

[0044] The microprocessor 46 then calls (P2) programming modules of the touch-sensitive input overlay module 54 to correlate the coordinates of the touch input to a location on the GUI created by the GUI module 52, and to determine the type and location of the touch-sensitive input overlay to present on the display device 56 with the control (now a focus). When the touch-sensitive input overlay is selected, graphics information (T1) is sent back to the microprocessor 46 so that it can be presented on the display device 56.

[0045] The microprocessor 46 then sends signals (P3) to the display device 56 so that the touch-sensitive input overlay is visible. When a subsequent touch input is received at the touch input device 58, a signal (D2) is again sent to the microprocessor 46. The microprocessor 46 forwards the signal (P42) to the touch-sensitive input overlay modules 54, which generate signals (T2) for the microprocessor 46, which, in turn sends signals (P5) to the GUI module 52. When the GUI module 52 receives the signals (P5), it will generate signals (G2) for the underlying application program (and display device 56) that indicate which functions should be performed in response to the touch input signals (D2). These signals (G2) are handled by the microprocessor 46, which sends display update signals (P6) to the display device 56 so that updated information is presented on the GUI.

[0046] According to an embodiment, from the perspective of the GUI module 52, there is no difference between a keyboard/mouse input and an input received and processed by the touch input device 58 and touch-sensitive input overlay module 54 (and interpreter 55)—these aspects are transparent to the GUI module 52.

[0047] According to an embodiment, I can modify a standard GUI that is not programmed for touch-screen input and use one or more data structures and processing techniques completely separate from the standard GUI and facilitate a touch-screen input. Thus, my invention works well with legacy windowing systems and graphical user interfaces and does not necessarily require modification of the standard GUI. However, while a standard GUI can be used according to an embodiment of the invention, the addition of certain data structures directly to the GUI or supplementing the GUI can be advantageous.

[0048] Turning to FIG. 3, it depicts a touch-sensitive input overlay selector data structure 64 that can be included to the program modules 51 to facilitate optimization of the overlay type (various types of overlays are presented below) and parameters. Again, however, this aspect is merely optional, as metadata from the GUI itself can be read to determine such parameters (e.g., by reading field properties or tags in the underlying GUI or application program).

[0049] Included in the data structure 64 are three fields. The first is the variable name 66, the second is the overlay type 68, and the third is the location indicator 70. The variable name field 66 is used to identify a particular control being operated on. The overlay type field 68 indicates which of a number of overlay types is best for entering data or selections into the control. For instance, a numeric pad may be best for data entry, or a scroll bar may be preferred for a list box. The location indicator field 70 is used to specify a preferred positioning or placement coordinates for the touch-sensitive input overlay when it is presented near the

control. The location indicator field 70 is most helpful where the GUI is complex, crowded, or prior control entries are helpful in making a current control entry into the touch-sensitive input overlay. For example, the location indicator field 70 can specify a region on the screen to place the touch-sensitive input overlay so it will not obstruct the view of other control fields or on-screen information, and so that it does not get placed out of view of the display area of the display device 56. As well, the location indicator can specify or cross-reference other variable names 66 or controls that are desired to be visible when the parent control has focus.

[0050] Additional or other attributes 72 can be specified too, such as special purpose touch buttons or options for particular control fields, such as default values, minimum values, maximum values, sub-touch-sensitive input overlay options (help menus, examples), etc. In particular the other attributes can include: information for a nested touch-input calculator within the touch-sensitive input overlay; a “transparent” mode button, which can allow for the touch-sensitive input overlay to become semi-transparent so data below the touch-sensitive input overlay is visible; or a touch-sensitive input overlay movement button, which can be employed by a user to manually reposition the touch-sensitive input overlay.

[0051] Further improvements on the touch-sensitive input overlay can include algorithms that account for various touch-input gestures received at the touch-sensitive input overlay, such as special “drag-and-drop”, movement, and character entry processing algorithms. For example, a first touch gesture on a specified region of the touch-sensitive input overlay 16 on the touch-input device 58, followed immediately by a second touch gesture, e.g., a continuous sweeping motion, followed next by a release of second touch gesture on the touch input device 58, can be perceived by the touch-sensitive input overlay module 54 as a traditional “drag-and-drop” function that is performed by a mouse, resulting in a change in placement or movement of the touch-sensitive input overlay 16 relative to its initial starting position.

[0052] FIG. 4 is a flowchart detailing computer implemented acts corresponding to implementing the touch-sensitive input overlay in an embodiment. According to an embodiment, the acts are stored as one or more sequences of instructions in a computer readable medium (or computer program modules). The sequences of instructions are typically stored in a persistent memory, such as memory 48, and just prior to execution they are copied or downloaded (e.g. from a network computer readable medium into a volatile execution memory area, such as memory 50, where they are executed by one or more microprocessors, such as microprocessor 46. While most of the acts are to be carried out by the touch-sensitive input overlay module 54, others can be distributed among other resources, such as through corresponding improvements to a standard GUI module 52, or in the underlying operating system or application program, if one is employed.

[0053] In act 80, the GUI is operating in normal mode—presenting text and graphics to a user on a display device 56, which can be replied to using a standard keyboard or mouse. An interrupt driven routine determines whether a touch input is received at the touch input device 58 in act 82. If no touch