US 20060026535A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2006/0026535 A1**

Hotelling et al. (43) **Pub. Date: Feb. 2, 2006**

(54) **MODE-BASED GRAPHICAL USER INTERFACES FOR TOUCH SENSITIVE INPUT DEVICES**

(75) Inventors: **Steve Hotelling**, San Jose, CA (US); **Brian Q. Huppi**, San Francisco, CA (US); **Joshua A. Strickon**, San Jose, CA (US); **Duncan Robert Kerr**, San Francisco, CA (US); **Bas Ording**, San Francisco, CA (US); **Imran Chaudhri**, San Francisco, CA (US); **Greg Christie**, San Jose, CA (US); **Jonathan P. Ive**, San Francisco, CA (US)

Correspondence Address:
BEYER WEAVER & THOMAS LLP
P.O. BOX 70250
OAKLAND, CA 94612-0250 (US)

(73) Assignee: **Apple Computer Inc.**

(21) Appl. No.: **11/038,590**

(22) Filed: **Jan. 18, 2005**

Related U.S. Application Data

(63) Continuation-in-part of application No. 10/903,964, filed on Jul. 30, 2004.

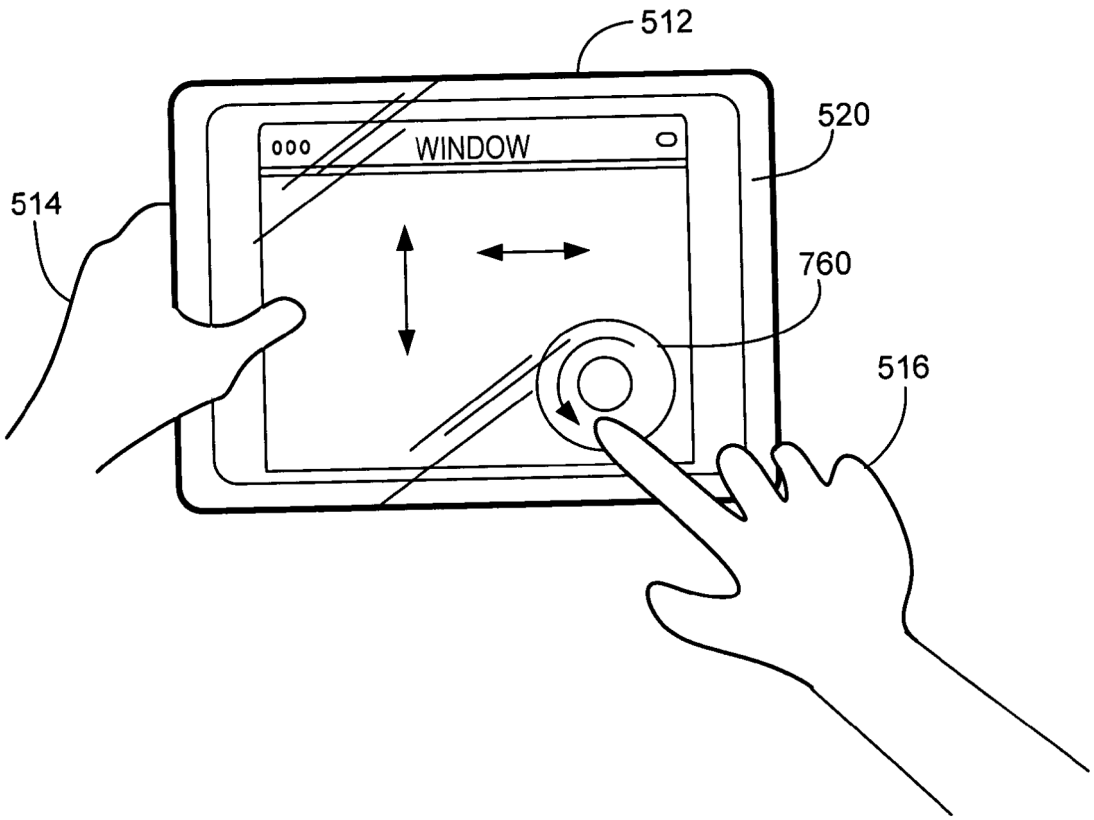
Publication Classification

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

(52) **U.S. Cl.** **715/863**

(57) **ABSTRACT**

A user interface method is disclosed. The method includes detecting a touch and then determining a user interface mode when a touch is detected. The method further includes activating one or more GUI elements based on the user interface mode and in response to the detected touch.



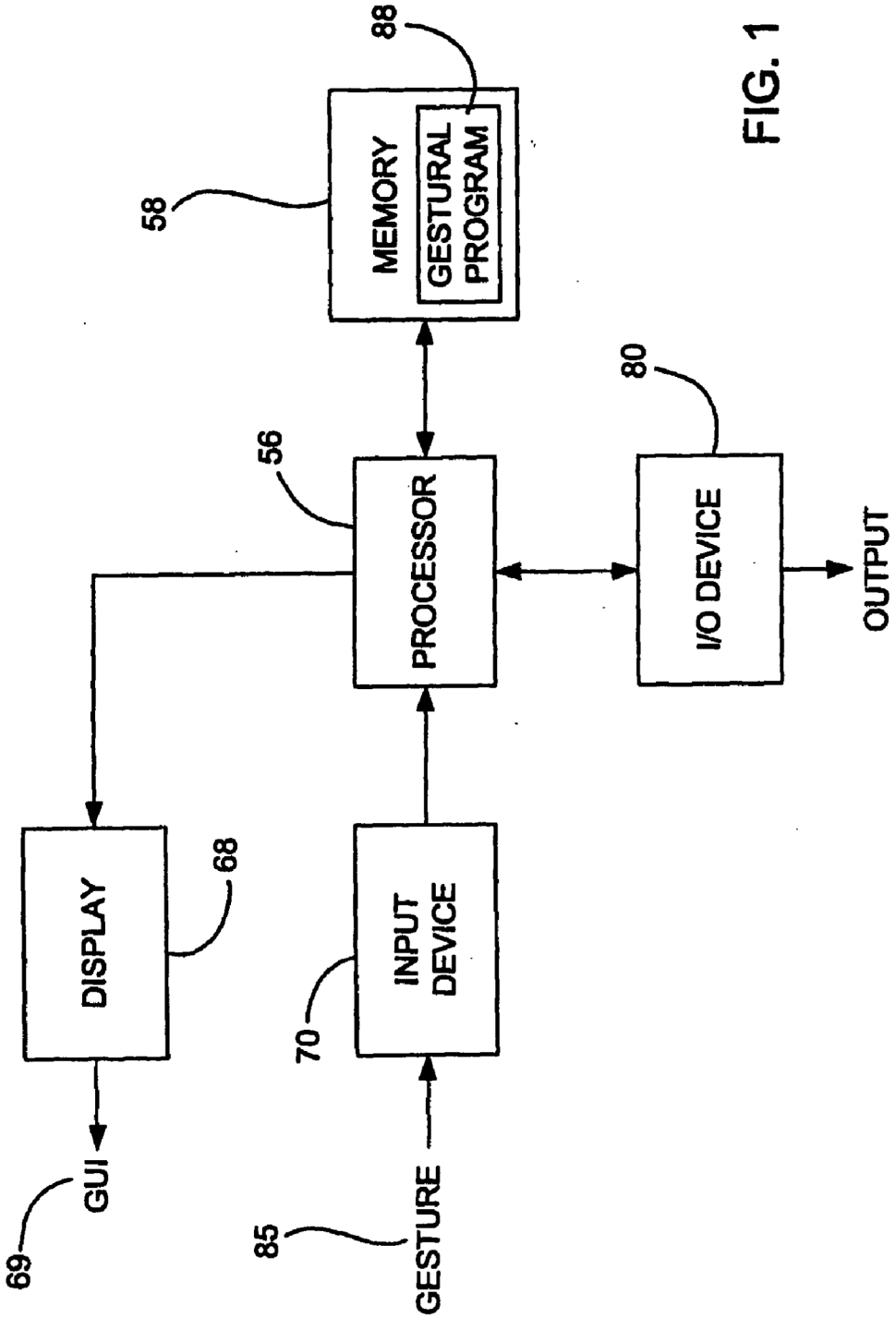

88
MEMORY
GESTURAL PROGRAM
58
56
PROCESSOR
80
I/O DEVICE
OUTPUT
DISPLAY
68
INPUT DEVICE
70
GUI
69
GESTURE
85

FIG. 1

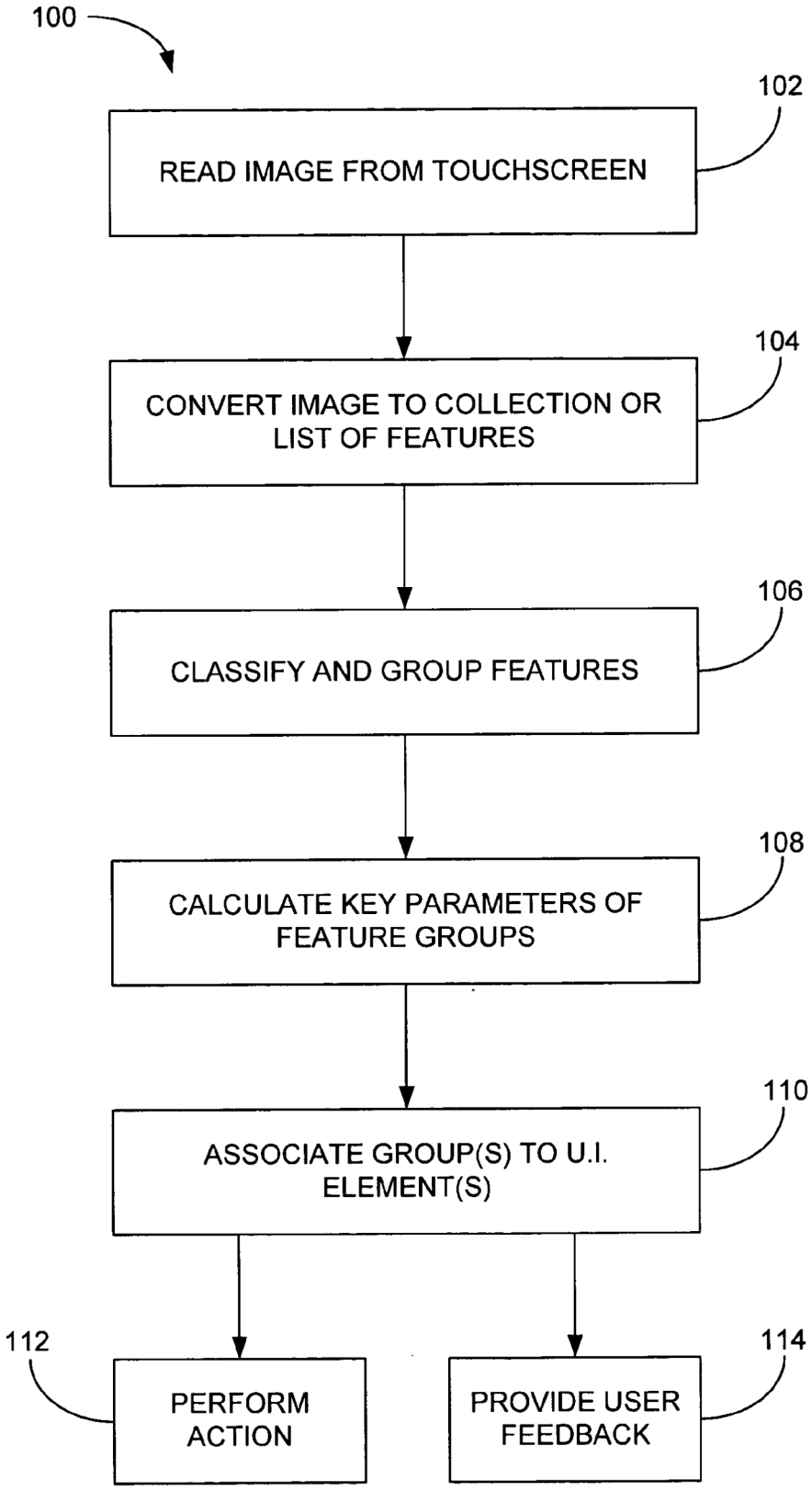

100
102
READ IMAGE FROM TOUCHSCREEN
104
CONVERT IMAGE TO COLLECTION OR LIST OF FEATURES
106
CLASSIFY AND GROUP FEATURES
108
CALCULATE KEY PARAMETERS OF FEATURE GROUPS
110
ASSOCIATE GROUP(S) TO U.I. ELEMENT(S)
112
PERFORM ACTION
114
PROVIDE USER FEEDBACK

FIG. 2

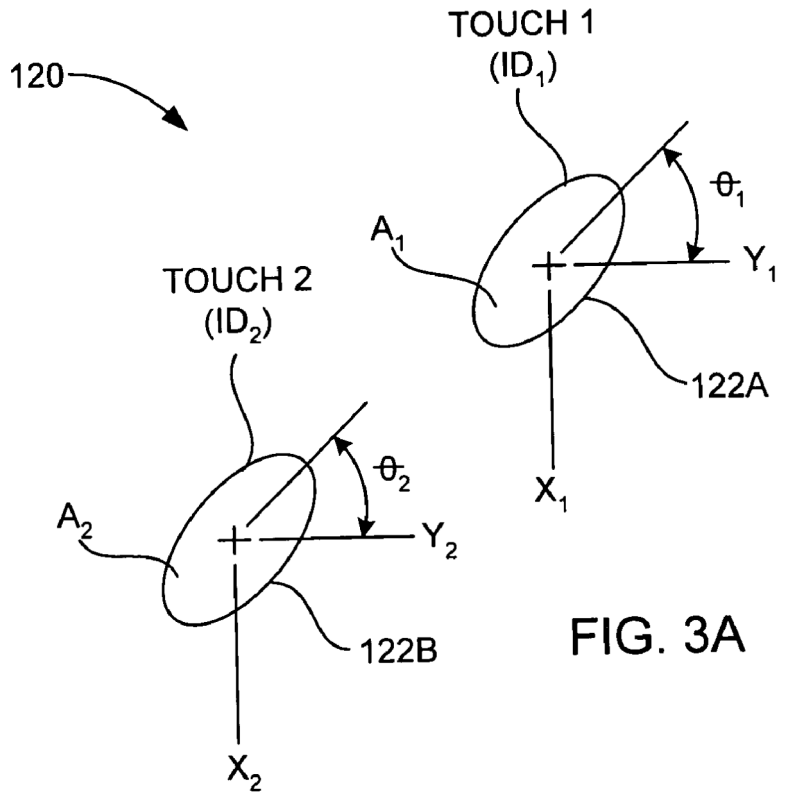

120
TOUCH 1
(ID1)
θ1
A1
Y1
TOUCH 2
(ID2)
θ2
A2
Y2
122A
122B
X1
X2

FIG. 3A

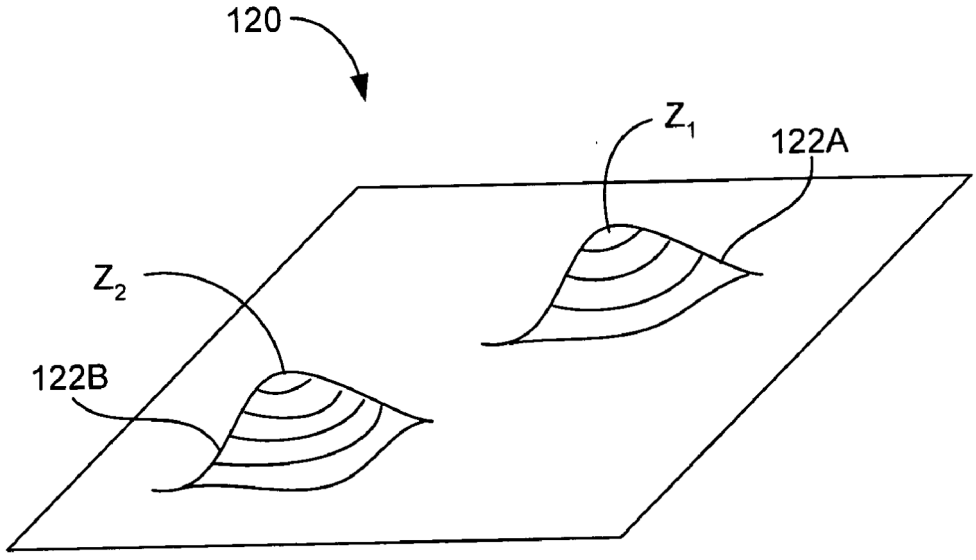

120
Z1
122A
Z2
122B

FIG. 3B

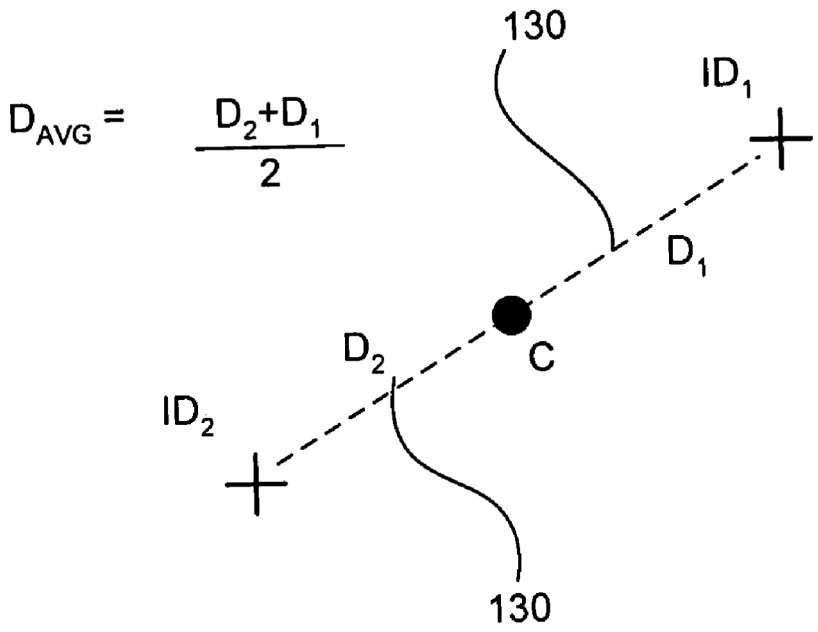


FIG. 4

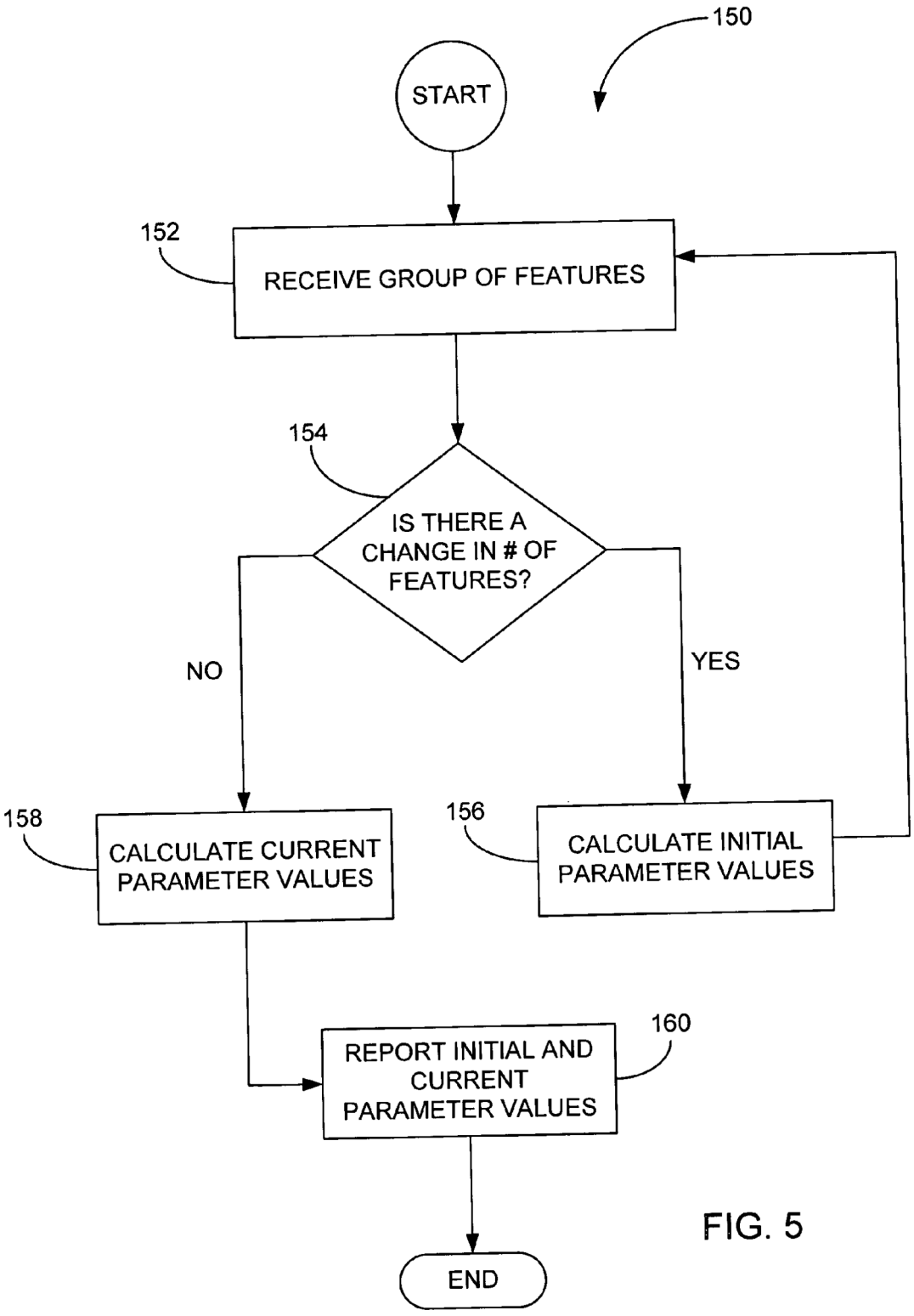

150
START
152
RECEIVE GROUP OF FEATURES
154
IS THERE A CHANGE IN # OF FEATURES?
NO
YES
158
CALCULATE CURRENT PARAMETER VALUES
156
CALCULATE INITIAL PARAMETER VALUES
160
REPORT INITIAL AND CURRENT PARAMETER VALUES
END

FIG. 5

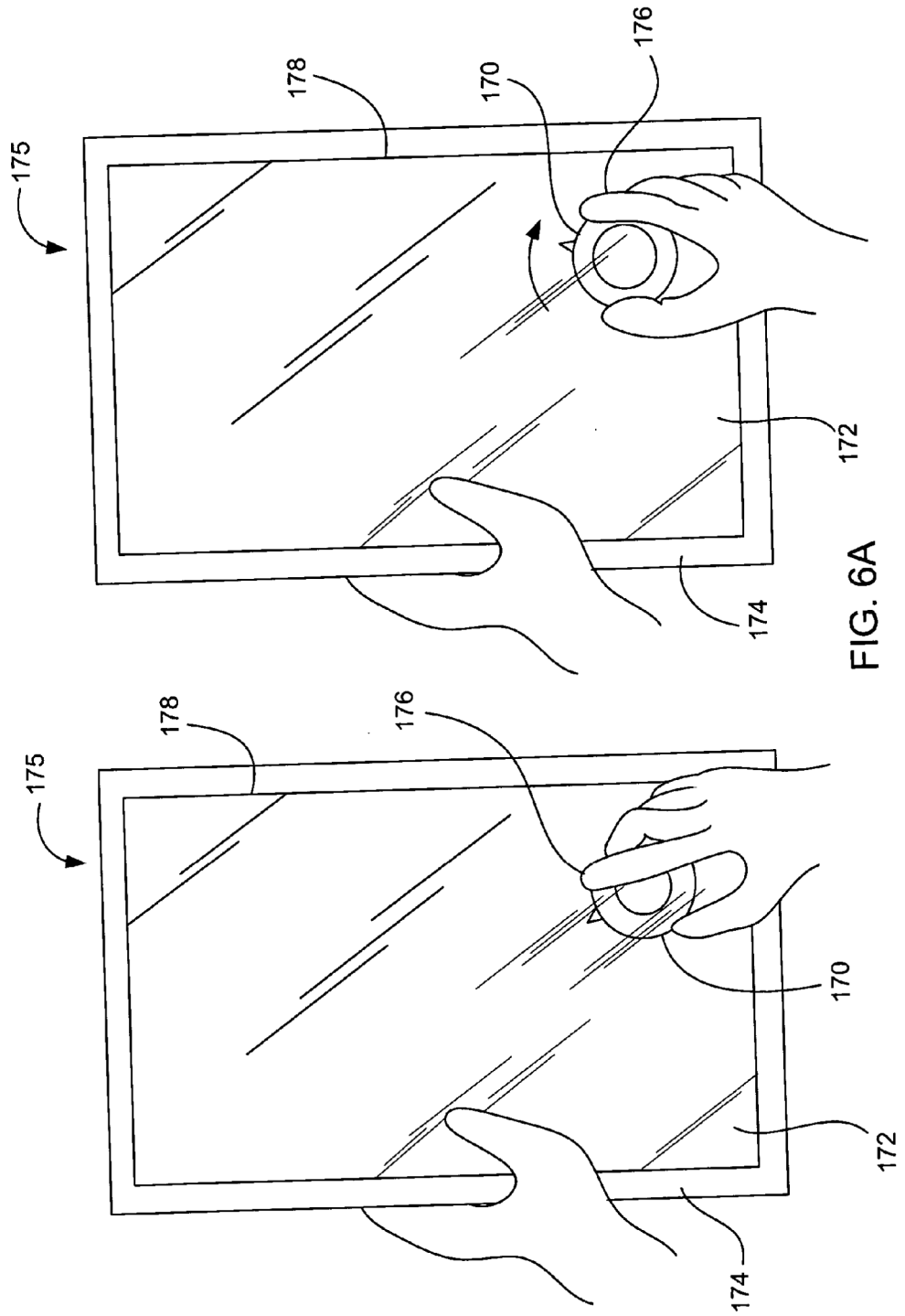


FIG. 6A

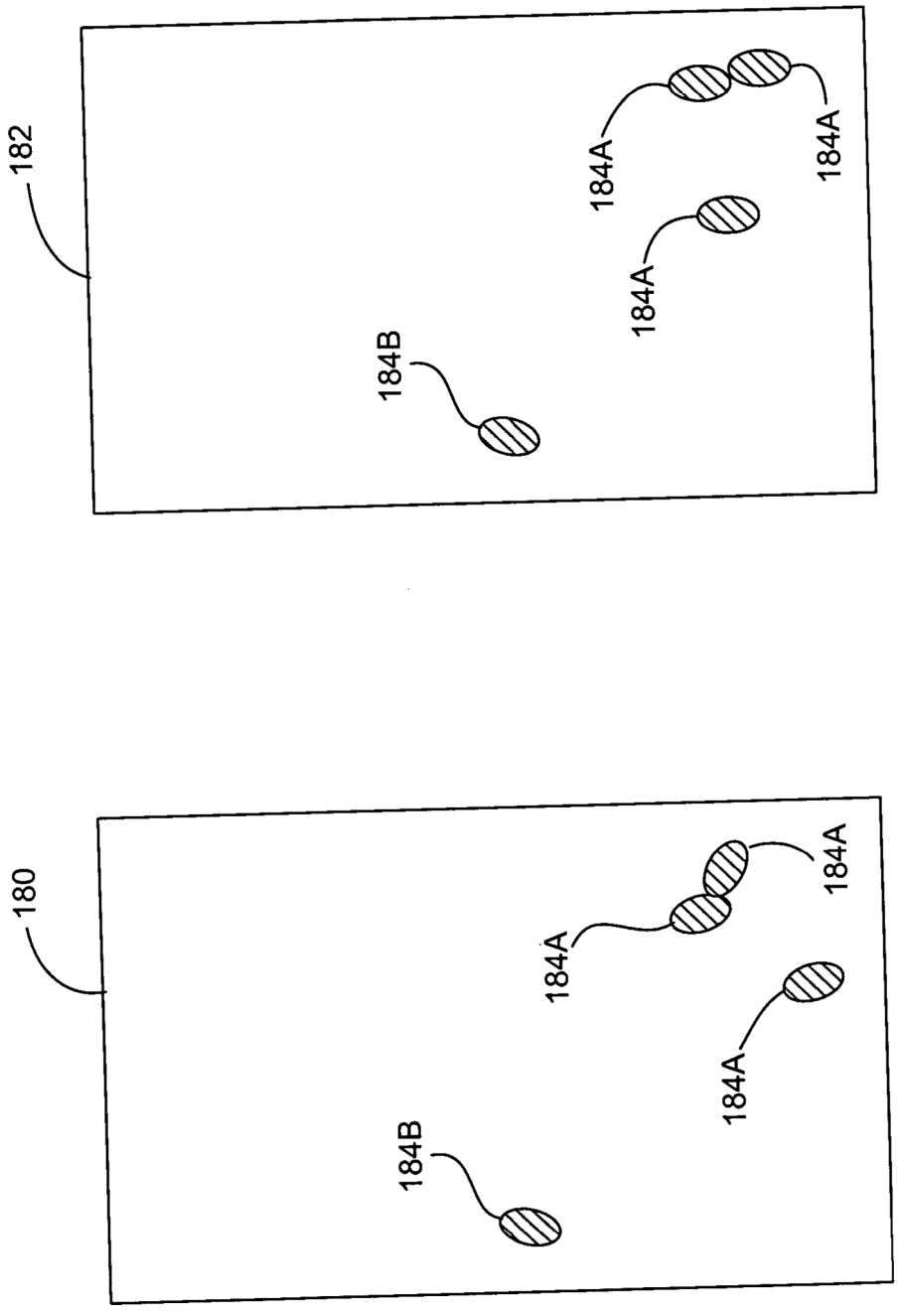


FIG. 6B

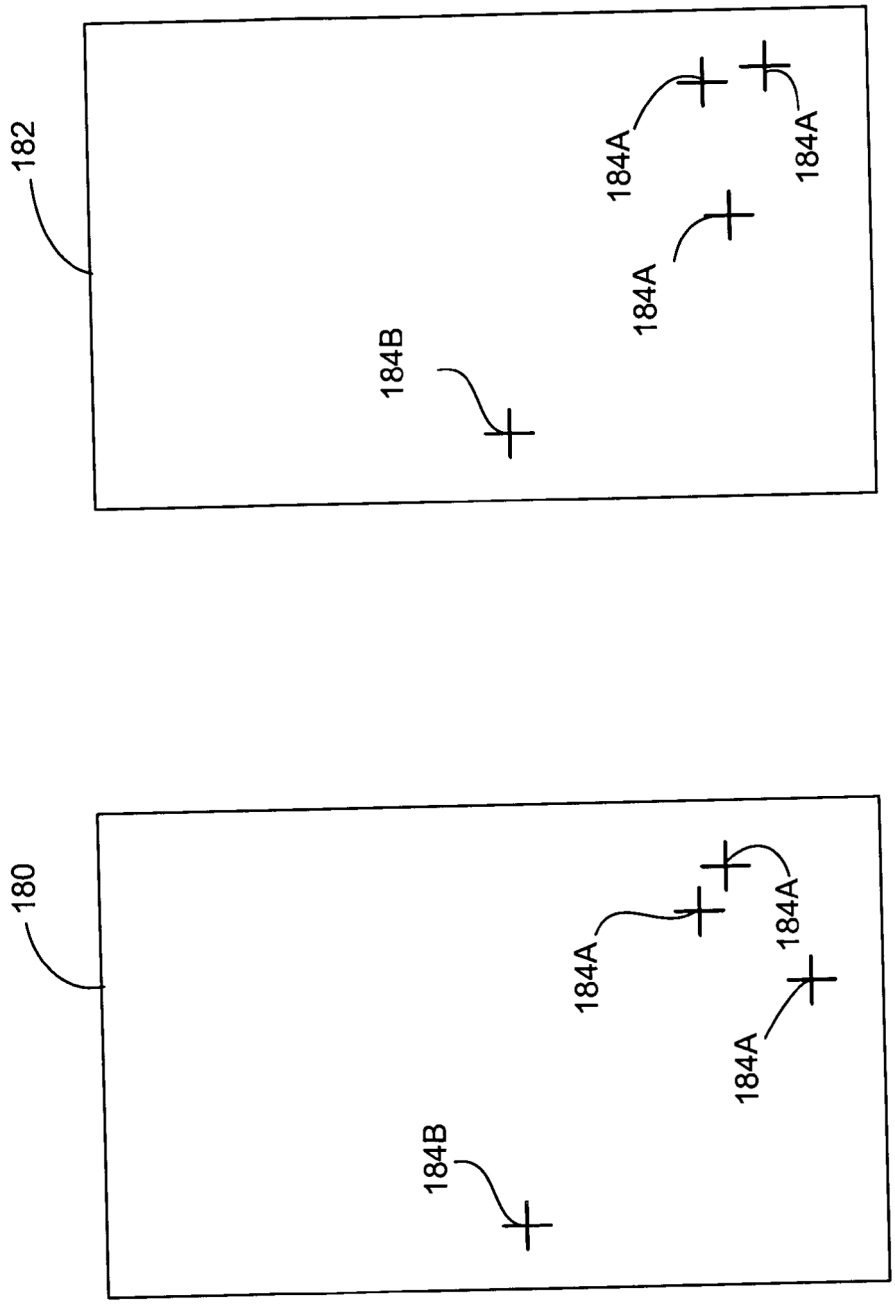


FIG. 6C

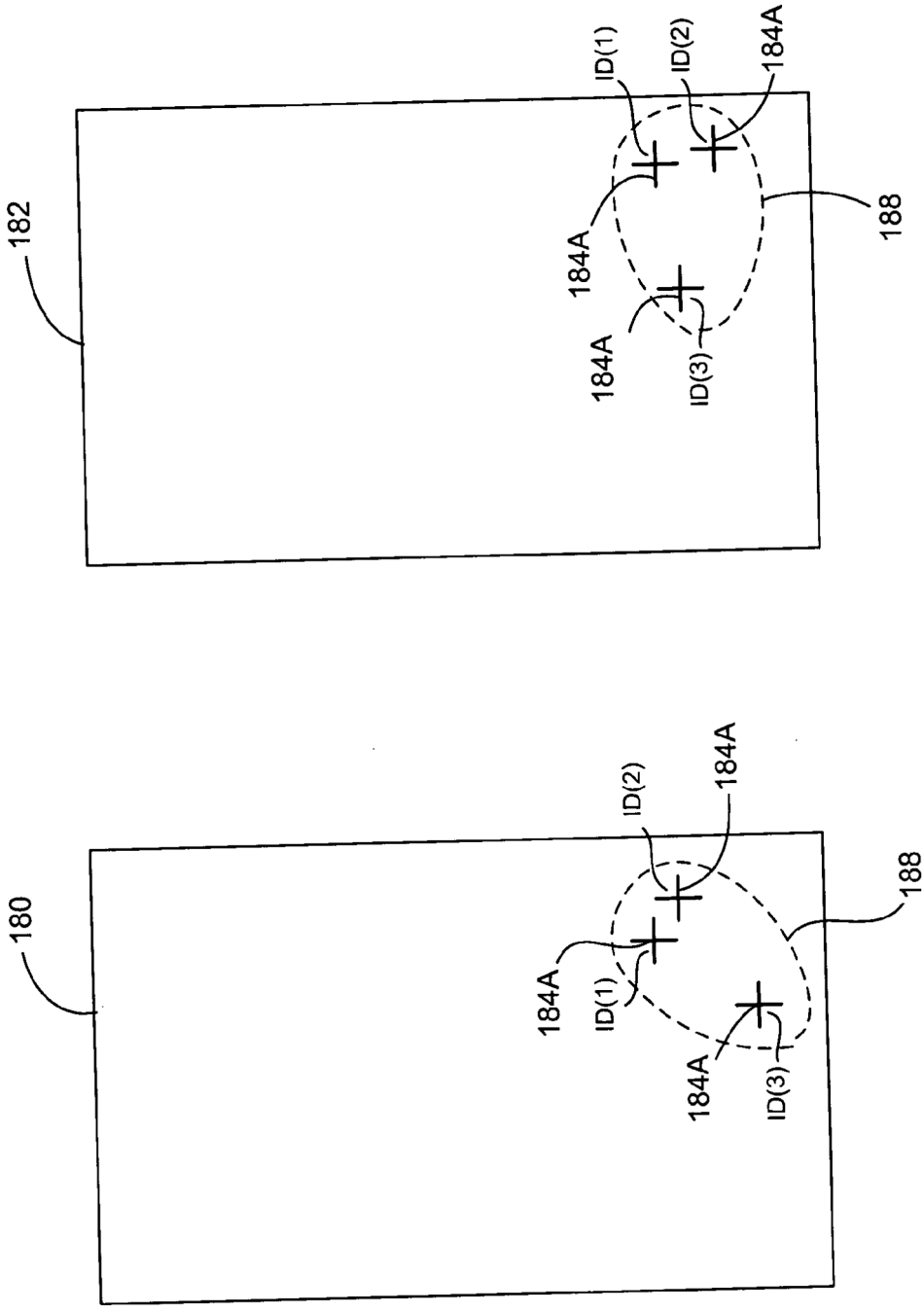

182
ID(1)
ID(2)
184A
188
184A
184A
ID(3)
180
ID(2)
184A
188
184A
ID(1)
184A
ID(3)

FIG. 6D

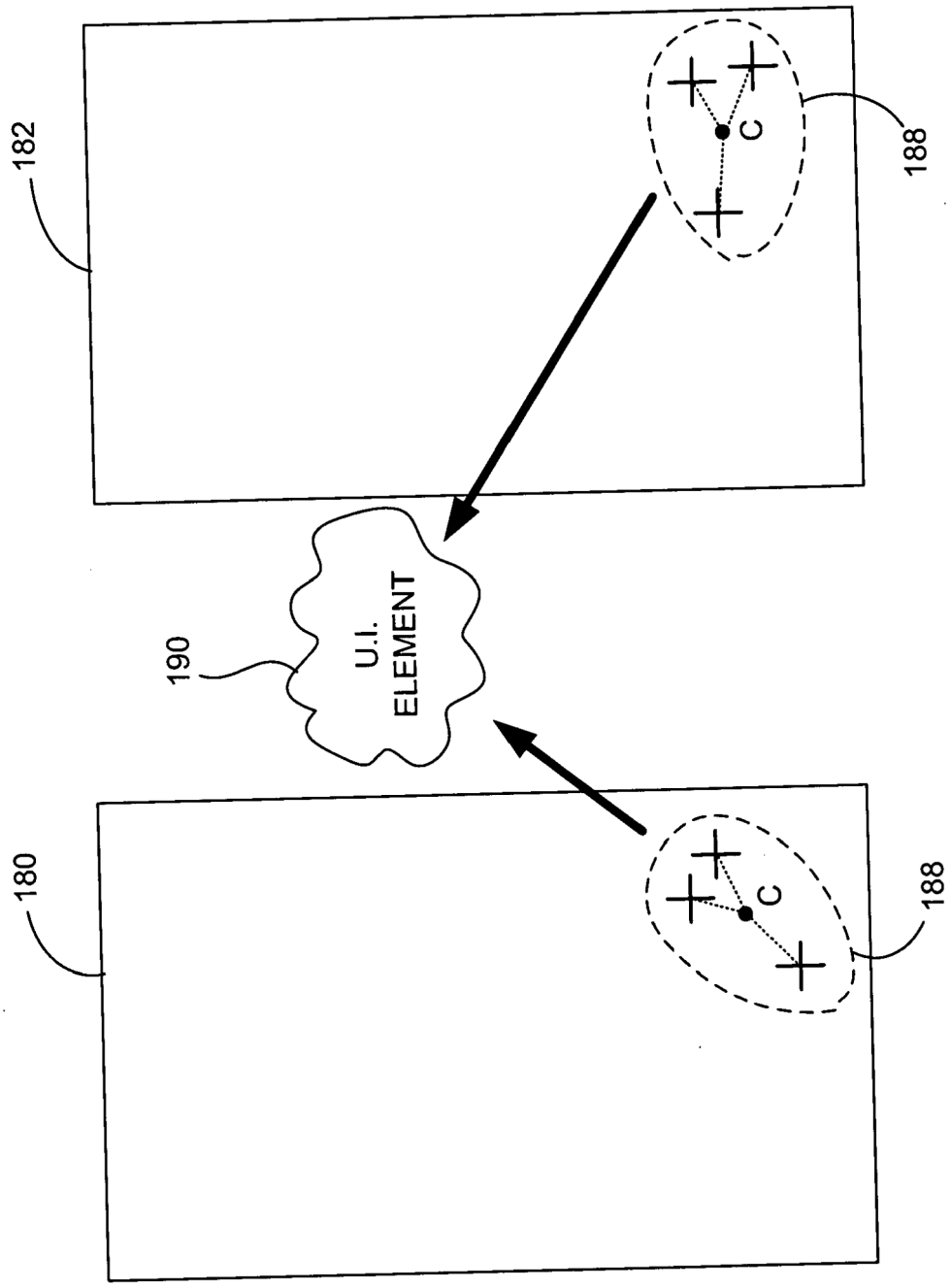


FIG. 6E

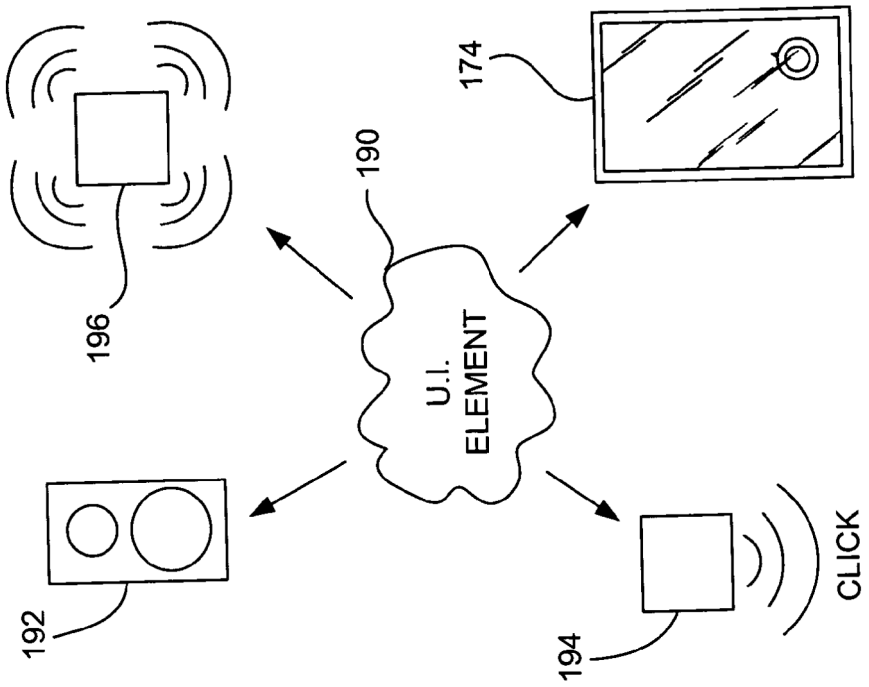


FIG. 6G

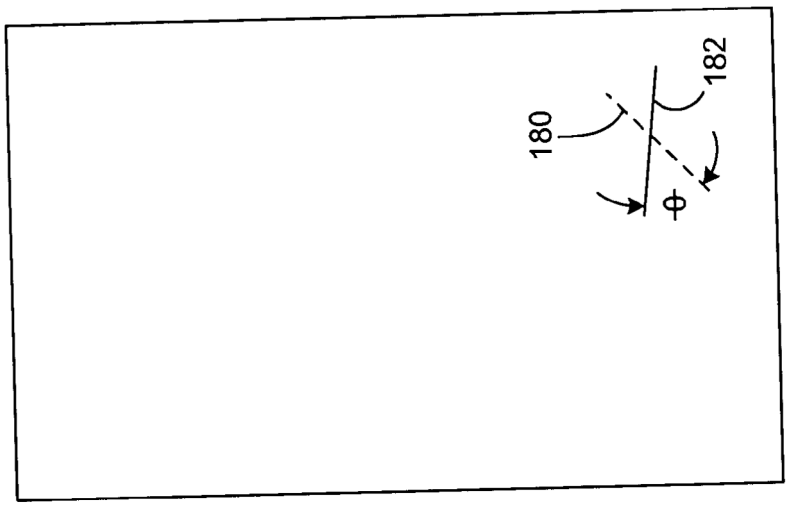


FIG. 6F

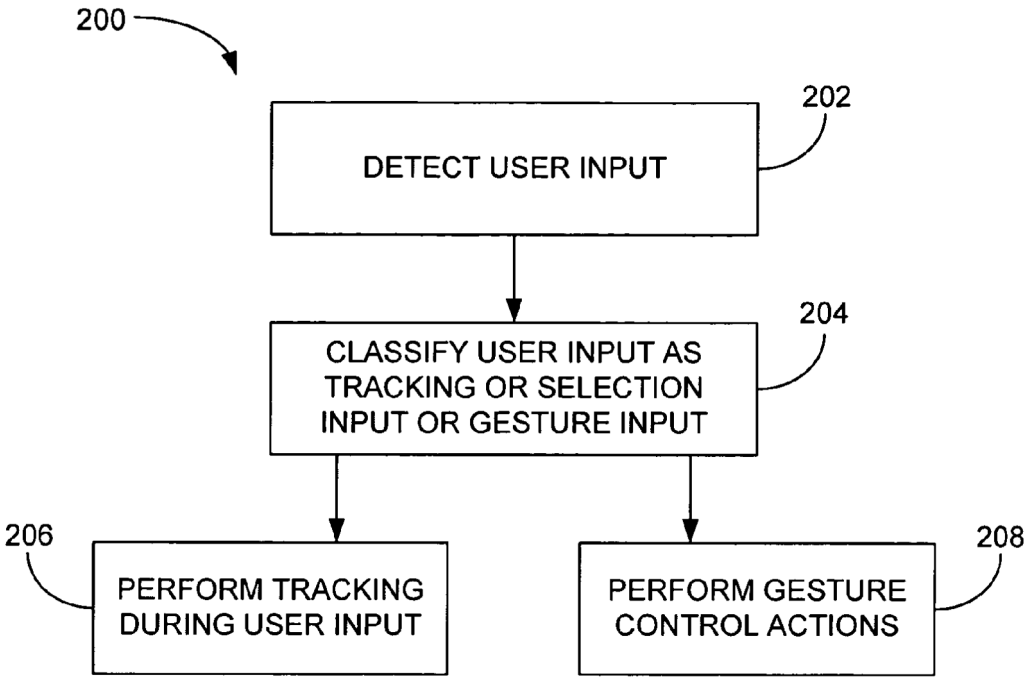

200
DETECT USER INPUT
202
CLASSIFY USER INPUT AS TRACKING OR SELECTION INPUT OR GESTURE INPUT
204
206
PERFORM TRACKING DURING USER INPUT
PERFORM GESTURE CONTROL ACTIONS
208

FIG. 7

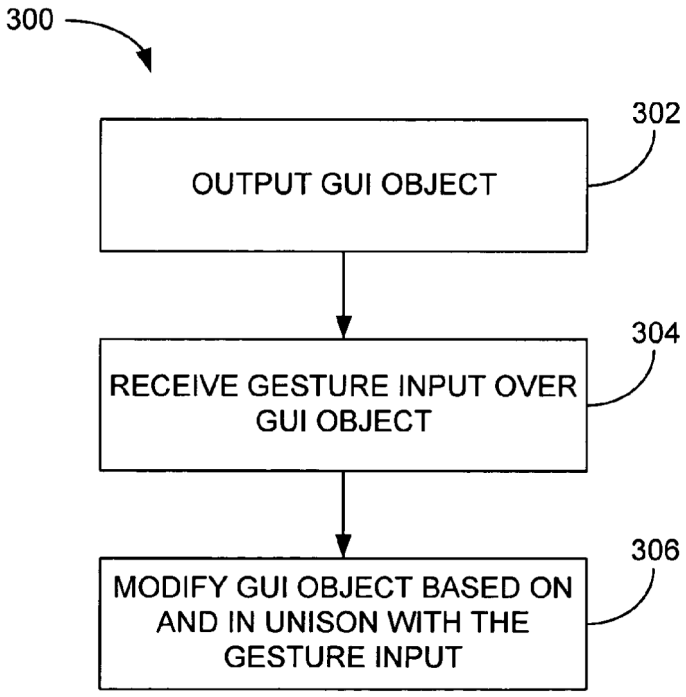

300
OUTPUT GUI OBJECT
302
RECEIVE GESTURE INPUT OVER GUI OBJECT
304
MODIFY GUI OBJECT BASED ON AND IN UNISON WITH THE GESTURE INPUT
306

FIG. 9

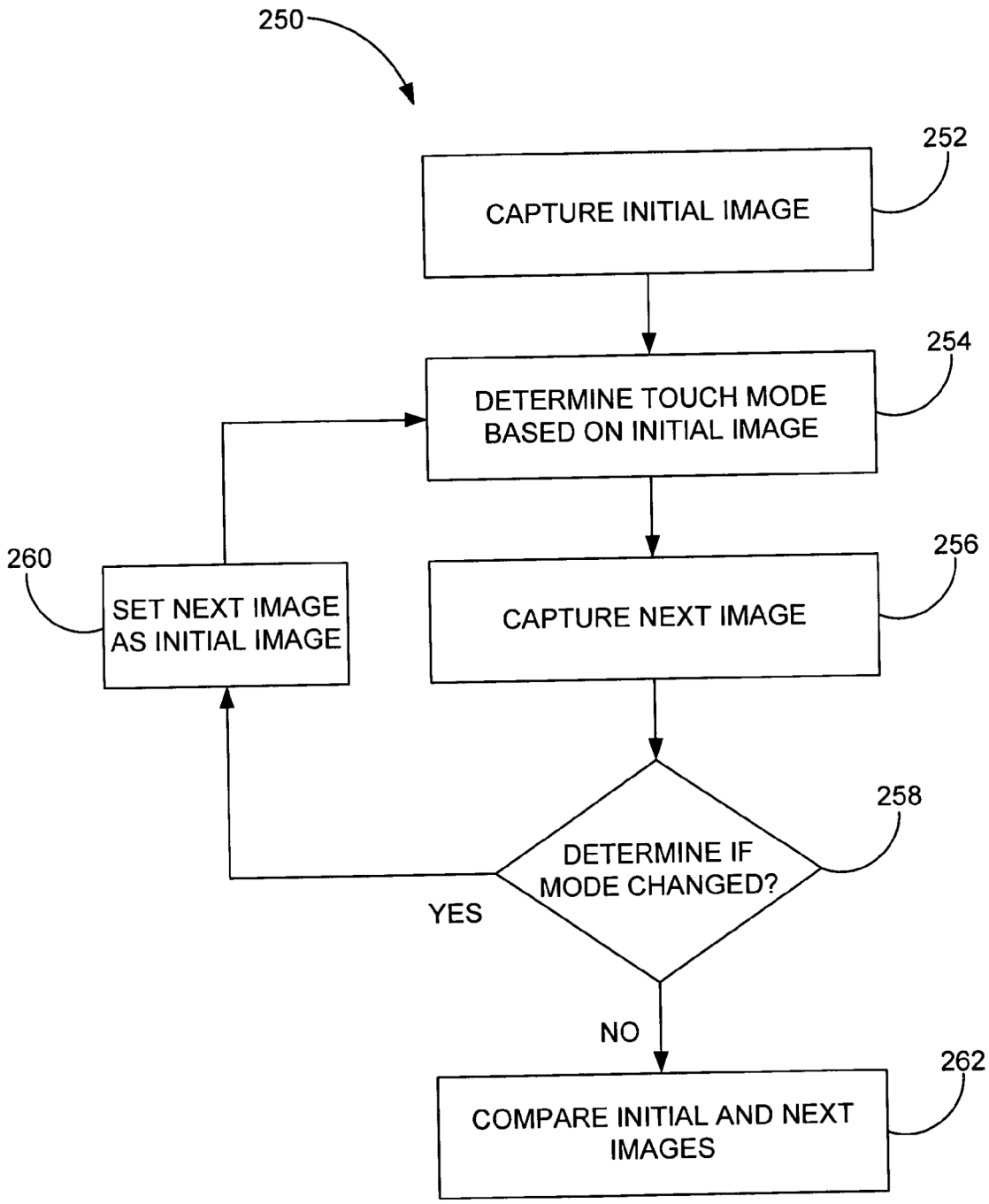

250
252
CAPTURE INITIAL IMAGE
254
DETERMINE TOUCH MODE BASED ON INITIAL IMAGE
256
CAPTURE NEXT IMAGE
258
DETERMINE IF MODE CHANGED?
YES
NO
260
SET NEXT IMAGE AS INITIAL IMAGE
262
COMPARE INITIAL AND NEXT IMAGES

FIG. 8

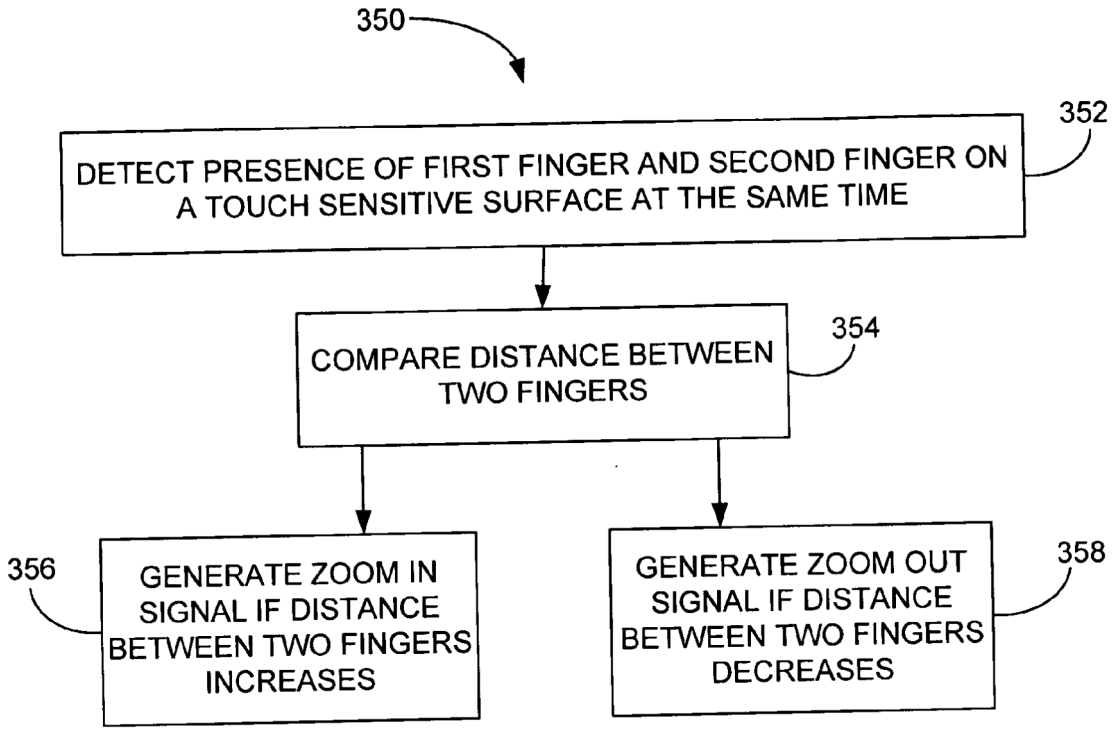

350
DETECT PRESENCE OF FIRST FINGER AND SECOND FINGER ON A TOUCH SENSITIVE SURFACE AT THE SAME TIME
352
COMPARE DISTANCE BETWEEN TWO FINGERS
354
GENERATE ZOOM IN SIGNAL IF DISTANCE BETWEEN TWO FINGERS INCREASES
356
GENERATE ZOOM OUT SIGNAL IF DISTANCE BETWEEN TWO FINGERS DECREASES
358

FIG. 10

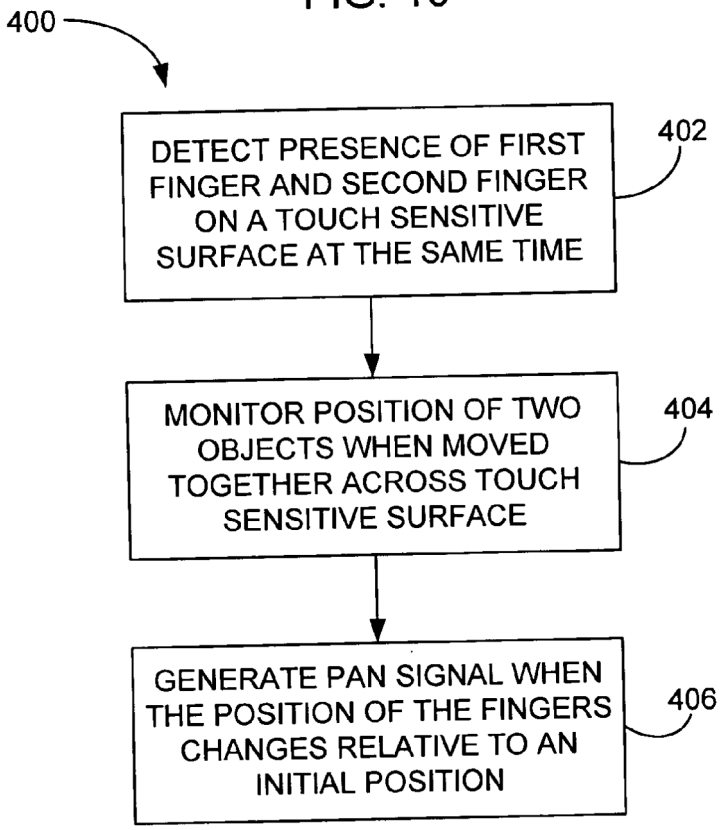

400
DETECT PRESENCE OF FIRST FINGER AND SECOND FINGER ON A TOUCH SENSITIVE SURFACE AT THE SAME TIME
402
MONITOR POSITION OF TWO OBJECTS WHEN MOVED TOGETHER ACROSS TOUCH SENSITIVE SURFACE
404
GENERATE PAN SIGNAL WHEN THE POSITION OF THE FINGERS CHANGES RELATIVE TO AN INITIAL POSITION
406

FIG. 12

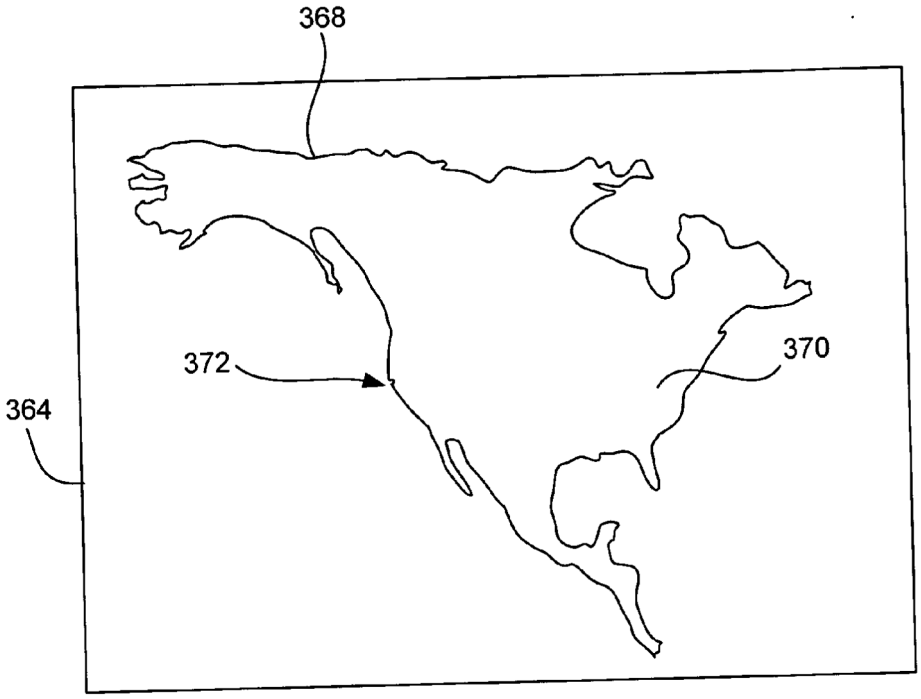


FIG. 11A

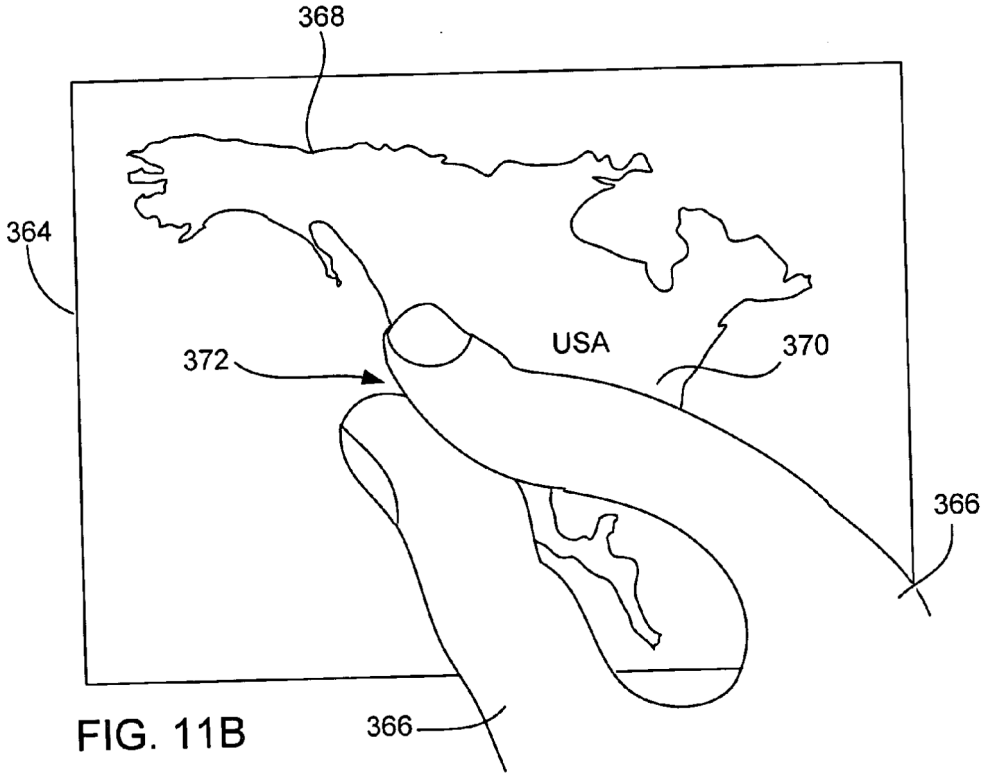


FIG. 11B

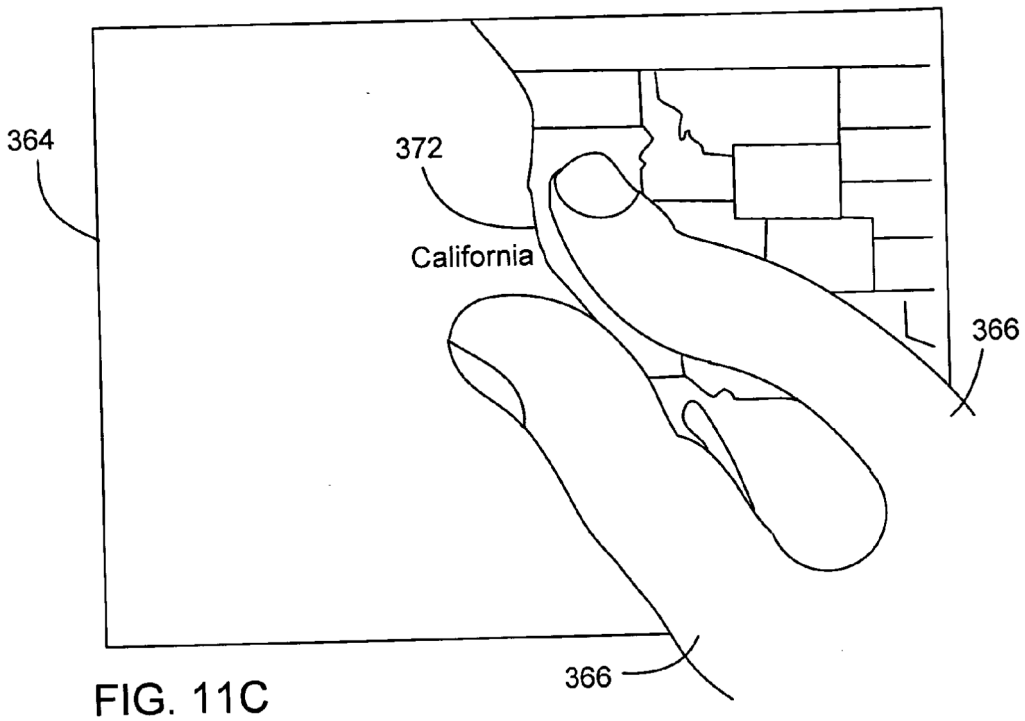


FIG. 11C

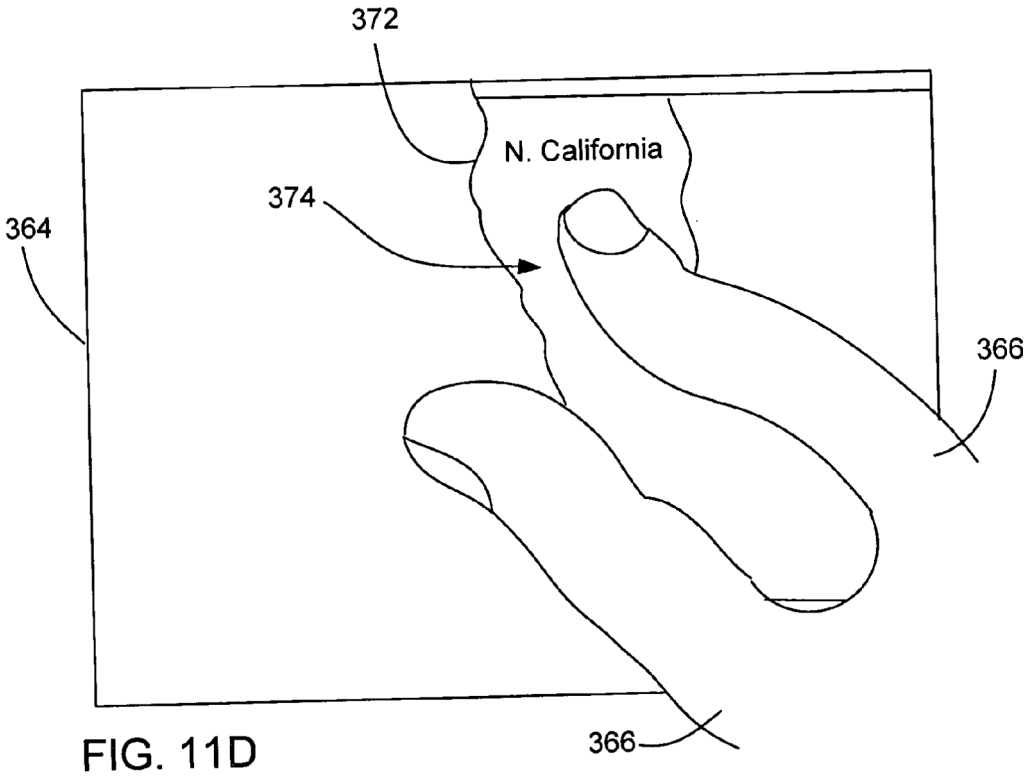


FIG. 11D

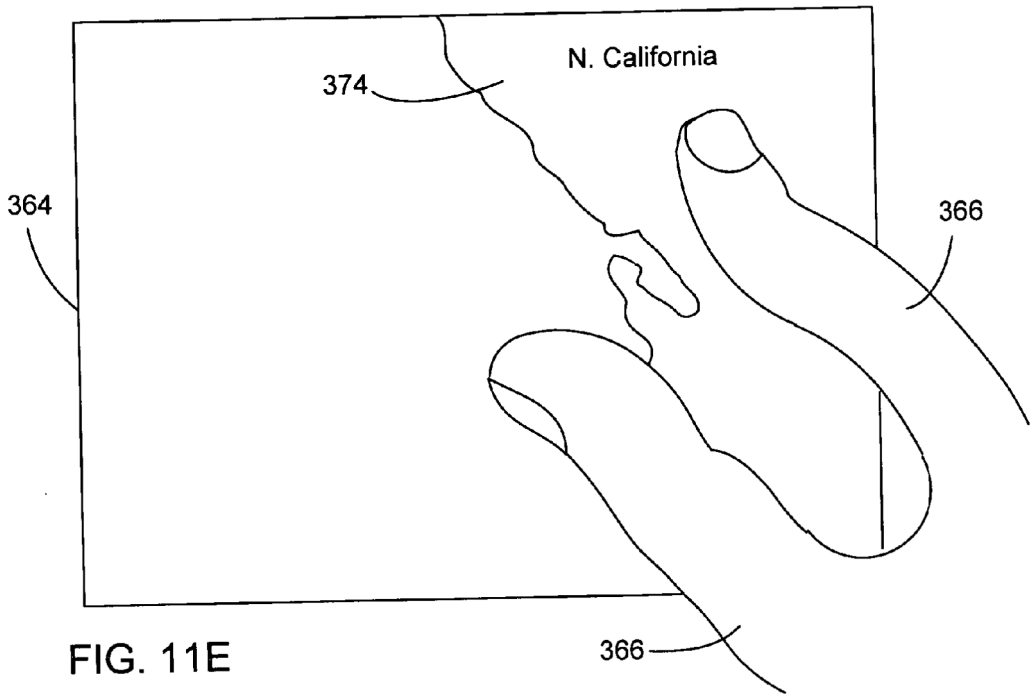


FIG. 11E

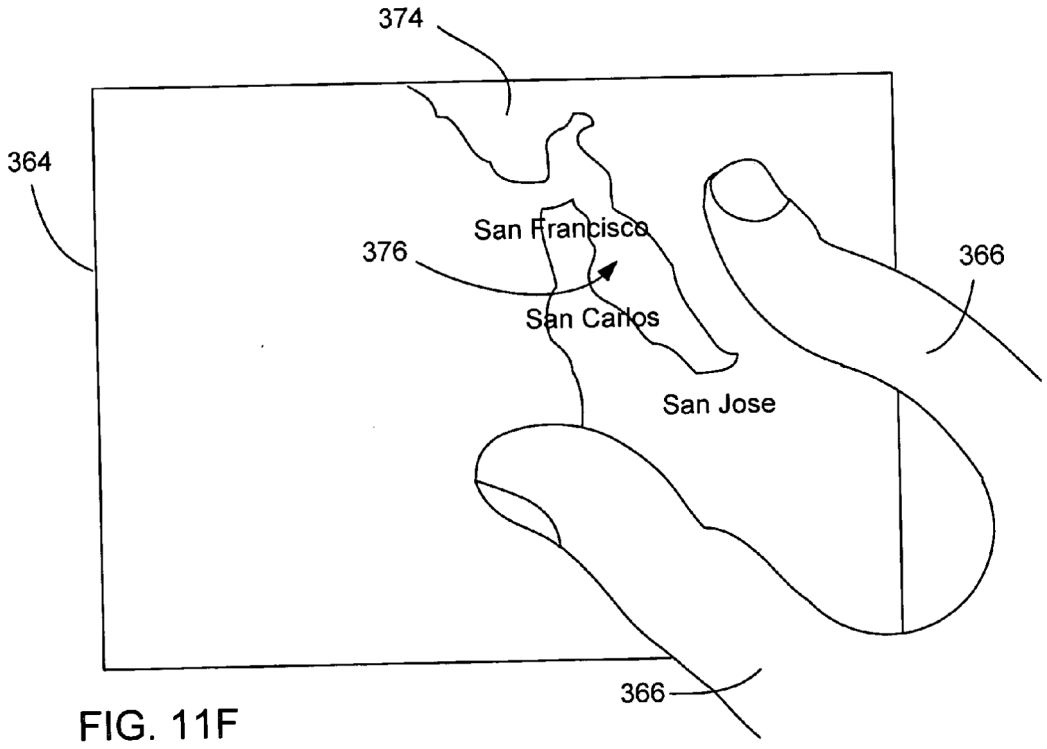


FIG. 11F

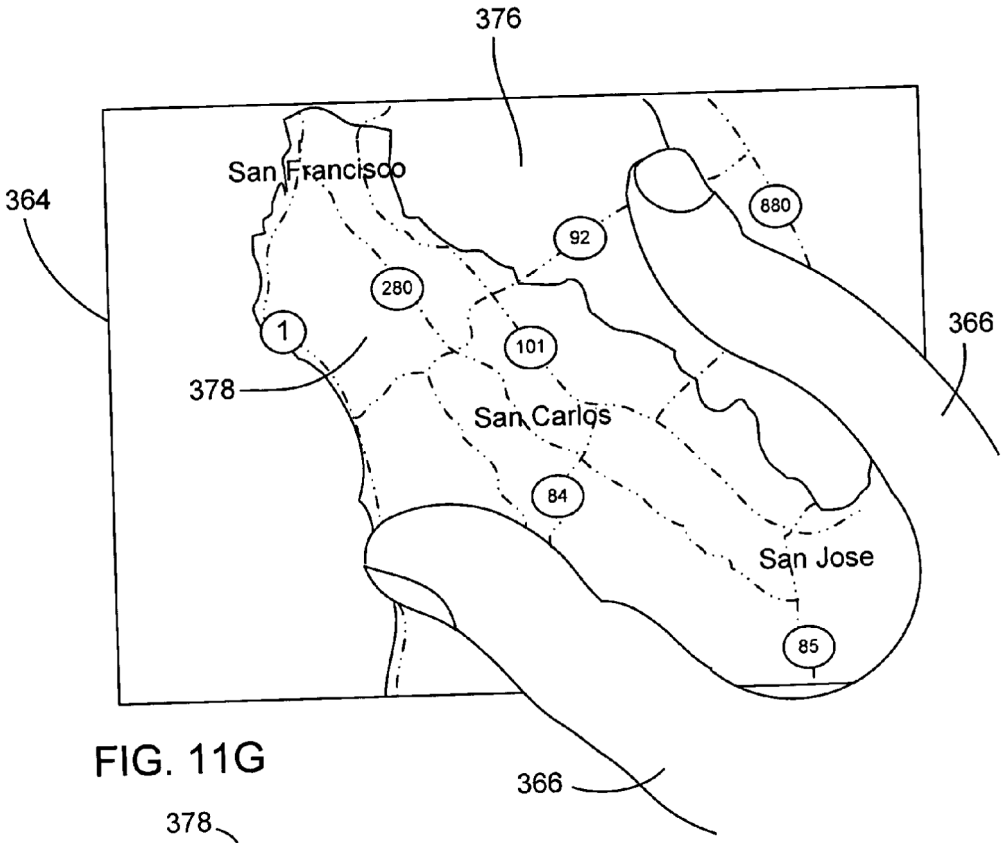


FIG. 11G

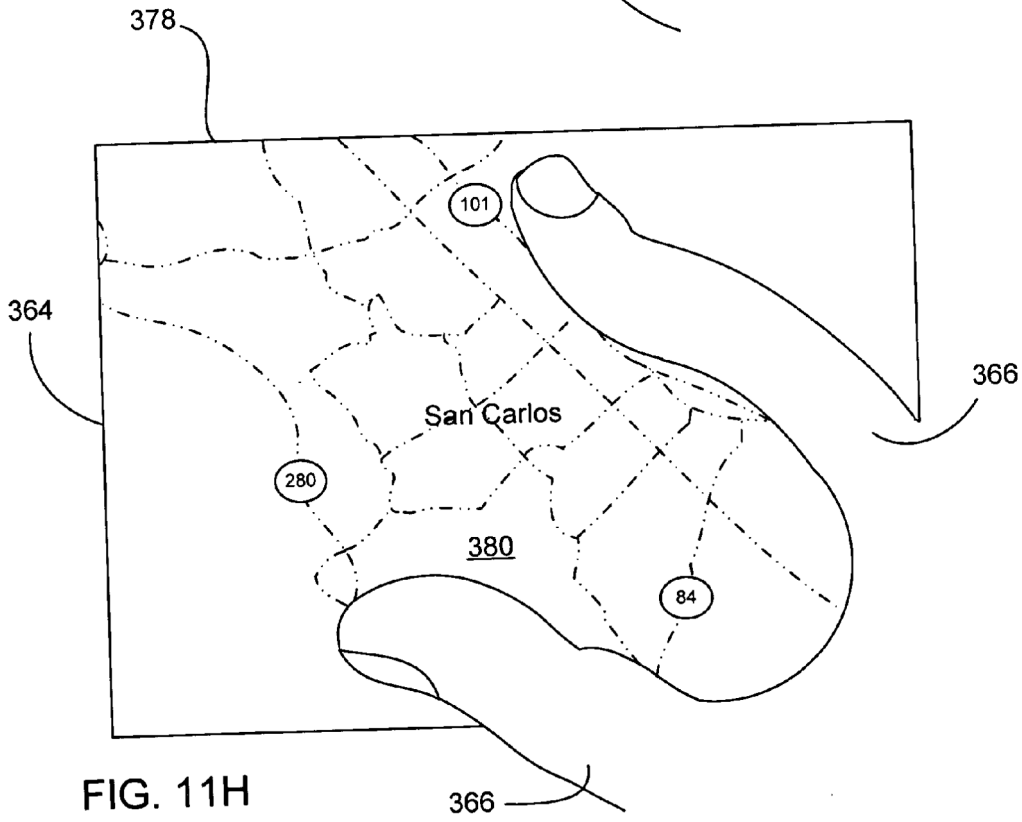


FIG. 11H

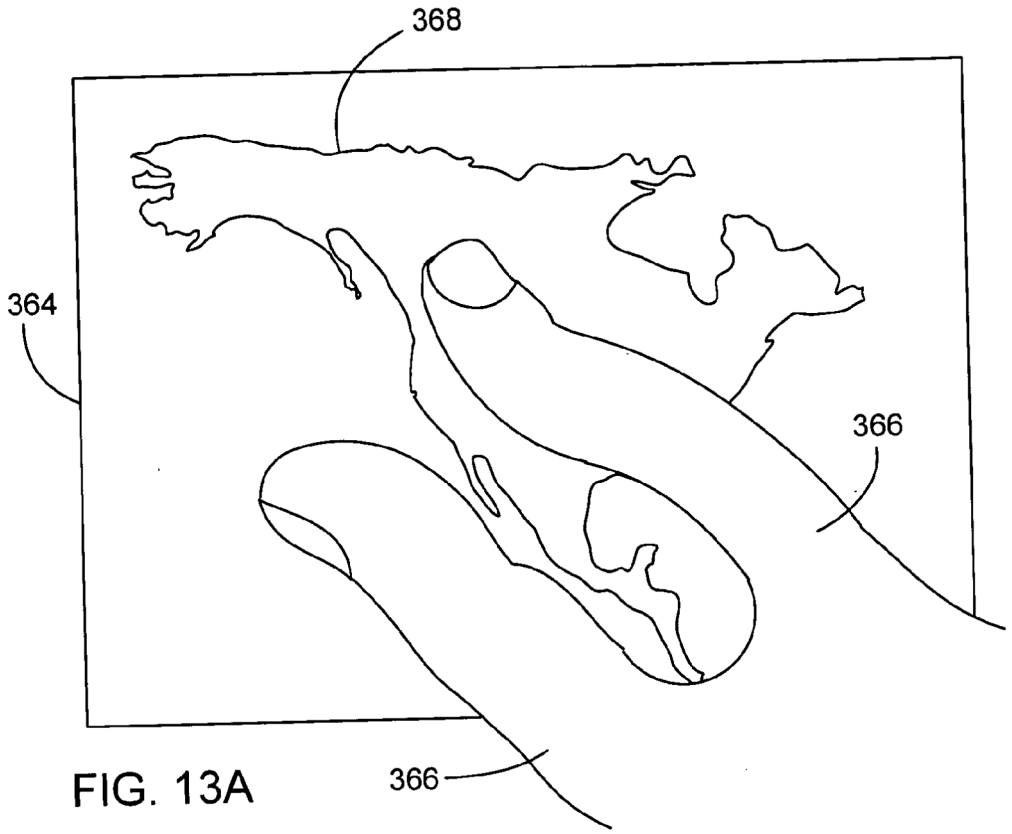


FIG. 13A

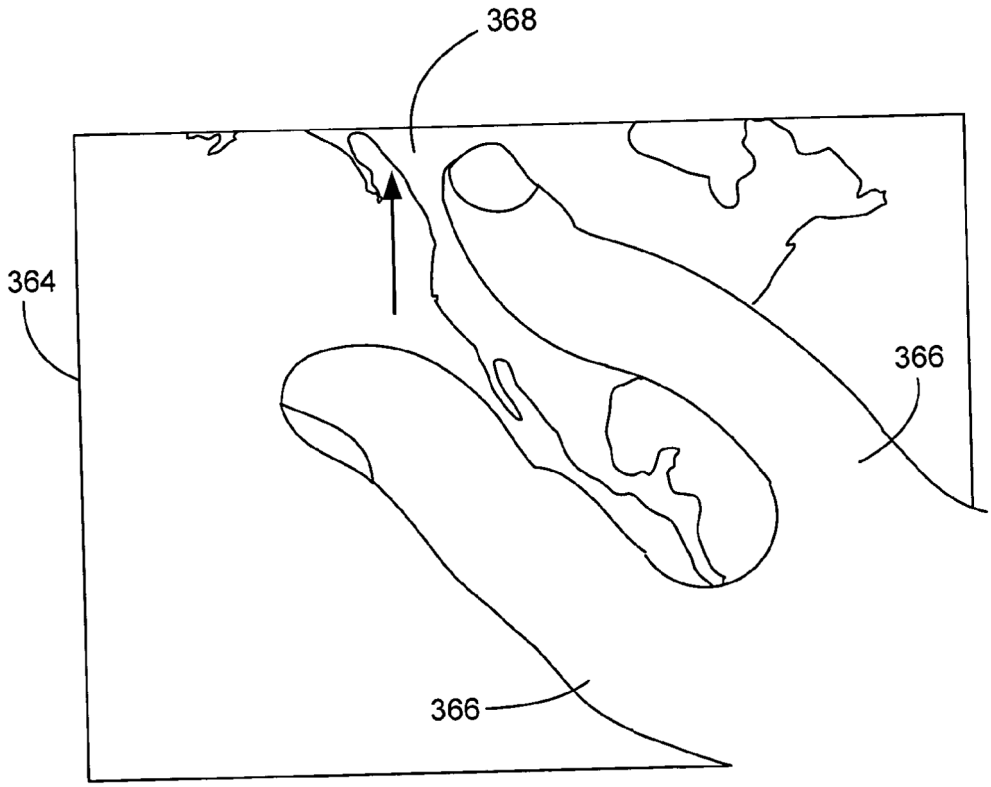


FIG. 13B

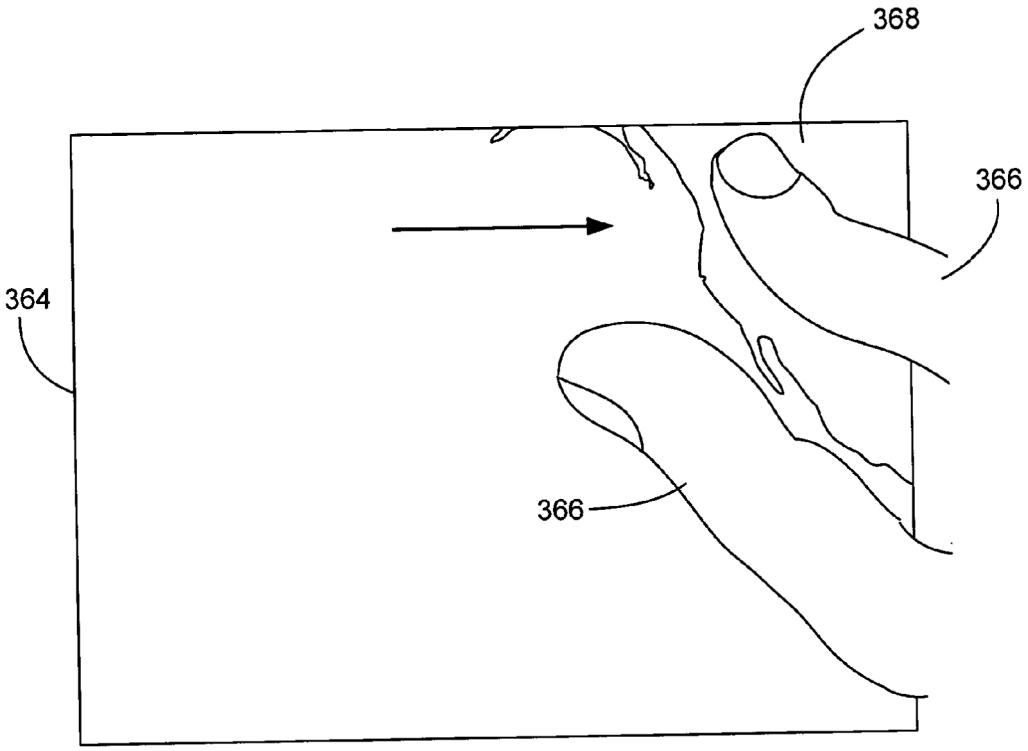


FIG. 13C

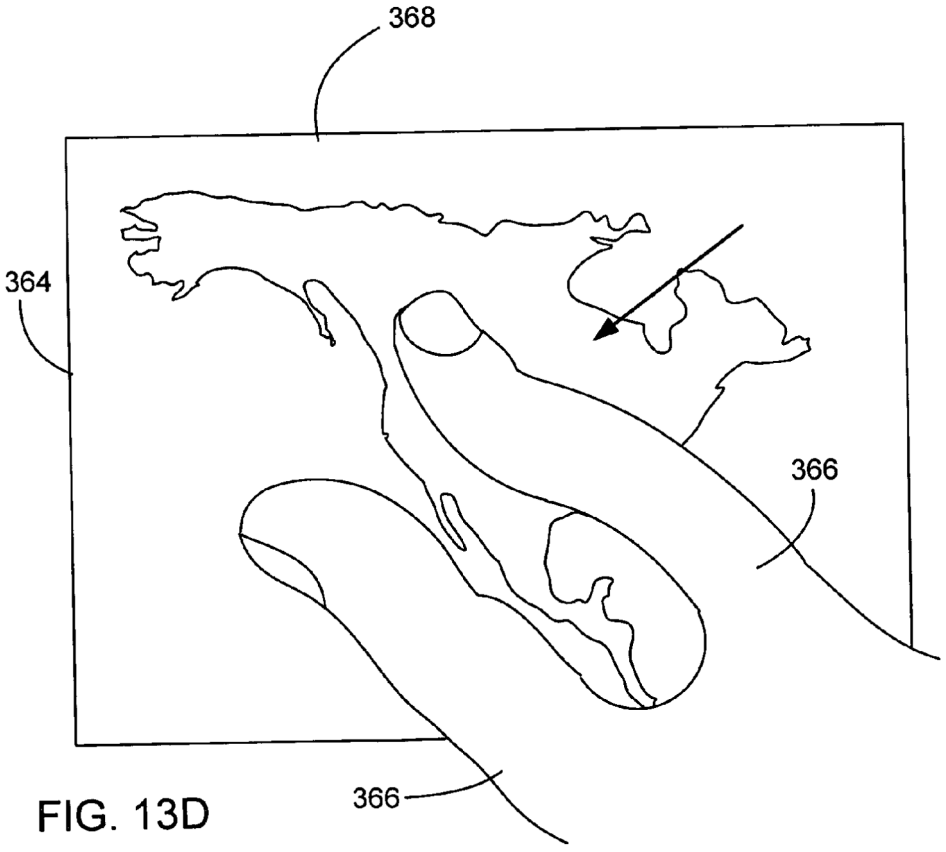


FIG. 13D

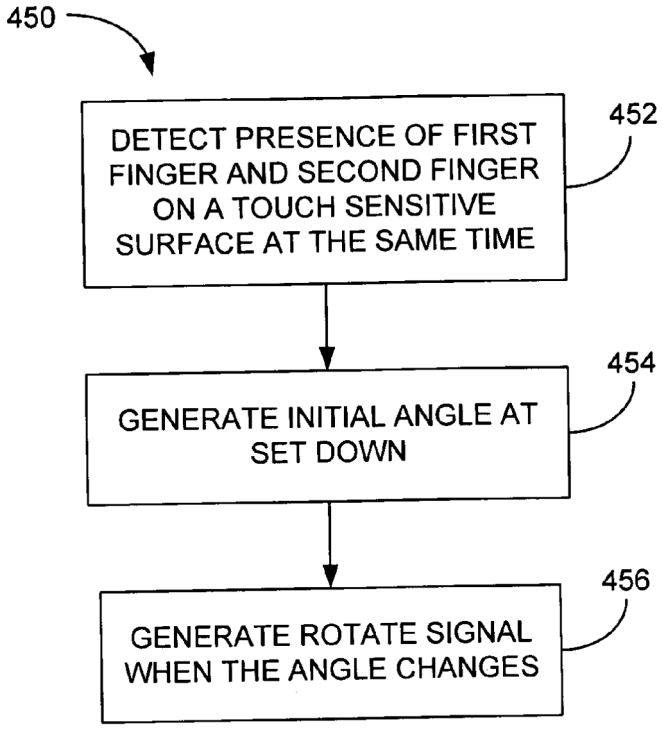

450
DETECT PRESENCE OF FIRST FINGER AND SECOND FINGER ON A TOUCH SENSITIVE SURFACE AT THE SAME TIME
452
GENERATE INITIAL ANGLE AT SET DOWN
454
GENERATE ROTATE SIGNAL WHEN THE ANGLE CHANGES
456

FIG. 14

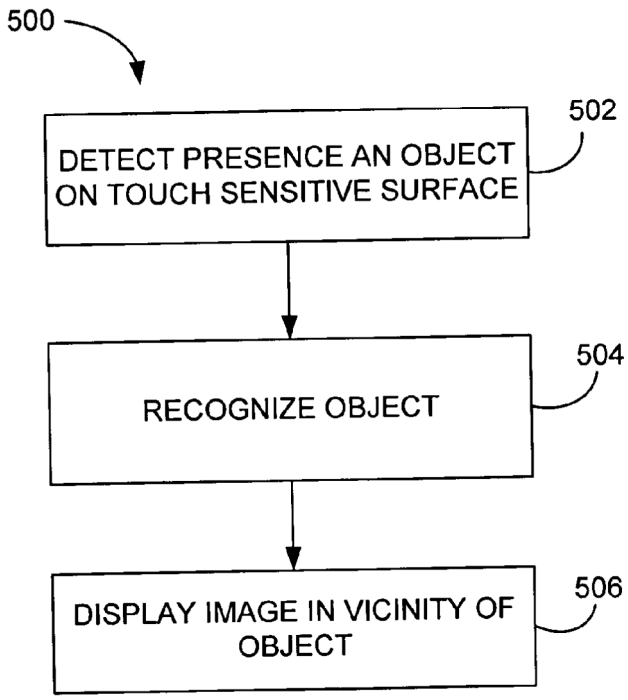

500
DETECT PRESENCE AN OBJECT ON TOUCH SENSITIVE SURFACE
502
RECOGNIZE OBJECT
504
DISPLAY IMAGE IN VICINITY OF OBJECT
506

FIG. 16

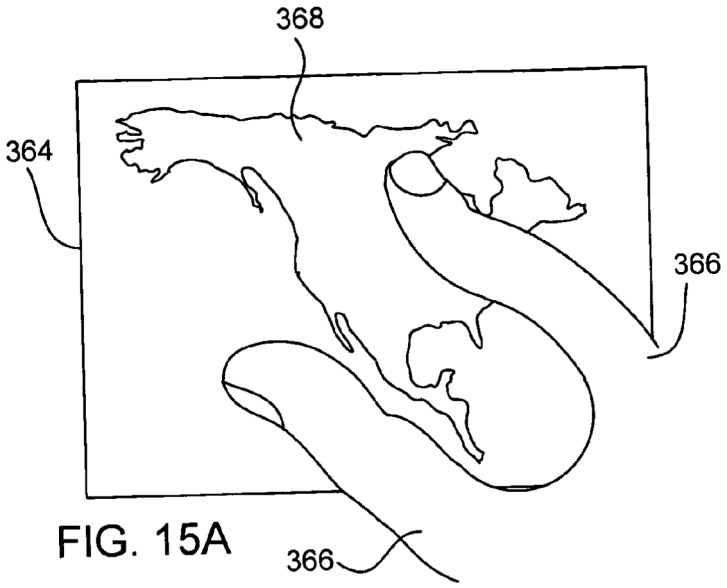


FIG. 15A

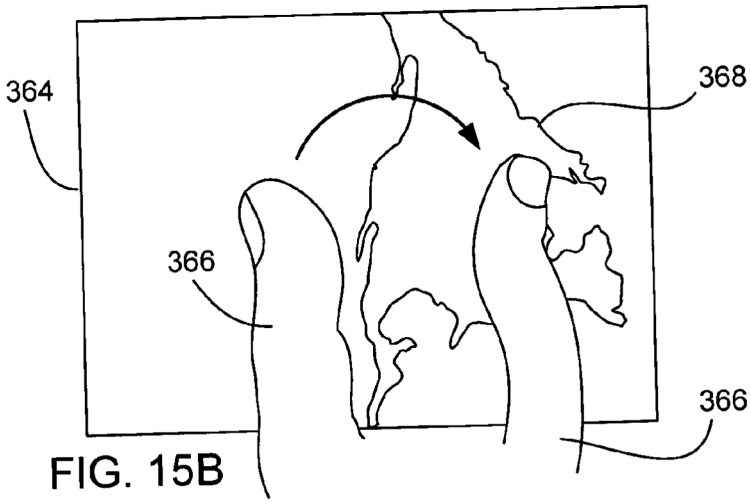


FIG. 15B

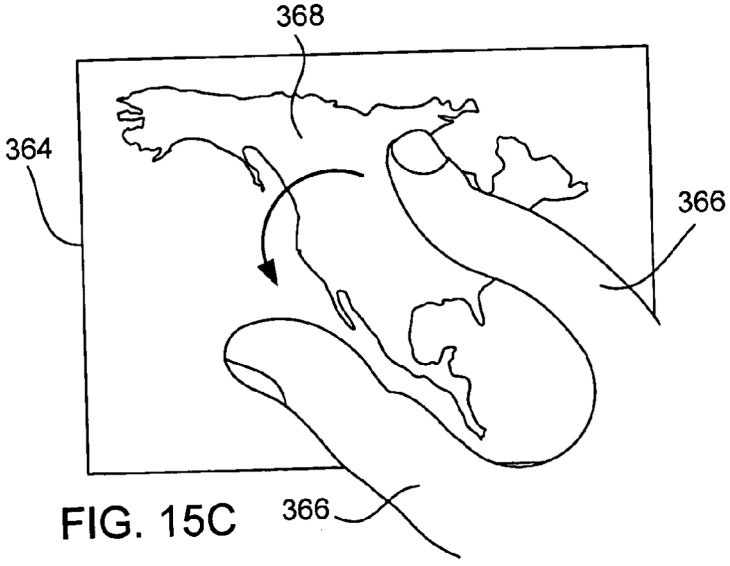


FIG. 15C

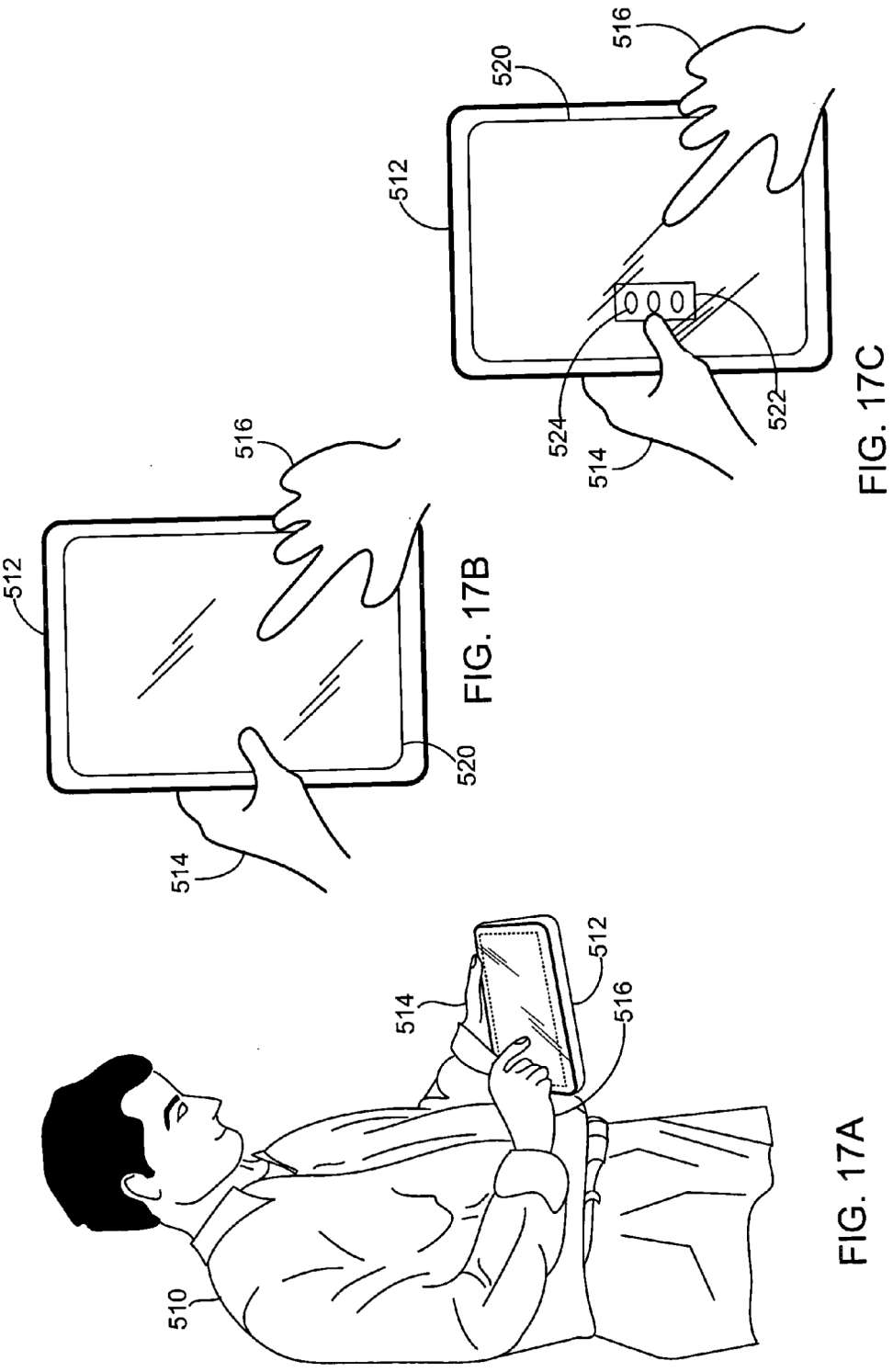


FIG. 17C

FIG. 17B

FIG. 17A

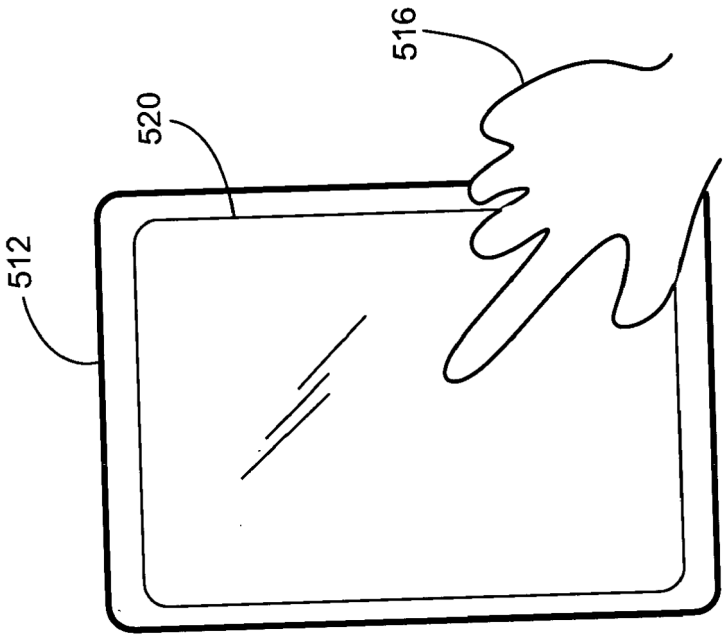


FIG. 17E

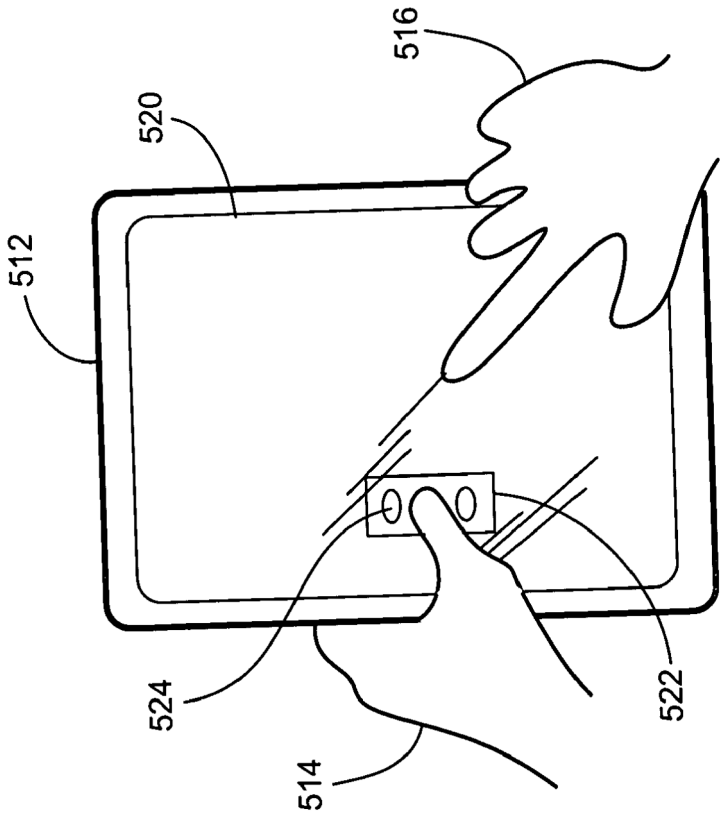


FIG. 17D

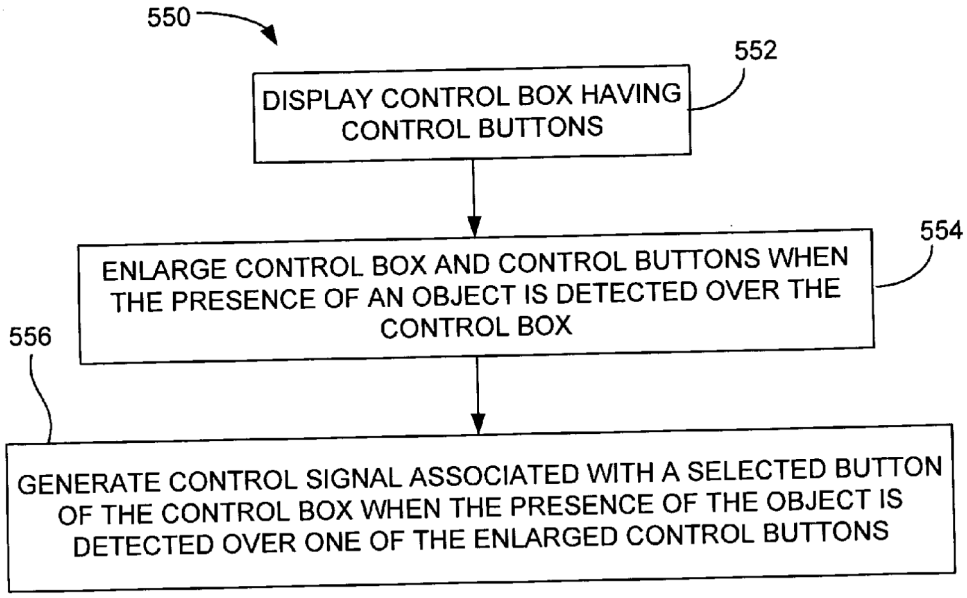

550
552
DISPLAY CONTROL BOX HAVING CONTROL BUTTONS
554
ENLARGE CONTROL BOX AND CONTROL BUTTONS WHEN THE PRESENCE OF AN OBJECT IS DETECTED OVER THE CONTROL BOX
556
GENERATE CONTROL SIGNAL ASSOCIATED WITH A SELECTED BUTTON OF THE CONTROL BOX WHEN THE PRESENCE OF THE OBJECT IS DETECTED OVER ONE OF THE ENLARGED CONTROL BUTTONS

FIG. 18

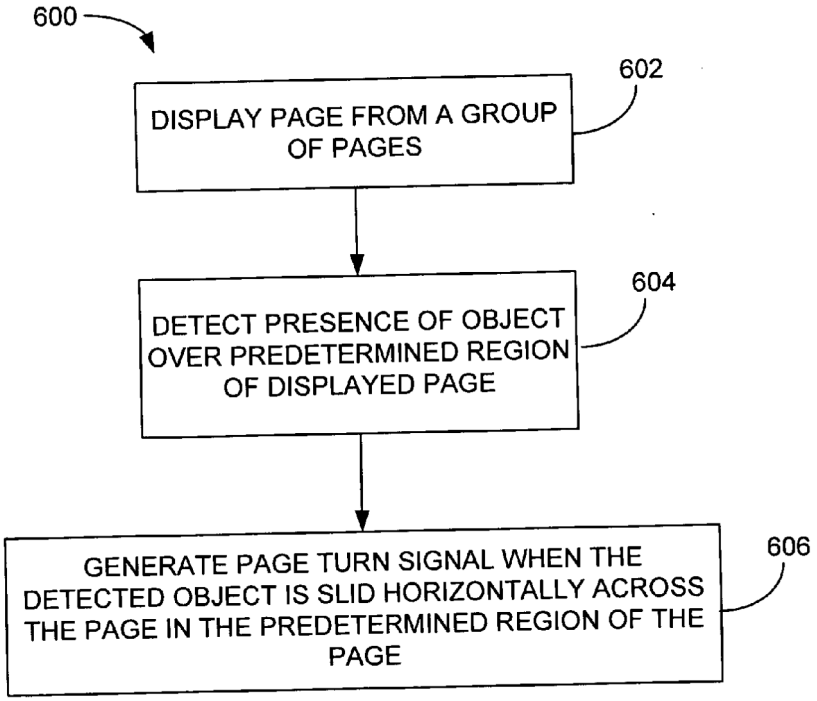

600
602
DISPLAY PAGE FROM A GROUP OF PAGES
604
DETECT PRESENCE OF OBJECT OVER PREDETERMINED REGION OF DISPLAYED PAGE
606
GENERATE PAGE TURN SIGNAL WHEN THE DETECTED OBJECT IS SLID HORIZONTALLY ACROSS THE PAGE IN THE PREDETERMINED REGION OF THE PAGE

FIG. 20

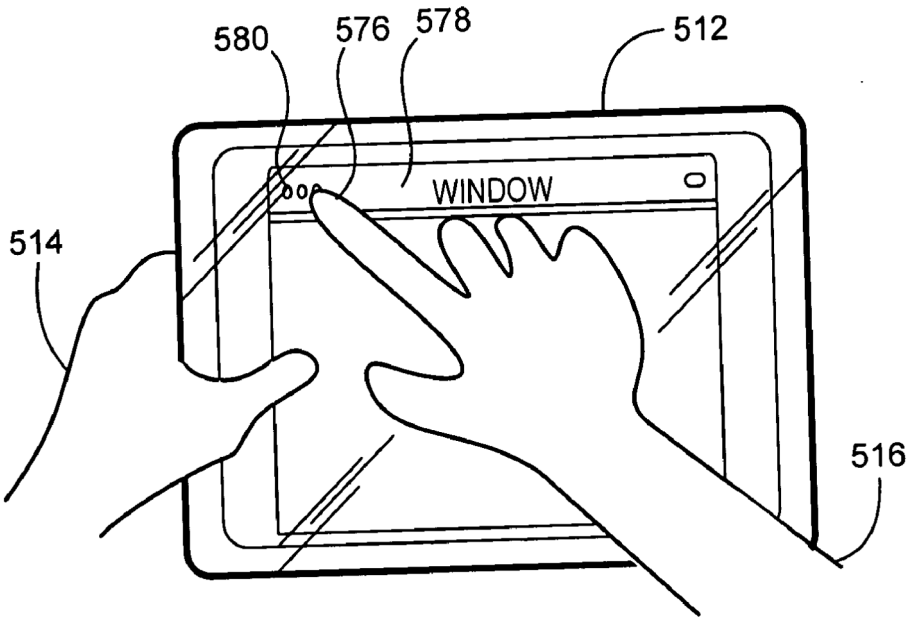


FIG. 19A

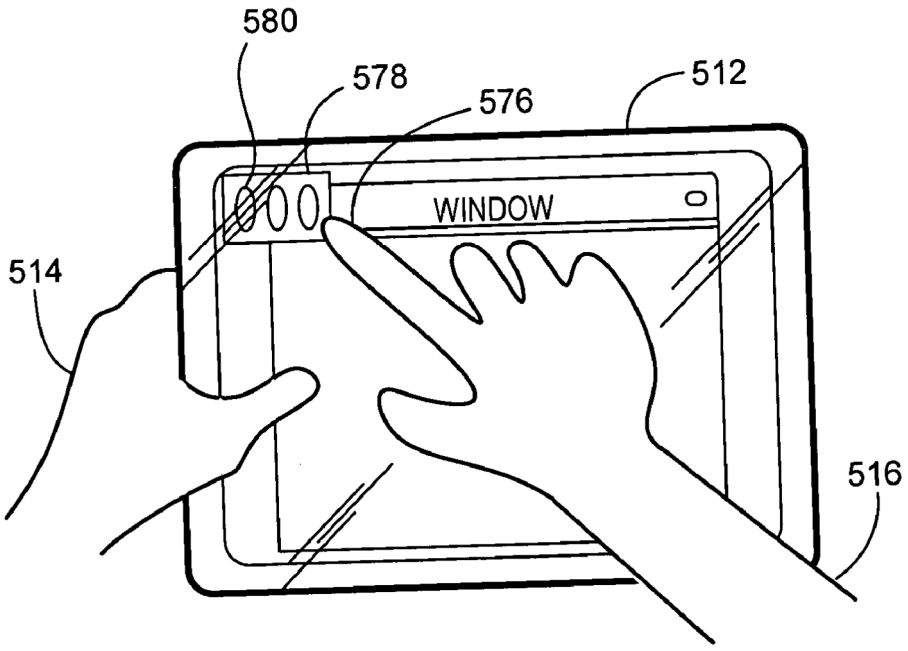


FIG. 19B

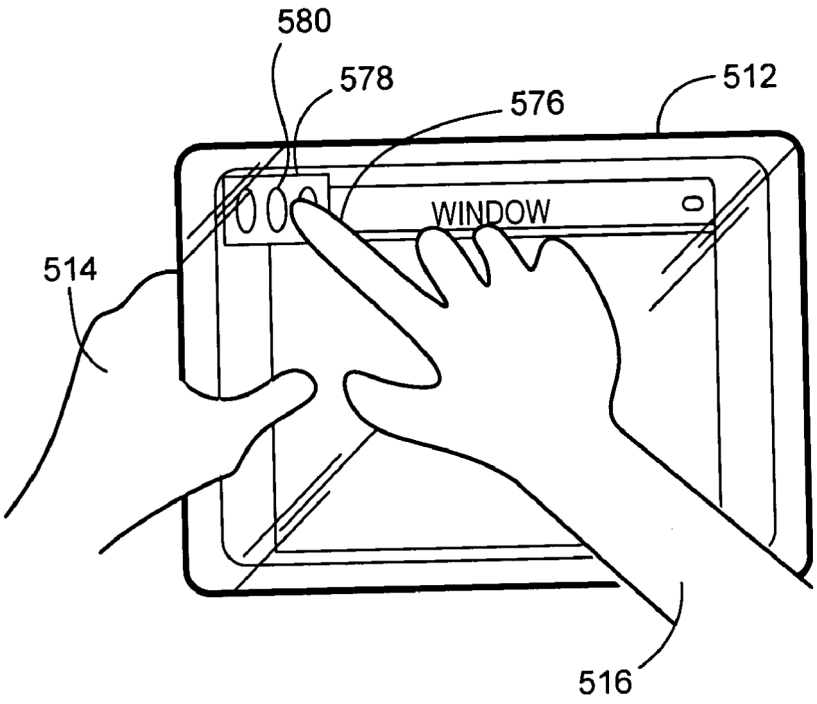


FIG. 19C

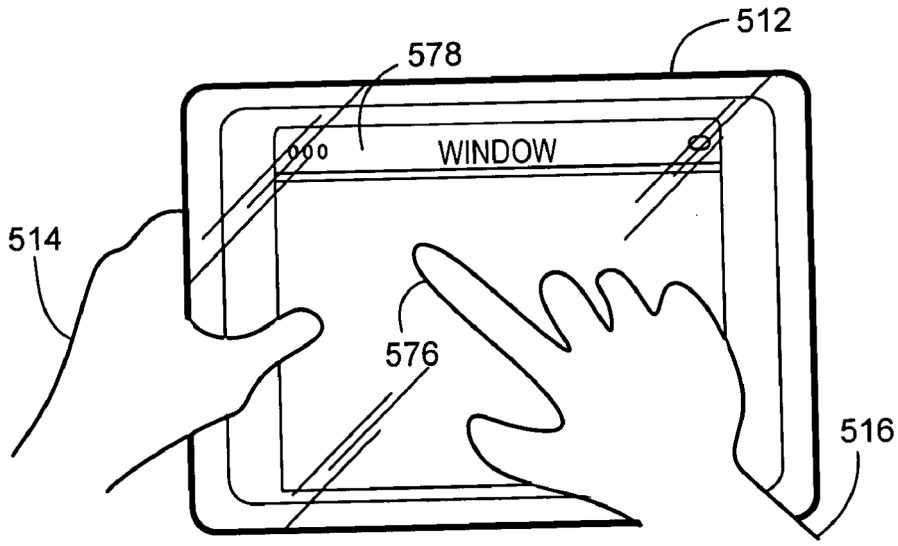


FIG. 19D

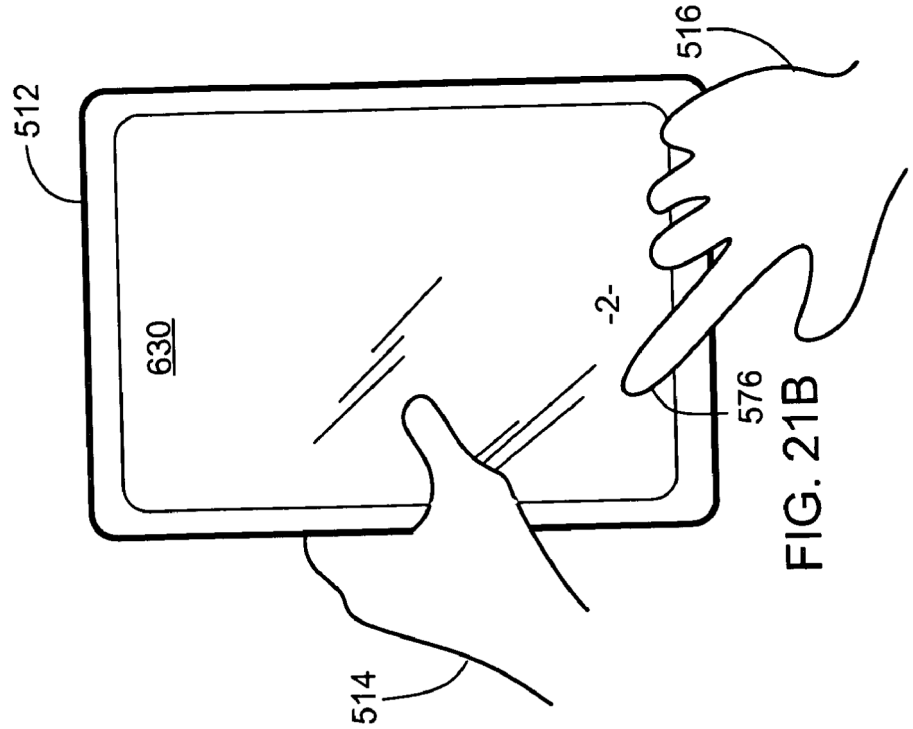


FIG. 21B

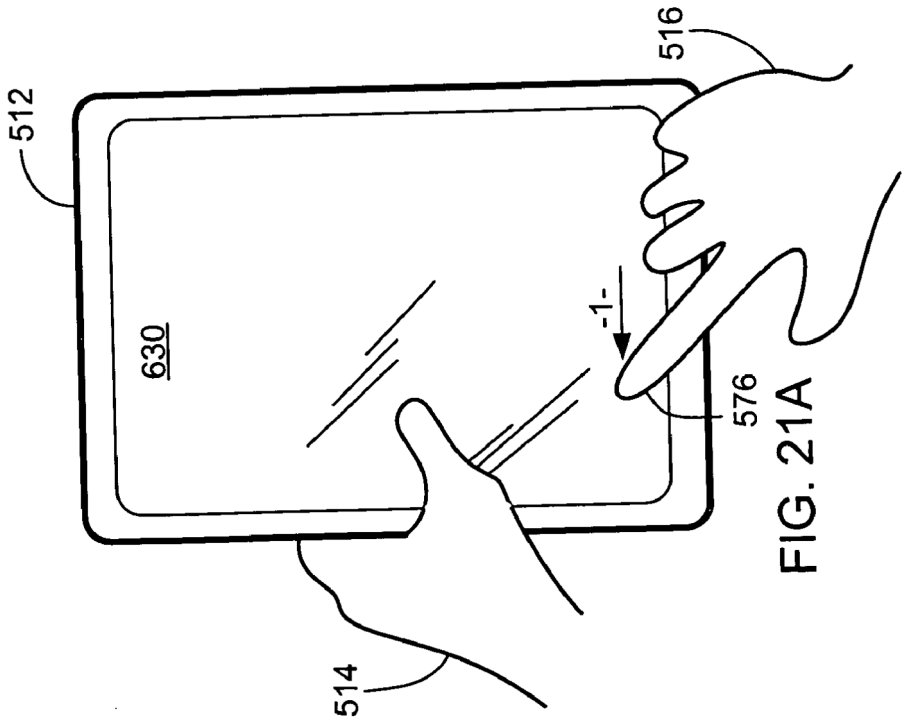


FIG. 21A

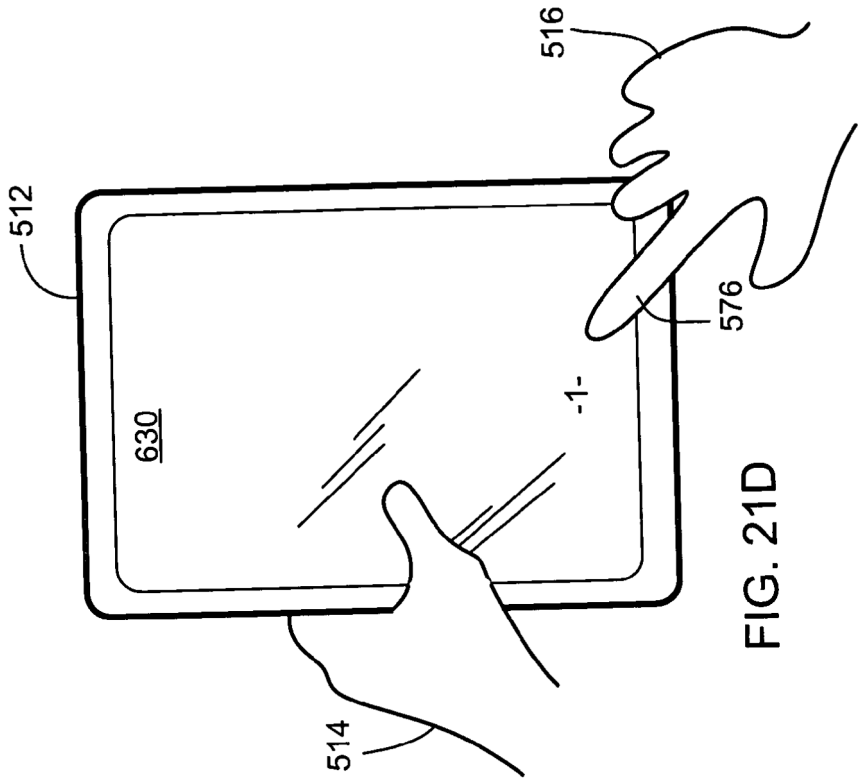


FIG. 21D

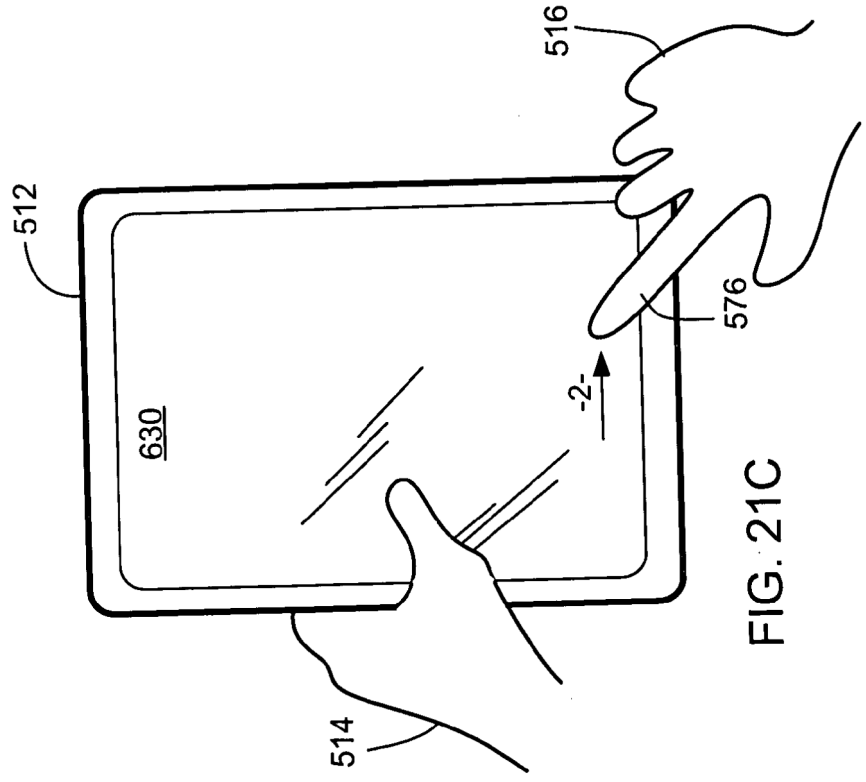


FIG. 21C

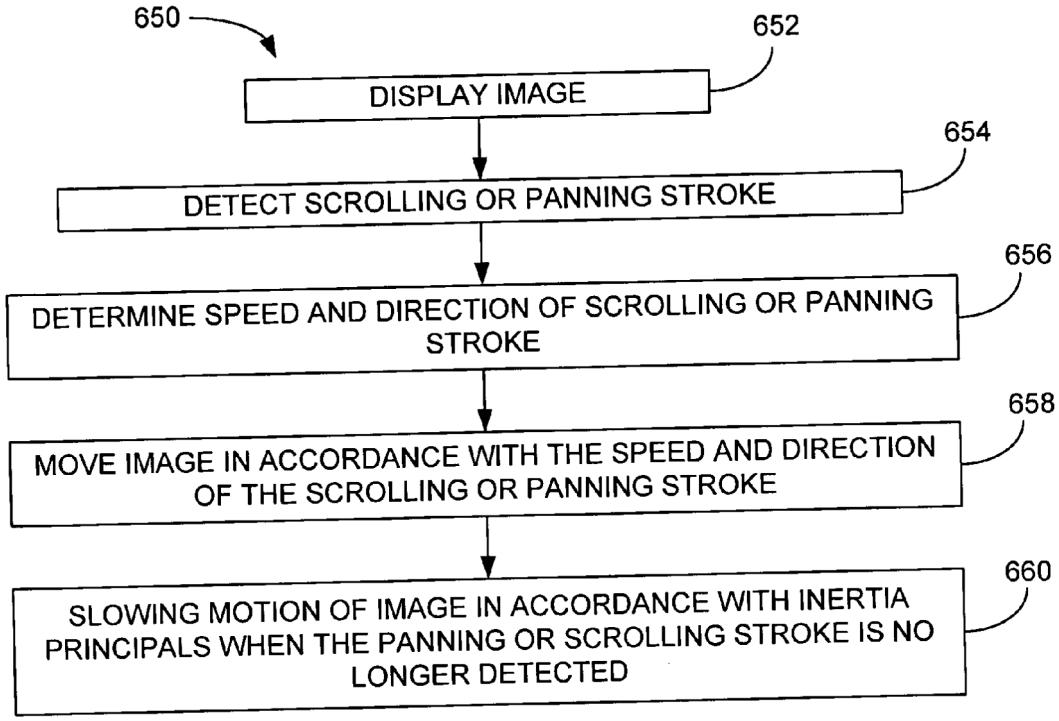

650
652
DISPLAY IMAGE
654
DETECT SCROLLING OR PANNING STROKE
656
DETERMINE SPEED AND DIRECTION OF SCROLLING OR PANNING STROKE
658
MOVE IMAGE IN ACCORDANCE WITH THE SPEED AND DIRECTION OF THE SCROLLING OR PANNING STROKE
660
SLOWING MOTION OF IMAGE IN ACCORDANCE WITH INERTIA PRINCIPALS WHEN THE PANNING OR SCROLLING STROKE IS NO LONGER DETECTED

FIG. 22

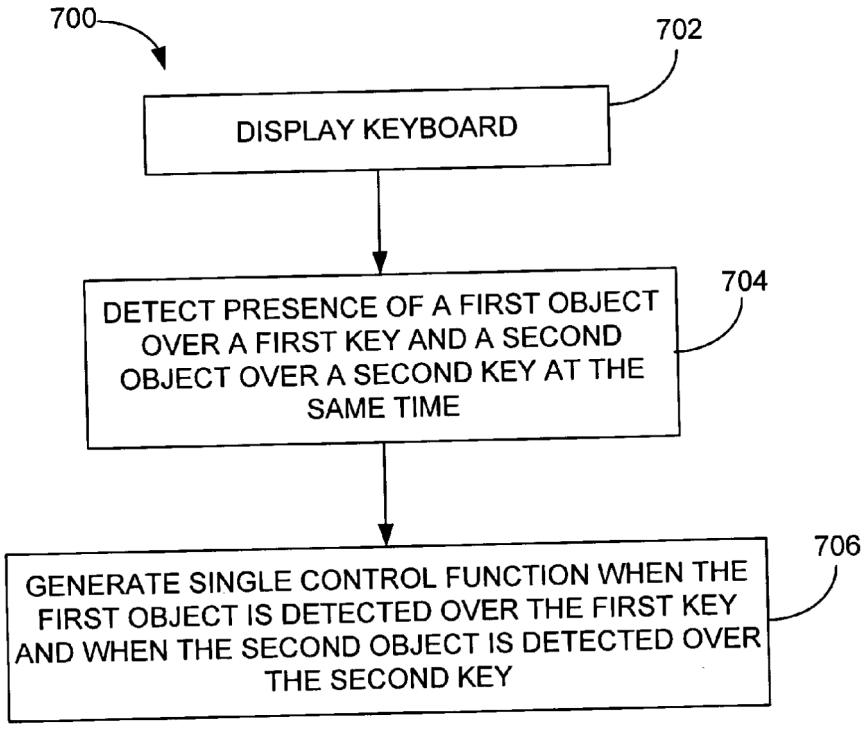

700
702
DISPLAY KEYBOARD
704
DETECT PRESENCE OF A FIRST OBJECT OVER A FIRST KEY AND A SECOND OBJECT OVER A SECOND KEY AT THE SAME TIME
706
GENERATE SINGLE CONTROL FUNCTION WHEN THE FIRST OBJECT IS DETECTED OVER THE FIRST KEY AND WHEN THE SECOND OBJECT IS DETECTED OVER THE SECOND KEY

FIG. 24

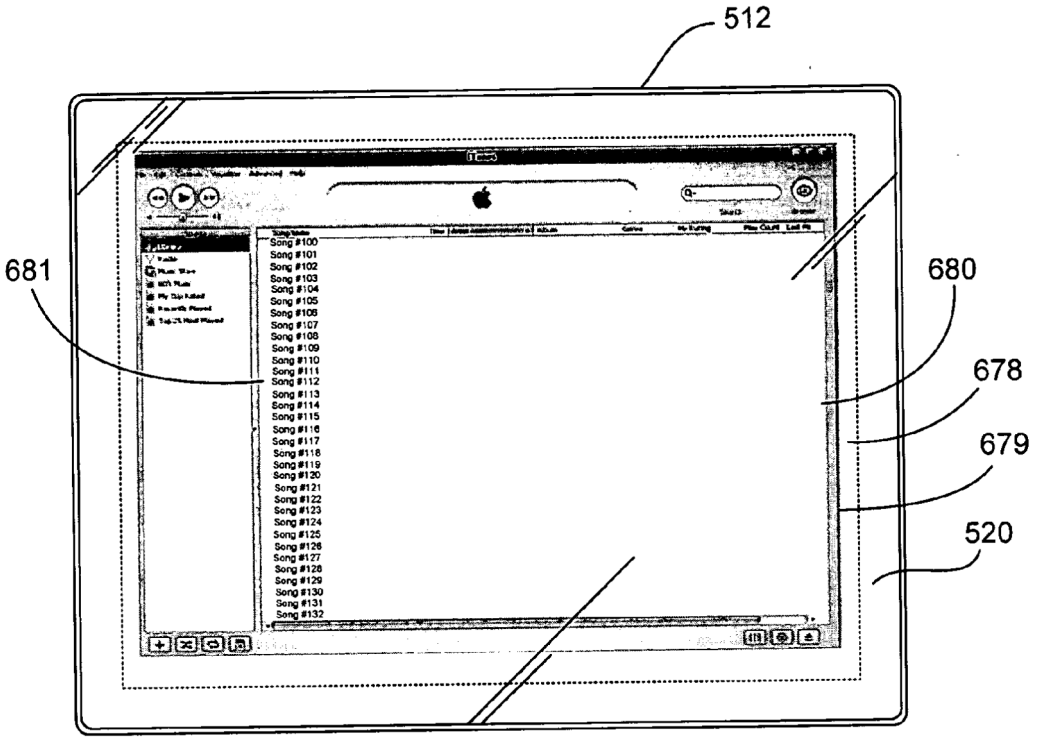


FIG. 23A

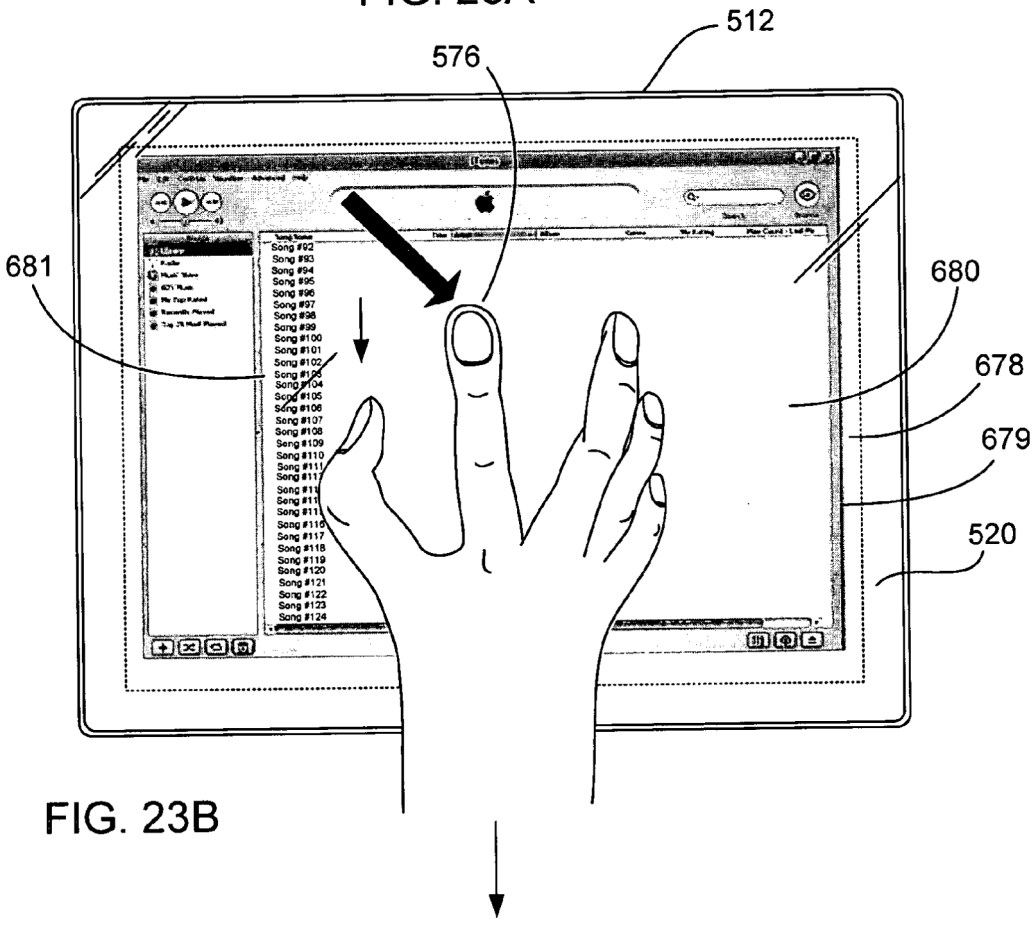


FIG. 23B

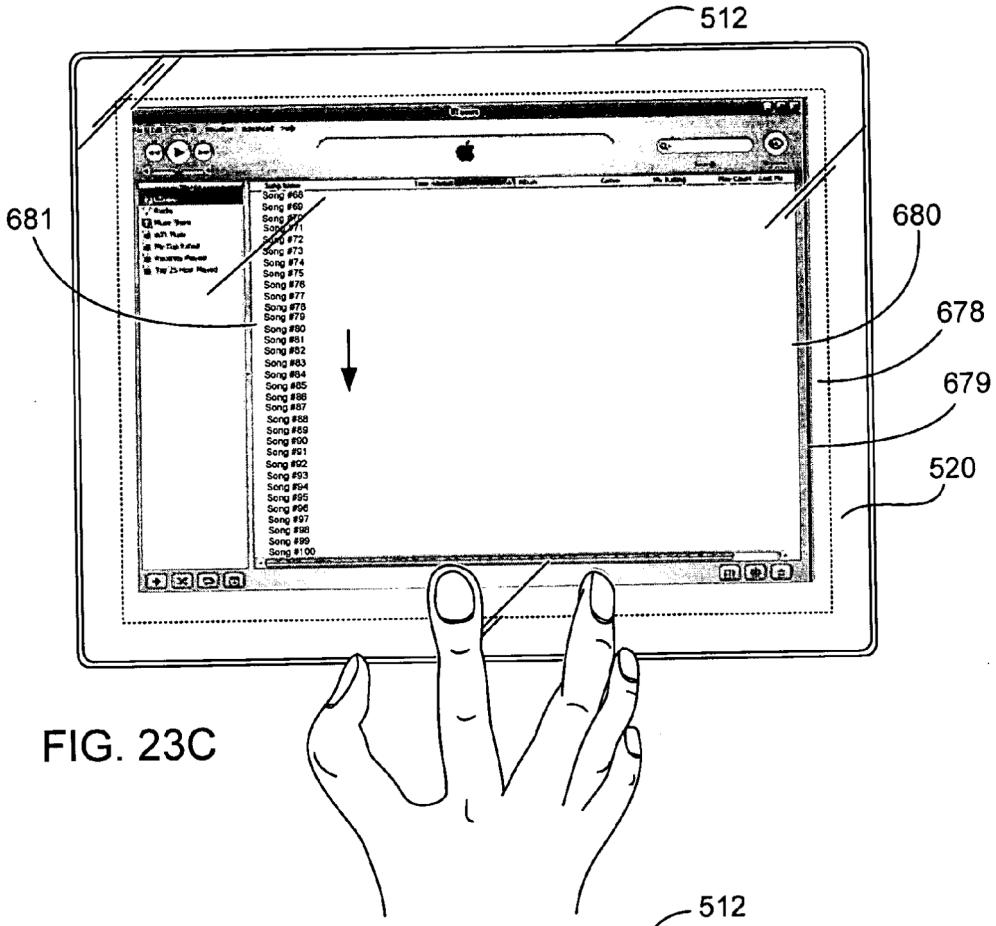


FIG. 23C

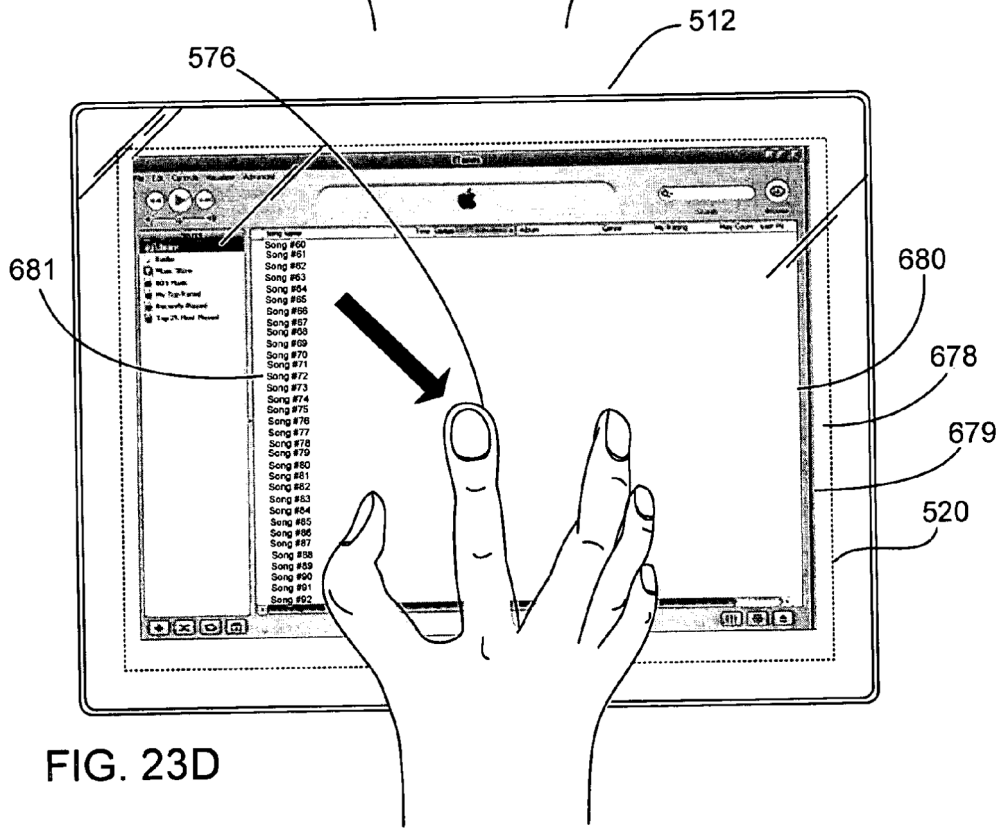


FIG. 23D

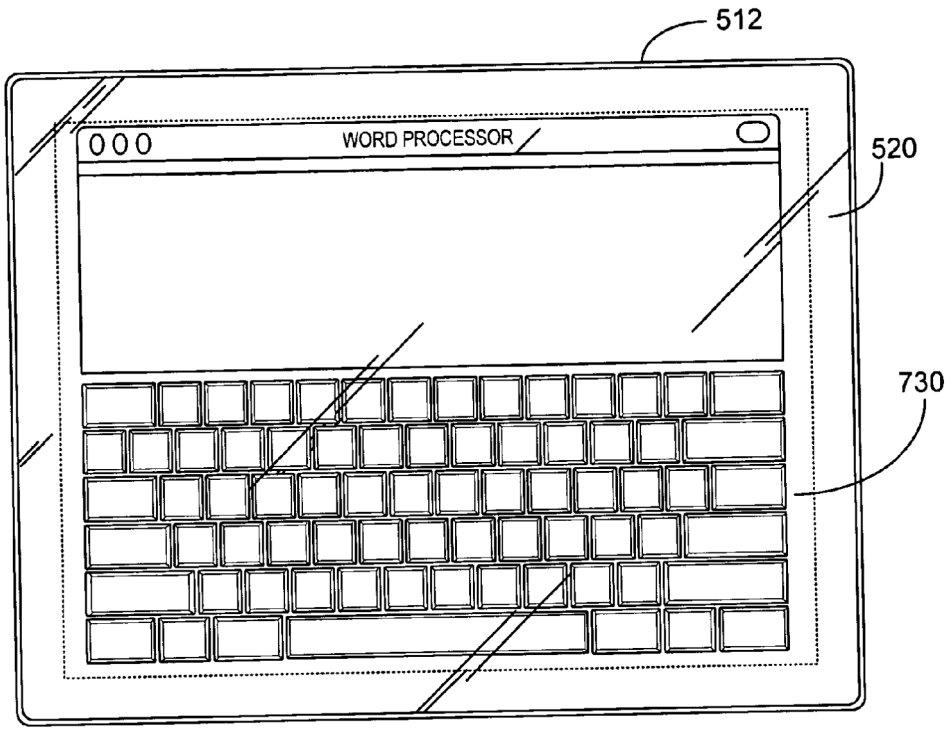


FIG. 25A

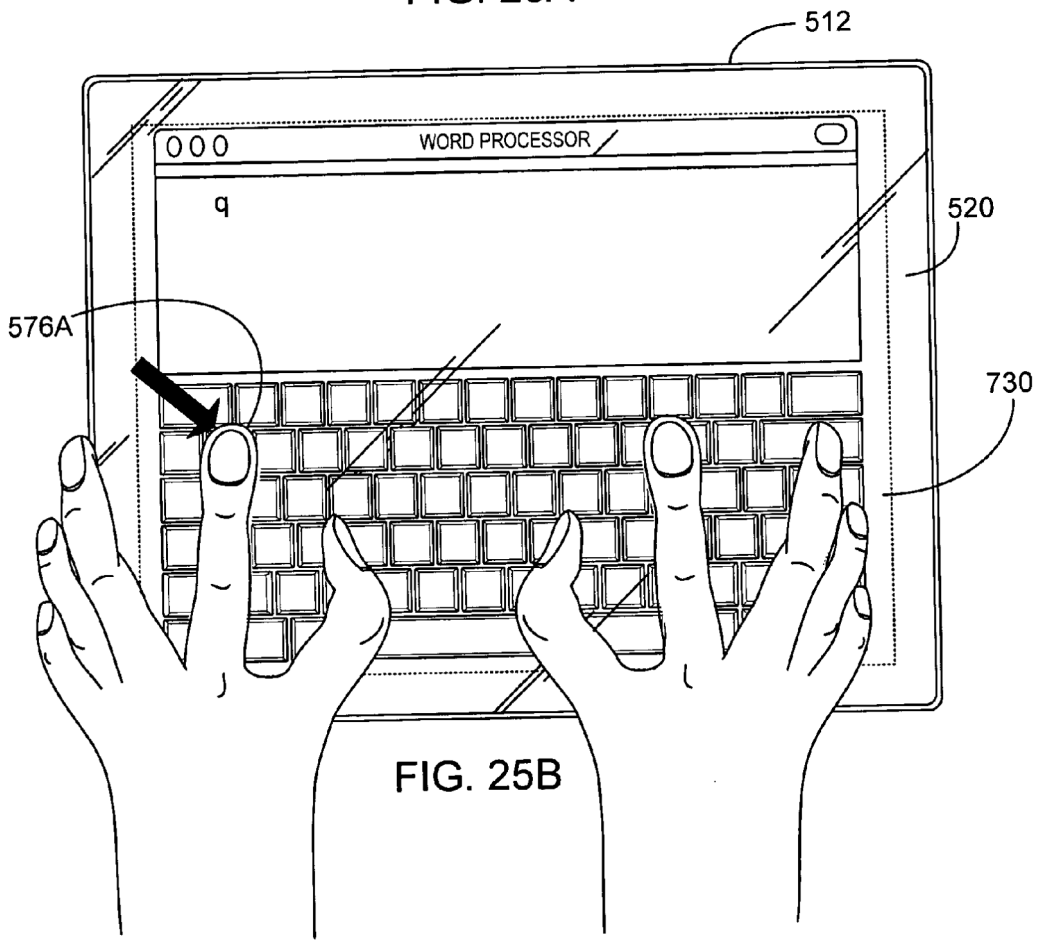


FIG. 25B

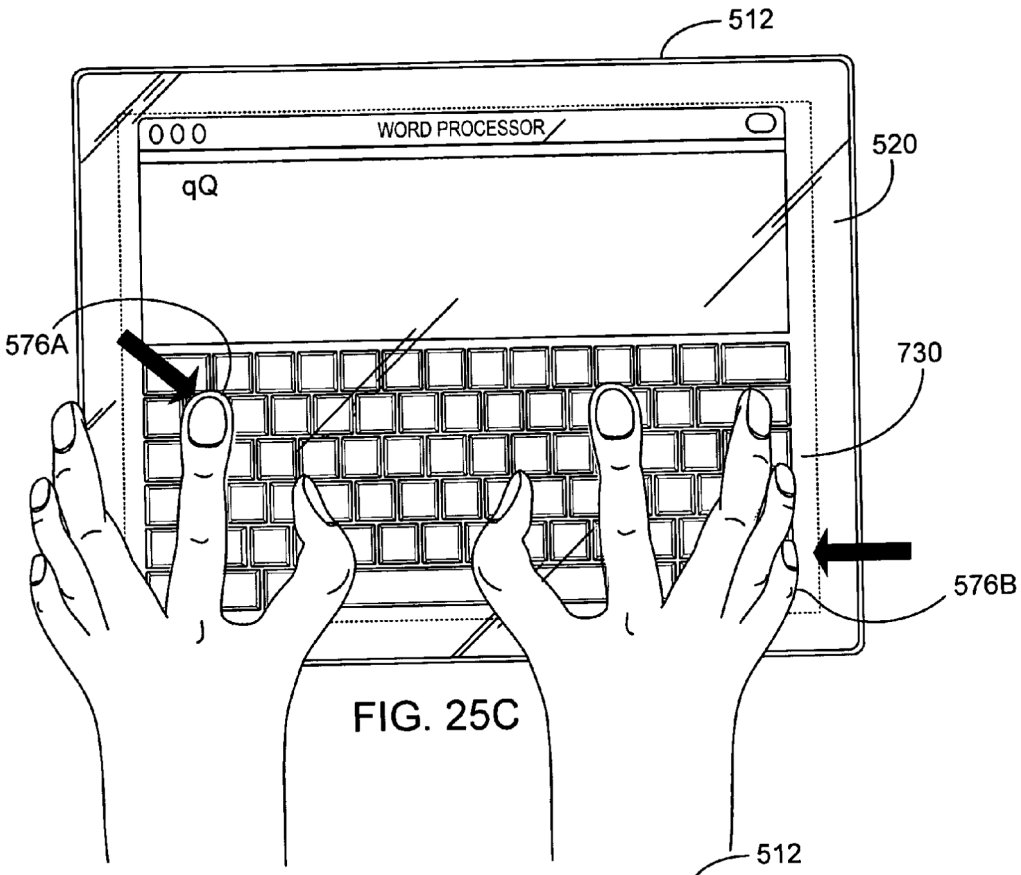


FIG. 25C

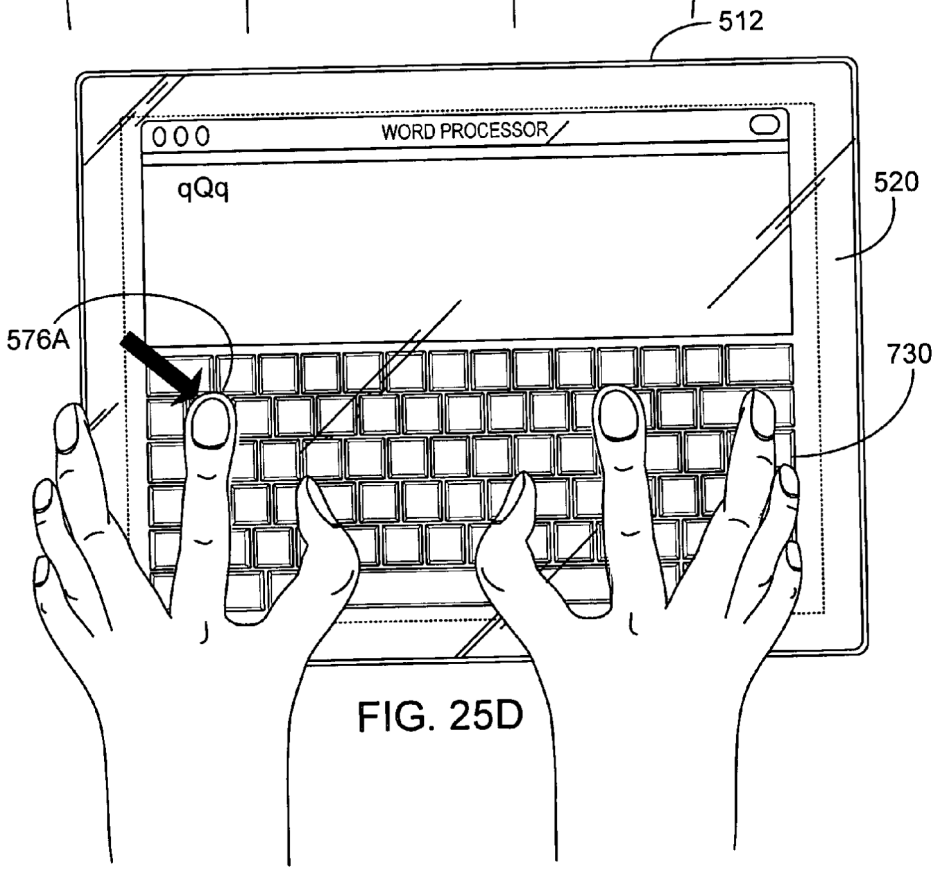


FIG. 25D

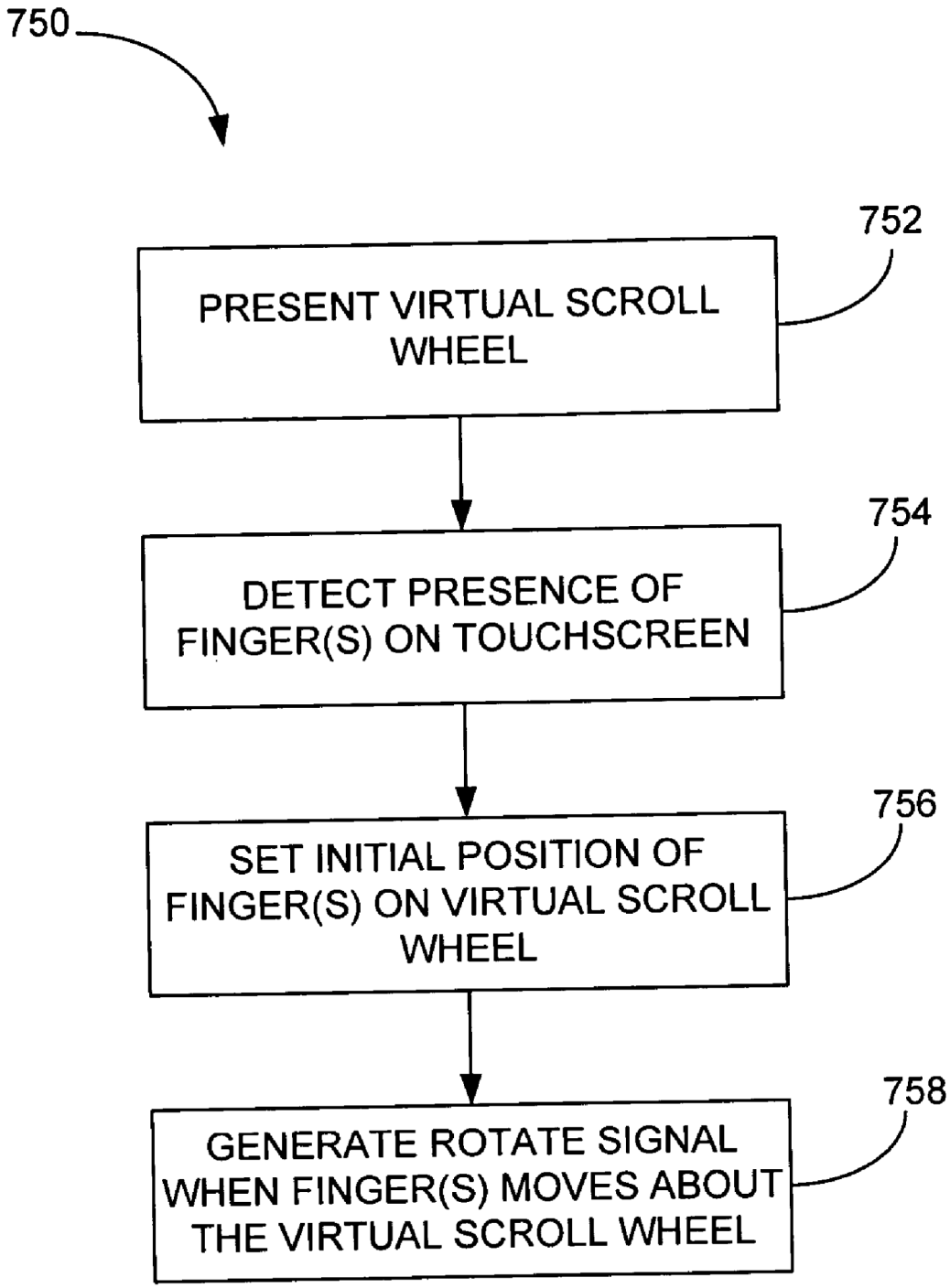

750
752
PRESENT VIRTUAL SCROLL WHEEL
754
DETECT PRESENCE OF FINGER(S) ON TOUCHSCREEN
756
SET INITIAL POSITION OF FINGER(S) ON VIRTUAL SCROLL WHEEL
758
GENERATE ROTATE SIGNAL WHEN FINGER(S) MOVES ABOUT THE VIRTUAL SCROLL WHEEL

FIG. 26

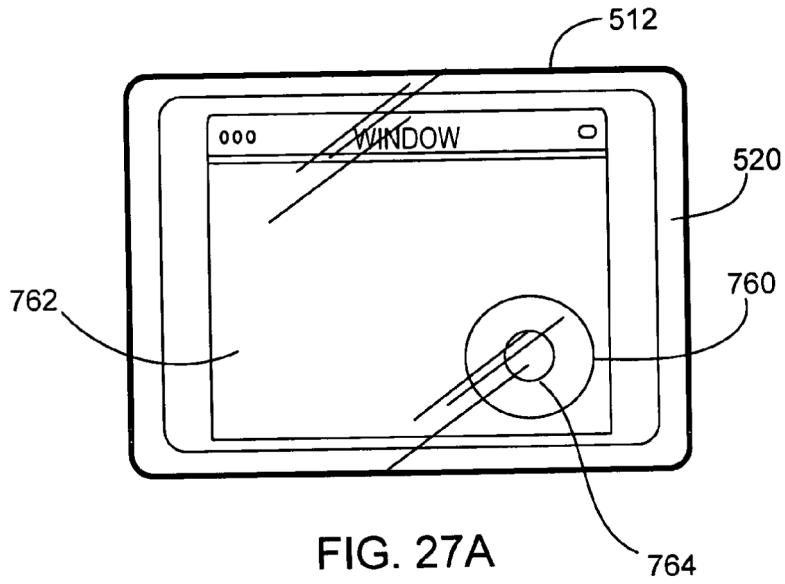


FIG. 27A

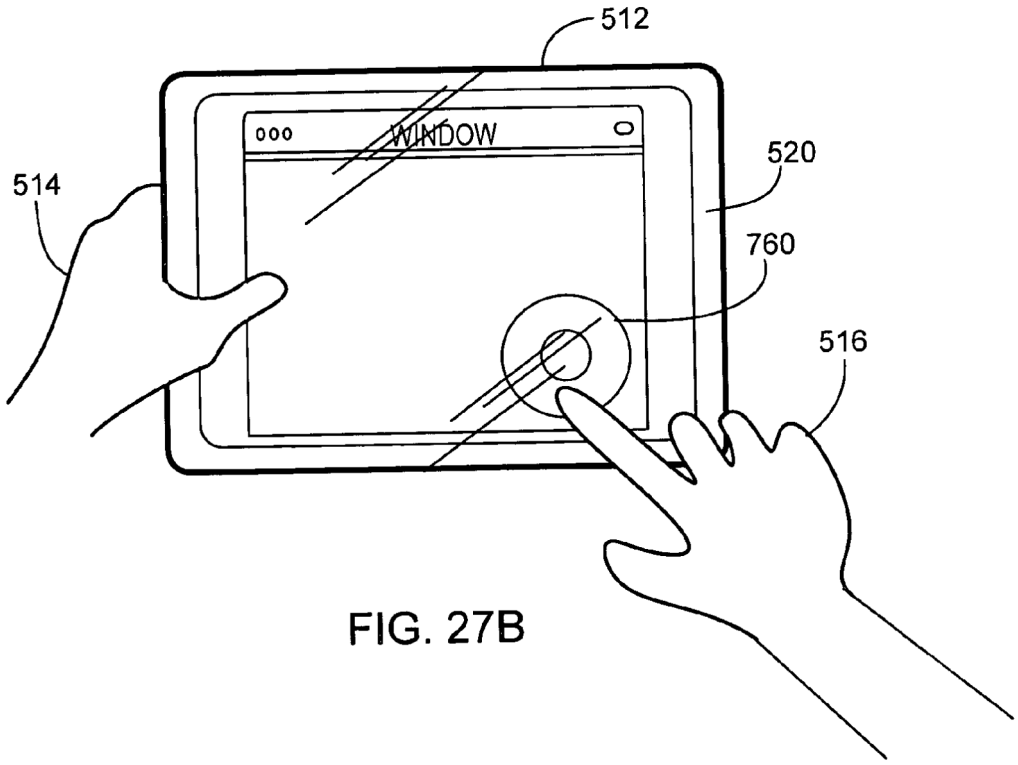


FIG. 27B

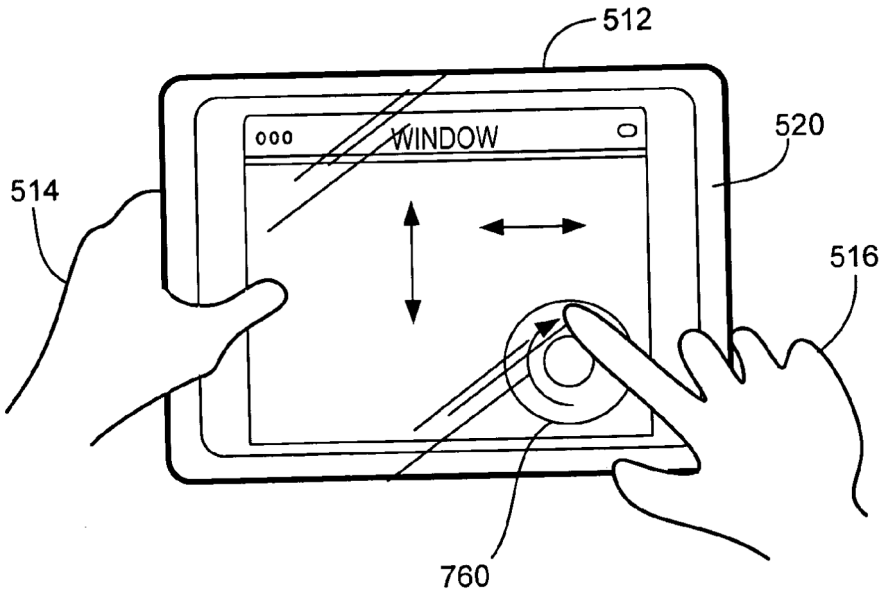


FIG. 27C

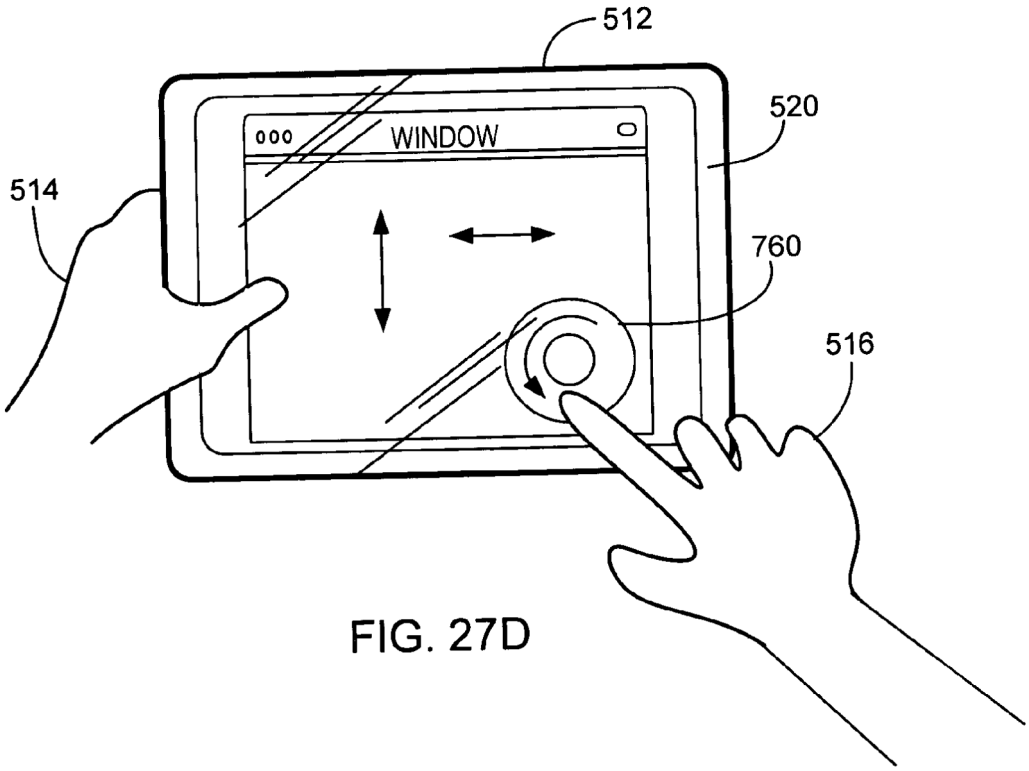


FIG. 27D

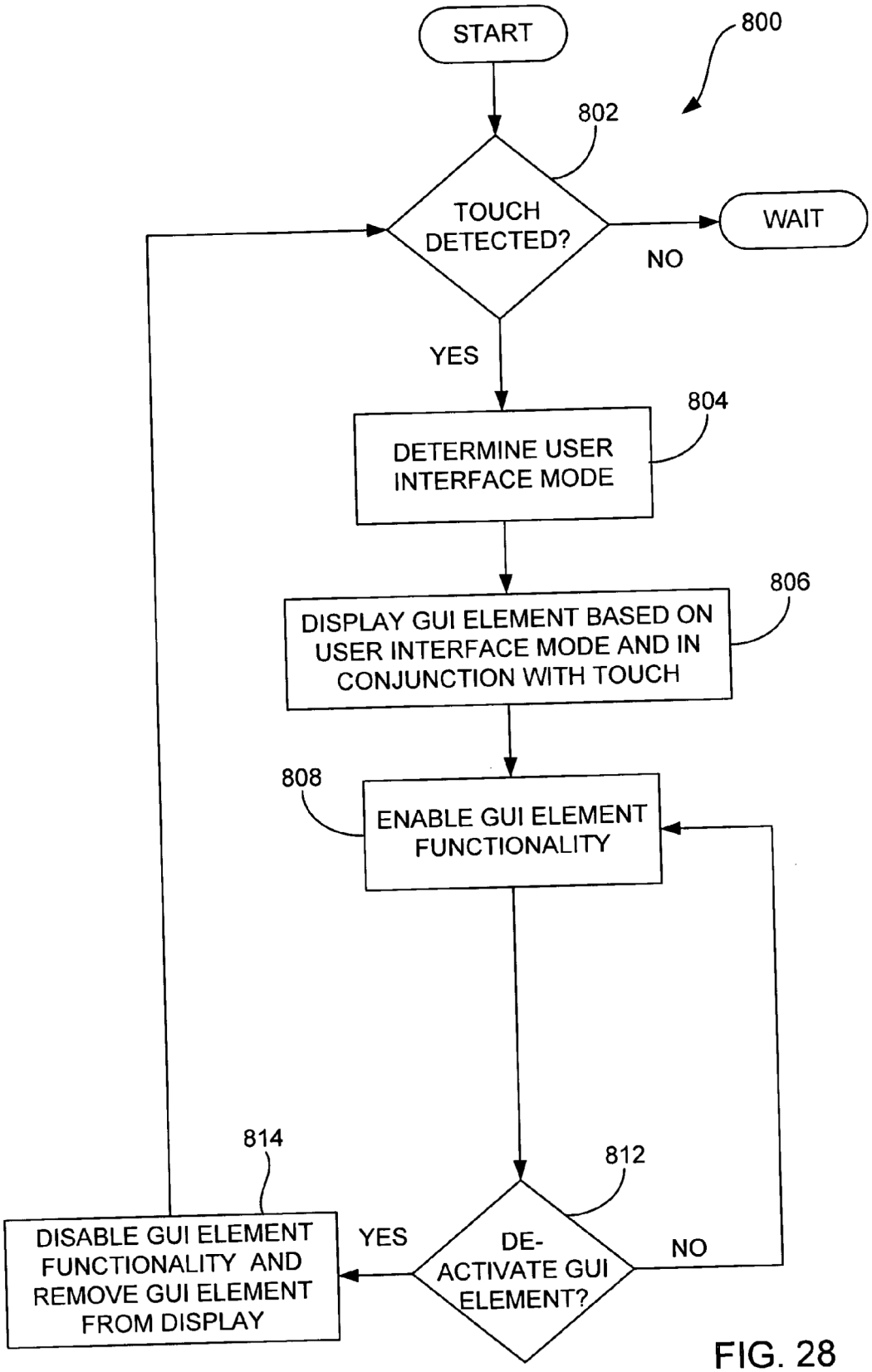

START
800
802
TOUCH DETECTED?
WAIT
NO
YES
804
DETERMINE USER INTERFACE MODE
806
DISPLAY GUI ELEMENT BASED ON USER INTERFACE MODE AND IN CONJUNCTION WITH TOUCH
808
ENABLE GUI ELEMENT FUNCTIONALITY
812
DE-ACTIVATE GUI ELEMENT?
YES
NO
814
DISABLE GUI ELEMENT FUNCTIONALITY AND REMOVE GUI ELEMENT FROM DISPLAY

FIG. 28

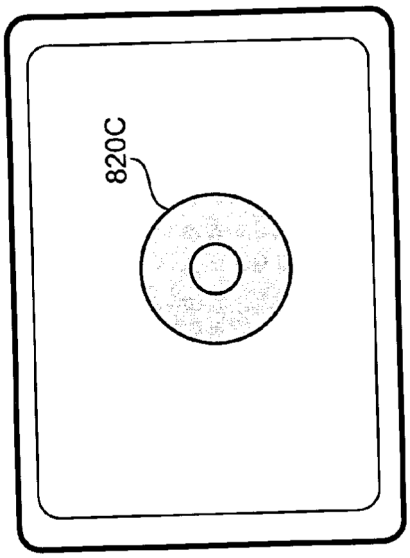


FIG. 29C

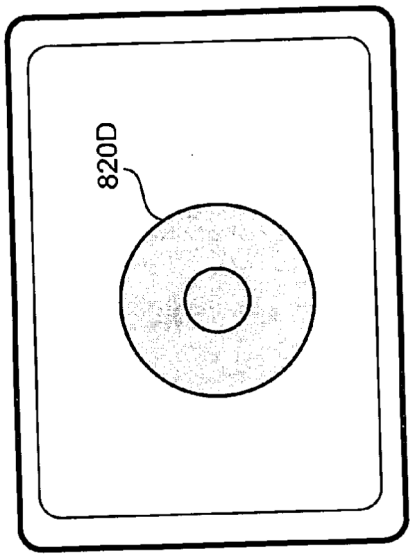


FIG. 29D

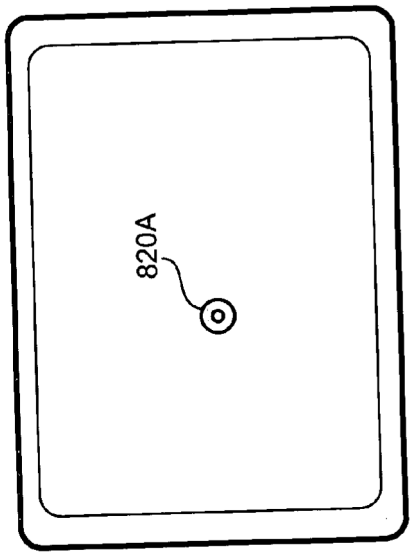


FIG. 29A

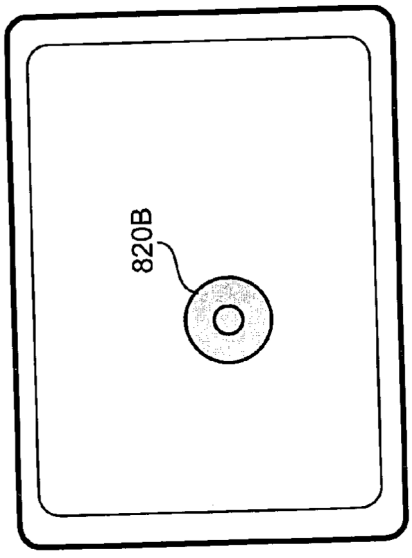


FIG. 29B

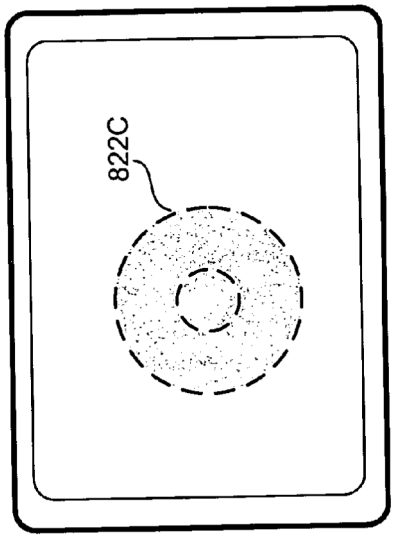


FIG. 30C

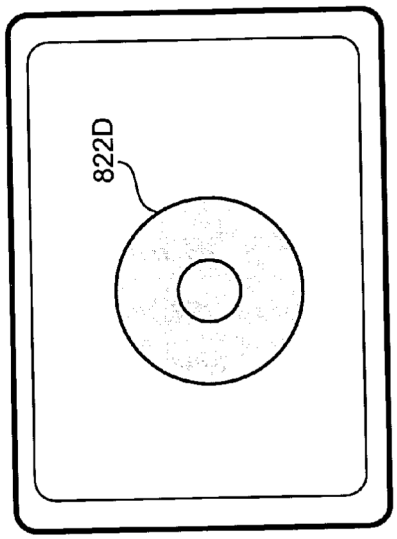


FIG. 30D

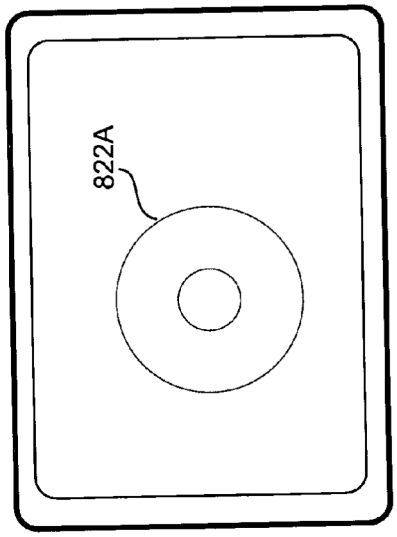


FIG. 30A

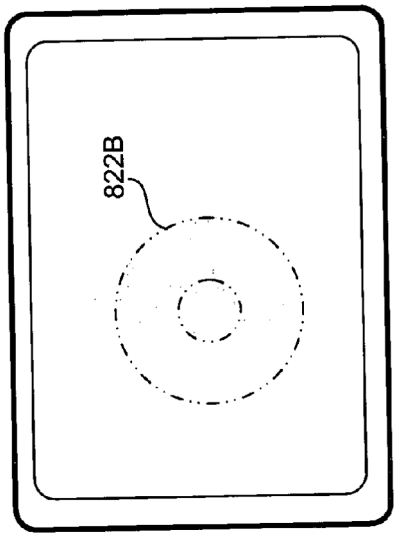


FIG. 30B

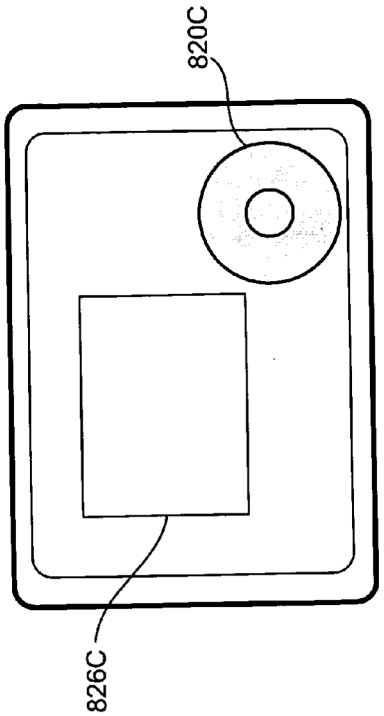


FIG. 31C

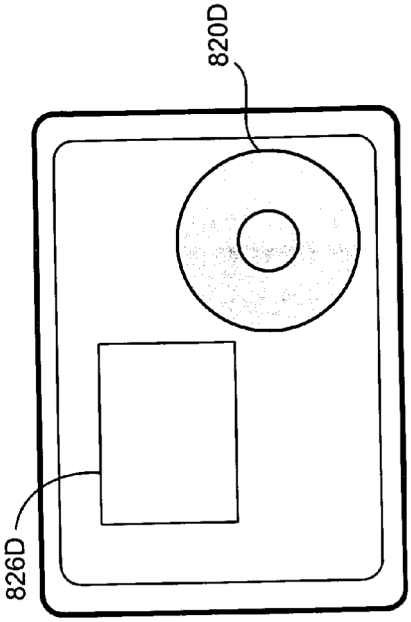


FIG. 31D

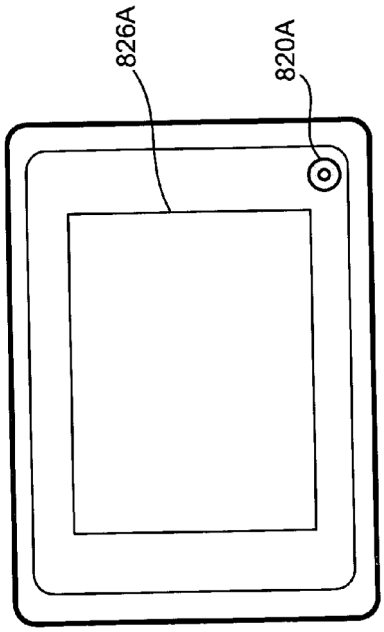


FIG. 31A

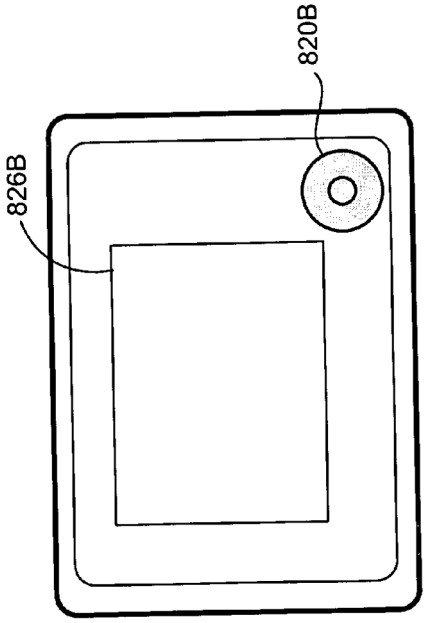


FIG. 31B

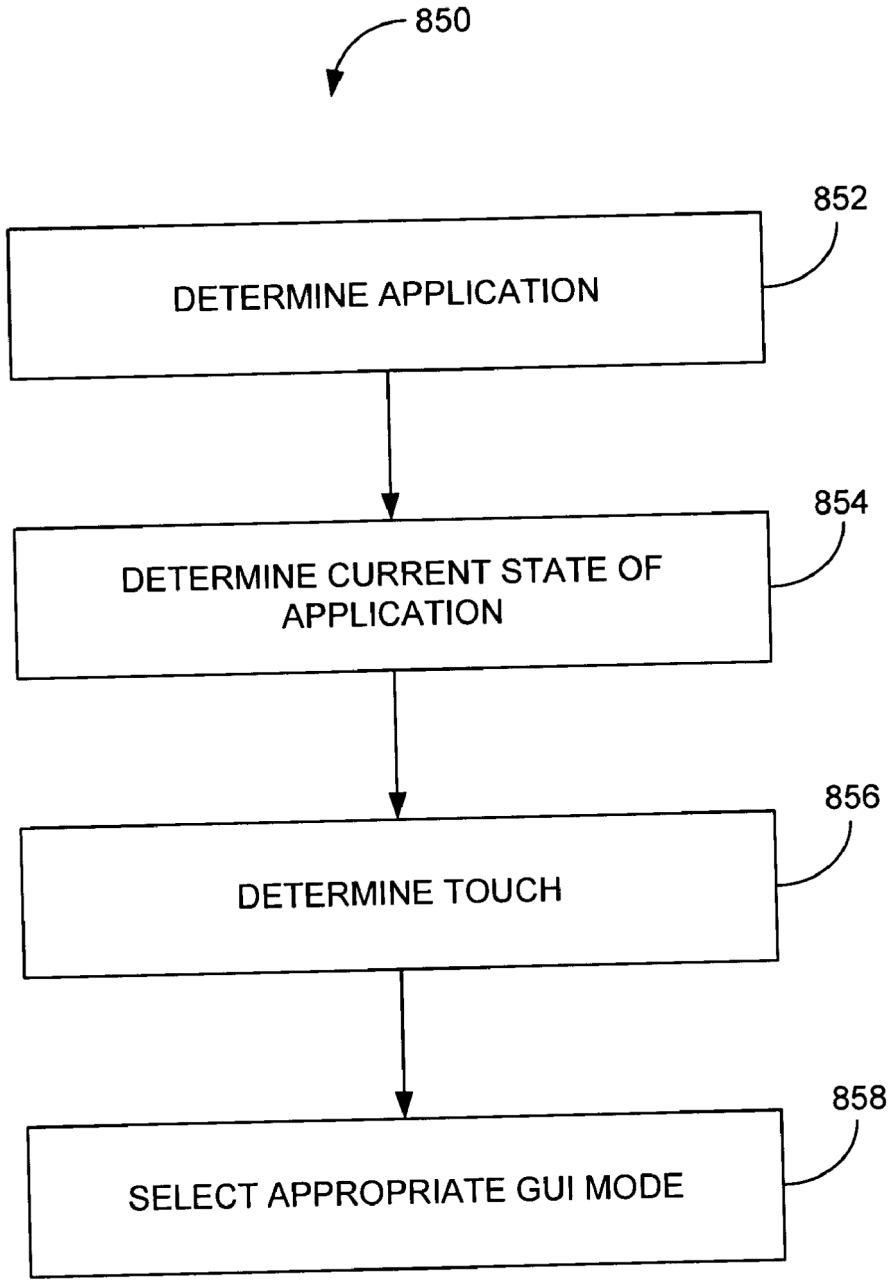

850
852
DETERMINE APPLICATION
854
DETERMINE CURRENT STATE OF APPLICATION
856
DETERMINE TOUCH
858
SELECT APPROPRIATE GUI MODE

FIG. 32

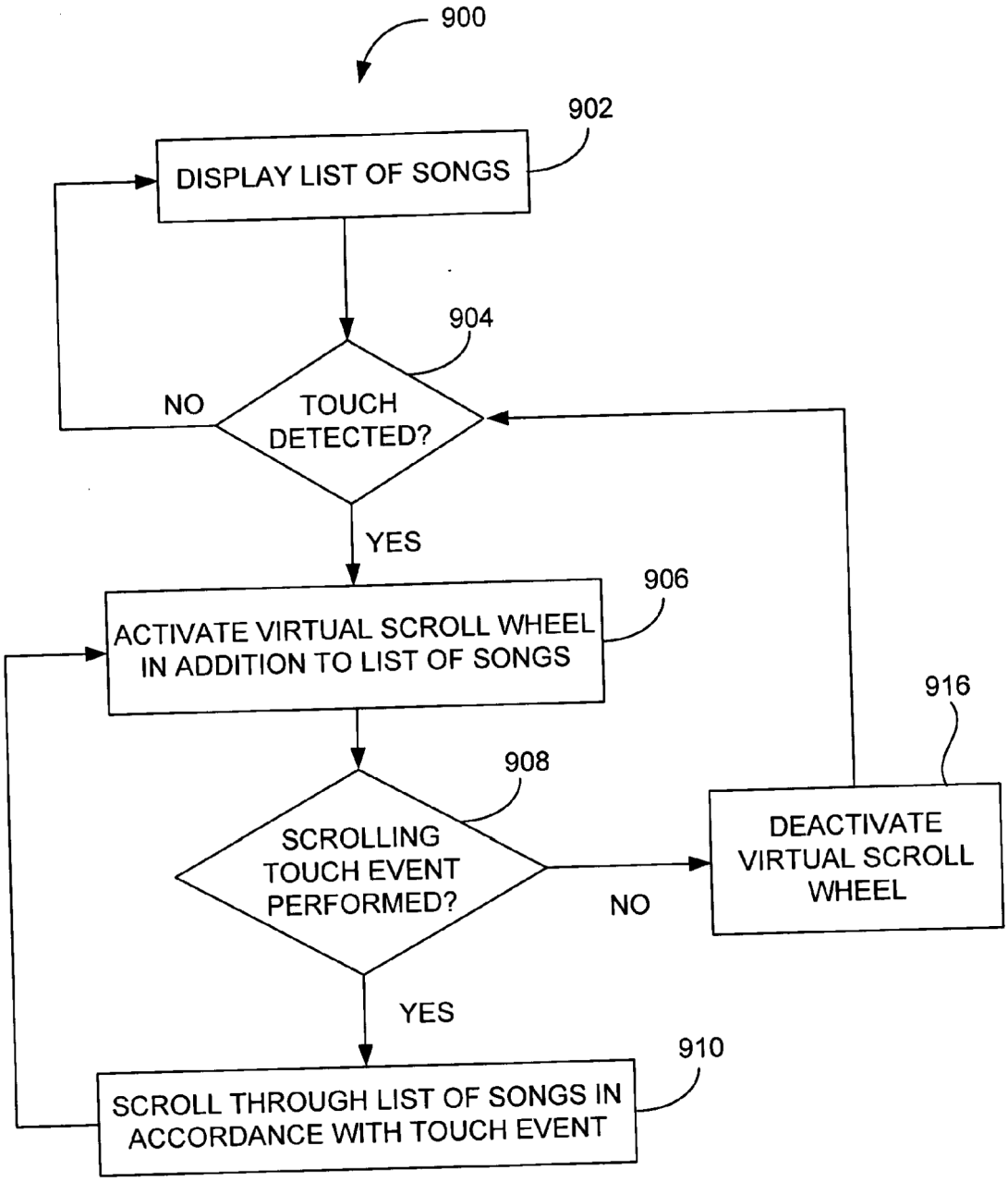

900
902
DISPLAY LIST OF SONGS
904
NO
TOUCH DETECTED?
YES
906
ACTIVATE VIRTUAL SCROLL WHEEL IN ADDITION TO LIST OF SONGS
916
908
SCROLLING TOUCH EVENT PERFORMED?
NO
DEACTIVATE VIRTUAL SCROLL WHEEL
YES
910
SCROLL THROUGH LIST OF SONGS IN ACCORDANCE WITH TOUCH EVENT

FIG. 33

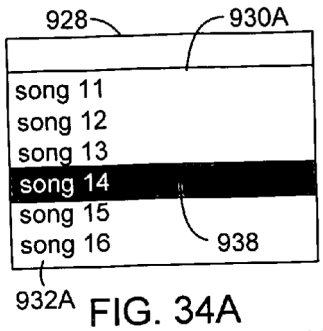


FIG. 34A

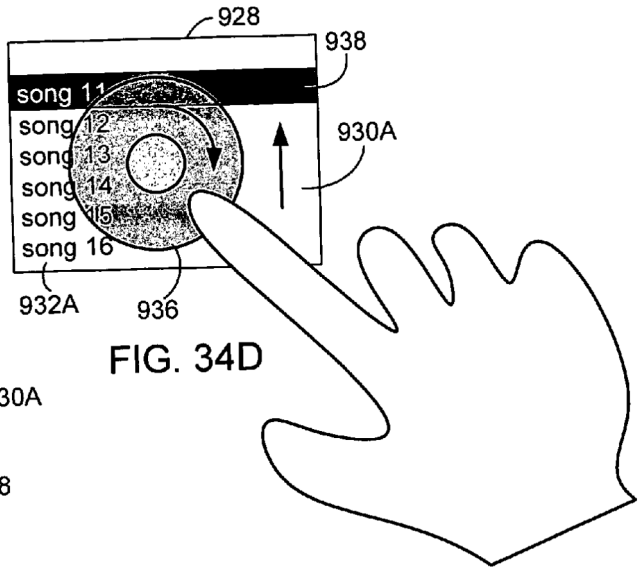


FIG. 34D

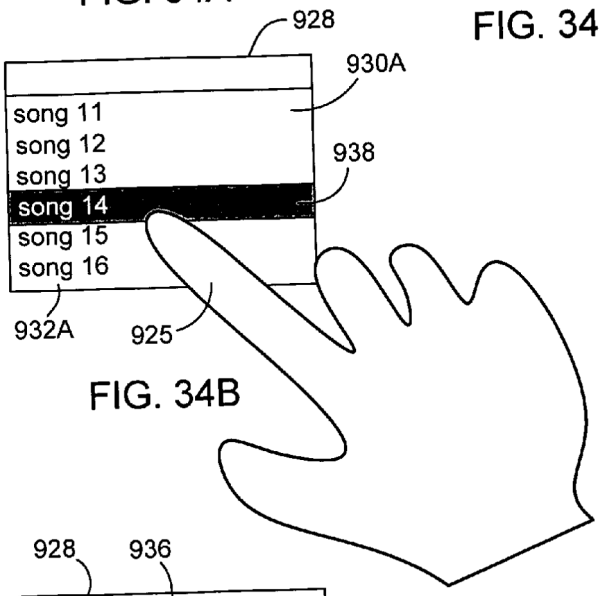


FIG. 34B

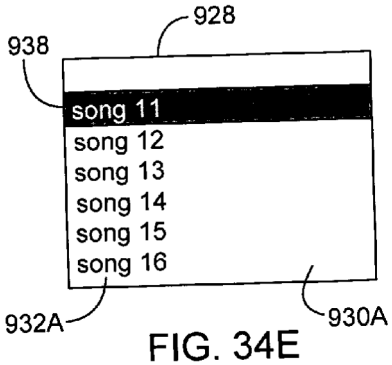


FIG. 34E

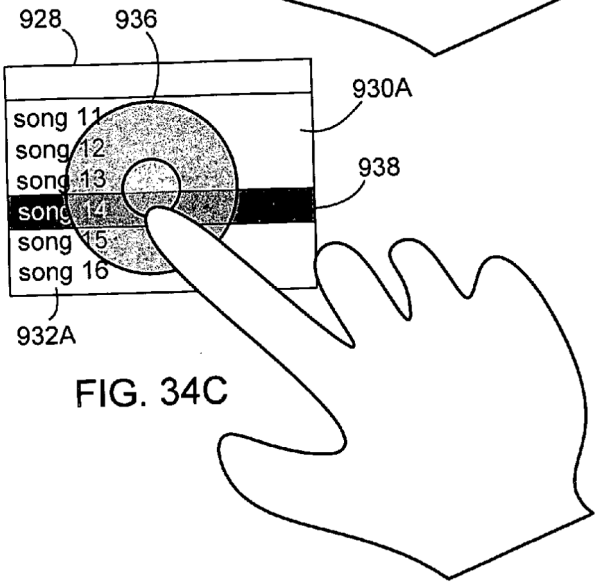


FIG. 34C

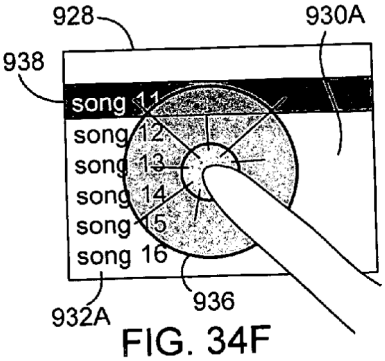


FIG. 34F

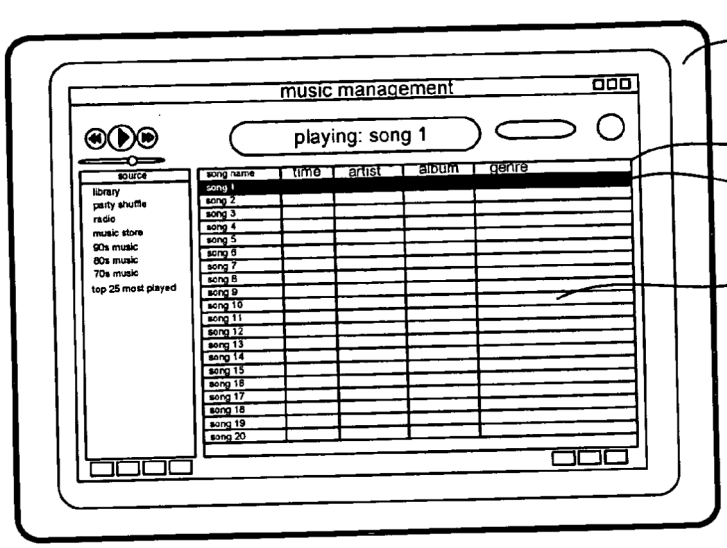


FIG. 35A

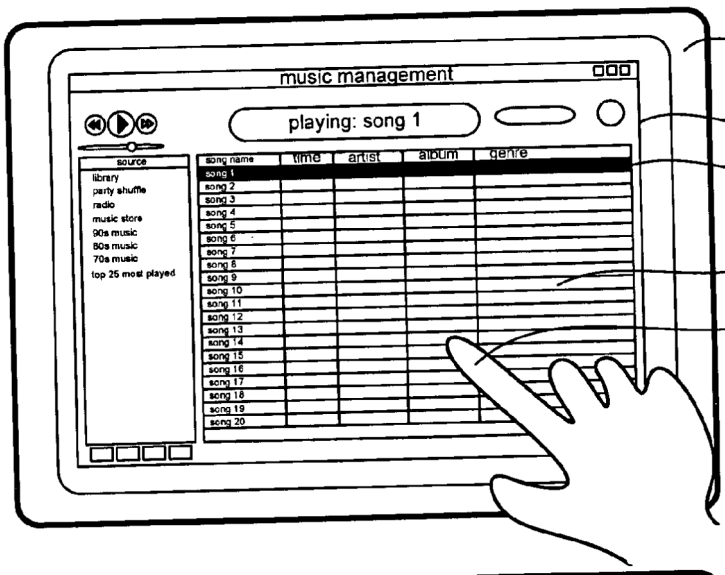


FIG. 35B

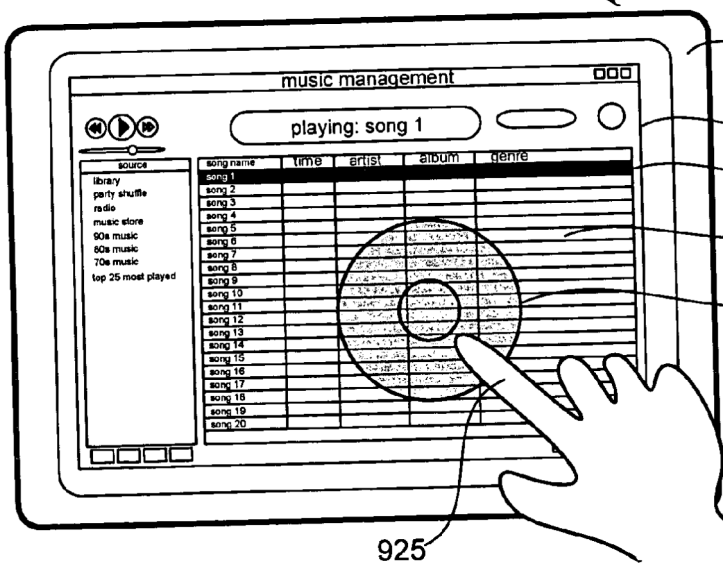


FIG. 35C

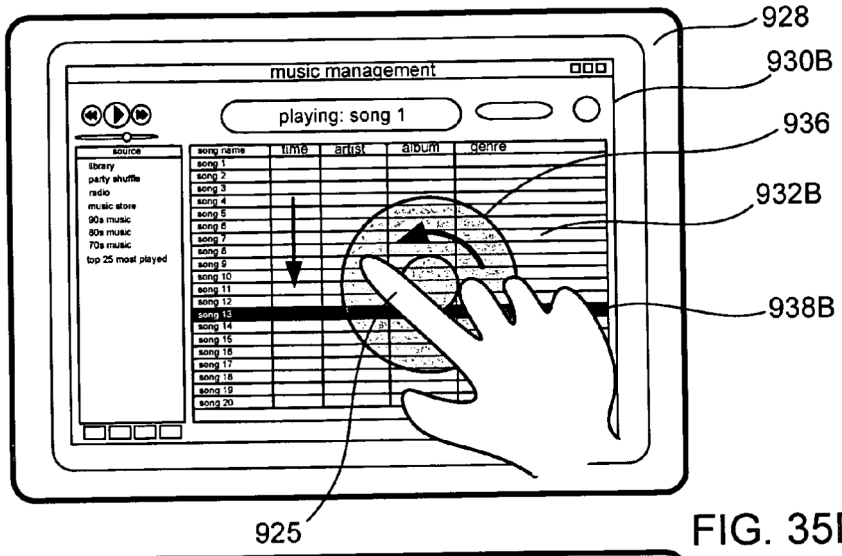


FIG. 35D

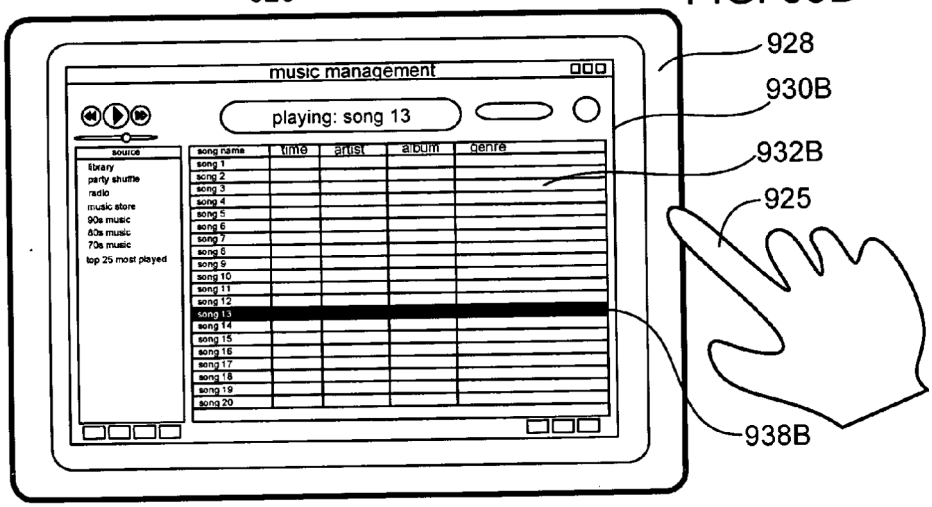


FIG. 35E

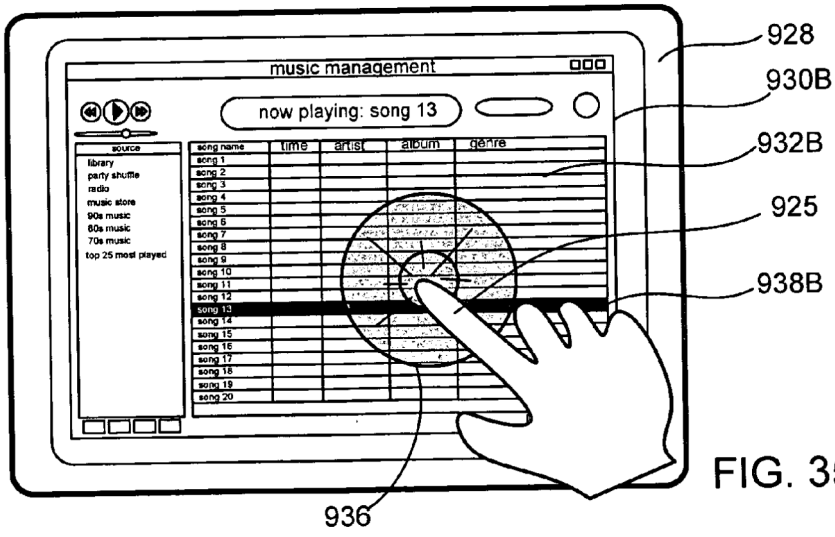


FIG. 35F

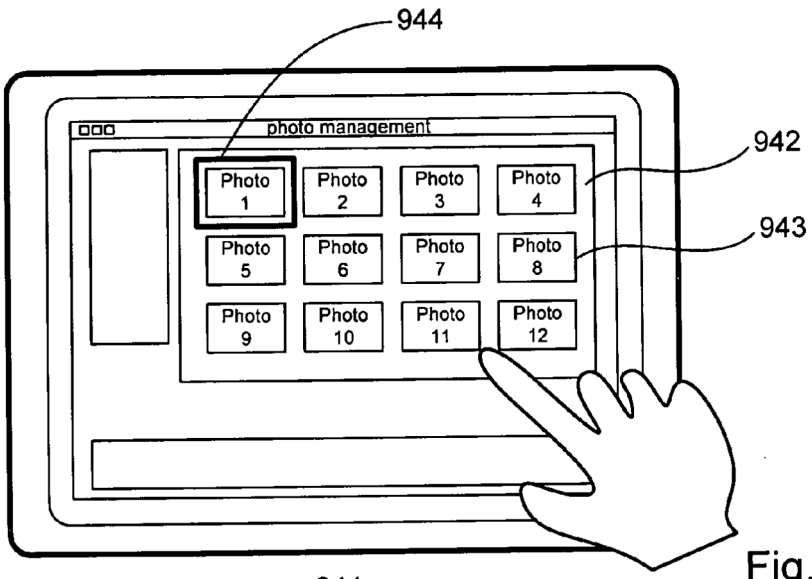


Fig. 36A

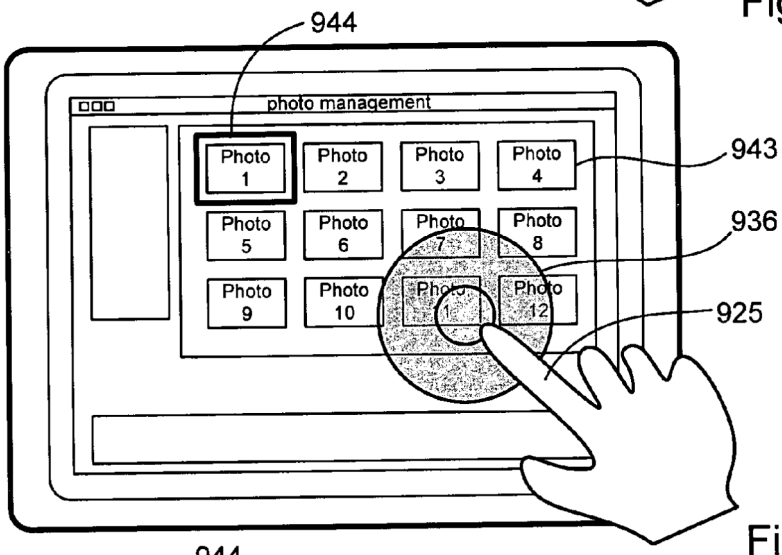


Fig. 36B

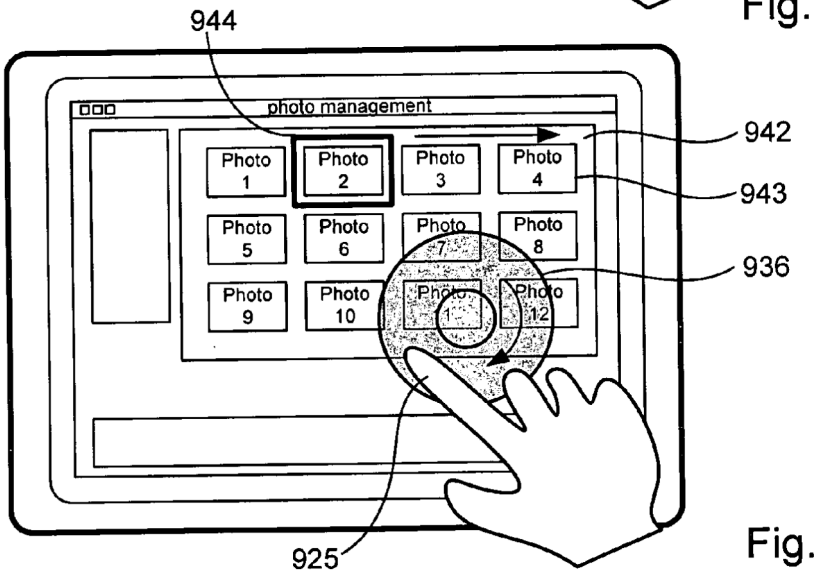


Fig. 36C

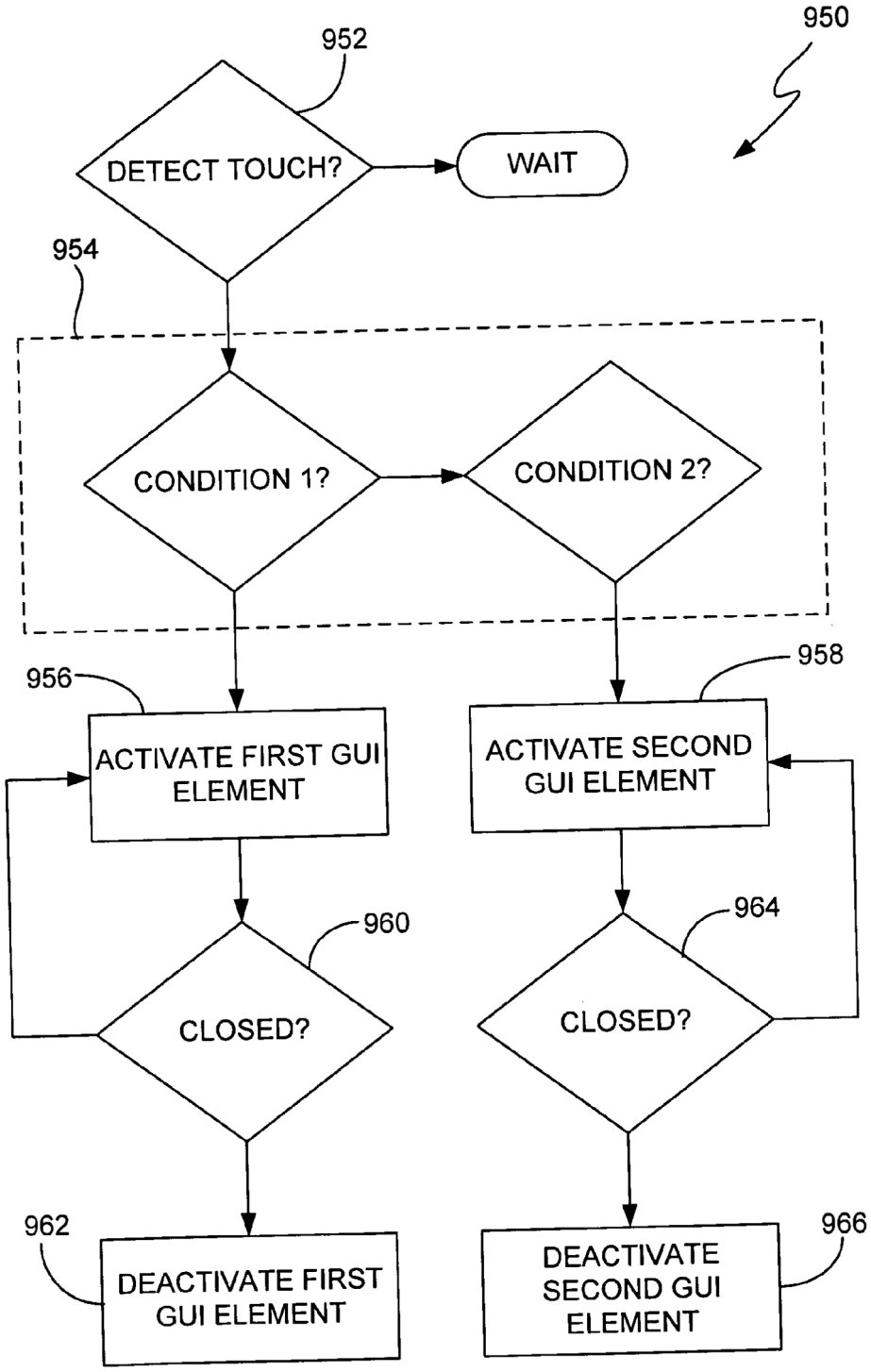

950
952
DETECT TOUCH?
WAIT
954
CONDITION 1?
CONDITION 2?
956
ACTIVATE FIRST GUI ELEMENT
958
ACTIVATE SECOND GUI ELEMENT
960
CLOSED?
964
CLOSED?
962
DEACTIVATE FIRST GUI ELEMENT
966
DEACTIVATE SECOND GUI ELEMENT

FIG. 37

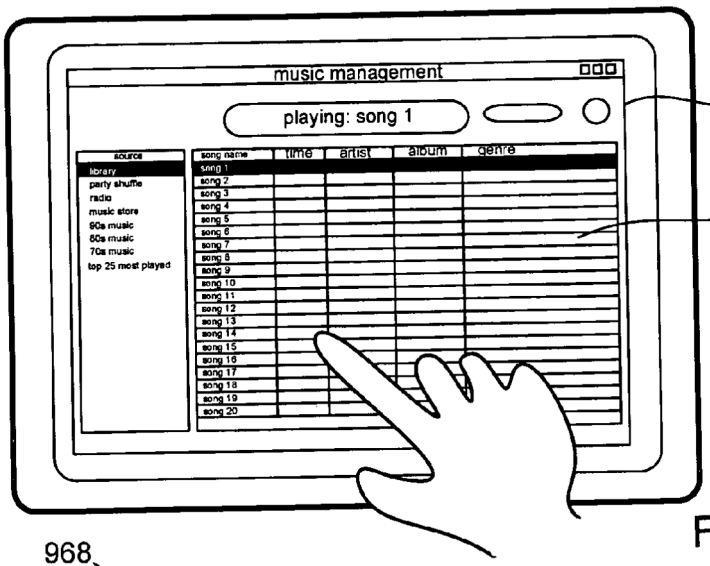


FIG. 38A

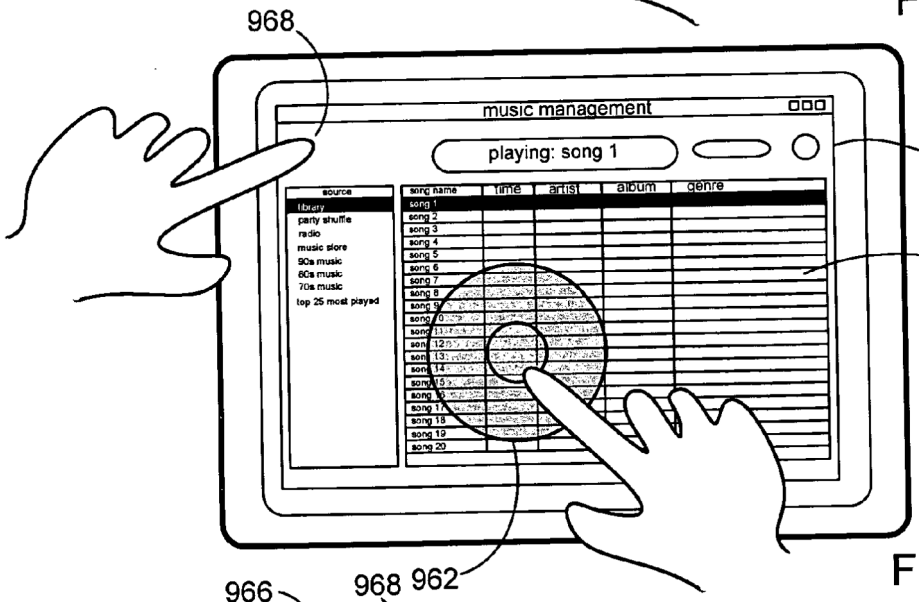


FIG. 38B

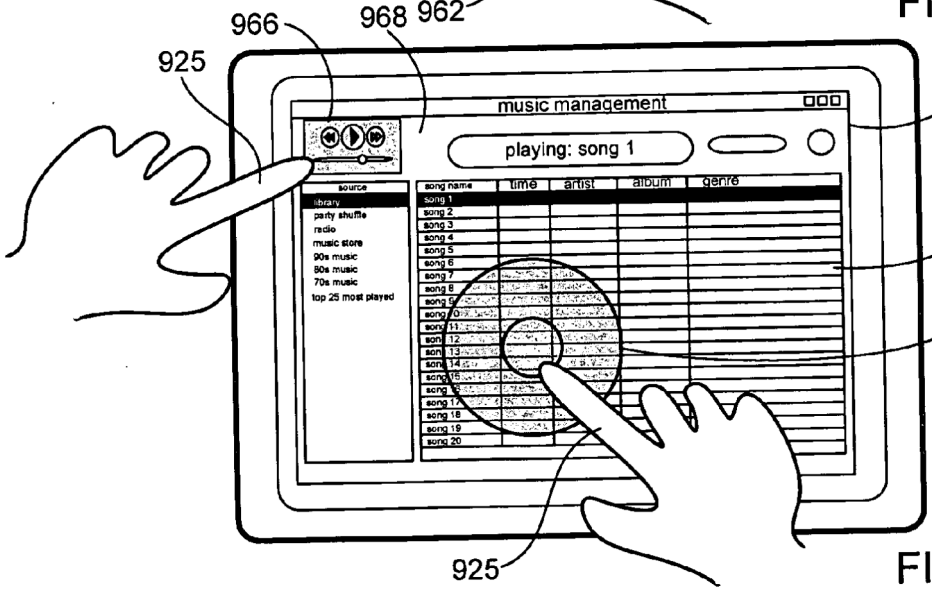


FIG. 38C

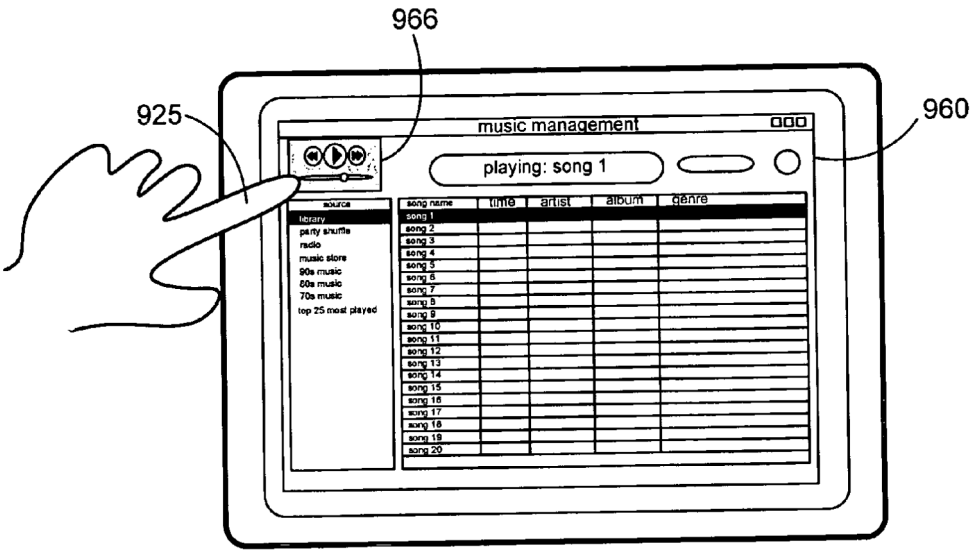


FIG. 38D

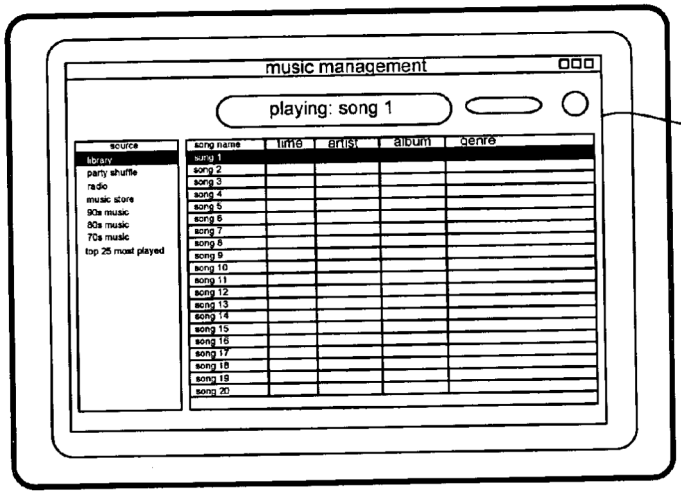


FIG. 38E

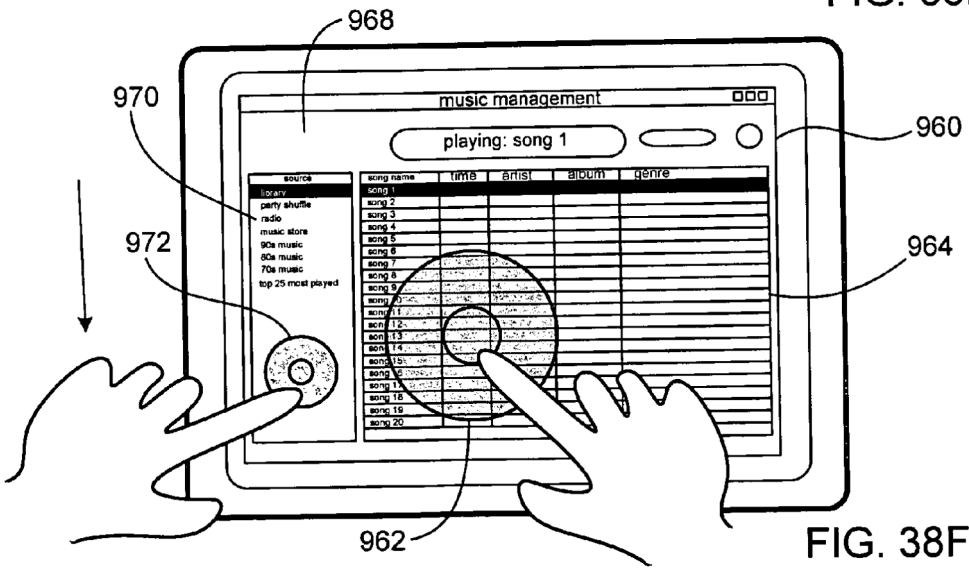


FIG. 38F

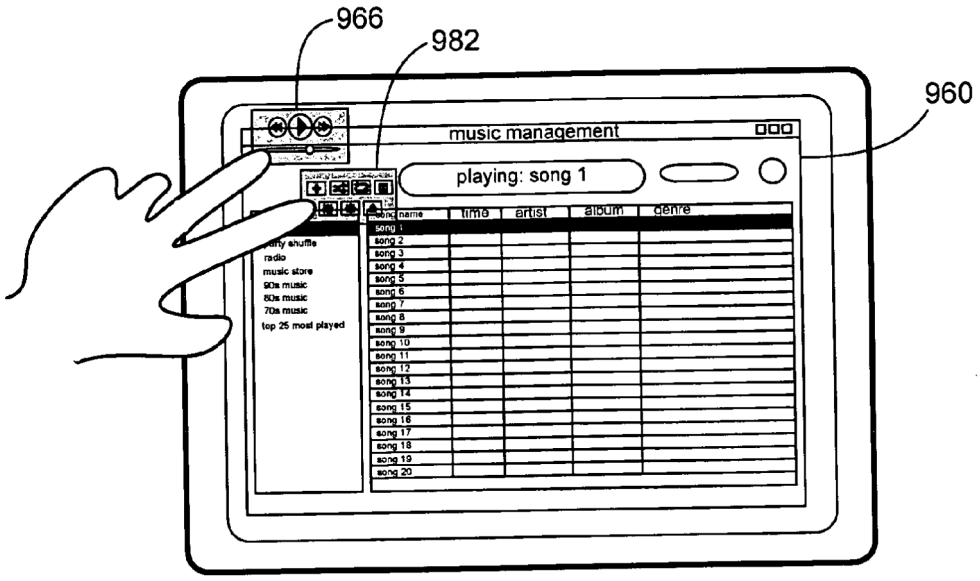


FIG. 38G

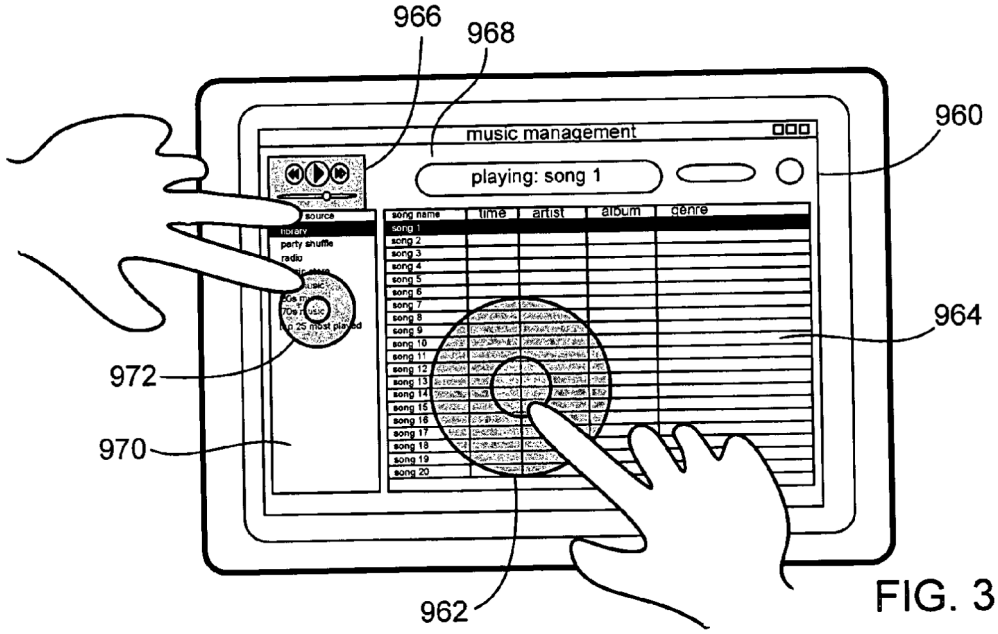


FIG. 38H

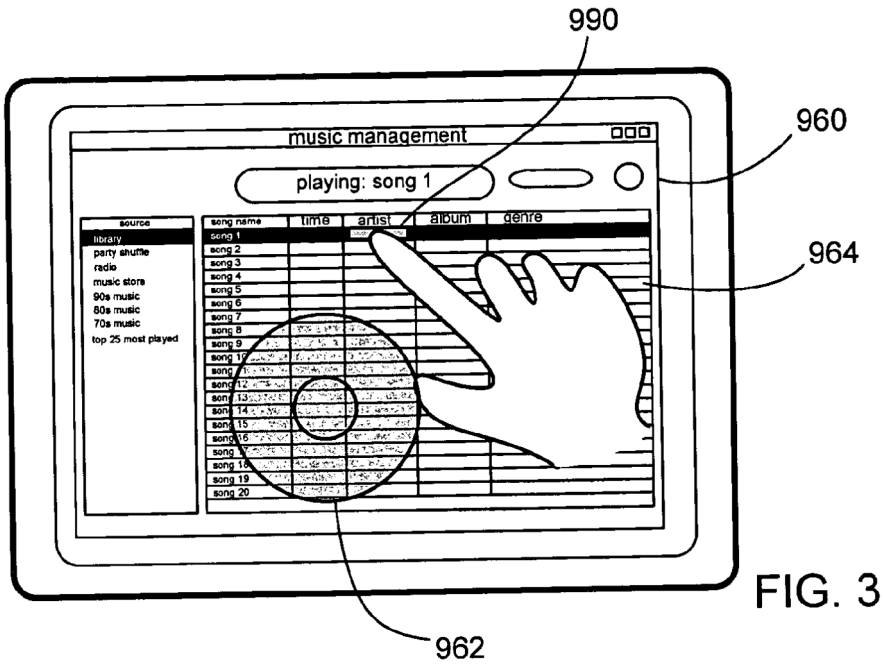


FIG. 38I

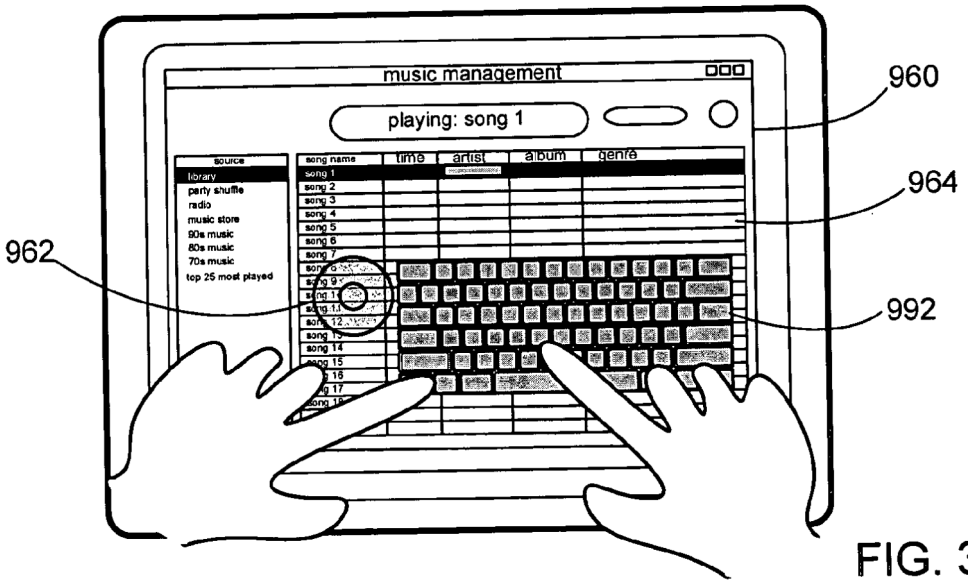


FIG. 38J

MODE-BASED GRAPHICAL USER INTERFACES FOR TOUCH SENSITIVE INPUT DEVICES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is a Continuation-in-Part of U.S. patent application Ser. No. 10/903,964, filed on Jul. 30, 2004, entitled "GESTURES FOR TOUCH SENSITIVE INPUT DEVICES," and which is herein incorporated by reference.

[0002] This application is also related to: (i) U.S. patent application Ser. No. 10/256,716 filed, on Sep. 26, 2002 and entitled "METHOD AND APPARATUS FOR ACCELERATED SCROLLING"; (ii) U.S. patent application Ser. No. 10/259,159, filed on Sep. 26, 2002 and entitled "METHOD AND APPARATUS FOR USE OF ROTATIONAL USER INPUT"; (iii) U.S. patent application Ser. No. 10/072,765, filed on Feb. 7, 2002 and entitled "MOUSE HAVING A ROTARY DIAL"; and (iv) U.S. patent application Ser. No. 10/840,862, filed on May 6, 2004 and entitled "MULTIPOINT TOUCH SCREEN". Each of these related application are hereby incorporated herein by reference

BACKGROUND OF THE INVENTION

[0003] 1. Field of the Invention

[0004] The present invention relates generally to gesturing associated with touch sensitive devices.

[0005] 2. Description of the Related Art

[0006] There exist today many styles of input devices for performing operations in a computer system. The operations generally correspond to moving a cursor and making selections on a display screen. The operations may also include paging, scrolling, panning, zooming, etc. By way of example, the input devices may include buttons, switches, keyboards, mice, trackballs, touch pads, joy sticks, touch screens and the like. Each of these devices has advantages and disadvantages that are taken into account when designing the computer system.

[0007] Buttons and switches are generally mechanical in nature and provide limited control with regards to the movement of the cursor and making selections. For example, they are generally dedicated to moving the cursor in a specific direction (e.g., arrow keys) or to making specific selections (e.g., enter, delete, number, etc.).

[0008] In mice, the movement of the input pointer corresponds to the relative movements of the mouse as the user moves the mouse along a surface. In trackballs, the movement of the input pointer corresponds to the relative movements of a ball as the user moves the ball within a housing. Mice and trackballs also include one or more buttons for making selections. Mice may also include scroll wheels that allow a user to move through the GUI by simply rolling the wheel forward or backward.

[0009] With touch pads, the movement of the input pointer corresponds to the relative movements of the user's finger (or stylus) as the finger is moved along a surface of the touch pad. Touch screens, on the other hand, are a type of display screen that has a touch-sensitive transparent panel covering the screen. When using a touch screen, a user makes a selection on the display screen by pointing directly to GUI objects on the screen (usually with a stylus or finger).

[0010] In order to provide additionally functionality, gestures have been implemented with some of these input devices. By way of example, in touch pads, selections may be made when one or more taps are detected on the surface of the touch pad. In some cases, any portion of the touch pad may be tapped, and in other cases a dedicated portion of the touch pad may be tapped. In addition to selections, scrolling may be initiated by using finger motion at the edge of the touch pad.

[0011] U.S. Pat. Nos. 5,612,719 and 5,590,219, assigned to Apple Computer, Inc. describe some other uses of gesturing. U.S. Pat. No. 5,612,719 discloses an onscreen button that is responsive to at least two different button gestures made on the screen on or near the button. U.S. Pat. No. 5,590,219 discloses a method for recognizing an ellipse-type gesture input on a display screen of a computer system.

[0012] In recent times, more advanced gestures have been implemented. For example, scrolling may be initiated by placing four fingers on the touch pad so that the scrolling gesture is recognized and thereafter moving these fingers on the touch pad to perform scrolling events. The methods for implementing these advanced gestures, however, has several drawbacks. By way of example, once the gesture is set, it cannot be changed until the user resets the gesture state. In touch pads, for example, if four fingers equals scrolling, and the user puts a thumb down after the four fingers are recognized, any action associated with the new gesture including four fingers and the thumb will not be performed until the entire hand is lifted off the touch pad and put back down again (e.g., reset). Simply put, the user cannot change gesture states midstream. Along a similar vein, only one gesture may be performed at any given time. That is, multiple gestures cannot be performed simultaneously.

[0013] Based on the above, there is a need for improvements in the way gestures are performed on touch sensitive devices.

SUMMARY OF THE INVENTION

[0014] The invention pertains to gestures and methods of implementing gestures with touch sensitive devices. Examples of touch sensitive devices include touch screens and touch pads. The invention also pertains to user interfaces and methods of implementing user interfaces with displays.

[0015] The invention can be implemented in numerous ways, including as a method, graphical user interface, computing device, or a computer readable medium. Several embodiments of the invention are discussed below.

[0016] The invention relates, in one embodiment, to a user interface method. The method includes detecting a touch. The method also includes determining a user interface mode when a touch is detected. The method further includes activating one or more GUI elements based on the user interface mode and in response to the detected touch.

[0017] The invention relates, in one embodiment, to a user interface method. The method includes displaying a group of media items. The method also includes detecting a touch over the group of media items that are displayed. The method further includes activating a virtual scroll wheel

when a touch is detected over the displayed media items. The activation of the virtual scroll wheel includes displaying and enabling the functionality of the virtual scroll wheel, the virtual scroll wheel providing a touch region where a user swirls their finger in order to traverse through the group of media items. The method additionally includes determining if a touch event is performed relative to the touch region of the virtual scroll wheel. Moreover, the method includes scrolling through the group of media items when a scrolling touch event is performed.

[0018] The invention relates, in one embodiment, to a method performed on a user operated electronic device having a display and a touch screen. The method includes determining if a touch is detected. The method also includes monitoring and analyzing the current operating conditions when a touch is detected. The method also includes activating a first GUI element for a first set of operating conditions. The method additionally includes activating a second GUI element for a second set of operating conditions.

[0019] The invention relates, in one embodiment, to a method performed on a computing device having a display and a touch sensing input device. The method includes sensing touches. The method also includes displaying and enabling a GUI element when a touch is detected. The GUI element is based on at least one of the following: (a) the application currently running on the computing device; (b) the current state of the application; and/or (c) one or more characteristics of the touch. The characteristics include, for example, touch location, touch ID, number of touches, and touch motion. The method further includes disabling and removing the GUI element from display when one of the following events occurs: (a) the touch is no longer detected, (b) a touch has not been detected for a preset amount of time, (c) a certain amount of time has gone by since the step of displaying and enabling, (d) a user selection.

[0020] The invention relates, in another embodiment, to a computing system. The computing system includes a display device configured to display a graphical user interface. The system also includes a touch screen positioned over the display device. The touch screen is configured to detect touches that occur over the display device. The system further includes a processor operatively coupled to the display device and the touch screen. The processor instructs the display device to display one or more GUI elements in response to a touch, and performs actions associated with the GUI element when touch events are detected relative to the displayed GUI elements.

[0021] The invention relates, in another embodiment, to a computing device. The computing device includes a processor. The computing device also includes a touch screen capable of sensing touch events. The computing device further includes a display configured to simultaneously display a plurality of media items and a virtual scroll wheel. The virtual scroll wheel provides a region where touch events are performed in order to implement a scrolling action. The scrolling action allows a user to traverse through the plurality of media items.

[0022] Other aspects and advantages of the invention will become apparent from the following detailed description, taken in conjunction with the accompanying drawings, illustrating by way of example the principles of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

[0023] The invention will be readily understood by the following detailed description in conjunction with the accompanying drawings, wherein like reference numerals designate like structural elements, and in which:

[0024] **FIG. 1** is a block diagram of a computer system, in accordance with one embodiment of the present invention.

[0025] **FIG. 2** is a multipoint processing method, in accordance with one embodiment of the present invention.

[0026] **FIGS. 3A** and B illustrate an image, in accordance with one embodiment of the present invention.

[0027] **FIG. 4** illustrates a group of features, in accordance with one embodiment of the present invention.

[0028] **FIG. 5** is a parameter calculation method, in accordance with one embodiment of the present invention.

[0029] **FIGS. 6A-6G** illustrate a rotate gesture, in accordance with one embodiment of the present invention.

[0030] **FIG. 7** is a diagram of a touch-based method, in accordance with one embodiment of the present invention.

[0031] **FIG. 8** is a diagram of a touch-based method, in accordance with one embodiment of the present invention.

[0032] **FIG. 9** is a diagram of a touch-based method, in accordance with one embodiment of the present invention.

[0033] **FIG. 10** is a diagram of a zoom gesture method, in accordance with one embodiment of the present invention.

[0034] **FIGS. 11A-11H** illustrates a zooming sequence, in accordance with one embodiment of the present invention.

[0035] **FIG. 12** is a diagram of a pan method, in accordance with one embodiment of the present invention.

[0036] **FIGS. 13A-13D** illustrate a panning sequence, in accordance with one embodiment of the present invention.

[0037] **FIG. 14** is a diagram of a rotate method, in accordance with one embodiment of the present invention.

[0038] **FIGS. 15A-15C** illustrate a rotating sequence, in accordance with one embodiment of the present invention.

[0039] **FIG. 16** is a diagram of a GUI operational method, in accordance with one embodiment of the present invention.

[0040] **FIGS. 17A-17E** illustrate a floating control sequence, in accordance with one embodiment of the present invention.

[0041] **FIG. 18** is a diagram of a GUI operational method, in accordance with one embodiment of the present invention.

[0042] **FIGS. 19A-19D** illustrate a zooming target sequence, in accordance with one embodiment of the present invention.

[0043] **FIG. 20** is a diagram of a GUI operational method, in accordance with one embodiment of the present invention.

[0044] **FIGS. 21A-21D** illustrate a page turning sequence, in accordance with one embodiment of the present invention.

[0045] **FIG. 22** is a diagram of a GUI operational method, in accordance with one embodiment of the present invention.

[0046] **FIGS. 23A-23D** illustrate an inertia sequence, in accordance with one embodiment of the present invention.

[0047] **FIG. 24** is a diagram of a GUI operational method, in accordance with one embodiment of the present invention.

[0048] **FIGS. 25A-25D** illustrates a keyboard sequence, in accordance with one embodiment of the present invention.

[0049] **FIG. 26** is a diagram of a GUI operational method, in accordance with one embodiment of the present invention.

[0050] **FIGS. 27A-27D** illustrates a scroll wheel sequence, in accordance with one embodiment of the present invention.

[0051] **FIG. 28** is user interface method, in accordance with one embodiment of the present invention.

[0052] **FIGS. 29A-29D** illustrate a transition effect, in accordance with one embodiment of the present invention.

[0053] **FIGS. 30A-30D** illustrate a transition effect, in accordance with another embodiment of the present invention.

[0054] **FIGS. 31A-31D** illustrate a transition effect, in accordance with another embodiment of the present invention.

[0055] **FIG. 32** is a determination method, in accordance with one embodiment of the present invention.

[0056] **FIG. 33** is a user interface method, in accordance with one embodiment of the present invention.

[0057] **FIG. 34A-34F** illustrates a sequence associated with the method shown in **FIG. 33**, in accordance with one embodiment of the present invention.

[0058] **FIG. 35A-35F** illustrates a sequence associated with the method shown in **FIG. 33**, in accordance with one embodiment of the present invention.

[0059] **FIGS. 36A-36C** illustrate a user interface sequence, in accordance with one embodiment of the present invention.

[0060] **FIG. 37** is a user interface method, in accordance with one embodiment of the present invention **FIGS. 38A-38J** illustrate a user interface sequence, in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0061] The invention pertains to gestures and methods of implementing gestures with touch sensitive devices. Examples of touch sensitive devices include touch screens and touch pads. The invention also pertains to user interfaces and methods of implementing user interfaces with displays. One aspect of the invention relates to determining a user interface mode based on one or more conditions. Another aspect of the invention relates to activating one or more GUI elements based on the user interface mode (e.g., displaying a GUI element). Another aspect of the invention relates to a virtual scroll wheel.

[0062] These and other aspects of the invention are discussed below with reference to **FIGS. 1-38J**. However, those skilled in the art will readily appreciate that the detailed description given herein with respect to these figures is for explanatory purposes as the invention extends beyond these limited embodiments.

[0063] **FIG. 1** is a block diagram of an exemplary computer system **50**, in accordance with one embodiment of the present invention. The computer system **50** may correspond to a personal computer system, such as a desktops, laptops, tablets or handheld computer. The computer system may also correspond to a computing device, such as a cell phone, PDA, dedicated media player, consumer electronic device, and the like.

[0064] The exemplary computer system **50** shown in **FIG. 1** includes a processor **56** configured to execute instructions and to carry out operations associated with the computer system **50**. For example, using instructions retrieved for example from memory, the processor **56** may control the reception and manipulation of input and output data between components of the computing system **50**. The processor **56** can be implemented on a single-chip, multiple chips or multiple electrical components. For example, various architectures can be used for the processor **56**, including dedicated or embedded processor, single purpose processor, controller, ASIC, and so forth.

[0065] In most cases, the processor **56** together with an operating system operates to execute computer code and produce and use data. Operating systems are generally well known and will not be described in greater detail. By way of example, the operating system may correspond to OS/2, DOS, Unix, Linux, Palm OS, and the like. The operating system can also be a special purpose operating system, such as may be used for limited purpose appliance-type computing devices. The operating system, other computer code and data may reside within a memory block **58** that is operatively coupled to the processor **56**. Memory block **58** generally provides a place to store computer code and data that are used by the computer system **50**. By way of example, the memory block **58** may include Read-Only Memory (ROM), Random-Access Memory (RAM), hard disk drive and/or the like. The information could also reside on a removable storage medium and loaded or installed onto the computer system **50** when needed. Removable storage mediums include, for example, CD-ROM, PC-CARD, memory card, floppy disk, magnetic tape, and a network component.

[0066] The computer system **50** also includes a display device **68** that is operatively coupled to the processor **56**. The display device **68** may be a liquid crystal display (LCD) (e.g., active matrix, passive matrix and the like). Alternatively, the display device **68** may be a monitor such as a monochrome display, color graphics adapter (CGA) display, enhanced graphics adapter (EGA) display, variable-graphics-array (VGA) display, super VGA display, cathode ray tube (CRT), and the like. The display device may also correspond to a plasma display or a display implemented with electronic inks.

[0067] The display device **68** is generally configured to display a graphical user interface (GUI) **69** that provides an

easy to use interface between a user of the computer system and the operating system or application running thereon. Generally speaking, the GUI **69** represents, programs, files and operational options with graphical images. The graphical images may include windows, fields, dialog boxes, menus, icons, buttons, cursors, scroll bars, etc. Such images may be arranged in predefined layouts, or may be created dynamically to serve the specific actions being taken by a user. During operation, the user can select and activate various graphical images in order to initiate functions and tasks associated therewith. By way of example, a user may select a button that opens, closes, minimizes, or maximizes a window, or an icon that launches a particular program. The GUI **69** can additionally or alternatively display information, such as non interactive text and graphics, for the user on the display device **68**.

[**0068**] The computer system **50** also includes an input device **70** that is operatively coupled to the processor **56**. The input device **70** is configured to transfer data from the outside world into the computer system **50**. The input device **70** may for example be used to perform tracking and to make selections with respect to the GUI **69** on the display **68**. The input device **70** may also be used to issue commands in the computer system **50**. The input device **70** may include a touch sensing device configured to receive input from a user's touch and to send this information to the processor **56**. By way of example, the touch-sensing device may correspond to a touchpad or a touch screen. In many cases, the touch-sensing device recognizes touches, as well as the position and magnitude of touches on a touch sensitive surface. The touch sensing means reports the touches to the processor **56** and the processor **56** interprets the touches in accordance with its programming. For example, the processor **56** may initiate a task in accordance with a particular touch. A dedicated processor can be used to process touches locally and reduce demand for the main processor of the computer system. The touch sensing device may be based on sensing technologies including but not limited to capacitive sensing, resistive sensing, surface acoustic wave sensing, pressure sensing, optical sensing, and/or the like. Furthermore, the touch sensing means may be based on single point sensing or multipoint sensing. Single point sensing is capable of only distinguishing a single touch, while multipoint sensing is capable of distinguishing multiple touches that occur at the same time.

[**0069**] The input device **70** may be a touch screen that is positioned over or in front of the display **68**. The touch screen **70** may be integrated with the display device **68** or it may be a separate component. The touch screen **70** has several advantages over other input technologies such as touchpads, mice, etc. For one, the touch screen **70** is positioned in front of the display **68** and therefore the user can manipulate the GUI **69** directly. For example, the user can simply place their finger over an object to be controlled. In touch pads, there is no one-to-one relationship such as this. With touchpads, the touchpad is placed away from the display typically in a different plane. For example, the display is typically located in a vertical plane and the touchpad is typically located in a horizontal plane. This makes its use less intuitive, and therefore more difficult when compared to touch screens. In addition to being a touch screen, the input device **70** can be a multipoint input device. Multipoint input devices have advantages over conventional singlepoint devices in that they can distinguish more than one object (finger). Singlepoint devices are simply incapable of distinguishing multiple objects. By way of example, a multipoint touch screen, which can be used herein, is shown and described in greater detail in copending and commonly assigned U.S. patent application Ser. No. 10/840,862, which is hereby incorporated herein by reference.

[**0070**] The computer system **50** also includes capabilities for coupling to one or more I/O devices **80**. By way of example, the I/O devices **80** may correspond to keyboards, printers, scanners, cameras, speakers, and/or the like. The I/O devices **80** may be integrated with the computer system **50** or they may be separate components (e.g., peripheral devices). In some cases, the I/O devices **80** may be connected to the computer system **50** through wired connections (e.g., cables/ports). In other cases, the I/O devices **80** may be connected to the computer system **80** through wireless connections. By way of example, the data link may correspond to PS/2, USB, IR, RF, Bluetooth or the like.

[**0071**] In accordance with one embodiment of the present invention, the computer system **50** is designed to recognize gestures **85** applied to the input device **70** and to control aspects of the computer system **50** based on the gestures **85**. In some cases, a gesture is defined as a stylized interaction with an input device that is mapped to one or more specific computing operations. The gestures **85** may be made through various hand, and more particularly finger motions. Alternatively or additionally, the gestures may be made with a stylus. In all of these cases, the input device **70** receives the gestures **85** and the processor **56** executes instructions to carry out operations associated with the gestures **85**. In addition, the memory block **58** may include a gesture operational program **88**, which may be part of the operating system or a separate application. The gestural operation program **88** generally includes a set of instructions that recognizes the occurrence of gestures **85** and informs one or more software agents of the gestures **85** and/or what action(s) to take in response to the gestures **85**.

[**0072**] When a user performs one or more gestures, the input device **70** relays gesture information to the processor **56**. Using instructions from memory **58**, and more particularly, the gestural operational program **88**, the processor **56** interprets the gestures **85** and controls different components of the computer system **50**, such as memory **58**, a display **68** and I/O devices **80**, based on the gestures **85**. The gestures **85** may be identified as commands for performing actions in applications stored in the memory **58**, modifying GUI objects shown on the display **68**, modifying data stored in memory **58**, and/or for performing actions in I/O devices **80**. By way of example, the commands may be associated with zooming, panning, scrolling, paging, rotating, sizing, and the like. As further examples, the commands may also be associated with launching a particular program, opening a file or document, viewing a menu, making a selection, executing instructions, logging onto the computer system, permitting authorized individuals access to restricted areas of the computer system, loading a user profile associated with a user's preferred arrangement of the computer desktop, and/or the like.

[**0073**] A wide range of different gestures can be utilized. By way of example, the gestures may be single point or multipoint gestures; static or dynamic gestures; continuous

or segmented gestures; and/or the like. Single point gestures are those gestures that are performed with a single contact point, e.g., the gesture is performed with a single touch as for example from a single finger, a palm or a stylus. Multipoint gestures are those gestures that can be performed with multiple points, e.g., the gesture is performed with multiple touches as for example from multiple fingers, fingers and palms, a finger and a stylus, multiple styli and/or any combination thereof. Static gestures are those gestures that do not include motion, and dynamic gestures are those gestures that do include motion. Continuous gestures are those gestures that are performed in a single stroke, and segmented gestures are those gestures that are performed in a sequence of distinct steps or strokes.

[**0074**] In one embodiment, the computer system **50** is configured to register multiple gestures at the same time, i.e., multiple gestures can be performed simultaneously. By way of example, a zoom gesture may be performed at the same time as a rotate gesture, or a rotate gesture may be performed at the same time as a pan gesture. In one particular implementation, zoom, rotate and pan gestures can all occur simultaneously in order to perform zooming, rotating and panning at the same time.

[**0075**] In another embodiment, the system is configured to immediately recognize the gestures so that actions associated with the gestures can be implemented at the same time as the gesture, i.e., the gesture and action simultaneously occur side by side rather than being a two-step process. By way of example, during a scrolling gesture, the screen moves with the finger motion.

[**0076**] In another embodiment, an object presented on a display **68** continuously follows the gesture occurring on a touch screen. There is a one to one relationship between the gesture being performed and the objects shown on the display **68**. For example, as the gesture is performed, modifications simultaneously occur to the objects located underneath the gesture. For example, during a zooming gesture, the fingers may spread apart or close together in order to cause the object shown on the display **68** to zoom in during the spreading and zoom out during the closing. During this operation, the computer system **50** recognizes the user input as a zoom gesture, determines what action should be taken, and outputs control data to the appropriate device, in this case the display **68**.

[**0077**] In another embodiment, the computer system **50** provides region sensitivity where gestures mean different things when implemented over different areas of the input device **68**. For example, a rotation gesture over a volume knob causes volume increase/decrease, whereas a rotation gesture over a photo causes rotation of the photo.

[**0078**] In another embodiment, the number of fingers in contact with the touch screen may indicate an input mode. For example, a single touch as for example by a single finger may indicate the desire to perform tracking, i.e., pointer or cursor movements, or selections, whereas multiple touches as for example by a group of fingers may indicate the desire to perform gesturing. The number of fingers for implementing gesturing may be widely varied. By way of example, two fingers may indicate a first gesture mode, three fingers may indicate a third gesture mode, etc. Alternatively, any number of fingers, i.e., more than one, may be used for the same gesture mode, which can include one ore more gesture controls. The orientation of the fingers may similarly be used to denote the desired mode. The profile of the finger may be detected to permit different modal operations based on whether the user has used his thumb or index finger, for example.

[**0079**] In another embodiment, an input can be changed while making a continuous stroke on the input device without stopping the stroke (e.g., lifting off the touch sensitive surface). In one implementation, the user can switch from a tracking (or selection) mode to gesturing mode while a stroke is being made. For example, tracking or selections may be associated with a single finger and gesturing may be associated with multiple fingers; therefore, the user can toggle between tracking/selection and gesturing by picking up and placing down a second finger on the touch screen. In another implementation, the user can switch from one gesture mode to another gesture mode while a stroke is being made. For example, zooming may be associated with spreading a pair of fingers and rotating may be associated with rotating the pair of fingers; therefore, the user can toggle between zooming and rotating by alternating the movement of their fingers between spreading and rotating. In yet another implementation, the number of gesture inputs can be changed while a stroke is being made (e.g., added or subtracted). For example, during zooming where the fingers are spread apart, the user may further rotate their fingers to initiate both zooming and rotation. Furthermore during zooming and rotation, the user can stop spreading their fingers so that only rotation occurs. In other words, the gesture inputs can be continuously input, either simultaneously or consecutively.

[**0080**] In one particular embodiment, a single finger initiates tracking (or selection) and two or more fingers in close proximity to one another initiates scrolling or panning. Two fingers is generally preferred so as to provide easy toggling between one and two fingers, i.e., the user can switch between modes very easily by simply picking or placing an additional finger. This has the advantage of being more intuitive than other forms of mode toggling. During tracking, cursor movement is controlled by the user moving a single finger on the touch sensitive surface of a touch sensing device. The sensor arrangement of the touch sensing device interprets the finger motion and generates signals for producing corresponding movement of the cursor on the display. During scrolling, screen movement is controlled by the user moving dual fingers on the touch sensitive surface of the touch sensing device. When the combined fingers are moved in the vertical direction, the motion is interpreted as a vertical scroll event, and when the combined fingers are moved in the horizontal direction, the motion is interpreted as a horizontal scroll event. The same can be said for panning although panning can occur in all directions rather than just the horizontal and vertical directions.

[**0081**] The term "scrolling" as used herein generally pertains to moving displayed data or images (e.g., text or graphics) across a viewing area on a display screen so that a new set of data (e.g., line of text or graphics) is brought into view in the viewing area. In most cases, once the viewing area is full, each new set of data appears at the edge of the viewing area and all other sets of data move over one position. That is, the new set of data appears for each set of data that moves out of the viewing area. In essence, the scrolling function allows a user to view consecutive sets of

data currently outside of the viewing area. The viewing area may be the entire viewing area of the display screen or it may only be a portion of the display screen (e.g., a window frame).

[**0082**] As mentioned above, scrolling may be implemented vertically (up or down) or horizontally (left or right). In the case of vertical scrolling, when a user scrolls down, each new set of data appears at the bottom of the viewing area and all other sets of data move up one position. If the viewing area is full, the top set of data moves out of the viewing area. Similarly, when a user scrolls up, each new set of data appears at the top of the viewing area and all other sets of data move down one position. If the viewing area is full, the bottom set of data moves out of the viewing area.

[**0083**] By way of example, the display screen, during operation, may display a list of media items (e.g., songs). A user is able to linearly scroll through the list of media items by moving his or her finger across a touch screen. As the finger moves across the touch screen, the displayed items from the list of media items are varied such that the user is able to effectively scroll through the list of media items. In most cases, the user is able to accelerate their traversal of the list of media items by moving his or her finger at greater speeds. Some embodiments, which may be related to the above example, are described in greater detail below. See for example **FIGS. 6, 23, 27**.

[**0084**] **FIG. 2** is a multipoint processing method **100**, in accordance with one embodiment of the present invention. The multipoint processing method **100** may for example be performed in the system shown in **FIG. 1**. The multipoint processing method **100** generally begins at block **102** where images are read from a multipoint input device, and more particularly a multipoint touch screen. By way of example, the multipoint touch screen may generally correspond to the multipoint touch screen disclosed in copending U.S. patent application Ser. No. 10/840,862, which is hereby incorporated herein by reference. Although the term "image" is used it should be noted that the data may come in other forms. In most cases, the image read from the touch screen provides magnitude (Z) as a function of position (x and y) for each sensing point or pixel of the touch screen. The magnitude may, for example, be reflect the capacitance measured at each point.

[**0085**] Following block **102**, multipoint processing method **100** proceeds to block **104** where the image is converted into a collection or list of features. Each feature represents a distinct input such as a touch. In most cases, each feature includes its own unique identifier (ID), x coordinate, y coordinate, Z magnitude, angle θ, area A, and the like. By way of example, **FIGS. 3A and 3B** illustrate a particular image **120** in time. In image **120**, there are two features **122** based on two distinct touches. The touches may for example be formed from a pair of fingers touching the touch screen. As shown, each feature **122** includes unique identifier (ID), x coordinate, y coordinate, Z magnitude, angle θ, and area A. More particularly, the first feature **122**A is represented by ID_1, x_1, y_1, Z_1, θ_1, A_1 and the second feature **122**B is represented by ID_2, x_2, y_2, Z_2, θ_2, A_2. This data may be outputted for example using a multitouch protocol.

[**0086**] The conversion from data or images to features may be accomplished using methods described in copending U.S. patent application Ser. No. 10/840,862 which is hereby incorporated herein by reference. As disclosed therein, the raw data is received. The raw data is typically in a digitized form, and includes values for each node of the touch screen. The values may be between 0 and 256 where 0 equates to no touch pressure and 256 equates to full touch pressure. Thereafter, the raw data is filtered to reduce noise. Once filtered, gradient data, which indicates the topology of each group of connected points, is generated. Thereafter, the boundaries for touch regions are calculated based on the gradient data, i.e., a determination is made as to which points are grouped together to form each touch region. By way of example, a watershed algorithm may be used. Once the boundaries are determined, the data for each of the touch regions are calculated (e.g., x, y, Z, θ, A).

[**0087**] Following block **104**, multipoint processing method **100** proceeds to block **106** where feature classification and groupings are performed. During classification, the identity of each of the features is determined. For example, the features may be classified as a particular finger, thumb, palm or other object. Once classified, the features may be grouped. The manner in which the groups are formed can widely varied. In most cases, the features are grouped based on some criteria (e.g., they carry a similar attribute). For example, the two features shown in **FIGS. 3A and 3B** may be grouped together because each of these features is located in proximity to each other or because they are from the same hand. The grouping may include some level of filtering to filter out features that are not part of the touch event. In filtering, one or more features may be rejected because they either meet some predefined criteria or because they do not meet some criteria. By way of example, one of the features may be classified as a thumb located at the edge of a tablet PC. Because the thumb is being used to hold the device rather than being used to perform a task, the feature generated therefrom is rejected, i.e., is not considered part of the touch event being processed.

[**0088**] Following block **106**, the multipoint processing method **100** proceeds to block **108** where key parameters for the feature groups are calculated. The key parameters may include distance between features, x/y centroid of all features, feature rotation, total pressure of the group (e.g., pressure at centroid), and the like. As shown in **FIG. 4**, the calculation may include finding the centroid C, drawing a virtual line **130** to each feature from the centroid C, defining the distance D for each virtual line (D_1 and D_2), and then averaging the distances D_1 and D_2. Once the parameters are calculated, the parameter values are reported. The parameter values are typically reported with a group identifier (GID) and number of features within each group (in this case three). In most cases, both initial and current parameter values are reported. The initial parameter values may be based on set down, i.e., when the user sets their fingers on the touch screen, and the current values may be based on any point within a stroke occurring after set down. As should be appreciated, blocks **102-108** are repetitively performed during a user stroke thereby generating a plurality of sequentially configured signals. The initial and current parameters can be compared in later steps to perform actions in the system.

[**0089**] Following block **108**, the process flow proceeds to block **110** where the group is or associated to a user interface (UI) element. UI elements are buttons boxes, lists, sliders,

wheels, knobs, etc. Each UI element represents a component or control of the user interface. The application behind the UI element(s) has access to the parameter data calculated in block **108**. In one implementation, the application ranks the relevance of the touch data to the UI element corresponding there to. The ranking may be based on some predetermine criteria. The ranking may include producing a figure of merit, and whichever UI element has the highest figure of merit, giving it sole access to the group. There may even be some degree of hysteresis as well (once one of the UI elements claims control of that group, the group sticks with the UI element until another UI element has a much higher ranking). By way of example, the ranking may include determining proximity of the centroid (or features) to the GUI object associated with the UI element.

[**0090**] Following block **110**, the multipoint processing method **100** proceeds to blocks **112** and **114**. The blocks **112** and **114** can be performed approximately at the same time. From the user perspective, in one embodiment, the blocks **112** and **114** appear to be performed concurrently. In block **112**, one or more actions are performed based on differences between initial and current parameter values as well as the UI element to which they are associated. In block **114**, user feedback pertaining to the one ore more action being performed is provided. By way of example, user feedback may include display, audio, tactile feedback and/or the like.

[**0091**] **FIG. 5** is a parameter calculation method **150**, in accordance with one embodiment of the present invention. The parameter calculation method **150** may, for example, correspond to block **108** shown in **FIG. 2**. The parameter calculation method **150** generally begins at block **152** where a group of features is received. Following block **152**, the parameter calculation method **150** proceeds to block **154** where a determination is made as to whether or not the number of features in the group of features has changed. For example, the number of features may have changed due to the user picking up or placing an additional finger. Different fingers may be needed to perform different controls (e.g., tracking, gesturing). If the number of features has changed, the parameter calculation method **150** proceeds to block **156** where the initial parameter values are calculated. If the number stays the same, the parameter calculation method **150** proceeds to block **158** where the current parameter values are calculated. Thereafter, the parameter calculation method **150** proceeds to block **150** where the initial and current parameter values are reported. By way of example, the initial parameter values may contain the average initial distance between points (or Distance (AVG) initial) and the current parameter values may contain the average current distance between points (or Distance (AVG) current). These may be compared in subsequent steps in order to control various aspects of a computer system.

[**0092**] The above methods and techniques can be used to implement any number of GUI interface objects and actions. For example, gestures can be created to detect and effect a user command to resize a window, scroll a display, rotate an object, zoom in or out of a displayed view, delete or insert text or other objects, etc. Gestures can also be used to invoke and manipulate virtual control interfaces, such as volume knobs, switches, sliders, handles, knobs, doors, and other widgets that may be created to facilitate human interaction with the computing system.

[**0093**] To cite an example using the above methodologies, and referring to **FIGS. 6A-6G**, a rotate gesture for controlling a virtual volume knob **170** on a GUI interface **172** of a display **174** of a tablet PC **175** will be described. In order to actuate the knob **170**, the user places their fingers **176** on a multipoint touch screen **178**. The virtual control knob may already be displayed, or the particular number, orientation or profile of the fingers at set down, or the movement of the fingers immediately thereafter, or some combination of these and other characteristics of the user's interaction may invoke the virtual control knob to be displayed. In either case, the computing system associates a finger group to the virtual control knob and makes a determination that the user intends to use the virtual volume knob. This association may also be based in part on the mode or current state of the computing device at the time of the input. For example, the same gesture may be interpreted alternatively as a volume know gesture if a song is currently playing on the computing device, or as a rotate command if an object editing application is being executed. Other user feedback may be provided, including for example audible or tactile feedback.

[**0094**] Once knob **170** is displayed as shown in **FIG. 6A**, the user's fingers **176** can be positioned around the knob **170** similar to if it were an actual knob or dial, and thereafter can be rotated around the knob **170** in order to simulate turning the knob **170**. Again, audible feedback in the form of a clicking sound or tactile feedback in the form of vibration, for example, may be provided as the knob **170** is "rotated." The user may also use their other hand to hold the tablet PC **175**.

[**0095**] As shown in **FIG. 6B**, the multipoint touch screen **178** detects at least a pair of images. In particular, a first image **180** is created at set down, and at least one other image **182** is created when the fingers **176** are rotated. Although only two images are shown, in most cases there would be many more images that incrementally occur between these two images. Each image represents a profile of the fingers in contact with the touch screen at a particular instant in time. These images can also be referred to as touch images. It will be understood that the term "image" does not mean that the profile is displayed on the screen **178** (but rather imaged by the touch sensing device). It should also be noted that although the term "image" is used, the data may be in other forms representative of the touch plane at various times.

[**0096**] As shown in **FIG. 6C**, each of the images **180** and **182** is converted to a collection of features **184**. Each feature **184** is associated with a particular touch as for example from the tips each of the fingers **176** surrounding the knob **170** as well as the thumb of the other hand **177** used to hold the tablet PC **175**.

[**0097**] As shown in **FIG. 6D**, the features **184** are classified, i.e., each finger/thumb is identified, and grouped for each of the images **180** and **182**. In this particular case, the features **184**A associated with the knob **170** are grouped together to form group **188** and the feature **184**B associated with the thumb is filtered out. In alternative arrangements, the thumb feature **184**B may be treated as a separate feature by itself (or in another group), for example, to alter the input or operational mode of the system or to implement another gesture, for example, a slider gesture associated with an equalizer slider displayed on the screen in the area of the thumb (or other finger).

[0098] As shown in **FIG. 6E**, the key parameters of the feature group **188** are calculated for each image **180** and **182**. The key parameters associated with the first image **180** represent the initial state and the key parameters of the second image **182** represent the current state.

[0099] Also as shown in **FIG. 6E**, the knob **170** is the UI element associated with the feature group **188** because of its proximity to the knob **170**. Thereafter, as shown in **FIG. 6F**, the key parameter values of the feature group **188** from each image **180** and **182** are compared to determine the rotation vector, i.e., the group of features rotated five (5) degrees clockwise from the initial to current state. In **FIG. 6F**, the initial feature group (image **180**) is shown in dashed lines while the current feature group (image **182**) is shown in solid lines.

[0100] As shown in **FIG. 6G**, based on the rotation vector the speaker **192** of the tablet PC **175** increases (or decreases) its output in accordance with the amount of rotation of the fingers **176**, i.e., increase the volume by 5% based on rotation of 5 degrees. The display **174** of the tablet PC can also adjust the rotation of the knob **170** in accordance with the amount of rotation of the fingers **176**, i.e., the position of the knob **170** rotates five (5) degrees. In most cases, the rotation of the knob occurs simultaneously with the rotation of the fingers, i.e., for every degree of finger rotation the knob rotates a degree. In essence, the virtual control knob follows the gesture occurring on the screen. Still further, an audio unit **194** of the tablet PC may provide a clicking sound for each unit of rotation, e.g., provide five clicks based on rotation of five degrees. Sill yet further, a haptics unit **196** of the tablet PC **175** may provide a certain amount of vibration or other tactile feedback for each click thereby simulating an actual knob.

[0101] It should be noted that additional gestures can be performed simultaneously with the virtual control knob gesture. For example, more than one virtual control knob can be controlled at the same time using both hands, i.e., one hand for each virtual control knob. Alternatively or additionally, one or more slider bars can be controlled at the same time as the virtual control knob, i.e., one hand operates the virtual control knob, while at least one finger and maybe more than one finger of the opposite hand operates at least one slider and maybe more than one slider bar, e.g., slider bar for each finger.

[0102] It should also be noted that although the embodiment is described using a virtual control knob, in another embodiment, the UI element can be a virtual scroll wheel. As an example, the virtual scroll wheel can mimic an actual scroll wheel such as those described in U.S. Patent Publication Nos.: 2003/0076303A1, 2003/0076301A1, 2003/0095096A1, which are all herein incorporated by reference. For example, when the user places their finger on the surface of the virtual scroll wheel and makes a swirling, rotational or tangential gesture motion, a scrolling action can be performed with respect to a list of items displayed in a window.

[0103] **FIG. 7** is a diagram of a touch-based method **200** in accordance with one embodiment of the present invention. The method generally begins at block **202** where a user input that occurs over a multipoint sensing device is detected. The user input includes one or more touch inputs, with each touch input having a unique identifier. Following block **202**, the touch-based method **200** proceeds to block **204** where the user input is classified as a tracking or selection input when the user input includes a single unique identifier (one touch input), or is classified as a gesture input when the user input includes at least two unique identifiers (more than one touch input). If the user input is classified as a tracking input, the touch-based method **200** proceeds to block **206** where tracking is performed corresponding to the user input. If the user input is classified as a gesture input, the touch-based method **200** proceeds to block **208** where one or more gesture control actions are performed corresponding the user input. The gesture control actions are based at least in part on changes that occur with or between the at least two unique identifiers.

[0104] **FIG. 8** is a diagram of a touch-based method **250** in accordance with one embodiment of the present invention. The touch-based method **250** generally begins at block **252** where an initial image is captured during an input stroke on a touch sensitive surface. Following block **252**, the touch-based method **250** proceeds to block **254** where the touch mode is determined based on the initial image. For example, if the initial image includes a single unique identifier then the touch mode may correspond to a tracking or selection mode. On the other hand, if the image includes more than one unique identifier, then the touch mode may correspond to a gesture mode. Following block **254**, the touch-based method **250** proceeds to block **256** where a next image is captured during the input stroke on the touch sensitive surface. Images are typically captured sequentially during the stroke and thus the there may be a plurality of images associated with the stroke. Following block **256**, touch-based method **250** proceeds to block **258** where a determination is made as to whether the touch mode changed between capture of the initial image and capture of the next image. If the touch mode changed, the touch-based method **250** proceeds to block **260** where the next image is set as the initial image and thereafter the touch mode is again determined at block **254** based on the new initial image. If the touch mode stayed the same, the touch-based method **250** proceeds to block **262** where the initial and next images are compared and one or more control signals are generated based on the comparison.

[0105] **FIG. 9** is a diagram of a touch-based method **300** in accordance with one embodiment of the present invention. The touch-based method **300** begins at block **302** where a GUI object is output. For example, a processor may instruct a display to display a particular GUI object. Following block **302**, the touch-based method **300** proceeds to block **304** where a gesture input is received over the GUI object. For instance, a user may set or move their fingers in a gestural way on the surface of the touch screen and while over the displayed GUI object. The gestural input may include one or more single gestures that occur consecutively or multiple gestures that occur simultaneously. Each of the gestures generally has a particular sequence, motion, or orientation associated therewith. For example, a gesture may include spreading fingers apart or closing fingers together, rotating the fingers, translating the fingers, and/or the like.

[0106] Following block **304** the touch-based method **300** proceeds to block **306** where the GUI object is modified based on and in unison with the gesture input. By modified, it is meant that the GUI object changes according to the particular gesture gestures being performed. By in unison, it

is meant that the changes occur approximately while the gesture or gestures are being performed. In most cases, there is a one to one relationship between the gesture(s) and the changes occurring at the GUI object and they occur substantially simultaneously. In essence, the GUI object follows the motion of the fingers. For example, spreading of the fingers may simultaneously enlarge the object, closing of the fingers may simultaneously reduce the GUI object, rotating the fingers may simultaneously rotate the object, translating the fingers may allow simultaneous panning or scrolling of the GUI object.

[0107] In one embodiment, block **306** may include determining which GUI object is associated with the gesture being performed, and thereafter locking the displayed object to the fingers disposed over it such that the GUI object changes in accordance with the gestural input. By locking or associating the fingers to the GUI object, the GUI object can continuously adjust itself in accordance to what the fingers are doing on the touch screen. Often the determination and locking occurs at set down, i.e., when the finger is positioned on the touch screen.

[0108] **FIG. 10** is a diagram of a zoom gesture method **350**, in accordance with one embodiment of the present invention. The zoom gesture may be performed on a multipoint touch screen. The zoom gesture method **350** generally begins at block **352** where the presence of at least a first finger and a second finger are detected on a touch sensitive surface at the same time. The presence of at least two fingers is configured to indicate that the touch is a gestural touch rather than a tracking touch based on one finger. In some cases, the presence of only two fingers indicates that the touch is a gestural touch. In other cases, any number of more than two fingers indicates that the touch is a gestural touch. In fact, the gestural touch may be configured to operate whether two, three, four or more fingers are touching, and even if the numbers change during the gesture, i.e., only need a minimum of two fingers at any time during the gesture.

[0109] Following block **352**, the zoom gesture method **350** proceeds to block **354** where the distance between at least the two fingers is compared. The distance may be from finger to finger or from each finger to some other reference point as for example the centroid. If the distance between the two fingers increases (spread apart), a zoom-in signal is generated as shown in block **356**. If the distance between two fingers decreases (close together), a zoom-out signal is generated as shown in block **358**. In most cases, the set down of the fingers will associate or lock the fingers to a particular GUI object being displayed. For example, the touch sensitive surface can be a touch screen, and the GUI object can be displayed on the touch screen. This typically occurs when at least one of the fingers is positioned over the GUI object. As a result, when the fingers are moved apart, the zoom-in signal can be used to increase the size of the embedded features in the GUI object and when the fingers are pinched together, the zoom-out signal can be used to decrease the size of embedded features in the object. The zooming typically occurs within a predefined boundary such as the periphery of the display, the periphery of a window, the edge of the GUI object, and/or the like. The embedded features may be formed on a plurality of layers, each of which represents a different level of zoom. In most cases, the amount of zooming varies according to the distance between the two objects. Furthermore, the zooming typically can occur substantially simultaneously with the motion of the objects. For instance, as the fingers spread apart or closes together, the object zooms in or zooms out at the same time. Although this methodology is directed at zooming, it should be noted that it may also be used for enlarging or reducing. The zoom gesture method **350** may be particularly useful in graphical programs such as publishing, photo, and drawing programs. Moreover, zooming may be used to control a peripheral device such as a camera, i.e., when the finger is spread apart, the camera zooms out and when the fingers are closed the camera zooms in.

[0110] **FIGS. 11A-11H** illustrate a zooming sequence using the method described above. **FIG. 11A** illustrates a display presenting a GUI object **364** in the form of a map of North America with embedded levels which can be zoomed. In some cases, as shown, the GUI object is positioned inside a window that forms a boundary of the GUI object **364**. **FIG. 11B** illustrates a user positioning their fingers **366** over a region of North America **368**, particularly the United States **370** and more particularly California **372**. In order to zoom in on California **372**, the user starts to spread their fingers **366** apart as shown in **FIG. 11C**. As the fingers **366** spread apart further (distance increases), the map zooms in further on Northern California **374**, then to a particular region of Northern California **374**, then to the Bay area **376**, then to the peninsula **378** (e.g., the area between San Francisco and San Jose Area), and then to the city of San Carlos **380** located between San Francisco and San Jose as illustrated in **FIGS. 11D-11H**. In order to zoom out of San Carlos **380** and back to North America **368**, the fingers **366** are closed back together following the sequence described above, but in reverse.

[0111] **FIG. 12** is a diagram of a pan method **400**, in accordance with one embodiment of the present invention. The pan gesture may be performed on a multipoint touch screen. The pan method **400** generally begins at block **402** where the presence of at least a first object and a second object are detected on a touch sensitive surface at the same time. The presence of at least two fingers is configured to indicate that the touch is a gestural touch rather than a tracking touch based on one finger. In some cases, the presence of only two fingers indicates that the touch is a gestural touch. In other cases, any number of more than two fingers indicates that the touch is a gestural touch. In fact, the gestural touch may be configured to operate whether two, three, four or more fingers are touching, and even if the numbers change during the gesture, i.e., only need a minimum of two fingers. Following block **402**, the pan method **400** proceeds to block **404** where the position of the two objects when the objects are moved together across the touch screen is monitored. Following block **404**, the pan method **400** proceeds to block **406** were a pan signal is generated when the position of the two objects changes relative to an initial position. In most cases, the set down of the fingers will associate or lock the fingers to a particular GUI object displayed on the touch screen. Typically, when at least one of the fingers is positioned over the image on the GUI object. As a result, when the fingers are moved together across the touch screen, the pan signal can be used to translate the image in the direction of the fingers. In most cases, the amount of panning varies according to the distance the two objects move. Furthermore, the panning typically can occur substantially simultaneously with the

motion of the objects. For instance, as the fingers move, the object moves with the fingers at the same time.

[0112] **FIGS. 13A-13D** illustrate a panning sequence based on the pan method **400** described above. Using the map of **FIG. 11**, **FIG. 13A** illustrates a user positioning their fingers **366** over the map. Upon set down, the fingers **366** are locked to the map. As shown in **FIG. 13B**, when the fingers **366** are moved vertically up, the entire map **364** is moved up thereby causing previously seen portions of map **364** to be placed outside the viewing area and unseen portions of the map **364** to be placed inside the viewing area. As shown in **FIG. 13C**, when the fingers **366** are moved horizontally sideways, the entire map **364** is moved sideways thereby causing previously seen portions of map **364** to be placed outside the vowing area and unseen portions of the map to be placed inside the viewing area. As shown in **FIG. 13D**, when the fingers **366** are moved diagonally, the entire map **364** is moved diagonally thereby causing previously seen portions of map **364** to be placed outside the viewing area and unseen portions of the map to be placed inside the viewing area. As should be appreciated, the motion of the map **364** follows the motion of the fingers **366**. This process is similar to sliding a piece of paper along a table. The pressure the fingers exert on the paper locks the paper to the fingers and when the fingers are slid across the table, the piece of paper moves with them.

[0113] **FIG. 14** is a diagram of a rotate method **450**, in accordance with one embodiment of the present invention. The rotate gesture may be performed on a multipoint touch screen. The rotate method **450** generally begins at block **452** where the presence of a first object and a second object are detected at the same time. The presence of at least two fingers is configured to indicate that the touch is a gestural touch rather than a tracking touch based on one finger. In some cases, the presence of only two fingers indicates that the touch is a gestural touch. In other cases, any number of more than two fingers indicates that the touch is a gestural touch. In fact, the gestural touch may be configured to operate whether two, three, four or more fingers are touching, and even if the numbers change during the gesture, i.e., only need a minimum of two fingers.

[0114] Following block **452**, the rotate method **450** proceeds to block **454** where the angle of each of the finger is set. The angles are typically determined relative to a reference point. Following block **454**, rotate method **450** proceeds to block **456** where a rotate signal is generated when the angle of at least one of the objects changes relative to the reference point. In most cases, the set down of the fingers will associate or lock the fingers to a particular GUI object displayed on the touch screen. Typically, when at least one of the fingers is positioned over the image on the GUI object, the GUI object will be associated with or locked to the fingers. As a result, when the fingers are rotated, the rotate signal can be used to rotate the object in the direction of finger rotation (e.g., clockwise, counterclockwise). In most cases, the amount of object rotation varies according to the amount of finger rotation, i.e., if the fingers move 5 degrees then so will the object. Furthermore, the rotation typically can occur substantially simultaneously with the motion of the fingers. For instance, as the fingers rotate, the object rotates with the fingers at the same time.

[0115] **FIGS. 15A-15C** illustrate a rotating sequence based on the method described above. Using the map of **FIG. 11**, **FIG. 15A** illustrates a user positioning their fingers **366** over the map **364**. Upon set down, the fingers **366** are locked to the map **364**. As shown in **FIG. 15B**, when the fingers **366** are rotated in a clockwise direction, the entire map **364** is rotated in the clockwise direction in accordance with the rotating fingers **366**. As shown in **FIG. 15C**, when the fingers **366** are rotated in a counterclockwise direction, the entire map **364** is rotated in the counter clockwise direction in accordance with the rotating fingers **366**.

[0116] It should be noted that the methods described in **FIGS. 10-15** can be implemented during the same gestural stroke. That is, zooming, rotating and panning can all be performed during the gestural stroke, which may include spreading, rotating and sliding fingers. For example, upon set down with at least two fingers, the displayed object (map) is associated or locked to the two fingers. In order to zoom, the user can spread or close their fingers. In order to rotate, the user can rotate their fingers. In order to pan, the user can slid their fingers. Each of these actions can occur simultaneously in a continuous motion. For example, the user can spread and close their fingers while rotating and sliding them across the touch screen. Alternatively, the user can segment each of these motions without having to reset the gestural stroke. For example, the user can first spread their fingers, then rotate their fingers, then close their fingers, then slide their fingers and so on.

[0117] **FIG. 16** is a diagram of a GUI operational method **500**, in accordance with one embodiment of the present invention. The GUI operational method **500** is configured for initiating floating controls in a GUI. The GUI operational method **500** generally begins at block **502** where the presence of an object such as a finger or thumb is detected. This may for example be accomplished using a touch screen. Following block **502**, the GUI operational method **500** proceeds to block **504** where the object is recognized (the identity of the object is found). The object may be recognized among a plurality of objects. For example, see block **104** of **FIG. 2** above.

[0118] Following block **504**, the GUI operational method **500** proceeds to block **506** where an image in the vicinity of the object is generated. The image is typically based on the recognized object. The image may include windows, fields, dialog boxes, menus, icons, buttons, cursors, scroll bars, etc. In some cases, the user can select and activate the image (or features embedded therein) in order to initiate functions and tasks. By way of example, the image may be a user interface element or a group of user interface elements (e.g., one or more buttons that open, close, minimize, or maximize a window). The image may also be one or more icons that launch a particular program or files that open when selected. The image may additionally correspond to non interactive text and graphics. In most cases, the image is displayed as long as the object is detected or it may be displayed for some preset amount of time, i.e., after a period of time it times out and is removed.

[0119] In one particular embodiment, the image includes one or more control options that can be selected by the user. The control options may include one or more control buttons for implementing various tasks. For example, the control option box may include music listening control buttons as for example, play, pause, seek and menu.

[0120] **FIGS. 17A-17E** illustrate a floating control sequence using the method described above. As shown in

FIG. 17A, a user **510** is using a tablet PC **512** and therefore is holding the tablet PC **512** with one hand **514** while navigating (e.g., tracking, gesturing) with the other hand **516**. As shown in **FIG. 17B**, which is a close up of the user holding the tablet PC **512**, a portion of the thumb of the holding hand **514** is positioned over the touch screen **520**. As shown in **FIG. 17C**, the tablet PC **512** recognizes the thumb and displays a control box **522** adjacent the thumb. The control box **522** includes various buttons **524**, which can be selected by the user's thumb to initiate tasks in the tablet PC **512**. As shown in **FIG. 17D**, while holding the tablet PC **512**, the thumb is extended over one of the buttons **524** and subsequently tapped thereby selecting the task associated with the button **524**. By way of example, the task may be associated with launching a program or gaining access to a network or changing the mode of operation of the device. The control box **522** and buttons **524** may be used to change the input mode of the touch screen **520** so that, for example, the identical gesture made with the fingers of the user's other hand may have multiple meanings depending on which of buttons **524** is selected. As shown in **FIG. 17E**, when the thumb is moved away from the touch screen **520**, the control box **522** may time out and disappear. Alternatively, the control box may be closed using conventional close icons or buttons.

[**0121**] **FIG. 18** is a diagram of a GUI operational method **550**, in accordance with one embodiment of the present invention. The GUI operational method **550** is configured for initiating zooming targets. The GUI operational method **550** generally begins at block **552** where a control box GUI element is displayed. The control box contains one or more control buttons, which are somewhat close together, and which can be used to perform actions. The control box may, for example, include control buttons such as maximize, minimize, close, and the like. Following block **552**, the GUI operational method **550** proceeds to block **554** where the control box is enlarged, or at least one of the control buttons is enlarged for a period of time when the presence of an object over the control box or one of the control buttons is detected. In the case where the control box is enlarged each of the control buttons is enlarged thereby making selection thereof much easier. In the case where only the control button is enlarged, the user would decide whether this is the correct button and if so select the enlarged control button, or restart the process so that the appropriate control button is presented. In most cases, the size of the control buttons corresponds to the size of the finger so that they may be easily selected by the object. Following block **554**, the GUI operational method **550** proceeds to block **556** where a control signal associated with the selected control button is generated when the presence of the object over one of the enlarged control buttons is detected.

[**0122**] **FIGS. 19A-19D** illustrate a zooming target sequence using the GUI operational method **550** described above. As shown in **FIG. 19A**, a user **510** places their finger **576** over a control box **578**. Because the buttons **580** of the control box **578** included therein are smaller than the finger **576** and located close together, it is difficult for the user **510** to make a selection directly without possibly pressing an undesirable button **580**, e.g., a button adjacent the desired button. By way of example, the finger **576** may cover two or more of the buttons **580**. As shown in **FIG. 19B**, at least a portion of the control box **578** is enlarged including the buttons **580** included therein when the user places their thumb over the control box. As shown in **FIG. 19C**, once the control box has reached its enlarged state, the user can select one of the enlarged buttons, which is now closer to the size of the thumb. By way of example, the user may tap on the desired control button. As shown in **FIG. 19D**, the control box reduces to its initial size after the button is selected or after a predetermined time period in which no selection was made (e.g., times out) or when the user moves their finger away from the control box.

[**0123**] **FIG. 20** is a diagram of a GUI operational method **600**, in accordance with one embodiment of the present invention. The GUI operational method **600** is configured for initiating a page turn. The GUI operational method **600** generally begins at block **602** where a page from a multitude of pages is displayed in a GUI. By way of example, the pages may be associated with an electronic book. Following block **602**, the GUI operational method **600** proceeds to block **604** where the presence of an object (or objects) in a predetermined region over the page is detected. The predetermined area may, for example, correspond to the area where the page number is displayed. Following block **604**, the GUI operational method **600** proceeds to block **606** where a page turn signal is generated when the object (or objects) is translated in the predetermined region. The translation is configured to simulate a finger turning the page in an actual paper bound book. The direction of the translation indicates whether to go to the next page or previous page in the list of pages. For example, if the finger is swiped right to left, then a page back signal is generated, and if the finger is swiped left to right, then a page up signal is generated. This GUI operational method **600** may be enhanced several ways. For instance, if multiple fingers are swiped, then this may create a paging signal greater than one page. For example, a two finger swipe equals two page turns, three finger swipe equals three page turns, etc. Or a two finger swipe equals ten page turns, three finger swipe equals 50 page turns, etc.

[**0124**] **FIGS. 21A-21D** illustrate a page turning sequence using the GUI operational method **600** described above. As shown in **FIG. 21A**, which is a close up of a user **510** holding the tablet PC **512**, the user swipes their finger over the page number in a direction to the left of the page **630**. As shown in **FIG. 21B**, the tablet PC **512** recognizes the swipe and direction of the swipe in the area of the page number and therefore the tablet PC **512** displays the next page in a group of pages. This can be performed repeatedly to whisk through the group of pages. As shown in **FIG. 21C**, the user swipes their finger **576** over the page number in a direction to the right of the page **630**. As shown in **FIG. 21D**, the tablet PC **512** recognizes the swipe and direction of the swipe in the area of the page number and therefore the tablet PC **512** displays the previous page in a group of pages. This can be performed repeatedly to whisk through the group of pages.

[**0125**] **FIG. 22** is a diagram of a GUI operational method **650**, in accordance with one embodiment of the present invention. The GUI operational method **650** is configured for initiating inertia typically during a scrolling or panning operation. Inertia is generally defined as the tendency of a body at rest to remain at rest or of a body in motion to stay in motion in a straight line unless disturbed by an external force. In this particular embodiment, the GUI or some portion thereof is associated with inertial properties, which is its resistance to rate of change in motion. For a GUI with

high inertia characteristics, the acceleration of the GUI will be small for a given input. On the other hand, if the GUI has low inertia characteristics, the acceleration will be large for a given input.

[0126] The GUI operational method **650** generally begins at block **652** where a graphical image is displayed on a GUI. Following block **652**, the GUI operational method **650** proceeds to block **654** where a scrolling or panning stroke on a touch sensitive surface is detected. By way of example, the stroke may be a linear or rotational stroke. During a linear stroke, the direction of scrolling or panning typically follows the direction of the stroke. During a rotational stroke (see **FIG. 6**), the rotational stroke is typically converted to a linear input where clockwise motion may correspond to vertical up and counterclockwise motion may correspond to vertical down. Following block **654** the process flow proceeds to block **656** where the speed and direction of the scrolling or panning stroke is determined. Following block **656**, the GUI operational method **650** proceeds to block **658** where the image is moved in accordance with the speed and direction of the scrolling or panning stroke as well as the associated inertial characteristics. Following block **658**, the GUI operational method **650** proceeds to block **660** where the motion of the image continues even when the panning or scrolling stroke is no longer detected. For example, when the user picks up their finger from the touch sensitive surface, the scrolling or panning function continues as if the scrolling or panning stroke was still being made. In some cases, the motion of the image continues infinitely until some braking (stopping or slowing) control is performed. This particular methodology simulates zero gravity. In other cases, the motion of the image is slowed in accordance with the associated inertia GUI operational method **650**. Metaphorically speaking, the image may correspond to a piece of paper moving over a desktop. In order to move the piece of paper, the user exerts a force on the paper in the desired direction. When the user lifts their finger off the paper, the paper will continue to slid along the desktop in the desired direction for some period of time. The amount it slides after lifting the finger generally depends on, among other things, its mass, the force applied by the finger, the friction force found between the paper and the desktop, etc. As should be appreciated, traditionally when scrolling and panning are implemented, the scrolling or panning stops when the fingers are picked up. In contrast, using the above mentioned methodology, the scrolling or panning continues to move when the fingers are picked up.

[0127] The GUI operational method **650** may additionally include blocks A and B. In block A, an object such as a finger is detected on the touch sensitive surface when the image is moving without the assistance of the object (block **660**). In block B, the motion of the image is stopped when the object is detected, i.e., the new touch serves as a braking means. Using the metaphor above, while the piece of paper is sliding across the desktop, the user presses their finger on the paper thereby stopping its motion.

[0128] **FIGS. 23A-23D** illustrate an inertia sequence using the method described above. **FIG. 23A** illustrates a display presenting a GUI **678** including a window **679** having a list **680** of media items **681**. The window **679** and list **680** may for example correspond to a control window and music list found in iTunes® manufactured by Apple Computer, Inc of Cupertino, Calif. As shown in **FIG. 23B**, when the user slides their finger or fingers **576** over the touch screen **520**, vertical scrolling, which moves media items up or down through the window, is implemented. The direction of scrolling may follow the same direction as finger movement (as shown), or it may go in the reverse direction. In one particular embodiment, a single finger is used for selecting the media items from the list, and two fingers are used to scroll through the list.

[0129] Scrolling generally pertains to moving displayed data or images (e.g., media items **681**) across a viewing area on a display screen so that a new set of data (e.g., media items **681**) is brought into view in the viewing area. In most cases, once the viewing area is full, each new set of data appears at the edge of the viewing area and all other sets of data move over one position. That is, the new set of data appears for each set of data that moves out of the viewing area. In essence, these functions allow a user to view consecutive sets of data currently outside of the viewing area. In most cases, the user is able to accelerate their traversal through the data sets by moving his or her finger at greater speeds. Examples of scrolling through lists can be found in U.S. Patent Publication Nos.: 2003/0076303A1, 2003/0076301A1, 2003/0095096A1, which are herein incorporated by reference.

[0130] As shown in **FIG. 23C**, the displayed data continues to move even when the finger is removed from the touch screen. The continuous motion is based at least in part on the previous motion. For example the scrolling may be continued in the same direction and speed. In some cases, the scrolling slow down over time, i.e., the speed of the traversal through the media items gets slower and slower until the scrolling eventually stops thereby leaving a static list. By way of example, each new media item brought into the viewing area may incrementally decrease the speed. Alternatively or additionally, as shown in **FIG. 23D**, the displayed data stops moving when the finger **576** is placed back on the touch screen **520**. That is, the placement of the finger back on the touch screen can implement braking, which stops or slows down the continuous acting motion. Although this sequence is directed at vertical scrolling it should be noted that this is not a limitation and that horizontal scrolling as well as panning may be performed using the methods described above.

[0131] **FIG. 24** is a diagram of a GUI operational method **700**, in accordance with one embodiment of the present invention. The method **700** is configured for simulating a keyboard. The method generally begins at block **702** where a keyboard is presented on the display. Following block **702**, the process flow proceeds to block **704** where the presence of a first object over a first key and a second object over a second key at the same time is detected on a touch screen. The touch screen is positioned over or in front of the display. By way of example, the display may be an LCD and the touch screen may be a multipoint touch screen. Following block **704**, the process flow proceeds to block **706** where one or more simultaneous control signals are generated when the first object is detected over the first key and when the second object is detected over the second key at the same time.

[0132] In one embodiment, only a single control signal is generated when the first object is detected over the first key and when the second object is detected over the second key at the same time. By way of example, the first key may be

a shift key and the second key may be a symbol key (e.g., letters, numbers). In this manner, the keyboard acts like a traditional keyboard, i.e., the user is allowed to select multiple keys at the same time in order to change the symbol, i.e., lower/upper case. The keys may also correspond to the control key, alt key, escape key, function key, and the like.

[0133] In another embodiment, a control signal is generated for each actuated key (key touch) that occurs at the same time. For example, groups of characters can be typed at the same time. In some cases, the application running behind the keyboard may be configured to determine the order of the characters based on some predetermined criteria. For example, although the characters may be jumbled, the application can determine that the correct order of characters based on spelling, usage, context, and the like.

[0134] Although only two keys are described, it should be noted that two keys is not a limitation and that more than two keys may be actuated simultaneously to produce one or more control signals. For example, control-alt-delete functionality may be implemented or larger groups of characters can be typed at the same time.

[0135] **FIGS. 25A-25D** illustrates a keyboard sequence using the method described above. **FIG. 25A** illustrates a display presenting a GUI object **730** in the form of a keyboard. As shown in **FIG. 25B**, a user positions their fingers **576** over the multipoint touch screen **520** over the keyboard **730** to enter data into a word processing program. By way of example, the user may place one of their fingers **576**A on the Q key in order to produce a lower case "q" in the word processing program. As shown in **FIG. 25C**, when the user decides that a letter should be in upper case, the user places one finger **576**B on the shift key and another finger **576**A on the desired letter (as indicated by the arrows). As shown in **FIG. 25D**, in order to continue typing in lower case, the user simply removes their finger **576**B from the shift key and places their finger **576**A over a desired letter (as indicated by the arrow).

[0136] **FIG. 26** is a diagram of a GUI operational method **750**, in accordance with one embodiment of the present invention. The method **750** is configured for simulating a scroll wheel such as those described in U.S. Patent Publication Nos.: 2003/0076303A1, 2003/0076301A1, 2003/0095096A1, all of which are herein incorporated by reference. The method generally begins at block **752** where a virtual scroll wheel is presented on the display. In some cases, the virtual scroll wheel may include a virtual button at its center. The virtual scroll wheel is configured to implement scrolling as for example through a list and the button is configured to implement selections as for example items stored in the list. Following block **752**, the method proceeds to block **754** where the presence of at least one finger and in some cases more than one finger, such as first and second fingers, over the virtual scroll wheel is detected on a touch screen. The touch screen is positioned over or in front of the display. By way of example, the display may be an LCD and the touch screen may be a multipoint touch screen. Following block **754**, the method proceeds to block **756** where the initial position of the fingers on the virtual scroll wheel is set. By way of example, the angle of the fingers relative to a reference point may be determined (e.g., 12 o clock, 6 o clock, etc.). In most cases, the set down of the finger(s) associate, link or lock the fingers (or finger) to the virtual scroll wheel when the fingers are positioned over the virtual scroll wheel.

[0137] Following block **756**, the method **750** proceeds to block **758** where a rotate signal is generated when the angle of the fingers change relative to the reference point. The rotate signal can be used to perform several actions including for example scrolling through a plurality of media items, and possibly moving the virtual scroll wheel with the finger(s). By way of example, the combination and frequency of the signals may be converted into distance, direction and speed necessary for moving a selector through the media items as well for moving the virtual scroll wheel around its axis. In most cases, the amount of scrolling and wheel rotation varies according to the amount of finger rotation. By way of example, if the fingers move 5 degrees then so will the wheel. Furthermore, the scrolling and rotation of the wheel typically occurs substantially simultaneously with the motion of the fingers. For instance, as the fingers rotate, both scrolling and rotation of the wheel are performed at the same time. Moreover, although not a requirement, the direction of scrolling and rotation of the wheel is generally the same as the direction of the finger motion. For example, the virtual scroll wheel rotates in the direction of finger rotation (e.g., clockwise, counterclockwise, etc.)

[0138] In some cases, the principals of inertia as described above can be applied to the virtual scroll wheel. In cases such as these, the virtual scroll wheel continues to rotate when the fingers (or one of the fingers) are lifted off of the virtual scroll wheel and slowly comes to a stop via virtual friction. Alternatively or additionally, the continuous rotation can be stopped by placing the fingers (or the removed finger) back on the scroll wheel thereby braking the rotation of the virtual scroll wheel.

[0139] It should be pointed out that a rotating virtual scroll wheel is not a limitation, and in some cases, the virtual scroll wheel may remain stationary (e.g., does not rotate with the fingers) so as to simulate a touch surface rather than a mechanical rotating wheel.

[0140] **FIGS. 27A-27D** illustrates a scroll wheel sequence using the method described above. **FIG. 27A** illustrates a display presenting a scroll wheel. The scroll wheel may be displayed automatically as part of a program or it may be displayed when a particular gesture is performed. By way of example, during the operation of a music program (such as iTunes® manufactured by Apple Computer Inc., of Cupertino, Calif.), the virtual scroll wheel may appear on the GUI of the music program when two fingers are placed on the touch screen rather than one finger which is typically used for tracking in the music program. In some cases, the virtual scroll wheel only appears when two fingers are placed on a predetermined area of the GUI. Alternatively, the appearance of the virtual scroll wheel may be based on something other than or in addition to the number of fingers. For example, the virtual scroll wheel may appear in response to any touch that occurs when the music program is running.

[0141] As shown in **FIG. 27B**, a user positions their fingers over the multipoint touch screen **520** over the scroll wheel. At some point, the fingers are locked to the scroll wheel. This can occur at set down for example. As shown in **FIG. 27C**, when the fingers are rotated in a clockwise

direction, the scroll wheel is rotated in the clockwise direction in accordance with the rotating fingers. As shown in **FIG. 27D**, when the fingers are rotated in a counterclockwise direction, the virtual scroll wheel is rotated in the counter clockwise direction in accordance with the rotating fingers. Alternatively, rotation of the virtual scroll wheel may also be rotated with linear motion of the fingers in a tangential manner.

[**0142**] **FIG. 28** is user interface method **800**, in accordance with one embodiment of the present invention. The user interface method **800** may for example be performed on a computing device having a display and a touch sensitive input device such as a touch screen. The user interface method **800** begins at block **802** where a touch is detected. This may be accomplished with the touch sensitive input device when an object such as a stylus or one or more fingers, is placed on the touch sensitive surface of the touch sensitive input device.

[**0143**] Once a touch is detected, the user interface method **800** proceeds to block **804** where a user interface (UI) mode is determined in response to the touch. The user interface mode may be widely varied. The user interface mode may include navigation modes, scroll modes, data entry modes, edit modes, control modes, information modes, display modes, etc. Each mode typically has one or more GUI interface elements associated therewith. By way of example, a virtual scroll wheel (e.g., **FIG. 27**) or slider bar may be associated with a scroll mode, a keyboard (e.g., **FIG. 25**) or keypad may be associated with data entry mode, a tool bar such as a formatting tool bar or drawing tool bar may be associated with an edit mode, a control panel including buttons may be associated with a control mode, a window may be associated with an information mode, etc.

[**0144**] The user interface mode may be determined at block **804** based on one or more conditions including for example, one or more applications currently running on the computing device, the current state or mode of the one or more applications and/or the touch characteristics associated with the touch. In fact, determining the user interface mode at block **804** may involve monitoring and analyzing one or more conditions.

[**0145**] The current applications may for example include operating systems (e.g., Mac OS), word processing programs, spreadsheet programs, draw editing programs, image editing programs, gaming programs, photo management programs (e.g., iPhoto), music management programs (e.g., iTunes), video editing programs (e.g., iMovie), movie management programs (e.g., QuickTime), music editing programs (e.g., GarageBand), Internet interface programs and/or the like.

[**0146**] The current state or mode of the applications may correspond to an active portion of the application (e.g., current window or windows within windows). For example, the active portion of a music management program may correspond to a music control mode, a playlist select mode, a menu mode, and/or the like. Further, the active portion of a photo management program may correspond to photo browsing mode or photo editing mode. Further still, the active portion of an internet interface program may correspond to a web mode or an email mode.

[**0147**] The touch characteristics on the other hand may, for example, correspond to touch location, touch ID, number of touches, etc, as described in many of the embodiments mentioned above.

[**0148**] With regards to applications, different applications may indicate different UI modes. For example, a word processing or spreadsheet application may indicate a data entry mode, while a music management program may indicate a control or scrolling mode. With regards to the current state of an application, different modes of the application may indicate different UI modes. For example, in a music management program, a menu window may indicate one UI mode, while a playlist window may indicate another UI mode.

[**0149**] With regards to the touch, the number of fingers may indicate different UI modes. For example, one finger may indicate a first mode while two fingers may indicate a second mode. In addition, the identity of the touch may indicate different UI modes. For example, a thumb may indicate a first UI mode and an index finger may indicate a second UI mode. Moreover, the location of the touch may indicate different UI modes. For example, a first touch location may indicate a first UI mode, while a second touch location may indicate a second UI mode (if the touch is located over a border of a music program a first UI mode may be implemented, and if the touch is located over a playlist or list of songs in the music program a second UI mode may be implemented.

[**0150**] In one embodiment, the user interface mode is based on only one of the conditions. For example, the user interface mode is only based on the application, the current state of the application or one of the various touch characteristics as described above. In another embodiment, the user interface mode is based on multiple conditions. For example, the user interface mode may be based on a combination of at least two selected from the application, the current state of the application and the various touch characteristics. By way of example, the application combined with a first touch characteristic may indicate a first UI mode and the same application combined with a second touch characteristic may indicate a second UI mode.

[**0151**] To cite a few examples, if the application is a word processing or spreadsheet program then the mode may be determined to be a data entry mode so that data can be entered into the spreadsheet (e.g., keyboard). If the application is a music management program and a playlist is currently showing (active portion), the mode may be determined to be a scroll mode so that the items in the list may be scrolled through in order to find a desired item (e.g., scroll wheel). Alternatively, if a song is playing (active portion), the mode may be determined to be a control mode so that the manner in which songs are played can be controlled (e.g., play, stop, seek and volume control options). Further, if the application is a photo management program and a particular photo is displayed (active portion), the mode may be determined to be a control mode so that the photo can be modified (e.g., converting to black and white, removing red eye, and rotate options).

[**0152**] After determining the user interface mode **804**, the user interface method **800** proceeds to block **806** where one or more GUI elements are displayed based on the user interface mode and in response to the touch(s). In some

cases, only one GUI element is displayed and in other cases multiple GUI elements are displayed. The GUI element is typically associated with a particular mode. For example, a slider bar or scroll wheel may be displayed in a scroll mode, a keyboard or keypad may be displayed in a data entry mode, a tool bar may be displayed in an edit mode, various buttons or a control panel may be displayed in a control mode, and information windows may be displayed in an information mode.

[**0153**] The GUI element may be displayed in a variety of ways. For example, it can be positioned over the currently displayed graphical images, or it can displace the currently displayed graphical images (e.g., minimize, shift, etc.). In some cases, the GUI element is made semi transparent so that the current graphical image disposed below the GUI element can be seen (thereby eliminating minimizing and shifting). This may be helpful when using a scroll wheel to traverse through a list disposed below the scroll wheel. Furthermore, the GUI element can be placed in the vicinity of the touch or it can be placed in some predetermined location. The predetermined location may be based on ergonomics, i.e., what is the best location for the user.

[**0154**] In addition to the above, the GUI element may be displayed using a transition effect such as growing, fading in, popping up, and in some cases may even pulsate, throb, etc. If the effect is popping up, the GUI element is immediately brought into view. If the effect is growing, as shown in **FIGS. 29A-29D**, a small GUI element **820**A (scroll wheel) is initially displayed, and thereafter the GUI element **820**A continuously enlarges through various sizes **820**B and **820** C until it reaches its desired size **820**D. The speed of growth may be based on the pressure of the touch. For example, if the touch pressure is low, the GUI element may grow slowly and if the touch pressure is high, the GUI element may grow more rapidly. In addition, the final size of the GUI element may be based on the length of the touch. For example, the GUI element stops growing when the touch is no longer detected. Alternatively, the speed and size may be user adjustable as for example through a control panel. If the effect is fading, as shown in **FIGS. 30A-30D**, the GUI element **822** is slowly brought into view from nothing, through various levels of distortion or transparency **822**A-**822**C, to a final complete image **822**D. The fading can be controlled similarly to growth. For example, the speed and level of fade may be controlled by the pressure and length of touch.

[**0155**] The transition effect may even carry over to the currently displayed images, i.e., the images currently displayed before the touch was detected. In one embodiment, the opposite effect happens to the currently displayed images. For example, as shown in **FIGS. 31A-31D**, the currently displayed graphical images **826** are minimized smaller and smaller as the GUI element **820** grows larger and larger. Alternatively, if the GUI element immediately pops in, the currently displayed graphical images can immediately pop out or be immediately minimized.

[**0156**] Once the GUI element is displayed (**806**), the user interface method **800** proceeds to block **808** where the functionality of the GUI element is enabled. For example, touch event is monitored relative to the GUI element and actions associated with the touch event are performed. The enablement of the GUI element may occur simultaneously with the display of the GUI element so that a user can immediately start using the GUI element once displayed. By way of example, in a scroll mode, a virtual scroll wheel may be displayed and when enabled, the touch events are monitored relative to the scroll wheel. During monitoring, control signals associated with the position of the finger on the scroll wheel are generated as the finger swirls around the virtual scroll wheel. These signals can be used to perform scrolling. For example, the number, combination and frequency of the signals may be converted into distance, direction and speed necessary for moving a selection bar through a list. By way of example, see **FIGS. 6, 26** and **27** for more detailed descriptions of virtual scroll wheels and how they function.

[**0157**] At some point after enabling and displaying the GUI element, a determination **812** is made as to whether or not to deactivate the GUI element. The determination **812** can be made in a variety of ways including, for example: 1) the touch is no longer detected, 2) a touch has not been detected for a preset amount of time, 3) a time out occurs (a preset amount of time has gone by since the GUI element was first displayed/enabled), or 4) a user selection (e.g., a user selects a button that closes the GUI element).

[**0158**] If the determination indicates deactivation, then the method proceeds to block **814** where the GUI element is disabled and removed from display. Once disabled, actions will no longer be performed when a touch event occurs. The removal of the GUI element from display may function similar to displaying the GUI element in that it can be removed using a transition effect such as slowly fading out, shrinking or immediately disappearing (popping out). The removal transition effect may work opposite the displaying transitioning effect. For example, the GUI element fades out similarly to fading in, shrinks similarly to growth or pops out similarly to popping in. Further, the GUI element can slowly recess and disappear from view while the displaced or shrunken current graphical images can slowly grow back to their original size and shape. If the determination does not indicate deactivation, then the method maintains the display of the GUI element as well as the enablement thereof.

[**0159**] **FIG. 32** is a determination method **850** in accordance with one embodiment of the present invention. The determination method may, for example, correspond to the block **804** in **FIG. 28**. The determination method begins at block **852** where the current application is determined. The determination method proceeds to block **854** where the current state of the application is determined. Following block **854**, the determination method proceeds to block **856** where the touch characteristics associated with the touch are determined. The determination method proceeds to block **860** where a UI mode is selected based on the results from blocks **852-858**. By way of example, a set of rules may indicate appropriate UI modes for a particular set of conditions.

[**0160**] **FIG. 33** is a user interface method **900** in accordance with one embodiment of the present invention. The method may, for example, be performed on a computing device having a display and a touch sensitive input device such as a touch screen. The interface method **900** begins at block **902** where a list of songs are displayed. **FIG. 34A** shows one example of a window **930**A including a list of songs **932**A, and **FIG. 35A** shows another example of a window **930**B including a list of songs **932**B. **FIG. 34A**

may, for example, be a graphical user interface displayed on an iPod® manufactured by Apple Computer of Cupertino, Calif., and **FIG. 35A** may for example be a graphical user interface associated with a music management program, such as iTunes® manufactured by Apple Computer of Cupertino, Calif.

[**0161**] Following block **902**, the user interface method **900** proceeds to block **904** where a touch is detected over the displayed list of songs (or window or entire GUI). This may be accomplished with the touch sensitive input device when an object such as a stylus or one or more fingers is placed on the touch sensitive surface of the touch sensitive input device such as a touch screen. **FIGS. 34B and 35B** show a finger **925** placed over the window **930** including the list of songs **932**.

[**0162**] Once a touch is detected, the user interface method **900** proceeds to block **906** where a virtual scroll wheel is activated. That is, a virtual scroll wheel is displayed in addition to the list of songs and its functionality is enabled. In essence, because the song list was touched, a scroll wheel that allows a user to traverse through the songs in the list of songs is provided. In some cases, the virtual scroll wheel displaces the media items, i.e., the media items are minimized or shifted to make room for the virtual scroll wheel. In other cases, the virtual scroll wheel is positioned or laid over the media items (the media items keep their current size, shape and position). The virtual scroll wheel can be made semi-transparent so that the media items can be viewed through the virtual scroll wheel. **FIGS. 34C and 35C** show a transparent virtual scroll **936** wheel laid over the window **930** including the list of songs **932**. Alternatively, a virtual slider bar may be displayed.

[**0163**] Once displayed, a determination **908** is made as to whether or not a scrolling touch event (or gesture) is performed relative to the virtual scroll wheel. For example, whether or not a finger is positioned over the scroll wheel and whether or not the finger is moved around the scroll wheel in a swirling fashion.

[**0164**] If a scrolling touch event is performed by the user, the user interface method **900** proceeds to block **910** where scrolling is implemented through the list of songs in accordance with the scrolling touch event. By way of example, a selector bar may be moved from one song to another as the finger is swirled around the virtual scroll wheel. **FIGS. 34D and 35D** show the finger **925** swirling around the virtual scroll wheel **936**, and a selector bar **938** moving linearly through the list of songs **932** in accordance with the swirling finger **925**. In the illustrated embodiments, the selector bar is moved linearly up when the finger is swirled in a clockwise fashion and linearly down when the finger is swirled in a counterclockwise fashion. It should be noted, however, that this is not a limitation. For example, the selector bar may moved linearly down when the finger is swirled in a clockwise fashion and linearly up when the finger is swirled in a counterclockwise fashion.

[**0165**] If a scrolling or select touch event is not performed, the user interface method **900** proceeds to block **916** where the virtual scroll wheel is deactivated. That is, the virtual scroll wheel is disabled and removed from the display. **FIGS. 34E and 35E** show the display **928** without the virtual scroll wheel **936**. Although the virtual scroll wheel **936** is removed, changes made to the list of songs, i.e., the position of the selector bar **938**, typically remain.

[**0166**] In some cases, the virtual scroll wheel may include button zones across its surface or a virtual button at its center or around its sides. The buttons and button zones may for example correspond to menu, play, seek, pause, and/or the like. In this particular embodiment, the method described above may include additional steps that occur before block **416**. For example, if a scrolling touch event is not performed, the user interface method **900** may include an additional block where a determination is made as to whether or not a selection touch event (or gesture) is performed relative to the virtual scroll wheel. The selection touch event may be implemented by tapping the button or by exerting increased or decreased pressure on the button rather than swirling around the surface of the virtual scroll wheel (see **FIGS. 34F and 35F**). If the button is a song select or enter button, the method include another block where the song with the selector bar disposed thereover is selected. That is, when the virtual button is tapped, or otherwise selected, the song currently covered by the selector bar is played and outputted for the user's enjoyment.

[**0167**] It should be noted that the methods described above are not limited to scrolling through a list of songs. Any media item as well as any group of elements can be scrolled through using the aforementioned technique. For example, in photo layout **942** as shown in **FIGS. 36A-36C**, the virtual scroll wheel **936** may appear when the user places their finger **925** over the photo layout **942** (or grouping), and thereafter it can be used to move a highlighter **944** through the various photos **943** in the layout **942**. By way of example, the photos may be thumbnails images that make traversing through a large number of images easier.

[**0168**] **FIG. 37** is a method **950**, in accordance with one embodiment of the present invention. The method begins at block **952** where it is determined if a touch is detected. If a touch is detected, the method proceeds to block **954** where the current operating conditions are monitored and analyzed. The conditions may for example correspond to the current application, the state of the application and/or the touch characteristics associated with the touch.

[**0169**] If a first set of conditions are implemented, the method proceeds to block **956** where a first GUI element is activated. For example, as shown in FIGS. **38**A-B, in an active window **960** of a music management program, a scroll wheel **962** may be activated when a user touches a playlist portion **964** of the active window **960**.

[**0170**] If a second set of conditions are implemented, the method proceeds to block **958** where a second GUI element is activated. For example, as shown in FIGS. **38**B-C, in the active window **960** of a music management program, a music control panel **966** may be activated when a user also touches a border **968** of the active window **960**. Although they work independent of one another, the first and second GUI elements may be activated at the same time if the first and second conditions occur simultaneously (**FIG. 34C**).

[**0171**] Following block **956**, the method proceeds to block **960** where it is determined if the first GUI element should be deactivated. If so, the method proceeds to block **962** where the GUI element is deactivated. For example, as shown in **FIG. 38D**, the first GUI element (scroll wheel **962**) is disabled and removed from display when the finger **925** is no longer detected over the playlist **962**. If not, the method maintains block **956**.

[0172] Similarly but independently, following block **958**, the method proceeds to block **964** where it is determined if the second GUI element should be deactivated. If so, the method proceeds to block **966** where the GUI element is deactivated. For example, as shown in **FIG. 38E**, the second GUI element (control panel **966**) is disabled and removed from display when the finger **925** is no longer detected over the border **968**. If not, the method maintains block **958**.

[0173] It should be noted that the method is not limited to only two GUI elements and that other GUI elements may be activated if other conditions are implemented. For example, a third GUI element may be activated when a third set of condition occurs and so on. By way of example, as shown in **FIG. 38F**, the user may slide their finger **925** from the border **968** to a menu portion **970** of the active window **960** thereby initiating a change from the control panel **966** to a scroll wheel **972** (e.g., while the second GUI element is being deactivated, the third GUI element is being activated).

[0174] Further, as shown in **FIG. 38G**, the user may add another finger **925** to the current touch thereby initiating a change from the first control panel **966** to a second control panel **982**. The first control panel **966** may include a first set of control options such as play, stop, seek and volume options and the second control panel **982** may include a second set of control options such as song playing order, song information, light effect options.

[0175] Moreover, as shown in **FIG. 38H**, the user may place one finger **925**A over the border **968**, another finger **925**B over a menu portion **970**, and yet another finger **925**C over the playlist portion **964** thereby initiating three different GUI elements, particularly, a control panel **966**, a first scroll wheel **972** for scrolling through the menu **970**, and a second scroll wheel **962** for scrolling through the playlist **964**.

[0176] In addition, multiple GUI elements can be activated in the same portion. For example, as shown in **FIGS. 38I and 38J**, if the user selects a particular box **990** in the playlist **964**, a keyboard **992** may be activated so that the user can enter data associated with the song (e.g., title, artist, genre, etc.). If the scroll wheel **962** is active at the same time as the keyboard **992**, the scroll wheel **962** may be minimized to accommodate the keyboard **992** as shown. Once the keyboard **992** is deactivated, the scroll wheel **962** reverts back to its original size.

[0177] The various aspects, embodiments, implementations or features of the invention can be used separately or in any combination.

[0178] The invention is preferably implemented by hardware, software or a combination of hardware and software. The software can also be embodied as computer readable code on a computer readable medium. The computer readable medium is any data storage device that can store data which can thereafter be read by a computer system. Examples of the computer readable medium include read-only memory, random-access memory, CD-ROMs, DVDs, magnetic tape, optical data storage devices, and carrier waves. The computer readable medium can also be distributed over network-coupled computer systems so that the computer readable code is stored and executed in a distributed fashion.

[0179] While this invention has been described in terms of several preferred embodiments, there are alterations, permutations, and equivalents, which fall within the scope of this invention. For example, although the invention has been primarily directed at touchscreens, it should be noted that in some cases touch pads may also be used in place of touchscreens. Other types of touch sensing devices may also be utilized. It should also be noted that there are many alternative ways of implementing the methods and apparatuses of the present invention. It is therefore intended that the following appended claims be interpreted as including all such alterations, permutations, and equivalents as fall within the true spirit and scope of the present invention.

What is claimed is:

1. A user interface method, comprising:

detecting a touch;

determining a user interface mode when a touch is detected;

displaying one or more GUI elements based on the user interface mode; and

enabling the functionality of the GUI element.

2. The user interface method as recited in claim 1 wherein the user interface mode is based on one or more applications.

3. The user interface method as recited in claim 2 wherein different applications indicate different user interface modes.

4. The user interface method as recited in claim 1 wherein the user interface mode is based on a current state of an application.

5. The user interface method as recited in claim 5 wherein different states of the application indicate different user interface modes.

6. The user interface method as recited in claim 1 wherein the user interface mode is based on one or more touch characteristics of the detected touch.

7. The user interface method as recited in claim 6 wherein different characteristics of the touch indicate different modes.

8. The user interface method as recited in claim 1 wherein the user interface mode is based on only one condition, the condition being selected from application, a state of an application, a touch characteristic of the detected touch.

9. The user interface method as recited in claim 1 wherein the user interface mode is based on multiple conditions, the conditions being selected from one or more applications, one or more states of an application, one or more touch characteristics of the detected touch.

10. The user interface method as recited in claim 1 wherein only one GUI element is displayed.

11. The user interface method as recited in claim 1 wherein multiple GUI elements are displayed.

12. The user interface method as recited in claim 1 wherein the user interface mode is a scroll mode, and wherein the GUI element is a virtual scroll wheel.

13. The user interface method as recited in claim 1 wherein the user interface mode is a data entry mode, and wherein the GUI element is a virtual keyboard.

14. The user interface method as recited in claim 1 wherein the user interface mode is an edit mode, and wherein the GUI element is a tool bar or virtual keyboard.

15. The user interface method as recited in claim 1 wherein the user interface mode is a control mode, and wherein the GUI element is a control panel.

16. The user interface method as recited in claim 1 wherein the user interface mode is an information mode, and wherein the GUI element is a window.

17. The user interface method as recited in claim 1 wherein the GUI element is displayed over a previously existing graphical image.

18. The user interface method as recited in claim 17 wherein the GUI element is semitransparent so that the previously existing graphical image disposed underneath the GUI element can be seen through the GUI element.

19. The user interface method as recited in claim 1 wherein the GUI element is configured to displace a previously existing graphical image.

20. The user interface method as recited in claim 1 wherein the GUI element is presented in the vicinity of the detected touch.

21. The user interface method as recited in claim 1 wherein the GUI element is presented at a predetermined location.

22. The user interface method as recited in claim 1 wherein the displaying the GUI element includes implementing a transition effect that transitions the GUI element from a first state to a second state.

23. The user interface method as recited in claim 22 wherein the transitioning effect is selected from popping, growing, or fading in and out.

24. The user interface method as recited in claim 22 wherein the speed of the transition effect is based on the pressure of the touch.

25. The user interface method as recited in claim 1 wherein enabling the GUI element includes monitoring a touch event relative to the GUI element, and performing actions associated with the touch event.

26. The user interface method as recited in claim 1 further comprising:

determining whether or not to deactivate the GUI element, deactivation including disabling the functionality of the GUI element and removing the GUI element from display.

27. The user interface method as recited in claim 26 wherein the GUI element is deactivated when a touch is no longer detected.

28. The user interface method as recited in claim 26 wherein the GUI element is deactivated when a touch has not been detected for a preset amount of time.

29. The user interface method as recited in claim 26 wherein the GUI element is deactivated after being activated for a preset amount of time.

30. The user interface method as recited in claim 26 wherein the GUI element is deactivated via a user selection.

31. The user interface method as recited in claim 1 wherein displaying the GUI element includes implementing a transition effect that transitions the GUI element from a first state to a second state, and wherein the step of removing the displayed GUI element includes implementing a reverse transition effect that transitions the GUI element from the second state to the first state.

32. The user interface method as recited in claim 1 wherein determining the user interface mode comprises:

determining a current application;

determining a current state of the current application; and

determining touch characteristics associated with the detected touch.

33. A method for scrolling through media items, the method comprising:

displaying a plurality of media items;

detecting a touch over at least one of the media items that are displayed;

activating a virtual scroll wheel when a touch is detected over the displayed media items;

determining if a touch event is performed relative to the touch region of the virtual scroll wheel; and

scrolling through the group of media items when a scrolling touch event is performed.

34. The method as recited in claim 33 wherein activating the virtual scroll wheel includes displaying and enabling the functionality of the virtual scroll wheel, the virtual scroll wheel providing a touch region where a user swirls their finger in order to traverse through the group of media items

35. The method as recited in claim 34 further comprising:

deactivating the virtual scroll wheel when a touch event is not performed, the deactivation of the virtual scroll wheel including disabling the functionality of the virtual scroll wheel and removing the virtual scroll wheel from display.

36. The method as recited in claim 35 wherein the display and the removal of the virtual scroll wheel from display are performed with a transition effect that transitions the virtual scroll wheel from a first state to a second state when displayed, and that transitions the virtual scroll wheel from the second state to the first state when removed.

37. The method as recited in claim 34 wherein the virtual scroll wheel is displayed with a transition effect that transitions the virtual scroll wheel from a first state to a second state.

38. The method as recited in claim 34 wherein the transition effect causes the virtual scroll wheel to grow, a small virtual scroll wheel being initially displayed, and thereafter the virtual scroll wheel continuously enlarging until the virtual scroll wheel reaches a final size.

39. The method as recited in claim 34 wherein the virtual scroll wheel is displayed over the plurality of media items, and wherein the virtual scroll wheel is semitransparent so that the plurality of media items disposed underneath the virtual scroll wheel can be seen through the virtual scroll wheel.

40. The method as recited in claim 33 wherein scrolling includes moving a selector bar linearly through the group of media items in accordance with the touch event occurring on the virtual scroll wheel.

41. The method as recited in claim 33 wherein the virtual scroll wheel provides one or more virtual buttons, and wherein the method further comprises:

determining if a selection touch event is performed relative to the one or more virtual buttons of the virtual scroll wheel; and

implementing an action associated with a particular button when the selection touch event is performed on the particular button.

42. The method as recited in claim 41 wherein the one or more virtual buttons include at least a center button that is surrounded by the touch region.

43. The method as recited in claim 42 wherein the virtual scroll wheel is circular.

44. The method as recited in claim 33 wherein the step of determining if a touch event is performed relative to the touch region of the virtual scroll wheel comprises:

detecting the presence of at least one finger over the touch region;

setting an initial position of the finger; and

monitoring finger movement relative to the initial position.

45. The method as recited in claim 33 wherein the media items are a list of songs.

46. A method performed on a user operated electronic device having a display and a touch screen, the method comprising:

determining if a touch is detected:

monitoring and analyzing the current operating conditions when a touch is detected;

activating a first GUI element for a first set of operating conditions; and

activating a second GUI element for a second set of operating conditions.

47. A method performed on a computing device having a display and a touch sensing input device, the method comprising:

sensing touches;

displaying and enabling a GUI element when a touch is detected, the GUI element being based on at least one of the following:

(a) the application currently running on the computing device;

(b) the current state of the application;

(c) one or more characteristics of the touch, the characteristics including,

(i) touch location,

(ii) touch ID

(iii) number of touches,

(iv) touch motion

disabling and removing the GUI element from display when one of the following events occurs,

(a) the touch is no longer detected,

(b) a touch has not been detected for a preset amount of time,

(c) a certain amount of time has gone by since the step of displaying and enabling,

(d) a user selection.

48. A computing system, comprising:

a display device configured to display a graphical user interface;

a touch screen positioned over the display device, the touch screen being configured to detect touches that occur over the display device; and

a processor operatively coupled to the display device and the touch screen, the processor instructing the display device to display one or more GUI elements in response to a touch, and performing actions associated with the GUI element when touch events are detected relative to the displayed GUI elements.

49. The system as recited in claim 42 wherein the processor is configured to determine a user interface mode in response to the touch, the user interface mode having one or more GUI elements associated therewith that are capable of being displayed, the user interface modes being based on at least one of a current application, a state of the application and a touch characteristics associated with the touches.

50. A computing device, comprising:

a processor;

a touch screen capable of sensing touch events; and

a display configured to simultaneously display a plurality of media items and a virtual scroll wheel, the virtual scroll wheel providing a region where touch events are performed in order to implement a scrolling action, the scrolling action allowing a user to traverse through the plurality of media items.

* * * * *