/335,282, Sepulveda et al., and Icon for a Display Screen or Portion Thereof, filed Apr. 13, 2009.

* cited by examiner



FIGURE

Exhibit 9

US00D602016S

(12) **United States Design Patent**
Andre et al.

(10) **Patent No.: US D602,016 S**
(45) **Date of Patent: ** Oct. 13, 2009**

(54) **ELECTRONIC DEVICE**

(75) Inventors: **Bartley K. Andre**, Menlo Park, CA (US); **Daniel J. Coster**, San Francisco, CA (US); **Daniele De Iuliis**, San Francisco, CA (US); **Richard P. Howarth**, San Francisco, CA (US); **Jonathan P. Ive**, San Francisco, CA (US); **Steve Jobs**, Palo Alto, CA (US); **Duncan Robert Kerr**, San Francisco, CA (US); **Shin Nishibori**, Portola Valley, CA (US); **Matthew Dean Rohrbach**, San Francisco, CA (US); **Peter Russell-Clarke**, San Francisco, CA (US); **Douglas B. Satzger**, Menlo Park, CA (US); **Christopher J. Stringer**, Woodside, CA (US); **Eugene Antony Whang**, San Francisco, CA (US); **Rico Zorkendorfer**, San Francisco, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(**) Term: **14 Years**

(21) Appl. No.: **29/319,377**

(22) Filed: **Jun. 6, 2008**

Related U.S. Application Data

(63) Continuation-in-part of application No. 29/306,334, filed on Apr. 7, 2008, and a continuation-in-part of application No. 29/306,950, filed on Apr. 18, 2008, which is a continuation of application No. 29/306,334, filed on Apr. 7, 2008.

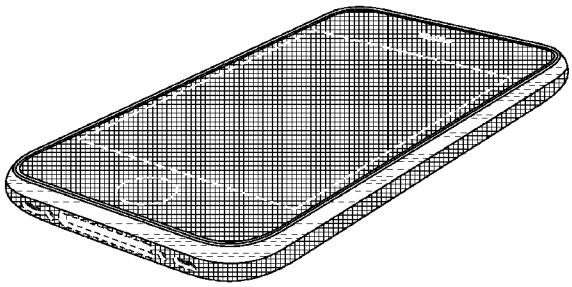
(51) **LOC (9) Cl.** **14-02**
(52) **U.S. Cl.** **D14/341**

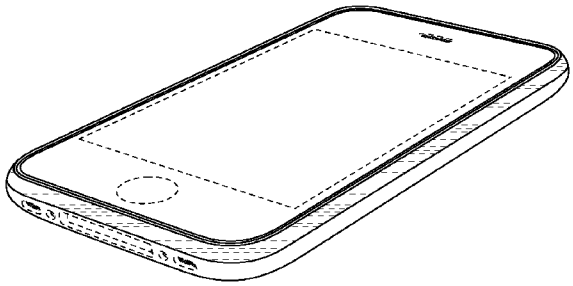
(58) **Field of Classification Search** D14/341–347, D14/420, 426, 427, 432, 439–441, 448, 496, D14/125, 129–130, 137, 138, 147, 156, 218, D14/247–248, 250, 389; D10/65, 104; D18/6–7; D21/329, 686; D6/596, 601, 605; 455/90.3, 455/6.1, 556.2, 575.1, 575.3, 575.4; 379/433.01, 379/433.04, 433.06, 433.07; 361/814; 341/22; 345/169; 346/173
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

D337,569	S		7/1993	Kando	
D420,354	S		2/2000	Morales	
D424,535	S		5/2000	Peltola	
6,297,795	B1	*	10/2001	Kato et al.	345/684
D498,754	S		11/2004	Blyth	
D504,889	S		5/2005	Andre et al.	
D505,950	S		6/2005	Summit et al.	
D514,590	S		2/2006	Naruki	
D519,116	S		4/2006	Tanaka et al.	
D520,020	S		5/2006	Senda et al.	
D527,379	S	*	8/2006	Oe et al.	D14/248
D528,542	S		9/2006	Luminosu et al.	
D528,561	S		9/2006	Ka-Wei et al.	
D529,045	S		9/2006	Shin	
D532,791	S		11/2006	Kim	
D534,143	S		12/2006	Lheem	
D535,281	S		1/2007	Yang	
D536,691	S		2/2007	Park	
D536,962	S		2/2007	Tanner	
D541,298	S		4/2007	Andre et al.	
D541,299	S	*	4/2007	Andre et al.	D14/496
D546,313	S		7/2007	Lheem	
D548,732	S		8/2007	Cebe et al.	
D548,747	S		8/2007	Andre et al.	
D556,211	S		11/2007	Howard	
D557,238	S		12/2007	Kim	
D557,259	S	*	12/2007	Hirsch	D14/217





D558,460	S		1/2008	Yu et al.	
D558,756	S		1/2008	Andre et al.	
D558,757	S		1/2008	Andre et al.	
D558,758	S		1/2008	Andre et al.	
D558,792	S		1/2008	Chigira	
D559,220	S		1/2008	Lee et al.	
D560,192	S		1/2008	Lee et al.	
D560,683	S		1/2008	Lee	
D560,686	S		1/2008	Kim	
D561,153	S		2/2008	Hong et al.	
D561,204	S		2/2008	Toh	
D561,782	S		2/2008	Kim	
D562,285	S		2/2008	Lim	
D563,432	S		3/2008	Kim	
D563,929	S		3/2008	Park	
D568,309	S	*	5/2008	Cebe et al.	D14/341
D575,259	S		8/2008	Kim et al.	
D578,500	S	*	10/2008	Lee et al.	D14/138 G
D579,898	S	*	11/2008	Lee et al.	D14/138 G
D580,387	S		11/2008	Andre et al.	
D581,384	S	*	11/2008	Kim et al.	D14/138 AD
D581,922	S		12/2008	Andre et al.	
D583,345	S		12/2008	Kim et al.	
D584,272	S	*	1/2009	Chung et al.	D14/138 G
D584,303	S	*	1/2009	Andre et al.	D14/341
D584,646	S	*	1/2009	Sanguinetti	D10/65
D584,711	S		1/2009	Kim et al.	
D584,738	S	*	1/2009	Kim et al.	D14/496
D585,411	S	*	1/2009	Eaton	D14/138 G
D588,146	S	*	3/2009	Tsuboi	D14/474
D591,252	S	*	4/2009	Kim et al.	D14/138 G
D592,628	S	*	5/2009	Kim et al.	D14/138 AD
2003/0125079	A1	*	7/2003	Park et al.	455/556
2004/0132499	A1		7/2004	Abe	
2004/0166907	A1		8/2004	Yajima	
2004/0223004	A1		11/2004	Lincke et al.	
2007/0082718	A1		4/2007	Yoon et al.	
2008/0004085	A1		1/2008	Jung et al.	

FOREIGN PATENT DOCUMENTS

JP	D1250487	9/2005
KR	30-0394921	12/2005

OTHER PUBLICATIONS

Samsung F700, announced Feb. 2007, [online], [retrieved on Feb. 8, 2007]. Retrieved from Internet ,<URL:http://www.gsmarena.com>.
Philips S900, announced Jun. 2006, [online], [retrieved on Feb. 20, 2007]. Retrieved from Internet ,<URL:http://www.gsmarena.com>.
LG KE850 Prada, announced Jan. 2007, [online], [retrieved on Feb. 20, 2007]. Retrieved from Internet ,<URL:http://www.gsmarena.com>.
Apple iPhone, announced Jan. 2007, [online], [retrieved on Mar. 12, 2007]. Retrieved from Internet ,<URL:http://www.gsmarena.com>.
Tinnos PDA, posted May 19, 2006, [online], [retrieved on Aug. 22, 2007]. Retrieved from Internet ,<URL:http://www.mobilewhack.com>.
Meizu M8, posted Jan., 29, 2007, [online], [retrieved on Sep. 13, 2007]. Retrieved from Internet ,<URL:http://www.engadget.com>.
Hilon LG DMB MP3 FM35,posted Jun. 26, 2006, [online], [retrieved on Dec. 26, 2008]. Retrieved from Internet, <URL: http://hilon.com.cn.autobak/a8810005549>.
eFashion Magazine, No. 12 (2004), p. 60.
eFashion Magazine, No. 119 (Jun. 1, 2005), p. 45.
eFashion Magazine, No. 172 (Apr. 2006), p. 26.
U.S. Appl. No. 29/282,831, Andre et al., Electronic Device, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,832, Andre et al., Electronic Device, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,833, Andre et al., Electronic Device, filed Jul. 30, 2007.
U.S. Appl. No. 29/284,187, Andre et al., Electronic Device, filed Aug. 31, 2007.
U.S. Appl. No. 29/284,188, Andre et al., Electronic Device, filed Aug. 31, 2007.
U.S. Appl. No. 29/284,269, Andre et al., Electronic Device, filed Sep. 4, 2007.
U.S. Appl. No. 29/284,271, Andre et al., Electronic Device, filed Sep. 4, 2007.
U.S. Appl. No. 29/284,272, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,276, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,288, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,308, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,310, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,312, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/306,334, Andre et al., Electronic Device, filed Apr. 7, 2008.
U.S. Appl. No. 29/319,239, Andre et al., Electronic Device, filed Jun. 5, 2008.
U.S. Appl. No. 29/319,433, Andre et al., Electronic Device, filed Jun. 9, 2008.
U.S. Appl. No. 29/306,950, Andre et al., Electronic Device, filed Apr. 18, 2008.

* cited by examiner

Primary Examiner—Cathron C Brooks
Assistant Examiner—Barbara Fox
(74) *Attorney, Agent, or Firm*—Sterne, Kessler, Goldstein & Fox P.L.L.C.

(57) **CLAIM**

The ornamental design for an electronic device, as shown and described.

DESCRIPTION

FIG. **1** is a front perspective view of an electronic device showing our new design;

FIG. **2** is a rear perspective view thereof;

FIG. **3** is a front view thereof;

FIG. **4** is a rear view thereof;

FIG. **5** is a side view thereof;

FIG. **6** is another side view thereof;

FIG. **7** is a top view thereof;

FIG. **8** is a bottom view thereof;

FIG. **9** is a front perspective view of another embodiment of an electronic device showing our new design;

FIG. **10** is a rear perspective view thereof;

FIG. **11** is a front view thereof;

FIG. **12** is a rear view thereof;

FIG. **13** is a side view thereof;

FIG. **14** is another side view thereof;

FIG. **15** is a top view thereof; and,

FIG. **16** is a bottom view thereof.

The face, back and side surfaces of the electronic device shown in the FIGS. **1**–**8** are illustrated with color designations. The face and the back surface are black. The side surfaces are silver, grey or chrome.

The face, back and side surfaces of the electronic device shown in the FIGS. **9**–**16** are illustrated with color designa-

tions. The face and the back surface are white. The side surfaces are silver, grey or chrome.

The broken lines in the Figures represent unclaimed portions of the electronic device andb form no part of the claimed design. The shade lines in the Figures show contour and not surface ornamentation.

1 Claim, 4 Drawing Sheets

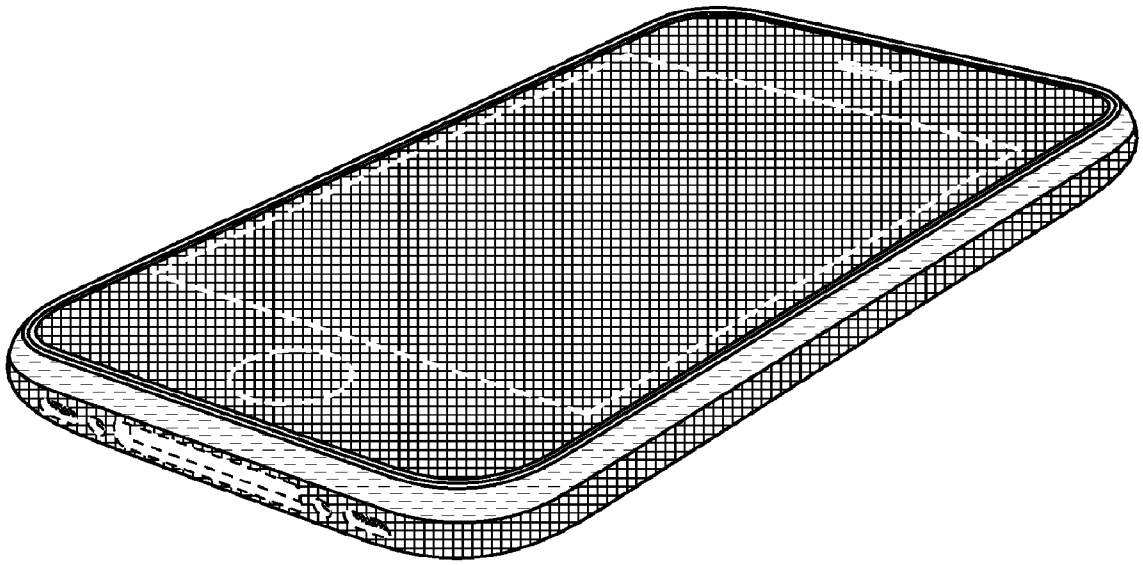


FIG. 1

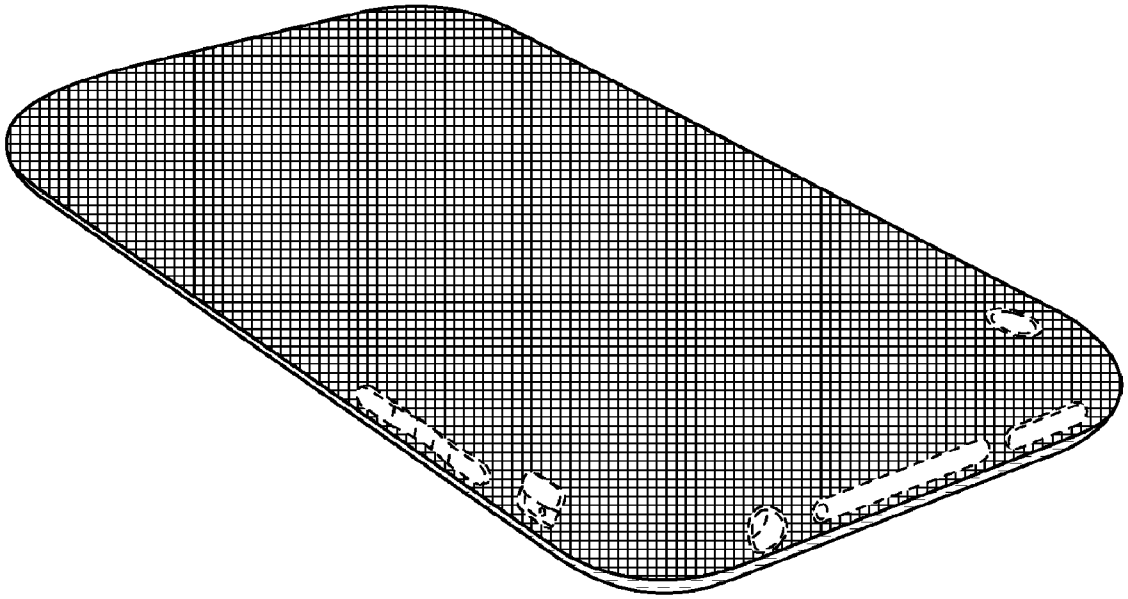


FIG. 2

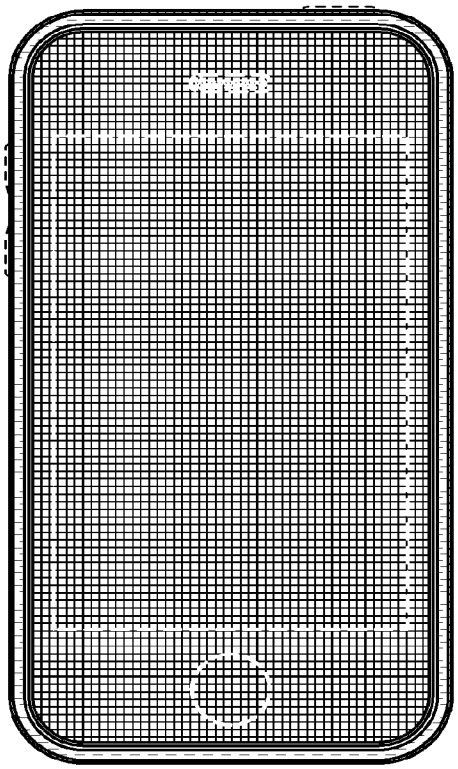


FIG. 3

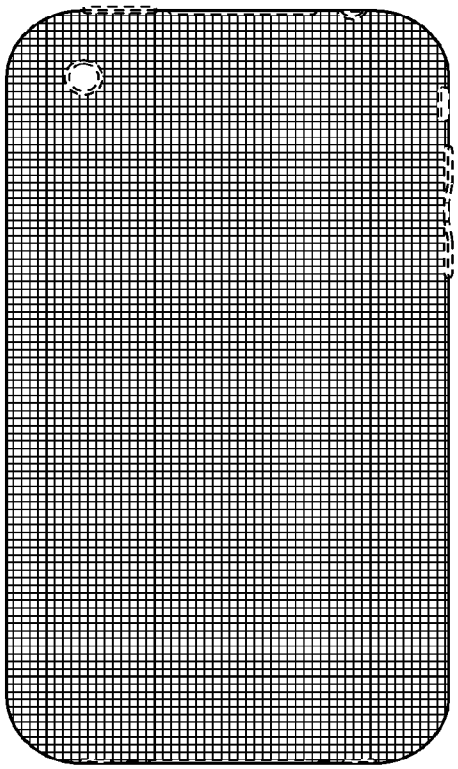


FIG. 4

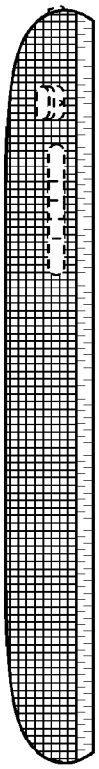


FIG. 5

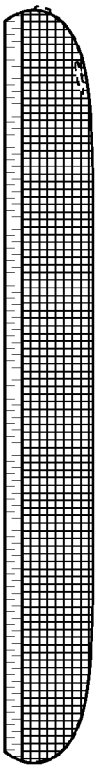


FIG. 6

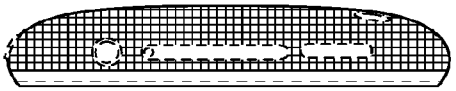


FIG. 7

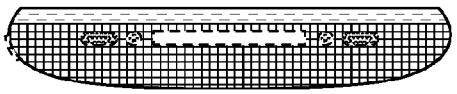


FIG. 8

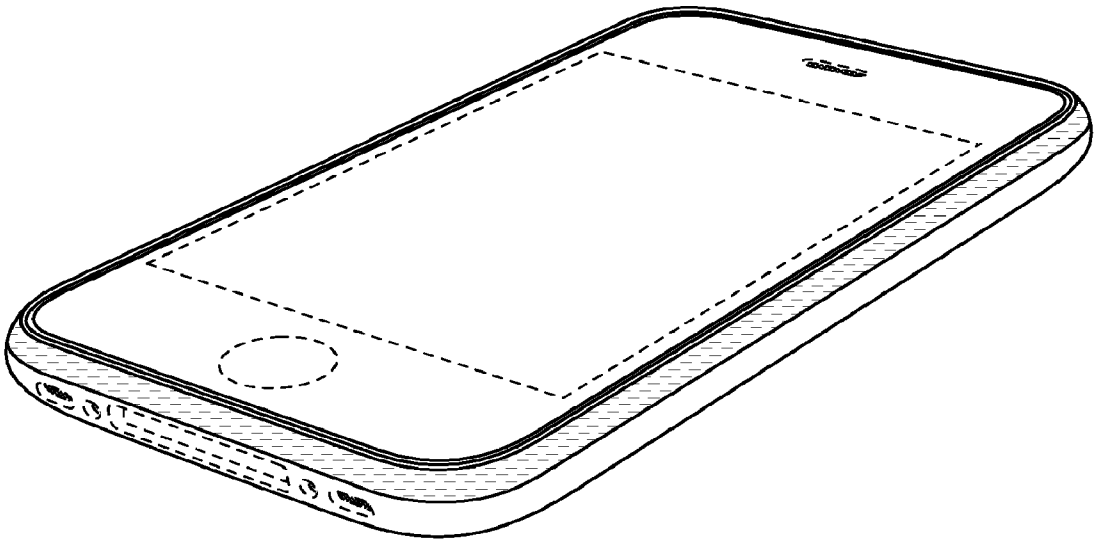


FIG. 9

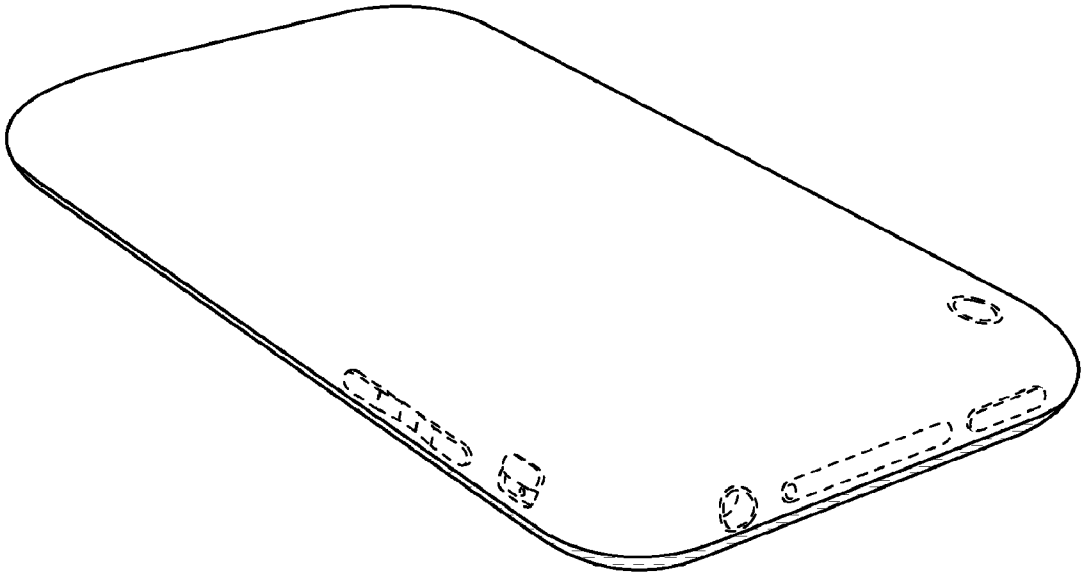


FIG. 10

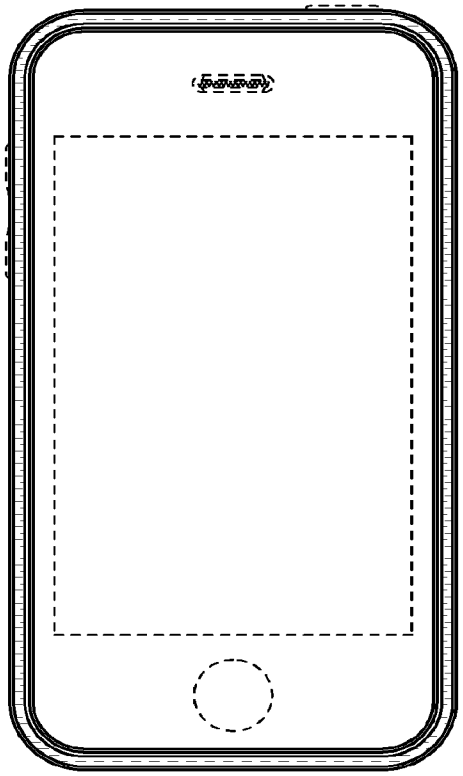


FIG. 11

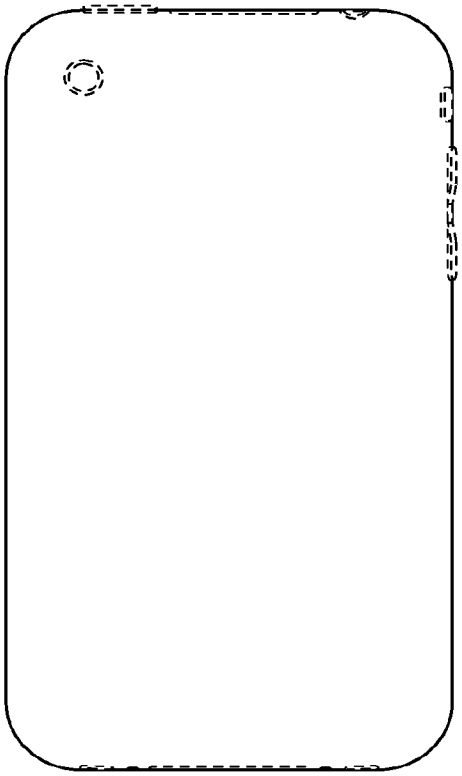


FIG. 12

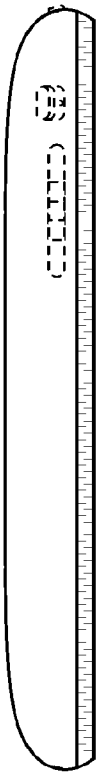


FIG. 13

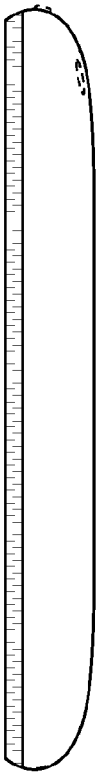


FIG. 14

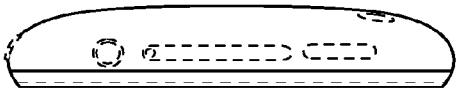


FIG. 15

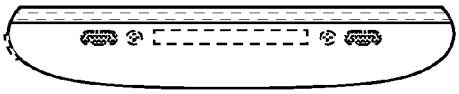


FIG. 16

Exhibit 10

US00D618677S

(12) United States Design Patent

Andre et al.

(10) Patent No.: US D618,677 S

(45) Date of Patent: ** *Jun. 29, 2010

(54) **ELECTRONIC DEVICE**

(75) Inventors: **Bartley K. Andre**, Menlo Park, CA (US); **Daniel J. Coster**, San Francisco, CA (US); **Daniele De Iuliis**, San Francisco, CA (US); **Richard P. Howarth**, San Francisco, CA (US); **Jonathan P. Ive**, San Francisco, CA (US); **Steve Jobs**, Palo Alto, CA (US); **Duncan Robert Kerr**, San Francisco, CA (US); **Shin Nishibori**, Portola Valley, CA (US); **Matthew Dean Rohrbach**, San Francisco, CA (US); **Douglas B. Satzger**, Menlo Park, CA (US); **Calvin Q. Seid**, Palo Alto, CA (US); **Christopher J. Stringer**, Woodside, CA (US); **Eugene Antony Whang**, San Francisco, CA (US); **Rico Zorkendorfer**, San Francisco, CA (US)

(73) Assignee: **Apple Inc.**, Cupertino, CA (US)

(*) Notice: This patent is subject to a terminal disclaimer.

(**) Term: **14 Years**

(21) Appl. No.: **29/328,018**

(22) Filed: **Nov. 18, 2008**

Related U.S. Application Data

(60) Division of application No. 29/282,834, filed on Jul. 30, 2007, now Pat. No. Des. 581,922, which is a continuation of application No. 29/270,888, filed on Jan. 5, 2007, now Pat. No. Des. 558,758.

(51) **LOC (9) Cl.** **14-02**

(52) **U.S. Cl.** **D14/341**; D14/248; D14/203.7

(58) **Field of Classification Search** D14/341, D14/342, 343, 344, 345, 346, 347, 420, 426, D14/427, 432, 439, 440, 441, 448, 496, 125, D14/137, 129, 130, 138, 250, 389, 147, 218, D14/247, 248, 156; D10/65, 104; D13/168; D18/6, 7; D21/329, 686; 455/90.3, 556.1, 455/556.2, 575.1, 575.3, 575.4; 379/433.01, 379/433.04, 433.06, 433.07; 361/814; 341/22; 345/169, 173; D6/596, 601, 605

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

D289,873 S 5/1987 Gemmell et al.

(Continued)

FOREIGN PATENT DOCUMENTS

EM 000569157-0005 5/2006

(Continued)

OTHER PUBLICATIONS

U.S. Appl. No. 29/282,831, Andre et al., Electronic Device, filed Jul. 30, 2007.

(Continued)

Primary Examiner—Cathron C Brooks

Assistant Examiner—Angela J Lee

(74) *Attorney, Agent, or Firm*—Sterne, Kessler, Goldstein & Fox PLLC

(57) **CLAIM**

The ornamental design of an electronic device, as shown and described.

DESCRIPTION

FIG. **1** is a front perspective view of an electronic device in accordance with the present invention;

FIG. **2** is a rear perspective view thereof;

FIG. **3** is a front view thereof;

FIG. **4** is a rear view thereof;

FIG. **5** is a top view thereof;

FIG. **6** is bottom view thereof;

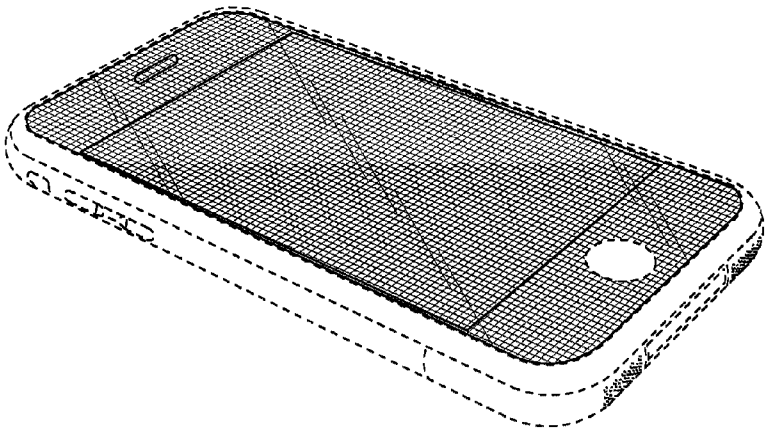
FIG. **7** is a left side view thereof; and,

FIG. **8** is a right side view thereof.

The claimed surface of the electronic device is illustrated with the color designation for the color black.

The electronic device is not limited to the scale shown herein. As indicated in the title, the article of manufacture to which the ornamental design has been applied is an electronic device, media player (e.g., music, video and/or game player), media storage device, a personal digital assistant, a communication device (e.g., cellular phone), a novelty item or toy.

1 Claim, 2 Drawing Sheets



U.S. PATENT DOCUMENTS

D337,569	S		7/1993	Kando	
D410,440	S	*	6/1999	Carnell	D14/346
D424,535	S		5/2000	Peltola	
D455,433	S	*	4/2002	Alviar et al.	D14/441
D456,023	S		4/2002	Andre et al.	
D458,252	S	*	6/2002	Palm et al.	D14/343
D469,413	S	*	1/2003	To et al.	D14/126
D489,731	S		5/2004	Huang	
D498,754	S		11/2004	Blyth	
D499,423	S		12/2004	Bahroocha et al.	
D502,173	S		2/2005	Jung et al.	
D504,889	S		5/2005	Andre et al.	
D505,950	S		6/2005	Summit et al.	
D507,003	S		7/2005	Pai et al.	
D514,121	S		1/2006	Johnson	
D514,590	S		2/2006	Naruki	
D519,116	S		4/2006	Tanaka et al.	
D519,523	S		4/2006	Chiu et al.	
D520,020	S		5/2006	Senda et al.	
7,042,712	B2	*	5/2006	Ghosh et al.	361/679.03
D528,542	S		9/2006	Luminosu et al.	
D528,561	S		9/2006	Ka-Wei et al.	
D529,045	S		9/2006	Shin	
D532,791	S		11/2006	Kim	
D534,143	S		12/2006	Lheem	
D535,281	S		1/2007	Yang	
D536,691	S		2/2007	Park	
D536,962	S		2/2007	Tanner	
D538,822	S		3/2007	Andre et al.	
D541,298	S		4/2007	Andre et al.	
D541,299	S		4/2007	Andre et al.	
D541,785	S		5/2007	Hwang et al.	
D546,313	S		7/2007	Lheem	
D548,732	S		8/2007	Cebe et al.	
D548,747	S		8/2007	Andre et al.	
D554,098	S		10/2007	Lee	
D556,211	S		11/2007	Howard	
D557,238	S		12/2007	Kim	
7,303,424	B2		12/2007	Tu et al.	
D558,460	S		1/2008	Yu et al.	
D558,756	S		1/2008	Andre et al.	
D558,757	S		1/2008	Andre et al.	
D558,758	S		1/2008	Andre et al.	
D558,792	S		1/2008	Chigira	
D559,220	S		1/2008	Lee et al.	
D560,192	S		1/2008	Lee et al.	
D560,683	S		1/2008	Lee	
D560,686	S		1/2008	Kim	
D561,153	S		2/2008	Hong et al.	
D561,204	S		2/2008	Toh	
D561,782	S		2/2008	Kim	
D562,285	S		2/2008	Lim	
D563,432	S		3/2008	Kim	
D563,929	S		3/2008	Park	
D568,309	S	*	5/2008	Cebe et al.	D14/341
D573,143	S	*	7/2008	Park et al.	D14/341
D574,015	S	*	7/2008	Masui	D14/496
7,409,059	B2	*	8/2008	Fujisawa	379/433.13
D579,930	S	*	11/2008	Maskatia	D14/341
D580,387	S		11/2008	Andre et al.	
D581,922	S		12/2008	Andre et al.	
D584,738	S	*	1/2009	Kim et al.	D14/496
D586,800	S	*	2/2009	Andre et al.	D14/341
D589,979	S	*	4/2009	Andre et al.	D14/496
2004/0132499	A1		7/2004	Abe	
2004/0166907	A1		8/2004	Yajima	
2004/0223004	A1		11/2004	Lincke et al.	
2005/0130715	A1		6/2005	Fujisawa	
2006/0281501	A1		12/2006	Zuo et al.	
2007/0082718	A1		4/2007	Yoon et al.	
2008/0004083	A1	*	1/2008	Ohki et al.	455/566
2008/0004085	A1		1/2008	Jung et al.	

FOREIGN PATENT DOCUMENTS

JP	D1159881	12/2002
JP	D1250487	9/2005
JP	2004 290256	10/2005
KR	30-0394921	12/2005
KR	30-0422221	8/2006

OTHER PUBLICATIONS

U.S. Appl. No. 29/282,832, Andre et al., Electronic Device, filed Jul. 30, 2007.
U.S. Appl. No. 29/282,833, Andre et al., Electronic Device, filed Jul. 30, 2007.
U.S. Appl. No. 29/284,187, Andre et al., Electronic Device, filed Aug. 31, 2007.
U.S. Appl. No. 29/284,188, Andre et al., Electronic Device, filed Aug. 31, 2007.
U.S. Appl. No. 29/284,269, Andre et al., Electronic Device, filed Sep. 4, 2007.
U.S. Appl. No. 29/284,272, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,276, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,288, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,308, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/284,310, Andre et al., Electronic Device, filed Sep. 5, 2007.
Hilon LG DMB MP3 FM35, posted on Jun. 26, 2006, [online], [retrieved on Dec. 26, 2008]. Retrieved from Internet, <URL:http://hilon.com.en.autobak/a8810005549>.
eFashion Magazine, 2004, vol. No. 12, p. 60, China.
eFashion Magazine, Jun. 1, 2005, vol. No. 119, p. 45, China.
eFashion Magazine, Apr. 2006, vol. No. 142, p. 26, China.
Notification and Request for Invalidation of Chinese Patent ZL200730148767.X, Patent Reexamination Board of the State of Intellectual Property Office of the PRC and English translation, mailed Dec. 11, 2008, 19 pages.
Notification and Request for Invalidation of Chinese Patent ZL200730148751.9, Patent Reexamination Board of the State of Intellectual Property Office of the PRC and English translation, mailed Dec. 11, 2008, 19 pages.
Notification and Request for Invalidation of Chinese Patent ZL200730148719.0, Patent Reexamination Board of the State of Intellectual Property Office of the PRC and English translation, mailed Dec. 11, 2008, 19 pages.
U.S. Appl. No. 29/306,334, Andre et al., Electronic Device, filed Apr. 7, 2008.
U.S. Appl. No. 29/306,950, Andre et al., Electronic Device, filed Apr. 18, 2008.
U.S. Appl. No. 29/319,239, Andre et al., Electronic Device, filed Jun. 5, 2008.
U.S. Appl. No. 29/319,433, Andre et al., Electronic Device, filed Jun. 9, 2008.
U.S. Appl. No. 29/324,130, Andre et al., Electronic Device, filed Sep. 5, 2008.
U.S. Appl. No. 29/324,137, Andre et al., Electronic Device, filed Sep. 6, 2008.
U.S. Appl. No. 29/324,262, Andre et al., Electronic Device, filed Sep. 9, 2008.
U.S. Appl. No. 29/284,271, Andre et al., Electronic Device, filed Sep. 4, 2007.
U.S. Appl. No. 29/284,312, Andre et al., Electronic Device, filed Sep. 5, 2007.
U.S. Appl. No. 29/319,377, Andre et al., Electronic Device, filed Jun. 6, 2008.
Samsung F700, announced Feb. 2007, [online], [retrieved on Feb. 8, 2007]. Retrieved from Internet, <URL:http://www.gsmarena.com>.

Philips S900, announced Jun. 2006, [online], [retrieved on Feb. 20, 2007]. Retrieved from Internet, <URL:http://www.gsmarena.com>.
LG DE850 Prada, announced Jan. 2007, [online], [retrieved on Feb. 20, 2007]. Retrieved from Internet, <URL:http://www.gsmarena.com>.
Apple iPhone, announced Jan. 2007, [online], [retrieved on Mar. 12, 2007]. Retrieved from Internet, <URL:http//www.gsmarena.com>.
Tinnos PDA, posted May 19, 2006, [online], [retrieved on Aug. 22, 2007]. Retrieved from Internet, <URL:http//www.mobilewhack.com>.
Meizu M8, posted Jan. 29, 2007, [online], [retrieved on Sep. 13, 2007]. Retrieved from Internet, <URL:http://www.engadget.com>.

* cited by examiner

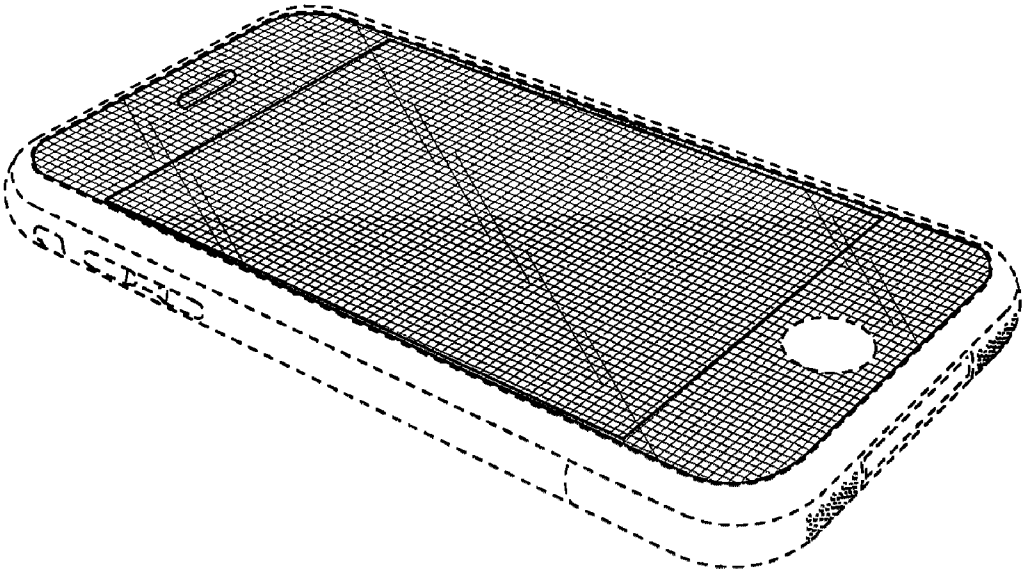


FIG. 1

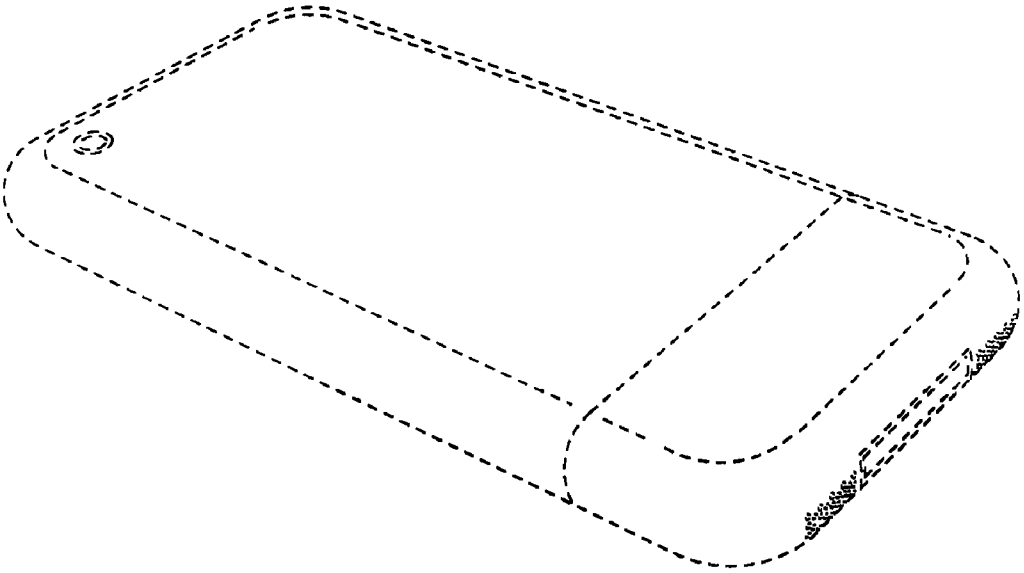


FIG. 2

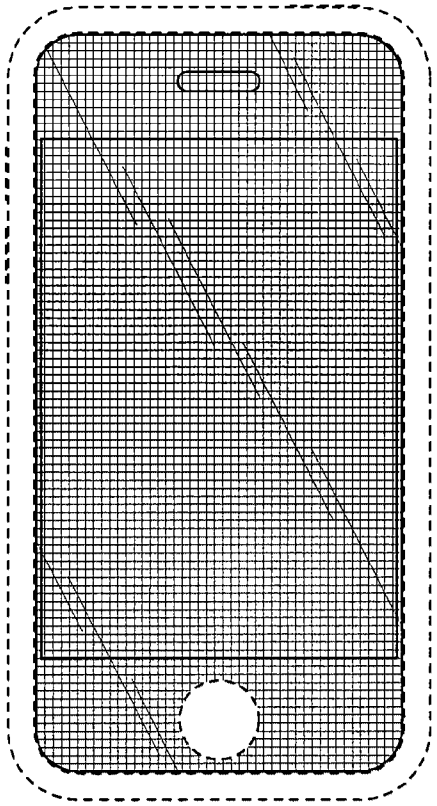


FIG. 3

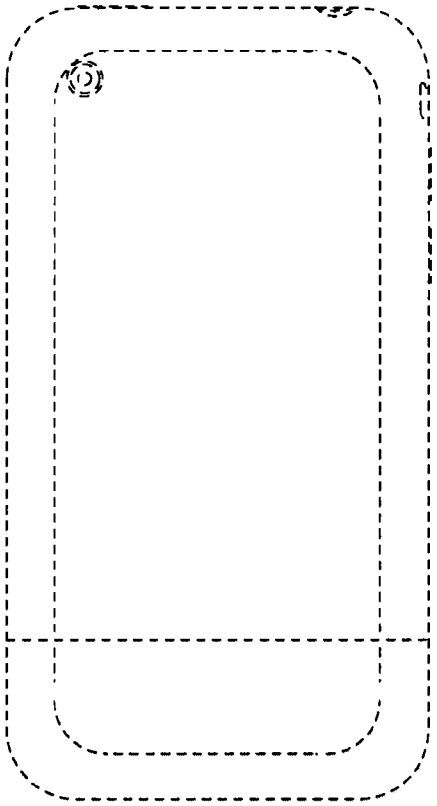


FIG. 4

FIG. 7

FIG. 8

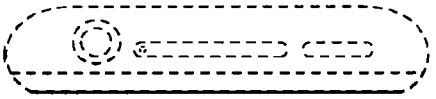


FIG. 5

FIG. 6

Ex. 11

Int. Cl.: 9

Prior U.S. Cls.: 21, 23, 26, 36 and 38

United States Patent and Trademark Office

Reg. No. 3,470,983

Registered July 22, 2008

TRADEMARK
PRINCIPAL REGISTER

APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: HANDHELD MOBILE DIGITAL ELECTRONIC DEVICES COMPRISED OF A MOBILE PHONE, DIGITAL AUDIO AND VIDEO PLAYER, HANDHELD COMPUTER, PERSONAL DIGITAL ASSISTANT, ELECTRONIC PERSONAL ORGANIZER, POCKET COMPUTER FOR NOTE-TAKING, ELECTRONIC CALENDAR, CALCULATOR, AND CAMERA, AND CAPABLE OF PROVIDING ACCESS TO THE INTERNET AND SENDING AND RECEIVING ELECTRONIC MAIL, DIGITAL AUDIO, VIDEO, TEXT, IMAGES, GRAPHICS AND MULTIMEDIA FILES, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 6-29-2007; IN COMMERCE 6-29-2007.

NO CLAIM IS MADE TO THE EXCLUSIVE RIGHT TO USE "SMS", APART FROM THE MARK AS SHOWN.

THE COLOR(S) BLACK, BLUE, BROWN, BROWN-GRAY, GRAY-GREEN, GREEN, ORANGE, RED, SILVER, TAN, WHITE AND YELLOW IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF THE CONFIGURATION OF A RECTANGULAR HANDHELD MOBILE DIGITAL ELECTRONIC DEVICE WITH ROUNDED SILVER EDGES, A BLACK FACE, AND AN ARRAY OF 16 SQUARE ICONS WITH ROUNDED EDGES. THE TOP 12 ICONS APPEAR ON A BLACK BACKGROUND, AND THE BOTTOM 4 APPEAR ON A SILVER BACKGROUND. THE FIRST ICON DEPICTS THE LETTERS "SMS" IN GREEN INSIDE A WHITE SPEECH BUBBLE ON A GREEN BACKGROUND; THE SECOND ICON IS WHITE WITH A THIN RED STRIPE AT THE TOP; THE THIRD ICON DEPICTS A SUNFLOWER WITH YELLOW PETALS, A BROWN CENTER, AND A GREEN STEM IN FRONT OF A BLUE SKY; THE FOURTH ICON DEPICTS A CAMERA LENS WITH A BLACK BARREL AND BLUE GLASS ON A SILVER BACKGROUND; THE FIFTH ICON DEPICTS A TAN TELEVISION CONSOLE WITH BROWN KNOBS AND A GRAY-GREEN SCREEN; THE SIXTH ICON DEPICTS A WHITE GRAPH LINE ON A BLUE BACKGROUND; THE SEVENTH ICON DEPICTS A MAP WITH YELLOW AND ORANGE ROADS, A PIN WITH A RED HEAD, AND A RED-AND- BLUE ROAD SIGN WITH THE NUMERAL "280" IN WHITE; THE EIGHTH ICON DEPICTS AN ORANGE SUN ON A BLUE BACKGROUND, WITH THE TEMPERATURE IN WHITE; THE NINTH ICON DEPICTS A WHITE CLOCK WITH BLACK AND RED HANDS AND NUMERALS ON A BLACK BACKGROUND; THE TENTH ICON DEPICTS THREE BROWN-GRAY CIRCLES AND ONE ORANGE CIRCLE ON A BLACK BACKGROUND WITH A WHITE BORDER, WITH THE MATHEMATICAL SYMBOLS FOR ADDITION, SUBTRACTION, MULTIPLICATION, AND THE EQUAL SIGN DISPLAYED IN WHITE ON THE CIRCLES; THE ELEVENTH ICON DEPICTS A PORTION OF A YELLOW NOTEPAD WITH BLUE AND RED RULING, WITH BROWN BINDING AT THE TOP; THE TWELFTH ICON DEPICTS THREE SILVER GEARS OVER A THATCHED BLACK-AND-SILVER BACKGROUND; THE THIRTEENTH ICON DEPICTS A WHITE TELEPHONE RECEIVER AGAINST A GREEN BACKGROUND; THE FOURTEENTH ICON DEPICTS A WHITE ENVELOPE OVER A BLUE SKY

WITH WHITE CLOUDS; THE FIFTEENTH ICON DEPICTS A WHITE COMPASS WITH A WHITE-AND-RED NEEDLE OVER A BLUE MAP; THE SIXTEENTH ICON DEPICTS THE DISTINCTIVE CONFIGURATION OF APPLICANT'S MEDIA PLAYER DEVICE IN WHITE OVER AN ORANGE BACKGROUND.

SEC. 2(F).

SER. NO. 77-303,282, FILED 10-12-2007.

SKYE YOUNG, EXAMINING ATTORNEY

Ex. 12

Int. Cl.: 9

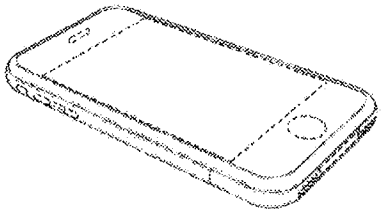
Prior U.S. Cls.: 21, 23, 26, 36 and 38

Reg. No. 3,457,218

United States Patent and Trademark Office

Registered July 1, 2008

TRADEMARK
PRINCIPAL REGISTER



APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP, MS: 3TM
CUPERTINO, CA 95014

FOR: HANDHELD MOBILE DIGITAL ELECTRONIC DEVICES COMPRISED OF A MOBILE PHONE, DIGITAL AUDIO AND VIDEO PLAYER, HANDHELD COMPUTER, PERSONAL DIGITAL ASSISTANT, ELECTRONIC PERSONAL ORGANIZER, POCKET COMPUTER FOR NOTE-TAKING, ELECTRONIC CALENDAR, CALCULATOR, AND CAMERA, AND CAPABLE OF PROVIDING ACCESS TO THE INTERNET AND SENDING AND RECEIVING ELECTRONIC MAIL, DIGITAL AUDIO, VIDEO, TEXT, IMAGES, GRAPHICS AND MULTIMEDIA FILES, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 1-9-2007; IN COMMERCE 6-29-2007.

THE MARK CONSISTS OF THE CONFIGURATION OF A RECTANGULAR HANDHELD MOBILE DIGITAL ELECTRONIC DEVICE WITH ROUNDED CORNERS. THE MATTER SHOWN IN BROKEN LINES IS NOT PART OF THE MARK.

SEC. 2(F).

SER. NO. 77-303,256, FILED 10-12-2007.

SKYE YOUNG, EXAMINING ATTORNEY

Ex. 13

Int. Cl.: 9

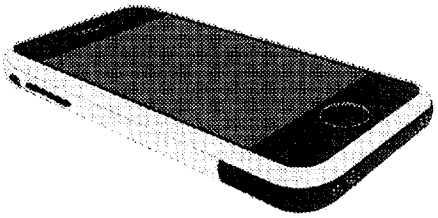
Prior U.S. Cls.: 21, 23, 26, 36 and 38

Reg. No. 3,475,327

United States Patent and Trademark Office

Registered July 29, 2008

TRADEMARK
PRINCIPAL REGISTER



APPLE, INC. (CALIFORNIA CORPORATION)

1 INFINITE LOOP

CUPERTINO, CA 95014

FOR: HANDHELD MOBILE DIGITAL ELECTRONIC DEVICES COMPRISED OF A MOBILE PHONE, DIGITAL AUDIO AND VIDEO PLAYER, HANDHELD COMPUTER, PERSONAL DIGITAL ASSISTANT, ELECTRONIC PERSONAL ORGANIZER, POCKET COMPUTER FOR NOTE-TAKING, ELECTRONIC CALENDAR, CALCULATOR, AND CAMERA, AND CAPABLE OF PROVIDING ACCESS TO THE INTERNET AND SENDING AND RECEIVING ELECTRONIC MAIL, DIGITAL AUDIO, VIDEO, TEXT, IMAGES, GRAPHICS AND MULTIMEDIA FILES, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 1-9-2007; IN COMMERCE 6-29-2007.

THE COLOR(S) GRAY, SILVER AND BLACK IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF THE CONFIGURATION OF A HANDHELD MOBILE DIGITAL ELECTRONIC DEVICE. THE MATERIAL SHOWN IN DOTTED LINES, NAMELY, THE BUTTONS AND OPENINGS ON THE DEVICE SHOW THE POSITION OF THE MARK IN RELATION TO THE DEVICE AND ARE NOT CONSIDERED A PART OF THE MARK. THE COLOR GRAY APPEARS AS A RECTANGLE AT THE FRONT, CENTER OF THE DEVICE. THE COLOR BLACK APPEARS ON THE FRONT OF THE DEVICE ABOVE AND BELOW THE GRAY RECTANGLE AND ON THE CURVED CORNERS OF THE DEVICE. THE COLOR SILVER APPEARS AS THE OUTER BORDER AND SIDES OF THE DEVICE. THE COLOR WHITE IS SHOWN SOLELY TO IDENTIFY PLACEMENT OF THE MARK AND IS NOT CLAIMED AS A PART OF THE MARK.

SEC. 2(F).

SER. NO. 77-303,049, FILED 10-12-2007.

SKYE YOUNG, EXAMINING ATTORNEY

Ex. 14

United States of America

United States Patent and Trademark Office

Reg. No. 3,886,196

Registered Dec. 7, 2010

Int. Cl.: 9

TRADEMARK

PRINCIPAL REGISTER

APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: TELEPHONY SOFTWARE, NAMELY, COMPUTER SOFTWARE FOR MAKING, MANAGING, AND RECEIVING PHONE CALLS, AND FOR REVIEWING, MANAGING, AND PLAYING ELECTRONIC VOICE MESSAGES , IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 6-29-2007; IN COMMERCE 6-29-2007.

OWNER OF U.S. REG. NO. 3,470,983.

THE COLOR(S) GREEN, LIGHT GREEN, DARK GREEN AND WHITE IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF A RECTANGLE WITH ROUNDED CORNERS DEPICTING A STYLIZED WHITE TELEPHONE RECEIVER AGAINST A STRIPED GREEN AND DARK GREEN BACKGROUND. A SHADE OF LIGHT GREEN COVERS THE UPPER HALF OF THE RECTANGLE DESIGN.

SER. NO. 85-019,804, FILED 4-21-2010.

ANDREW RHIM, EXAMINING ATTORNEY

David J. Kappos

Director of the United States Patent and Trademark Office

Ex. 15

United States of America

United States Patent and Trademark Office

Reg. No. 3,889,642

Registered Dec. 14, 2010

Int. Cl.: 9

TRADEMARK

PRINCIPAL REGISTER

APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: TEXT AND MULTIMEDIA MESSAGING SOFTWARE, NAMELY, SOFTWARE FOR PROCESSING IMAGES, GRAPHICS AND TEXT, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 6-29-2007; IN COMMERCE 6-29-2007.

OWNER OF U.S. REG. NO. 3,470,983.

THE COLOR(S) WHITE, GREEN, DARK GREEN AND LIGHT GREEN IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF A RECTANGLE WITH ROUNDED CORNERS DEPICTING A STYLIZED SPEECH BUBBLE ON A DIAGONAL STRIPED BACKGROUND. THE COLOR WHITE APPEARS IN THE SPEECH BUBBLE DESIGN; THE COLORS GREEN AND DARK GREEN APPEAR IN THE DIAGONAL STRIPES IN THE BACKGROUND OF THE RECTANGLE DESIGN; AND THE COLOR LIGHT GREEN APPEARS IN THE UPPER HALF OF THE RECTANGLE DESIGN.

SER. NO. 85-018,959, FILED 4-21-2010.

ANDREW RHIM, EXAMINING ATTORNEY

UNITED STATES PATENT AND TRADEMARK OFFICE
DEPARTMENT OF COMMERCE

David J. Kappos

Director of the United States Patent and Trademark Office

Ex. 16

United States of America

United States Patent and Trademark Office

Reg. No. 3,886,200

Registered Dec. 7, 2010

Int. Cl.: 9

TRADEMARK

PRINCIPAL REGISTER

APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: COMPUTER SOFTWARE FOR ORGANIZING, STORING, SHARING, AND VIEWING IMAGES, SOLD AS A FEATURE OF COMPUTERS AND HANDHELD MOBILE DIGITAL ELECTRONIC DEVICES COMPRISED OF MOBILE PHONES, DIGITAL AUDIO AND VIDEO PLAYERS, HANDHELD COMPUTERS, PERSONAL DIGITAL ASSISTANTS, AND ELECTRONIC PERSONAL ORGANIZERS, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 6-29-2007; IN COMMERCE 6-29-2007.

OWNER OF U.S. REG. NO. 3,470,983.

THE COLOR(S) YELLOW, BLUE, GREEN, BROWN, BLACK, GRAY AND WHITE IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF A GRAY, WHITE, AND BLUE RECTANGLE WITH ROUNDED CORNERS DEPICTING A STYLIZED FLOWER IN THE COLORS GREEN, YELLOW, BROWN, BLACK, WHITE AND GRAY.

SER. NO. 85-019,831, FILED 4-21-2010.

JAMES MACFARLANE, EXAMINING ATTORNEY

David J. Kappos

Director of the United States Patent and Trademark Office

Ex. 17

United States of America

United States Patent and Trademark Office

Reg. No. 3,889,685

Registered Dec. 14, 2010

Int. Cl.: 9

TRADEMARK

PRINCIPAL REGISTER

APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: COMPUTER SOFTWARE FOR MANAGING USER SYSTEM SETTINGS AND PREFERENCES SOLD AS A FEATURE OF COMPUTERS AND HAND-HELD MOBILE DIGITAL DEVICES COMPRISED OF MOBILE PHONES, DIGITAL AUDIO AND VIDEO PLAYERS, HAND-HELD COMPUTERS, PERSONAL DIGITAL ASSISTANTS, AND ELECTRONIC PERSONAL ORGANIZERS, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 6-29-2007; IN COMMERCE 6-29-2007.

OWNER OF U.S. REG. NOS. 3,470,983 AND 3,586,577.

THE COLOR(S) GRAY, WHITE, SILVER AND BLACK IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF PARTIAL IMAGES OF THREE GEARS SHOWN IN GRAY, WHITE AND SILVER, ON A BACKGROUND OF GRAY WITH BLACK DOTS, ALL CONTAINED WITHIN A RECTANGULAR GREY AND WHITE FRAME WITH ROUNDED CORNERS.

SER. NO. 85-020,006, FILED 4-21-2010.

FRED CARL, EXAMINING ATTORNEY

David J. Kappos

Director of the United States Patent and Trademark Office

Ex. 18

United States of America

United States Patent and Trademark Office

Reg. No. 3,886,169

Registered Dec. 7, 2010

Int. Cl.: 9

TRADEMARK

PRINCIPAL REGISTER

APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: COMPUTER SOFTWARE FOR COMPOSING, READING, SEARCHING, SHARING AND SYNCING TEXT FILES, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 6-29-2007; IN COMMERCE 6-29-2007.

OWNER OF U.S. REG. NO. 3,470,983.

THE COLOR(S) YELLOW, BROWN AND GRAY IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF A RECTANGLE WITH ROUNDED CORNERS DEPICTING A STYLIZED CROSS-SECTION OF A PAGE OF NOTEBOOK PAPER. THE COLOR YELLOW APPEARS IN THE NOTEBOOK PAPER; THE COLOR BROWN APPEARS AT THE TOP ABOVE THE NOTEBOOK PAPER AND IN THE VERTICAL LINES ON THE LEFT SIDE OF THE NOTEBOOK PAPER; AND THE COLOR GRAY APPEARS IN THE HORIZONTAL LINES ACROSS THE NOTEBOOK PAPER.

SER. NO. 85-019,396, FILED 4-21-2010.

ANDREW RHIM, EXAMINING ATTORNEY

David J. Kappos

Director of the United States Patent and Trademark Office

Ex. 19

United States of America

United States Patent and Trademark Office

Reg. No. 3,886,197

Registered Dec. 7, 2010

Int. Cl.: 9

TRADEMARK

PRINCIPAL REGISTER

APPLE INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: COMPUTER SOFTWARE FOR CONTACT INFORMATION MANAGEMENT SOLD AS A FEATURE OF COMPUTERS AND HANDHELD MOBILE DIGITAL ELECTRONIC DEVICES COMPRISED OF MOBILE PHONES, DIGITAL AUDIO AND VIDEO PLAYERS, HANDHELD COMPUTERS, PERSONAL DIGITAL ASSISTANTS, AND ELECTRONIC PERSONAL ORGANIZERS, IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 6-19-2009; IN COMMERCE 6-19-2009.

OWNER OF U.S. REG. NOS. 3,462,199, 3,462,224, AND 3,470,983.

THE COLOR(S) BROWN, WHITE AND GRAY IS/ARE CLAIMED AS A FEATURE OF THE MARK.

THE MARK CONSISTS OF A BROWN RECTANGLE WITH ROUNDED CORNERS DEPICTING A STYLIZED WIRE-BOUND BOOK WITH THE SILHOUETTE OF A MAN IN THE MIDDLE. THE WIRE BINDING APPEARS IN WHITE AND GRAY. THERE ARE BROWN TABS ON THE RIGHT OF THE BOOK WITH THE LETTERS "ABCDEF" IN GRAY AND WHITE.

SER. NO. 85-019,809, FILED 4-21-2010.

JAMES MACFARLANE, EXAMINING ATTORNEY

David J. Kappos

Director of the United States Patent and Trademark Office

Ex. 20

Thank you for your request. Here are the latest results from the TARR web server.

This page was generated by the TARR system on 2011-04-15 12:59:06 ET

Serial Number: 85041463 Assignment Information Trademark Document Retrieval

Registration Number: (NOT AVAILABLE)

Mark

Standard Character claim: No

Current Status: Review prior to publication completed.

Date of Status: 2011-03-16

Filing Date: 2010-05-18

The Information will be/was published in the Official Gazette on 2011-04-19

Transformed into a National Application: No

Registration Date: (DATE NOT AVAILABLE)

Register: Principal

Law Office Assigned: LAW OFFICE 101

Attorney Assigned:
RHIM ANDREW

Current Location: 650 -Publication And Issue Section

Date In Location: 2011-03-16

LAST APPLICANT(S)/OWNER(S) OF RECORD

---

1. Apple Inc.

Address:
Apple Inc.
1 Infinite Loop

Cupertino, CA 95014
United States
Legal Entity Type: Corporation
State or Country of Incorporation: California

GOODS AND/OR SERVICES

International Class: 009
Class Status: Active
Computer software for use in searching, browsing, reviewing, sampling, playing, purchasing, and downloading pre-recorded audio and video content
Basis: 1(a)
First Use Date: 2008-06-00
First Use in Commerce Date: 2008-06-00

ADDITIONAL INFORMATION

Color(s) Claimed: The color(s) purple, white and light purple is/are claimed as a feature of the mark.

Description of Mark: The mark consists of a rectangle with rounded corners depicting a stylized musical note. A circle surrounds the musical note. The color purple appears in the lower portion of the rectangle; the color white appears in the musical note and in the circle surrounding the musical note; and the color light purple appears in the upper portion of the rectangle.

Design Search Code(s):
24.17.13 - Clef symbol (musical); Musical notes; Musical staff; Musical symbols, including treble and bass clef symbols, sharp and flat symbols and notes
26.01.21 - Circles that are totally or partially shaded.
26.11.21 - Rectangles that are completely or partially shaded

Prior Registration Number(s):
2935038
3470983

MADRID PROTOCOL INFORMATION

USPTO Reference Number: A0022114
International Registration Number: 1057958
International Registration Date: 2010-11-12
Original Filing Date with USPTO: 2010-11-12
International Registration Status: Application For IR Registered By IB
Date of International Registration Status: 2010-12-09
International Registration Renewal Date: 2020-11-12
Irregularity Reply by Date: (DATE NOT AVAILABLE)

Madrid History:
12-09-2010 - 23:03:13 - Application For IR Registered By IB
11-13-2010 - 21:02:33 - IR Certified And Sent To IB
11-13-2010 - 05:51:19 - New Application For IR Received

11-13-2010 - 05:51:19 - Automatically Certified

PROSECUTION HISTORY

NOTE: To view any document referenced below, click on the link to "Trademark Document Retrieval" shown near the top of this page.

2011-03-30 - Notice of publication

2011-03-16 - Law Office Publication Review Completed

2011-03-16 - Assigned To LIE

2011-02-27 - Approved for Pub - Principal Register (Initial exam)

2011-02-03 - Teas/Email Correspondence Entered

2011-02-03 - Communication received from applicant

2011-02-03 - TEAS Response to Office Action Received

2010-08-03 - Non-final action mailed

2010-08-03 - Non-Final Action Written

2010-08-03 - Assigned To Examiner

2010-05-25 - Notice Of Design Search Code Mailed

2010-05-22 - New Application Office Supplied Data Entered In Tram

2010-05-21 - New Application Entered In Tram

ATTORNEY/CORRESPONDENT INFORMATION

Attorney of Record
Lisa G. Widup

Correspondent
LISA G. WIDUP
APPLE INC.
1 INFINITE LOOP # MS3TM
CUPERTINO, CA 95014-2083

Ex. 21

Int. Cl.: 9

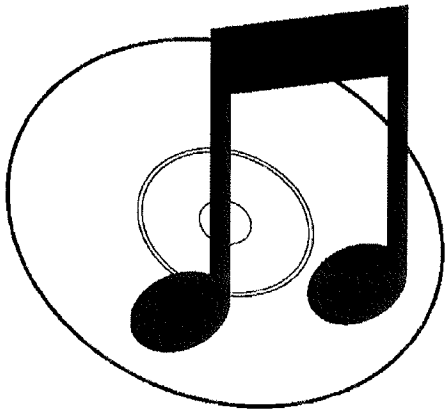
Prior U.S. Cls.: 21, 23, 26, 36 and 38

Reg. No. 2,935,038

United States Patent and Trademark Office

Registered Mar. 22, 2005

TRADEMARK
PRINCIPAL REGISTER



APPLE COMPUTER, INC. (CALIFORNIA CORPORATION)
1 INFINITE LOOP
CUPERTINO, CA 95014

FOR: COMPUTER SOFTWARE FOR USE IN PURCHASING, AUTHORING, DOWNLOADING, TRANSMITTING, RECEIVING, EDITING, EXTRACTING, ENCODING, DECODING, PLAYING, STORING AND ORGANIZING AUDIO DATA , IN CLASS 9 (U.S. CLS. 21, 23, 26, 36 AND 38).

FIRST USE 1-9-2001; IN COMMERCE 1-9-2001.

THE MARK CONSISTS OF A DESIGN OF A COMPACT DISC WITH TWO MUSICAL NOTES.

SER. NO. 78-382,867, FILED 3-11-2004.

HENRY S. ZAK, EXAMINING ATTORNEY