

[0093] It may be noted that the positions can comprise more than one possible user input point, as the equations may lead to non-integer position values. The non-integer values may be avoided by interpolating the position values or by using a touch area instead of a second position. The position resolution of the second point is decreased, as the positioning error of the calculated third point P_2 is increased by a factor of 3.

[0094] The dual point user input can be used for new user interface features such as two item selection, shift/alt/ctrl functionality in on screen keypads, drag & drop, keyboard shortcuts, etc . . . in the case when resistive touch pad technology is used. The operation principle is simple and implementation requires only a small modification into software (hardware driver module). The invention can also be implemented in a hardware module. The present invention allows the implementation of new user interface styles.

[0095] If the dual point input is activated and the middle point P_M is moved, the one to one relationship is no longer existent. If e.g. the middle point moves one step to the right, it can principally not be determined if the user has moved each point a single step to the right or one of them two steps to the right. In some cases it is however possible to determine which was the actual user input.

[0096] One possibility resides in that always the first point is used as a fixed reference point to calculate a movement of the second point according to the above equations. This possibility is very useful at the shift/alt/ctrl functionality, in on screen keypads, keyboard shortcuts, and all applications in which the first position is supposed to be stationary.

[0097] In case of the drag and drop feature, it is expected that a user first points to an item and then activates the drag functionality by pressing a second point on a touch pad or the touch screen. In this case, the calculated second point is supposed to be stationary. In contrast to an ad hoc approach the calculated second point is fixed and the motion of the first point can be calculated from the movement middle point. This may simply be implemented e.g. by exchanging the first and second points before setting the reference point and calculating the movement.

[0098] FIG. 8 is a flow chart of an implementation of the method of the present invention. The method starts with the detection of an input event at the position P_1 . In the next step the position change rate is determined, e.g. by determining 82 if the change rate exceeds a predetermined value. If the change rate does not exceed this value the change is regarded 83 as a conventional motion of the one-point user input at a point P_1 . This is the case if the point P_1 remains static or is moved over the surface of the touch-input device. The point P_1 is then reported 84 to the application using said touch input device as a user interface.

[0099] If the change rate exceeds the threshold value, the change is regarded as a discontinuous motion or a 'jump' of the one-point user input. Thus if the jump is detected, a new input event is detected 88 at the point P_M . The points P_1 and P_M are then used to calculate 90 a second input point P_2 analogue to the above equations. The new double or dual input points $\{P_1, P_2\}$ are generated 92 and reported 84 to the application using said touch input device as a user interface.

[0100] FIG. 9 depicts examples of how boundary areas can improve the accuracy of the recognition of a two-point

user input on a touch-input device. Boundary areas can be defined and used to exclude a number of falsely recognized two point user inputs. In FIG. 9 there are four different examples of boundary areas indicated in the 10x10 input matrices numbered 1 to 4.

[0101] In the matrix number 1 the point P_1 is positioned at near the lower left corner. If a discontinuous jump to the point P_M is detected, the point P_2 can easily be calculated. If the point P_M is instead detected e.g. at the position of P_2 , a respective calculated point would be positioned outside and not inside the matrix. To prevent that the calculated points are positioned outside the matrix, a dual-point input may only be detected if the new point P_M is detected within a boundary area 98 defined by the 'half edge distance' lines 94. The half edge distance lines 94 represent all points having equal distances to the edges of the touch pad and the first point P_1 . A combination of all half-edge distance lines 94 represent the boundary 96 of the boundary area 98. By using a boundary area 98, three quarters of the input area and therefore three quarters of the possible user inputs can be excluded from being recognized as possible dual point user input. A jump longer than a usual one (beyond the boundary area 98) excludes a dual point user input. It is to be noted that the position of this boundary area depends on the position of the first point P_1 and may have to be calculated.

[0102] In the matrix number 2 the point P_1 is also positioned to be near the lower left corner. The borderline 100 separates the border area 98' from the rest of the touch pad area. The border area 98' can contain user interface features such as the shift/alt/ctrl functionality, keyboard shortcuts, and the like. The border area 98' can be used as a boundary area for the point P_1 , when shift/alt/ctrl functionality, keyboard shortcuts input areas (not depicted) are located within said area 98'. The boundary area 98' can be used for e.g. right-handed persons, wherein it is supposed that that right handed person uses his non-dominant left hand to hold the device and uses the left thumb to press the shift/alt/ctrl functionality, while the right hand wields an input pen.

[0103] In case of a left handed person said shift/alt/ctrl functionality input areas should be (analogously) located on the right-hand side of the touch-input device. This is indicated by the interrupted line 100'. In a preferred embodiment the electronic device offers a possibility to 'reverse' the contents of e.g. a touch screen display to enable left-handed persons to use the device in an optimized way. The left-hand right-hand reversal may be implemented in a user selectable settings/user profile menu.

[0104] In the matrix number 3 the right hand borderline 100 separates the border area 98' for point P_1 and combines it with a half edge distance area 98, defined between the lines line 94 and 100. The matrix number 3 enables the recognition of a dual point input only when the point P_1 is located within the area 98' and when the point P_M is located within the area 98. That is there are two different position based constraints to enable a dual point user input, which in turn increases the accuracy of the recognition of a dual point input.

[0105] In the matrix number 4 there are different input areas 102 provided representing each an input feature as known from drag & drop-feature or the activation of different input styles as known from drawing programs. The input areas 102 can e.g. define a drawing—or an eraser-function-