(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
2 October 2014 (02.10.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/158221 A1

(51) International Patent Classification:
G06F 3/0482 (2013.01) *G06F 3/0485* (2013.01)
G06F 9/44 (2006.01) *G06F 3/0488* (2013.01)
G06F 3/0483 (2013.01) *G06F 3/0481* (2013.01)

(21) International Application Number:
PCT/US2013/059566

(22) International Filing Date:
13 September 2013 (13.09.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/853,957 29 March 2013 (29.03.2013) US

(71) Applicant: **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).

(72) Inventors: **GARSIDE, Adrian**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **GALLANT, Allison**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **ZAMAN, Nazia**; c/o Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

[Continued on next page]

(54) Title: START AND APPLICATION NAVIGATION

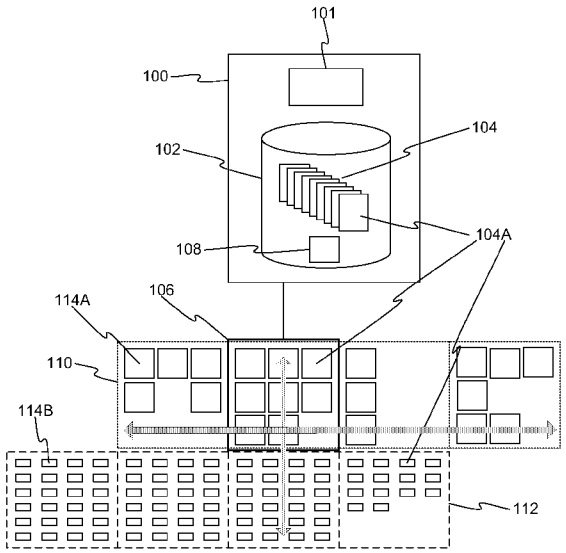

FIG. 1

(57) Abstract: Described herein are techniques to facilitate efficient application navigation by a user. In one embodiment, two or more scrollable surfaces contain application icons. One surface may have a first type of application icon and may be a user-customized surface akin to a desktop. Usually, only one of the surfaces is active and displayed at a given time. The user can swap one surface into view and the other surface out of view. The active surface is independently scrollable; when a given surface is swapped out and then swapped back in, despite any scrolling of the intervening surface, when the given surface is swapped back into view it returns at the same position it had before being swapped out. Another embodiment may provide optimizations to enhance performance and responsiveness by pre-rendering imagery of an off-screen surface. Caching logic may be used to assure that the imagery is correct.

WO 2014/158221 A1

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *with international search report (Art. 21(3))*

START AND APPLICATION NAVIGATION

BACKGROUND

[0001] It has long been known how to provide user interfaces for navigating and launching applications available on a computing device. In particular, some implementations have used scrollable desktops or user shells with icons for invoking applications. Recently, these types of user interfaces have become more complex and may handle an increasing number of applications. The number of icons or tiles representing respective applications may become cumbersome. A desktop, start screen, application management user interface, or functional equivalents, may allow a user to manage burgeoning application icons by manually configuring a desktop or the like. A user may choose to include only some icons that are frequently used. However, this may leave the user with only inefficient ways to access infrequently used applications that are not included in the application navigator or launcher, start screen, graphical user shell, etc.

[0002] Techniques related to efficient navigation between scrollable application surfaces are discussed below.

SUMMARY

[0003] The following summary is included only to introduce some concepts discussed in the Detailed Description below. This summary is not comprehensive and is not intended to delineate the scope of the claimed subject matter, which is set forth by the claims presented at the end.

[0004] Described herein are techniques to facilitate efficient application navigation by a user. In one embodiment, two or more scrollable surfaces contain application icons. One surface may have a first type of application icon and may be a user-customized surface akin to a desktop other user interfaces for application control and management (icon type differences may be visual, functional, or otherwise, as discussed further below). Usually, only one of the surfaces is active and displayed at a given time. The user can swap one surface into view and the other surface out of view. The active surface may be independently scrollable; when a given surface is swapped out and then swapped back in, despite any scrolling of the intervening surface, when the given surface is swapped back into view it returns at the same position it had before being swapped out. Another embodiment may provide optimizations to enhance performance and responsiveness by

pre-rendering imagery of an off-screen surface. Caching logic may be used to assure that the imagery is correct.

[0005] Many of the attendant features will be explained below with reference to the following detailed description considered in connection with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] The present description will be better understood from the following detailed description read in light of the accompanying drawings, wherein like reference numerals are used to designate like parts in the accompanying description.

[0007] Figure 1 shows a multi-surface user interface.

[0008] Figure 2 shows a sequence of a user alternating between surfaces.

[0009] Figure 3 shows a process for handling scrolling and surface-swapping.

[0010] Figure 4 shows detail of a surface swap transition.

[0011] Figure 5 shows scrolling a surface after a swap.

[0012] Figure 6 shows an example of how surfaces can pan independently.

[0013] Figure 7 shows an embodiment with image caching to enhance appearance during a surface swap.

[0014] Figure 8 shows a process for using cached image data to enhance swapping performance.

[0015] Figure 9 shows an example of a computing device.

DETAILED DESCRIPTION

[0016] Embodiments discussed below relate to efficient navigation between independently scrollable application surfaces. A general system with two independently scrollable surfaces will be described first. Details of how to alternate between the two independently scrollable surfaces will be discussed next, followed by discussion of optimizations to enhance performance and provide a fluid user experience.

[0017] Figure 1 shows a multi-surface user interface. A computing device 100 has a processor 101, storage 102 storing applications 104 installed on the computing device, and a display 106, which may integrated with or detached from the computing device. The computing device also has one or more input devices, such as a touch-sensitive or pressure-sensitive surface, a microphone, a mouse or equivalent, a camera-based motion recognition system, and so forth. A navigation application 108 (e.g., a user-shell, a window manager, one of the applications 104, etc.) is also installed. The navigation application 108 provides a user interface (UI) with user interface elements shown in the figure below the computing device 100.

[0018] The user interface may include, among other things, a first pannable or scrollable surface 110 and a second pannable or scrollable surface 112. For ease of discussion, only two surfaces are discussed herein, but the embodiments described below are trivially extensible to more than two surfaces; any embodiment or claim herein is considered to implicitly refer to two or more surfaces. Also, where the display 106 is described in relation to user interfaces, a sub-area of the display, such as a view, a window, a display area, etc., will be equally applicable. In other words, mention of display 106, according to context, also implicitly describes discrete managed sub-areas of the display.

[0019] The surfaces 110, 112, each have icons 114A, 114B representing respective applications 104. The icons 114A, 114B may also be referred to as graphical application representations, tiles, etc. Note that there might be more than one type of graphical element in one view or surface. In one embodiment, the icons 114A, 114B are merely interactive user interface elements that can be activated or selected by a user to launch corresponding applications ("launch" meaning bring to the fore and possibly also initiate execution). In other embodiments, either or both of the icons 114A, 114B have additional features for managing the icons, for managing the applications, or other related to the applications or icons. For example, an icon, may be interacted with to activate a menu with items such as "uninstall", "resize", "move", "pin", "unpin", "open location", "run as administrator", "restart", and so on. An icon in both surfaces may represent a same application, as indicated for example application 104A. Icons in both surfaces may also tie into a same underlying application management framework.

[0020] In one embodiment, the icons 114A in first surface 110 have different traits than the icons 114B in second surface 112. For example, the icons 114A in the first surface 110 may be dynamic tiles or icons that over time display updated content from the corresponding application. The icons 114B in the second surface 112 may be static tiles or icons that are smaller and do not display dynamic content, although, as noted above, in some embodiments there is no restriction on which types of tiles or icons can be added to which surfaces. In yet another embodiment the first surface 110 is configured manually by the user, for example by pinning applications to the first surface 110, removing icons, turning dynamic content on and off, etc., whereas the icons 114B in the second surface 112 represent "all" of the applications 104 or are icons that are automatically added whenever a new application is installed. Either or both surfaces may be equipped with functionality such as view filtering, view sorting, searching, and others. Any combination

of icon types and features, and any kind of surface behavior may be used in either or both of the two surfaces 110, 112.

[0021] Regarding scrolling or panning, at any given time when not transitioning between surfaces, one of the surfaces is currently active and displayed by the display 106 (or, as noted above, a display sub-area such as a programmatic view, a window, etc.) In the example shown in Figure 1 the first surface 110 is the active or current surface. The active surface may be panned or scrolled either incrementally between discrete sections indicated by the vertical dashed lines, or smoothly at any position between the ends of the surface. Typically, a user will provide input to scroll the surface. In the case of touch input or the like, the navigation application 108 handles the user input by scrolling the active surface, causing different of the icons 114A, 114B to scroll into and out of view on the display 106. As will be described next, the user may also provide inputs that are interpreted as indicating an intent to display an inactive or off-screen surface.

[0022] Figure 2 shows a sequence of a user alternating between surfaces. Starting chronologically at the top of Figure 2, initially, the first surface 110 is displayed in the display 106 and can be scrolled (either surface may be the default or initial view). The second surface 112 is not displayed. The user provides an input such as a stroke to the left, a keypress, a voice command, etc., which is interpreted as a "scroll left" command 130. In the second frame of Figure 2 the active first surface has scrolled left. Another user input is received, for example a stroke upward. This input is interpreted as a command 132 to make off-screen second surface the active surface. Consequently, as shown in the third frame, the second surface is now the active surface and available for scrolling input. Yet another input is received, for example a designated keypress or dragging of a scrollbar, and another scroll left command 134 is invoked, this time scrolling the second surface, as shown in the last frame of Figure 2.

[0023] Figure 3 shows a process for handling scrolling and surface-swapping. Initially, the navigation application 108 or other program awaits input at step 150. When an input is received, step 152 determines if the input is for scrolling or for swamping in a new active surface. If the input is for scrolling, then at step 152 the active surface is scrolled according to the input. If the input indicates that a surface is to be swapped into the view or display, then at step 154 a transition process begins. A transition, described in detail further below, may provide the illusion that the two swapping surfaces are conjoint. For example, a transition may show the current surface moving out of view as a surface targeted by the input moves into view. This might involve showing a leading edge of the

target surface (such as a frame or header, and/or a background image) adjacent to a departing edge of the active surface as the new edge peeks into the view.

[0024] At step 156, if the input is of a continuous type (e.g., holding a key, a movement or contact of a finger, a mouse drag, etc.), then when the input ends there a determination if a swap condition has been met. For example, a threshold distance might be reached. In another example, a threshold speed is detected regardless of whether input has ended. If the swap condition is not met then at step 158 the transition is visually reversed (if applicable) and process resumes at step 150 to await further input. If the swap condition is met then at step 160 the transition may complete automatically, resulting in the departing surface no longer being displayed and the target surface occupying the display or view and available to be scrolled.

[0025] Figure 4 shows detail of a surface swap transition. In this example, the lower surface is an "applications" surface that has static application icons and the upper surface is a similar type of surface that might have dynamic application icons, application representations, tiles, etc. It should be noted that either or both surfaces may be heterogeneous or homogenous with respect to the types of application representations contained therein. Responsive to a user input, the lower surface begins to peek up from the bottom of the display 106. If the lower surface has a static element such as a header 170, then the header begins to appear at the bottom of the display as the upper surface begins to move up the display. When the surfaces have moved the threshold distance 172 then the swap may automatically complete; the surfaces are automatically moved upward until the applications surface is fully displayed. Figure 5 shows scrolling a surface after a swap. Continuing from the example of Figure 4, after the applications surface occupies the display 106 (top of Figure 5) the user inputs a pan command and the applications surface pans in the display. The desktop surface remains in memory so that when desired it can be swapped back to the display; the same region that was in view before the swap may again be shown by the display, despite the panning of the applications surface.

[0026] Figure 6 shows an example of how surfaces pan independently. Initially at step 190 area A2 is displayed. Responsive to a user input the lower surface swaps into view at area B1. At step 192 a user input is received that causes the lower surface to pan to the left which brings section B2 into view. At step 194 another user input pulls area A2 of the upper surface into view and pushes area B2 of the active surface down the display and out of view. At step 196 the upper surface is again available for panning. In some

embodiments, including those described next, the surfaces may scroll in unison rather than independently.

[0027] Figure 7 shows an embodiment with image caching to enhance appearance during a surface swap. Some computing devices and program environments such as a managed code environment may not render a complex user interface (such as a pannable surface) with sufficient speed to be responsive to a swap. Program logic 210 may be implemented to enhance speed and reduce memory use.

[0028] Program logic 210 performs smart image caching. The program logic 210 performs two processes 212 and 214 to use and maintain data structures 216, 218 and image data 220, 222. The data structures 216, 218 track user interface elements of a surface. Briefly, the program logic 210 uses the data structures 216, 218 to render image data of surfaces and their content. The pre-rendered image data 220 222 facilitates rapid display of imagery of the surfaces when a swap begins. The actions of the program logic and how it optimizes performance are described next.

[0029] At an opportune time, for instance shortly after the navigation application 108 begins executing, the program logic 210 parses through the surfaces to identify the user interface elements thereof that are painted to the display when the surfaces are to be displayed. Such user interface elements may be frames and headers 224, search tools, hot buttons, surface backgrounds 226, and icons 114A, 114B, to name a few examples. The program logic 210 accumulates the relevant user interface elements into the data structures 216, 218. The data structures may be a collection, hierarchy, or other arrangement of references 228 to the user interface elements. The program logic 210 uses the data structures 216, 218 to pre-render image data mirroring appearance the surfaces would have if displayed by the graphics environment. The image data may be divided into parts so that there is separate image data for static elements such as headers and separate image data for dynamic elements such as application icons.

[0030] The program logic 210 may detect updates that affect dynamic content of a surface. For example, event handlers may be attached to objects that implement the surfaces. The program logic 210 may also receive notifications when application icons are rearranged, added, removed, etc. In effect, any event that can change the displayed appearance of a surface is detected. When such an event is detected the program logic 210 may perform various actions. For example, the data structures 216, 218 may be updated, image data might be re-rendered, and/or image data might be marked as stale. If appearance-affecting events occur frequently it may be most efficient to simply mark

image data as stale and re-render when needed. In other cases it may be preferable to re-render the image data every time surface appearance changes. Moreover, the appearance tracking may isolate only the portion of a surface that is active or would be active if a swap occurs. Operation of the program logic 210 for swapping is discussed next.

[0031] Figure 8 shows a process for using cached image data to enhance swapping performance. At step 240 an input initiates swapping, as previously described. At step 242 pre-rendered static image data such as an image of a surface header is retrieved and begins to be displayed immediately. That is, pre-rendered imagery of the target surface begins to be displayed at step 244 as emerging from whichever edge of the display 106 is indicated by the user input. At step 246 the process checks to see if image data for the dynamic surface elements is available and/or not marked stale. If the image data is not available or is stale then the image data is rendered from the data structures at step 248. At step 250 the image data of the target surface begins to be displayed. If the image data had to be newly rendered at step 248 and is not yet fully rendered then as the target surface pans into view the rendered (or partly rendered) dynamic image data may be faded into view as it moves and as the image data becomes available. Although the use of pre-rendered image data for the target surface has been discussed, pre-rendered image data for the active departing surface can similarly be used.

[0032] While the pre-rendered image data is being used an active display instance of the target surface may be being prepared. At step 252, optionally, the pre-rendered image data is replaced by display of the interactive target surface. This step may occur while transitioning or when transitioning is complete. Because the interactive target surface and the pre-rendered image data have the same appearance and the same (or nearly) location when switched, the use might not perceive that a static image of the target surface has been replaced by the interactive surface itself. In an implementation where such a switch occurs before the transition is complete, it might be the case that the pre-rendered image data (representing the emerging surface) has not reached a threshold distance, speed, or other condition that indicates that a swap has been triggered. If necessary, at step 256 any pre-rendered image data of the active (and returning to normal position) surface is switched out for the surface itself. Assuming that the surface swap is to be completed, then at step 258, upon or near completion of the swap (when the target surface or image data thereof occupies the display 106), the image data of the target surface is replaced with a display of the now-active target surface.

[0033] In one implementation, the thresholds mentioned earlier may act as a mandatory swap point. When such a threshold is reached automatic completion of the swap takes over regardless of the user input.

[0034] Among the various mechanisms for a user to trigger a surface swap, a self-revealing gesture may also be used. When a particular pre-defined user input occurs, such as a click on negative space of a surface (e.g., non-functional background space) may cause a user interface element to be displayed. When the user interface element is activated by the user the swap process begins and completes. Display of this self-revealing gesture may be conditioned on detecting user activity that indicates the user is not aware of a hidden surface, such as a fixed number of logins for a user without accessing the hidden surface. Also, as previously mentioned, panning of either a surface swap or a surface itself may be "on rails", for example, restricted to horizontal or vertical movement relative to the display.

[0035] Figure 9 shows an example of a computing device 300. The computing device 300 may have a display 106, as well as storage 302 and a processor 304. These elements may cooperate in ways well understood in the art of computing. In addition, input devices 306 may be integrated with or in communication with the computing device 300. The display 106 may be a touch-sensitive display that also functions as an input device. The computing device 300 may have any form factor or be used in any type of encompassing device. For example, touch-sensitive control panels are often used to control appliances, robots, and other machines. The computing device 300 may be in the form of a handheld device such as a smartphone, a tablet computer, a gaming device, a server, or others.

[0036] Embodiments and features discussed above can be realized in the form of information stored in volatile or non-volatile computer or device readable media (which does not include signals or energy per se). This is deemed to include at least media such as optical storage (e.g., compact-disk read-only memory (CD-ROM)), magnetic media, flash read-only memory (ROM), or any means of storing digital information in a physical device or media. The stored information can be in the form of machine executable instructions (e.g., compiled executable binary code), source code, bytecode, or any other information that can be used to enable or configure computing devices to perform the various embodiments discussed above. This is also deemed to include at least volatile memory (but not signals per se) such as random-access memory (RAM) and/or virtual memory storing information such as central processing unit (CPU) instructions during execution of a program carrying out an embodiment, as well as non-volatile media storing

information that allows a program or executable to be loaded and executed. The embodiments and features can be performed on any type of computing device discussed above.

CLAIMS

1. A method performed by a computing device comprising storage, a display, and a processor, the method for a user to selectively alternate between display a first surface and displaying a second surface, the method comprising:

displaying the first surface in a view area displayed by the display, the first surface comprising first graphic application representations respectively representing a first set of respective applications installed on the computing device, wherein the first surface is able to be interactively scrolled by the user; and

receiving input indicating that the second surface is to be activated, and in response displaying the first surface and/or an image thereof moving out of the view while displaying the second surface and/or an image thereof moving into the view, wherein the second surface comprises graphic application representations respectively representing a second set of respective applications installed on the computing device, wherein the second surface is able to be interactively scrolled by the user, wherein the second set of applications is a superset of the first set of applications, and wherein the first application representations are different type of representation than the second application representations.

2. A method according to claim 1, wherein the first application representations comprise dynamic tiles that dynamically display information according to execution of the respective applications, wherein when a first application representation is selected by the user the corresponding application is launched.

3. A method according to claim 2, wherein the second application representations comprises static icons that do not dynamically display information, wherein when a static icon is selected by the user the corresponding application is launched.

4. A method according to claim 3, wherein the displaying the second surface and/or the image thereof comprises displaying the first surface and/or image thereof concurrently with and adjacent to the second surface and/or image thereof.

5. A method according to claim 4, further comprising determining if a threshold distance or speed has been reached by the concurrent movement, wherein when the threshold is reached the concurrent movement continues until the second surface and/or image thereof is no longer displayed in the view, and wherein when the threshold is not reached when the single input ends the second surface and/or image thereof is moved out of the view.

6. A method according to claim 3, wherein the surfaces scroll independently such that scrolling a surface in the view does not change a given scroll position that the other surface had when the other surface was moved out of the view, and when the other surface is then moved into the view the other surface is displayed at the given scroll position.

7. A method of allowing a user to launch applications and navigate between a first and second app-launching pannable surfaces displayed on a display of a computing device, the method comprising:

allowing the user to scroll whichever app-launching pannable surface is currently displayed only in first opposing directions relative to the display, wherein panning the current app-launching pannable surface causes app-launching elements of the current surface to pan into and out of the display area, and the panning does not pan the other surface which is not displayed on the display; and

caching image data representing an appearance one of the app-launching pannable surfaces would have if displayed in the display area, wherein the image data corresponds to an appearance that the one of the app-launching pannable surfaces would have if displayed in the display area; and

responding to user input by using the cached image data to move the one of the app-launching pannable surfacesinto the display area while moving the other app-launching pannable surface out of the display area.

8. A method according to claim 7, wherein the caching comprises pre-rendering at least part of the one of the app-launching pannable surfaces to generate the image data and monitoring for changes affecting appearance of the one of the app-launching pannable surfaces.

9. A method according to claim 7, wherein the computing device comprises a touch-sensitive surface, the app-launching pannable surfaces are panned by strokes substantially in the first opposing directions, and the user input comprises a stroke substantially perpendicular to the first opposing directions, wherein the moving the pannable surfaces comprises moving the pannable surfaces and/or image data thereof only in a direction perpendicular to the first opposing directions.

10. A method according to claim 7, wherein the computing device comprises a keyboard and mouse, the app-launching pannable surfaces are panned by inputs inputted by the keyboard, the mouse, or both.

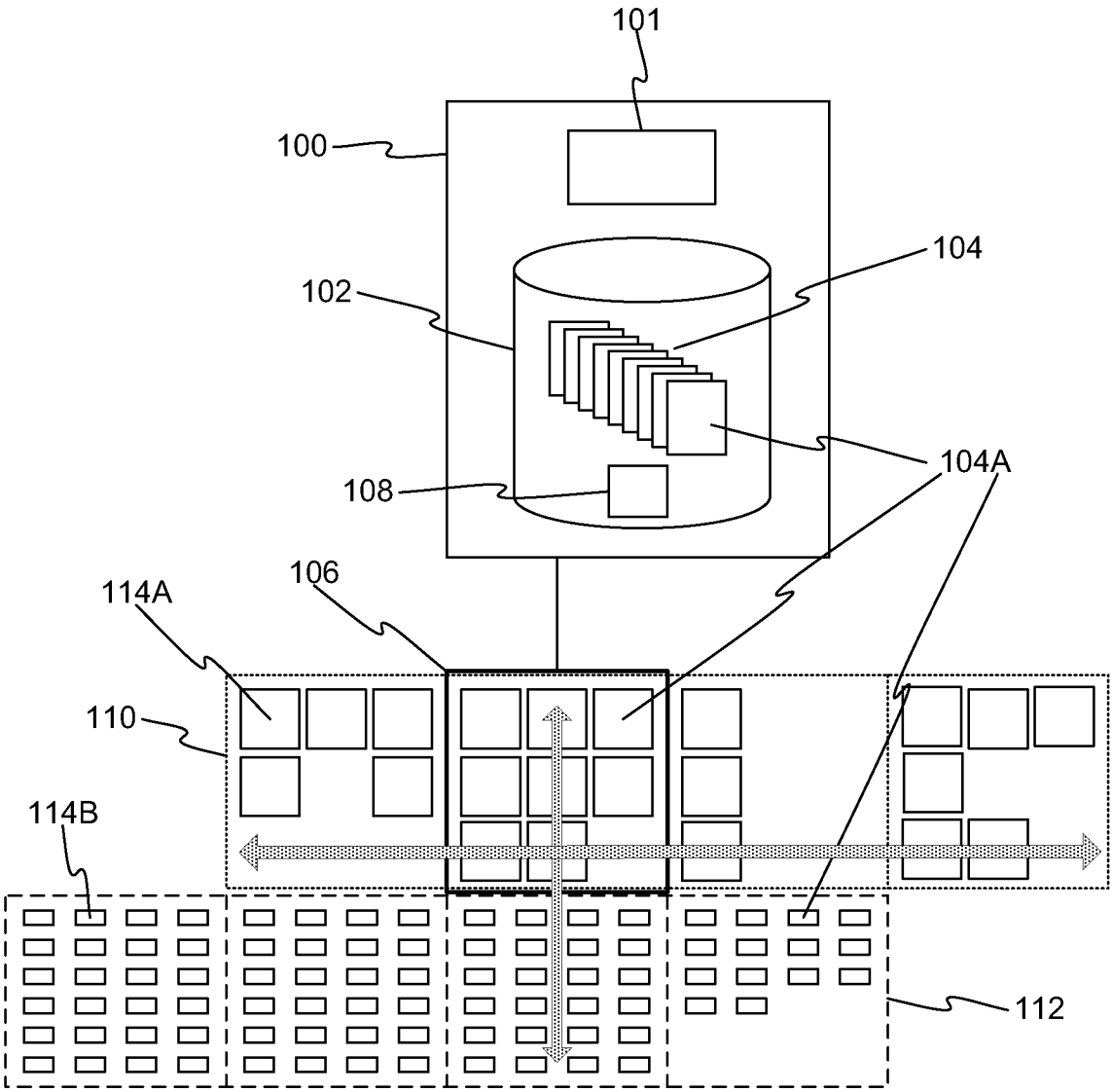

FIG. 1

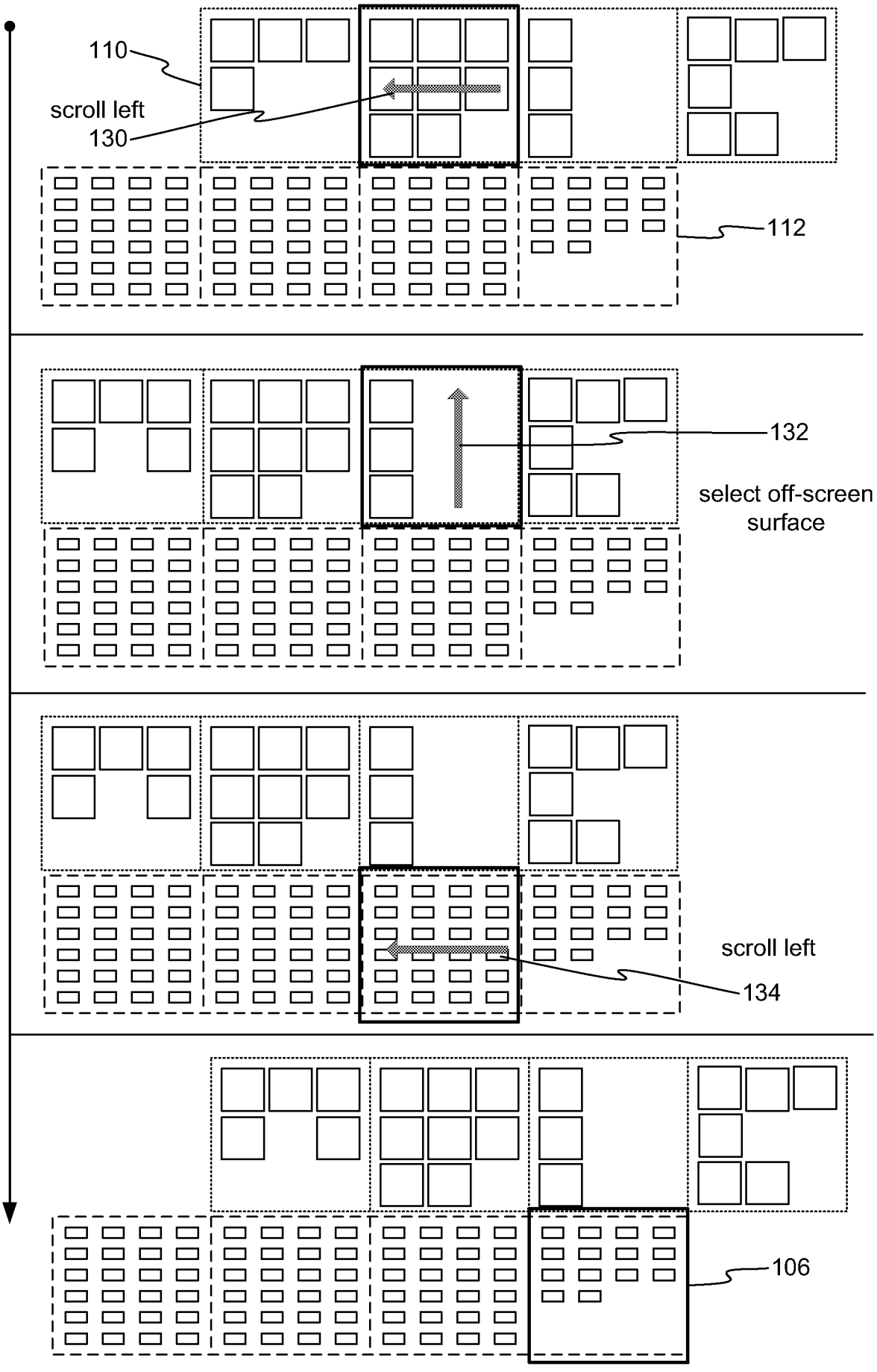
110
scroll left
130
112
132
select off-screen surface
scroll left
134
106

FIG. 2

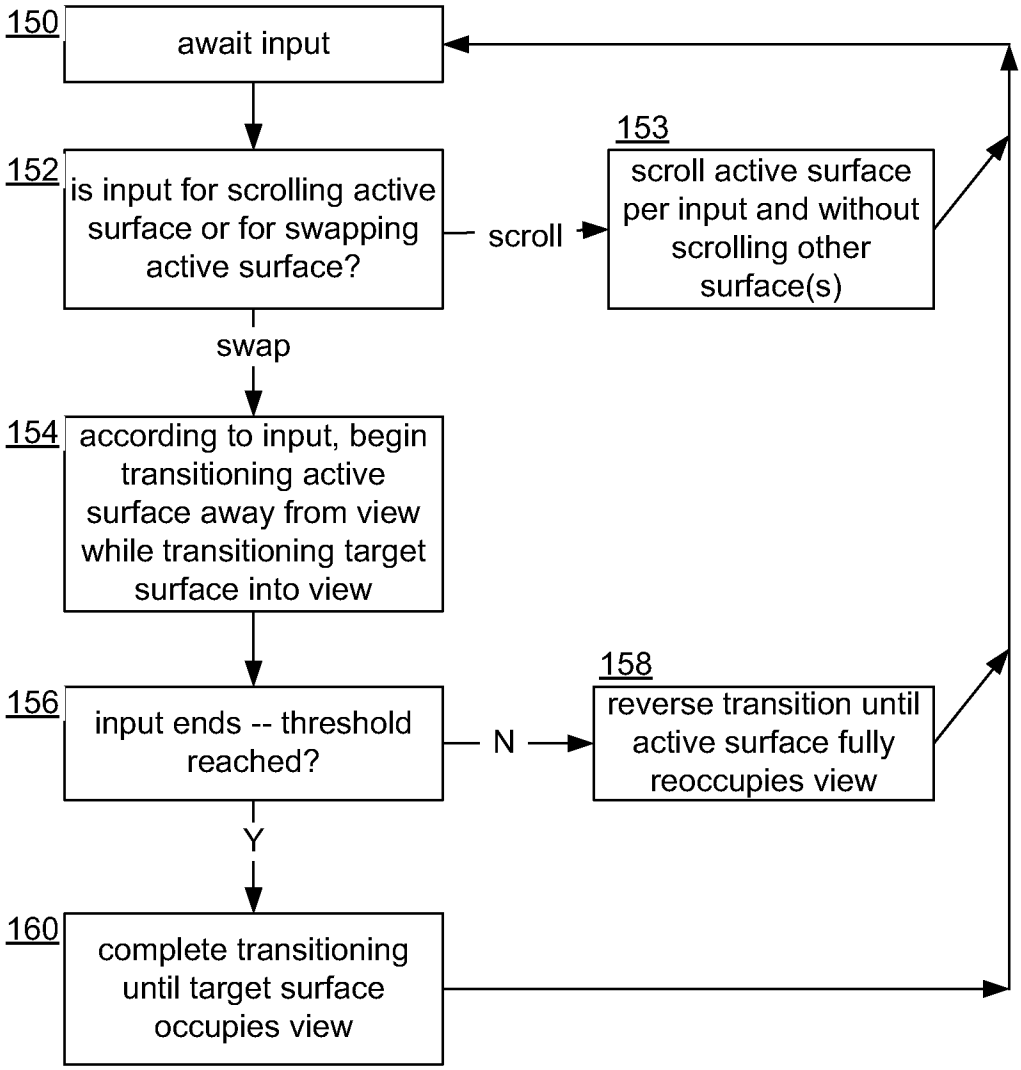
150
await input
152
is input for scrolling active surface or for swapping active surface?
scroll
153
scroll active surface per input and without scrolling other surface(s)
swap
154
according to input, begin transitioning active surface away from view while transitioning target surface into view
156
input ends -- threshold reached?
N
158
reverse transition until active surface fully reoccupies view
Y
160
complete transitioning until target surface occupies view

FIG. 3

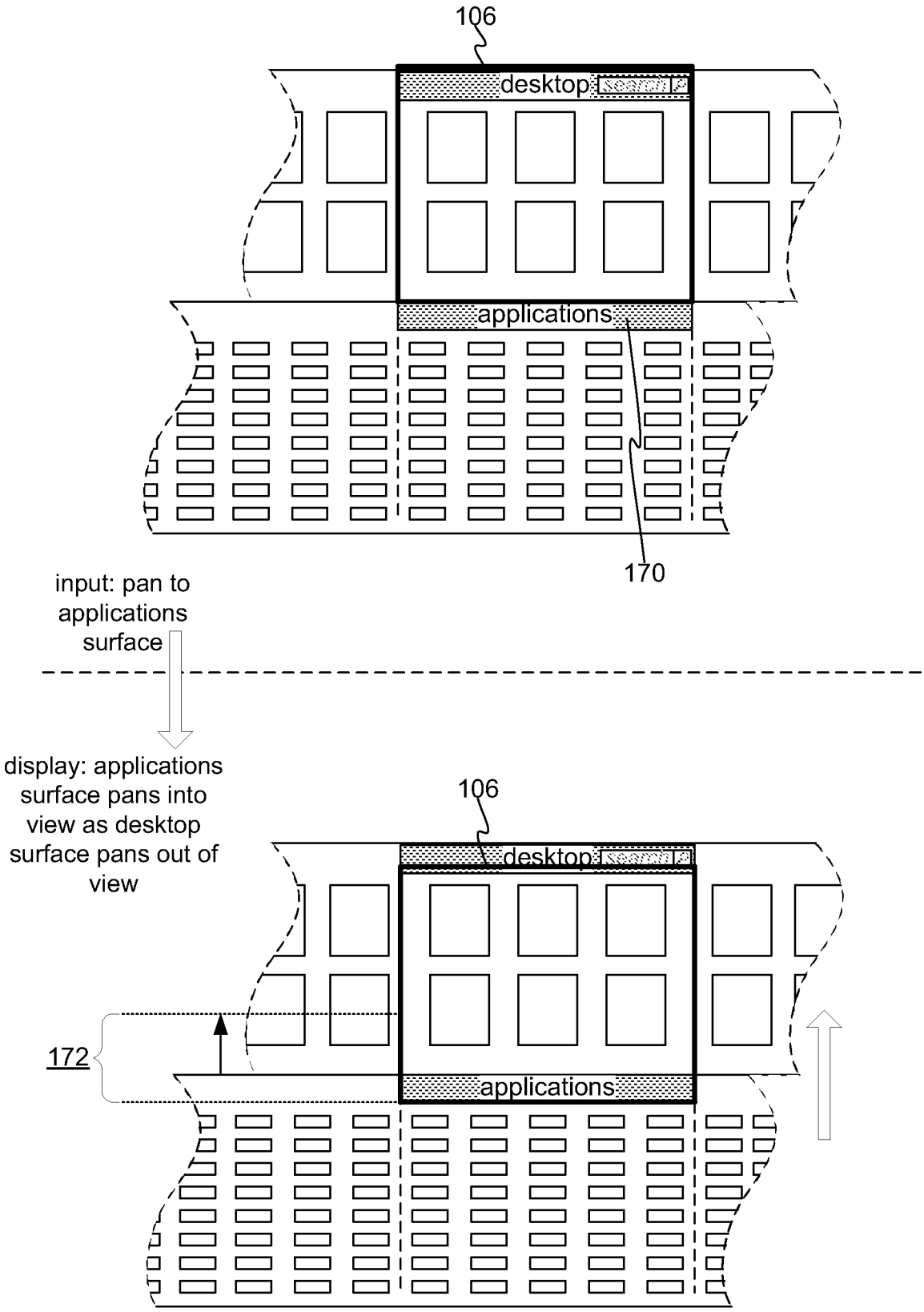
106
desktop
applications
170
input: pan to applications surface
display: applications surface pans into view as desktop surface pans out of view
106
desktop
172
applications

FIG. 4

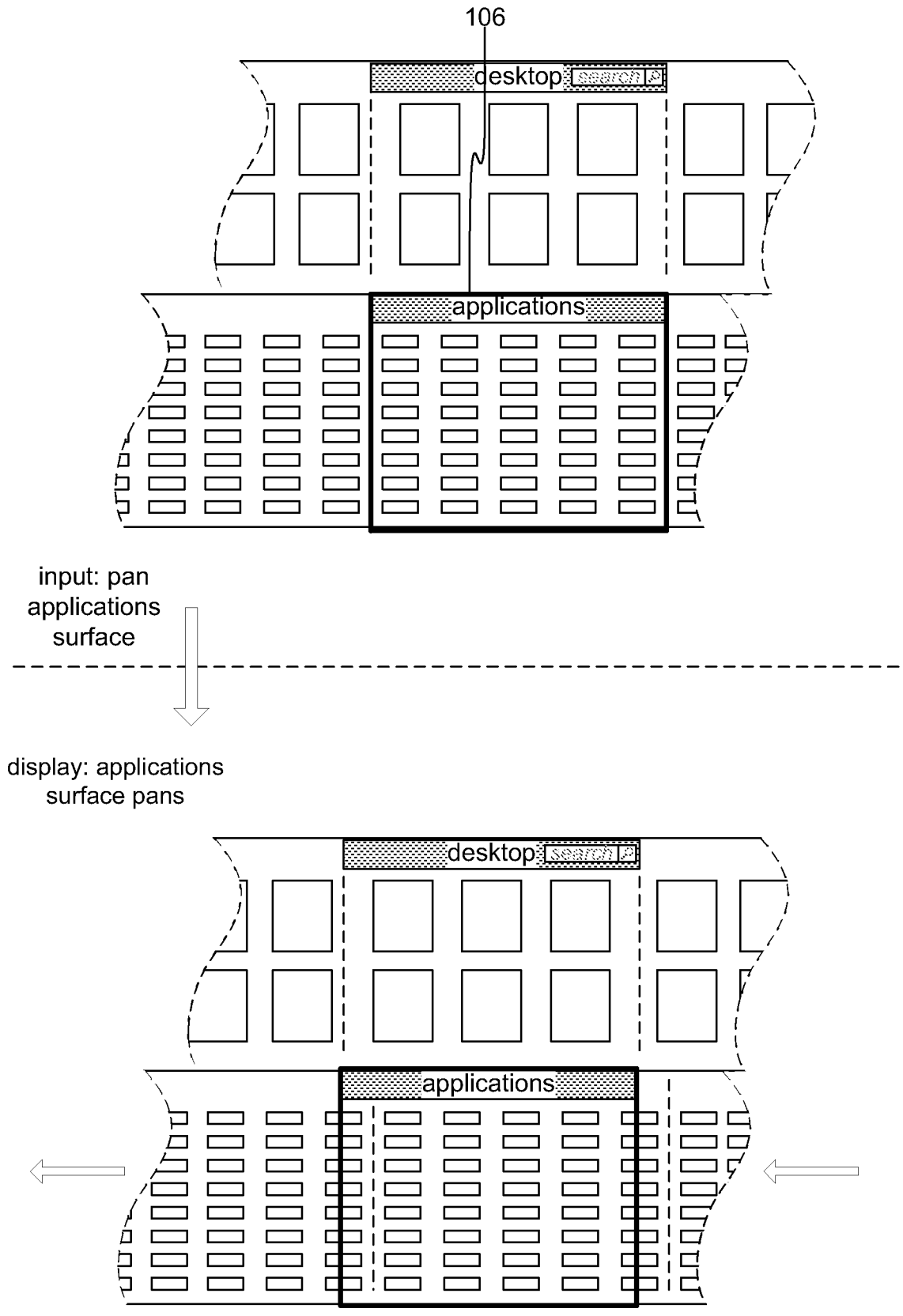
106
desktop
applications
input: pan
applications
surface
display: applications
surface pans
desktop
applications

FIG. 5

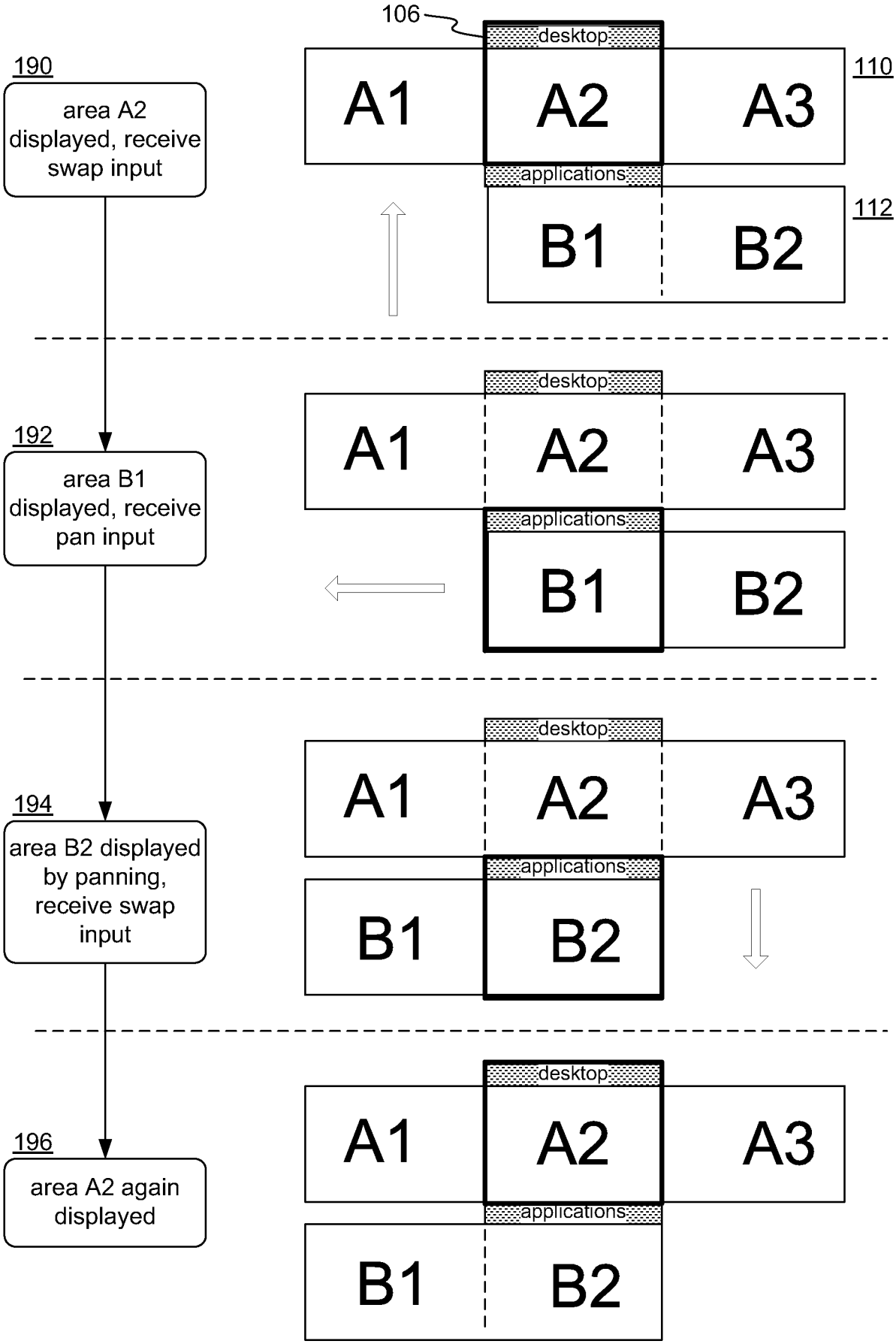
106
190
area A2 displayed, receive swap input
192
area B1 displayed, receive pan input
194
area B2 displayed by panning, receive swap input
196
area A2 again displayed
desktop
A1
A2
A3
110
applications
B1
B2
112

FIG. 6

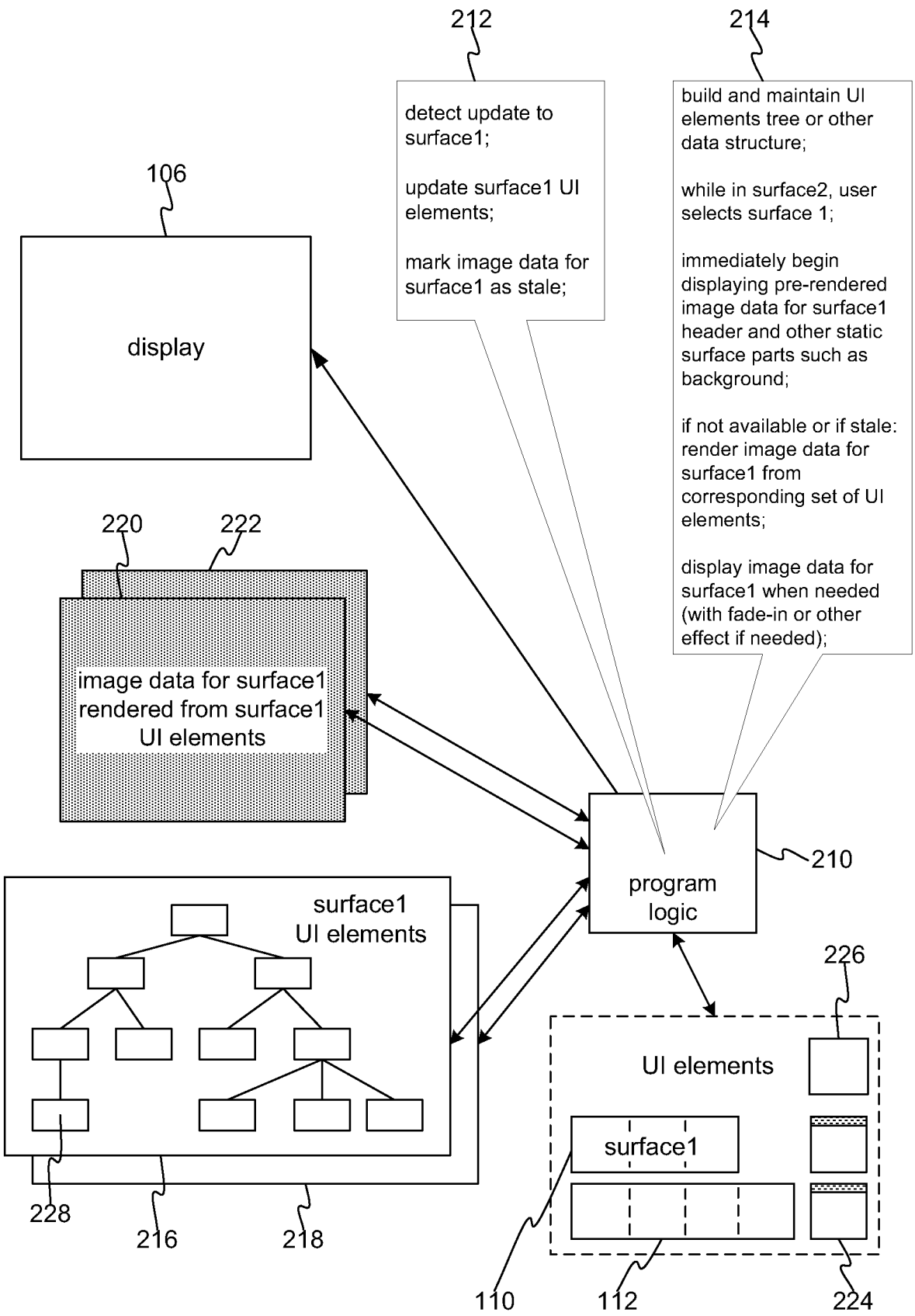
212
214
detect update to surface1;
update surface1 UI elements;
mark image data for surface1 as stale;
build and maintain UI elements tree or other data structure;
while in surface2, user selects surface 1;
immediately begin displaying pre-rendered image data for surface1 header and other static surface parts such as background;
if not available or if stale: render image data for surface1 from corresponding set of UI elements;
display image data for surface1 when needed (with fade-in or other effect if needed);
106
display
220
222
image data for surface1 rendered from surface1 UI elements
210
program logic
surface1 UI elements
226
UI elements
surface1
228
216
218
110
112
224

FIG. 7

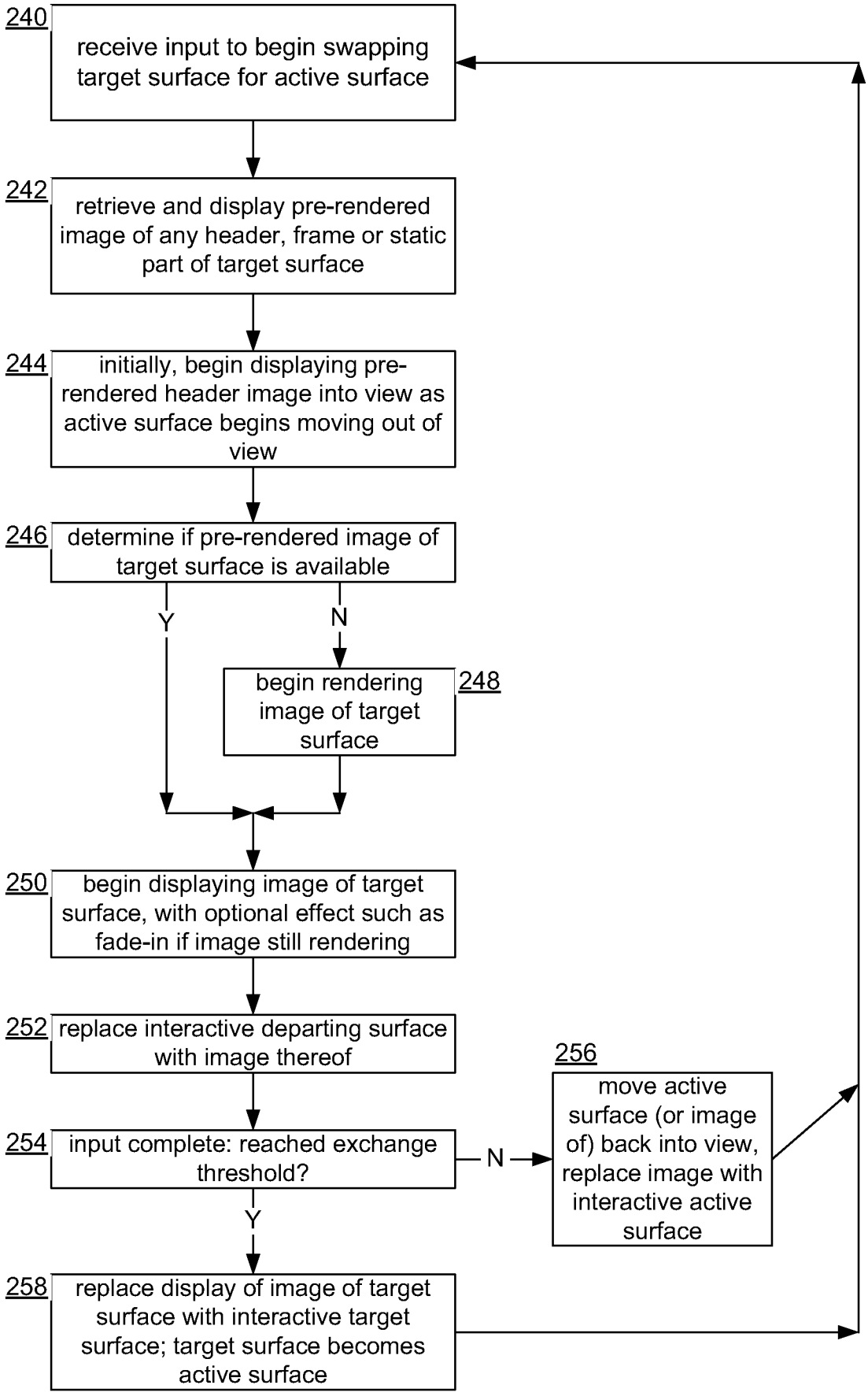
240
receive input to begin swapping target surface for active surface
242
retrieve and display pre-rendered image of any header, frame or static part of target surface
244
initially, begin displaying pre-rendered header image into view as active surface begins moving out of view
246
determine if pre-rendered image of target surface is available
Y
N
begin rendering image of target surface
248
250
begin displaying image of target surface, with optional effect such as fade-in if image still rendering
252
replace interactive departing surface with image thereof
254
input complete: reached exchange threshold?
N
256
move active surface (or image of) back into view, replace image with interactive active surface
Y
258
replace display of image of target surface with interactive target surface; target surface becomes active surface

FIG. 8

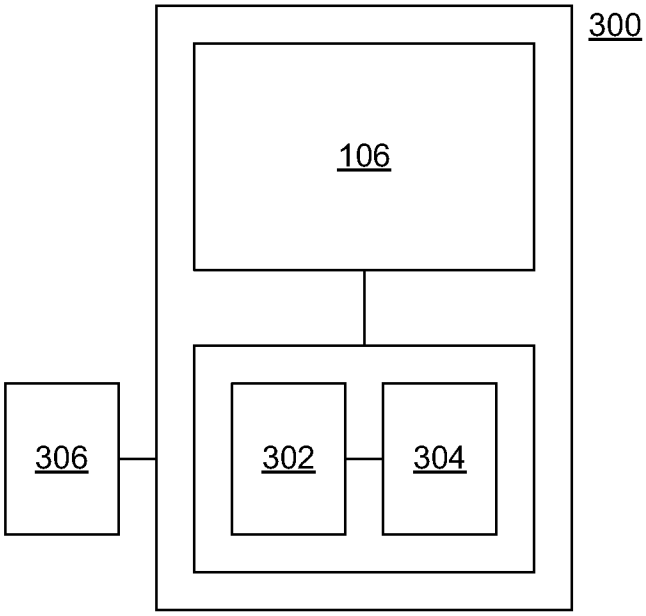

FIG. 9

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2013/059566

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F3/0482 G06F9/44 G06F3/0483 G06F3/0485 G06F3/0488
G06F3/0481
ADD.
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	EP 2 503 444 A2 (SAMSUNG ELECTRONICS CO LTD [KR]) 26 September 2012 (2012-09-26) abstract; figures 1-13 paragraphs [0006], [0031] - [0033], [0045] - [0047], [0052], [0055] - [0065] -----	1 2-10
Y	US 2012/313876 A1 (SMITH GEORGE CARL [US]) 13 December 2012 (2012-12-13) abstract; figures 1-9 paragraphs [0022] - [0038] -----	4,5,7-10
Y	US 2011/131532 A1 (RUSSELL DEBORAH C [US] ET AL) 2 June 2011 (2011-06-02) abstract; figures 1-13 paragraphs [0036] - [0039], [0044], [0049], [0052] -----	2-6

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
8 January 2014	15/01/2014

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Köhn, Andreas

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2013/059566

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 2503444 A2	26-09-2012	EP 2503444 A2	26-09-2012
		KR 20120109324 A	08-10-2012
		US 2012242598 A1	27-09-2012
US 2012313876 A1	13-12-2012	NONE	
US 2011131532 A1	02-06-2011	NONE	