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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 October 2013 (03.10.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/149164 A2

(51) International Patent Classification:
G06F 3/041 (2006.01) *G06F 3/044* (2006.01)

(21) International Application Number:
PCT/US2013/034635

(22) International Filing Date:
29 March 2013 (29.03.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/617,831 30 March 2012 (30.03.2012) US
13/830,891 14 March 2013 (14.03.2013) US

(71) Applicant: **MICROCHIP TECHNOLOGY INCORPORATED** [US/US]; 2355 West Chandler Blvd., Chandler, AZ 85224-6199 (US).

(72) Inventors: **LAMONT, Lance**; 12450 W. Center Street, Brookfield, WI 53005 (US). **HANAUER, Jerry**; N103W16737 County Court, Germantown, WI 53022 (US).

(74) Agent: **SLAYDEN, Bruce, W., II**; King & Spalding LLP, 401 Congress Ave., Suite 3200, Austin, TX 78701 (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, KM, 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, 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, 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))*

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

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

WO 2013/149164 A2

(54) Title: METHOD AND SYSTEM FOR MULTI-TOUCH DECODING

(57) Abstract: A touch sensor capable of detecting multiple touches thereto is coupled with a digital device having multi-touch decoding capabilities. These multi-touch decoding capabilities comprise touch data acquisition, touch identification, touch tracking and processed touch data output to a device associated with the touch sensor. Touch identification comprises touch location(s) peak detection, touch location(s) nudging and touch location(s) interpolation. Touch data acquisition locates potential touches on the touch sensor. Peak detection identifies where potential touch locations are on the touch sensor. Once a potential touch location(s) has been identified, touch location nudging examines each adjacent location thereto and interpolation examines the adjacent touch location values to generate a higher resolution location of the touch. Touch tracking compares time sequential "frames" of touch identification data and then determines which touches are associated between frames for further processing, e.g., determining gesturing actions.

METHOD AND SYSTEM FOR MULTI-TOUCH DECODING

RELATED PATENT APPLICATION

This application claims priority to commonly owned United States Provisional Patent Application Serial Number 61/617,831; filed March 30, 2012; which is hereby incorporated by reference herein for all purposes.

TECHNICAL FIELD

The present disclosure relates to decoding of capacitive touch sensing, in particular, multi-touch decoding.

BACKGROUND

Human interface devices include touch control systems that are based on touch sensing surfaces, *e.g.*, pads, screens, *etc.*, using capacitive sensors that change capacitance values when touched. Transforming the touch(es) on the touch sensor into one or more touch locations is non-trivial. Tracking one or more touches on the touch sensor is also challenging. Advanced touch control systems are capable of detecting not only a single touch and/or movement on a touch sensing surface such as a touch screen, but also so-called multi-touch scenarios in which a user touches more than one location and/or moves more than one finger over the respective touch sensing surface. *e.g.*, gesturing.

Key challenges of multi-touch systems are: limited processing speed of low cost systems, such as processing capabilities of, for example but not limited to, 8-bit microcontroller architectures as these architectures may be unable to do advanced math for processing the respective signals generated by the touch sensing device. There may also exist limited touch scanning performance, for example the entire system may be unable to reasonably sample the entire plane of the touch sensor or screen every "frame." Other challenges include having enough program memory space to provide for touch location determination programs that are concise, modular and general purpose. Limited random access memory (RAM) space may make the touch determination system unable to store multiple entire "images" of the touch detection and location(s) thereof simultaneously.

Hence, there exists a need to improve and simplify touch determination methods. Conventional solutions were threshold based and required complex computations. Hence, there is a need for touch determination methods that are more robust and less computation intensive. Furthermore, there exists a need for high quality multi-touch decoding, in

particular, a method and/or system that can be implemented with, for example but not limited to, a low-cost 8-bit microcontroller architecture.

SUMMARY

The aforementioned problems are solved, and other and further benefits achieved by the multi-touch decoding method and system disclosed herein.

According to an embodiment, a method for decoding multiple touches on a touch sensing surface may comprise the steps of: scanning a plurality of channels aligned on an axis for determining self values of the channels; comparing the at least one self value to determine which one of the channels may be a local maximum self value; scanning a plurality of nodes of the at least one channel that may have the local maximum self value for determining mutual values of the nodes; and comparing the mutual values to determine which one of the nodes may have the largest mutual value, wherein the node having the largest mutual value on the local maximum self value channel may be a potential touch location.

According to a further embodiment, the method may comprise the steps of: determining if at least one of the self values may be greater than a self touch threshold, wherein if yes then continue to the step of scanning a plurality of nodes of the at least one channel having the largest self value, and if no then end a touch detection frame as completed.

According to a further embodiment, the method may comprise the steps of: determining left and right slope values for the at least one self value, wherein: the left slope value may be equal to the at least one self value minus a self value of a channel to the left of the at least one channel, and the right slope value may be equal to the at least one self value minus a self value of a channel to the right of the at least one channel.

According to a further embodiment, the method may comprise the steps of: determining if the left slope value may be greater than zero (0) and the right slope value may be less than zero (0), wherein if yes then return to the step of scanning the plurality of nodes of the at least one channel, and if no then continue to next step; determining if the left slope value may be greater than zero (0) and greater than the right slope value, wherein if yes then return to the step of scanning the plurality of nodes of the at least one channel, and if no then continue to next step; determining if the left slope value may be less than zero (0) and greater than a percentage of the right slope value, wherein if yes then return to the step of scanning

the plurality of nodes of the at least one channel, and if no then continue to next step; determining if there may be another self value, wherein if yes then return to the step of determining if at least one of the self values may be greater than the self touch threshold value using the another self value, and if no then end a touch detection frame as completed.

According to a further embodiment, the method may comprise the steps of: determining if at least one of the mutual values may be greater than a mutual touch threshold, wherein if yes then continue to the step of scanning a plurality of nodes of the at least one channel having the largest self value, and if no then end the touch detection frame as completed.

According to a further embodiment, the method may comprise the steps of: determining a next slope value, wherein the next slope value may be equal to a current mutual value minus a next mutual value of a next node; and determining a previous slope value, wherein the previous slope value may be equal to the current mutual value minus a previous mutual value of a previous node.

According to a further embodiment, the method may comprise the steps of: determining if the next slope value may be less than zero (0) and the previous slope value may be greater than zero (0), wherein if yes then begin the step of validating the node, and if no then continue to next step; determining if the next slope value may be greater than zero (0) and less than a percentage of the previous slope value, wherein if yes then begin the step of validating the node, and if no then continue to next step; determining if the next slope value may be less than zero (0) and greater than the previous slope value, wherein if yes then begin the step of validating the node, and if no then continue to next step; determining if there may be another mutual value, wherein if yes then return to the step of determining if at least one of the mutual values may be greater than the mutual touch threshold, and if no then continue to the next step; and determining if there may be another self value, wherein if yes then examine another self value and return to the step of determining if at least one of the self values may be greater than a self touch threshold, and if no then end the touch detection frame as completed.

According to a further embodiment of the method, the step of validating the node may comprise the steps of: identifying the node having a local maximum mutual value as a current node; determining if there may be a valid node north of the current node, wherein if no then

continue to the step of determining if there may be a valid node south of the current node, and if yes then perform a mutual measurement on the north node and continue to the next step; determining if the north node may be greater then the current node, if yes then make the north node the current node and continue to the step of determining whether a touch point already exists at this node, and if no then continue to the next step; determining if there may be a valid node south of the current node, wherein if no then continue to the step of determining if there may be a valid node east of the current node, and if yes then perform a mutual measurement on the south node and continue to the next step; determining if the south node may be greater then the current node, wherein if yes then make the south node the current node and continue to the step of determining whether a touch point already exists at this node, and if no then continue to the next step; determining if there may be a valid node east of the current node, wherein if no then continue to the step of determining if there may be a valid node west of the current node, and if yes then perform a mutual measurement on the east node and continue to the next step; determining if the east node may be greater then the current node, if yes then make the east node the current node and continue to the step of determining whether a touch point already exists at this node, and if no then continue to the next step; determining if there may be a valid node west of the current node, wherein if no then continue to the step of determining if there may be a valid node left of the current node, and if yes then perform a mutual measurement on the west node and continue to the next step; determining if the west node may be greater then the current node, if yes then make the west node the current node and continue to the step of determining whether a touch point already exists at this node, and if no then continue to the next step; determining if there may be a valid node left of the current node, wherein if no then define a left mutual value as a center mutual value minus a right mutual value and continue to the step of determining a fine position for the node, and if yes then perform a mutual measurement on the left node and continue to the next step; determining if there may be a valid node right of the current node, wherein if no then define the mutual value as the center mutual value minus the left mutual value and continue to the step of determining the fine position for the node, and if yes then perform a mutual measurement on the right node and continue to the next step; defining a fine position of the node by subtracting the left value from the right value, dividing this difference by the center value and multiplying the result thereof by 64 and continue to the next step; and determining whether interpolation was performed for each axis, wherein if yes,

then add another touch point to a list of all detected touch points and return to the step of determining if there may be additional mutual values, and if no, then interpolate an other axis by using left and right nodes of the other axis for starting again at the step of determining if there may be a valid node left of the current node.

According to another embodiment, a method for tracking previously found and current touch locations on a touch sensing surface may comprise the steps of: determining if there may be at least one current touch location, wherein if yes then select one of the current touch locations, and if no then continue to next step; determining if there may be at least one previous touch location, wherein if no then end tracking, and if yes then select one of the previous touch locations and continue to next step; determining if the previous touch location may be associated with the current touch location, wherein if no then a touch may be no longer present at the previous touch location, stop tracking that previous touch location and continue to the step of determining if there may be at least one more previous touch location, and if yes continue to the next step; and determining if there may be at least one more previous touch location, wherein if yes then select a next previous touch location and continue to the step of determining if the previous touch location may be associated with the current touch location using the next previous touch location for the previous touch location, and if no then output touch locations tracked.

According to a further embodiment of the method, the step of selecting one of the current touch locations may comprises the steps of: determining if there may be at least one previous touch location, wherein if no then new touch to track at current touch location and continue to the step of determining if there may be at least one more current touch location, and if yes then set a temporary weight value to a maximum weight value, select a previous touch location and continue to next step; measuring a distance between the selected current touch location and the selected previous touch location, use this distance as a current weight value for determining pairing of the selected current touch location and the previous touch location, and continue to next step; determining if the current weight value may be less than the temporary weight value, wherein if no then continue to the step of determining if there may be at least one more previous touch location, and if yes then set the temporary weight value to the current weight value, record the selected previous touch location as a temporary touch location and continue to next step; determining if there may be at least one more previous touch location, wherein if yes then select the next previous touch location and return

to the step of measuring the distance between the selected current touch location and the selected previous touch location, and if no then continue to next step; determining if the temporary location may be already assigned to a different current location, wherein, if yes then calculate a next worst weight value for the current location and for an assigned current location then continue to the step of determining if the next worst weight value for the current location may be less than the next worst weight value for the assigned location, and if no then continue to next step; determining if the weight value may be below a maximum association threshold, wherein if yes then assigning the temporary location to the current location and continue to the step of determining if there may be at least one more current touch location, and if no then a new touch location may be identified for tracking thereof and continue to next step; determining if there may be at least one more current touch location, wherein if no then return to the step of determining if there may be at least one other previous touch locations, and if yes then select a next current touch location and return to the step of determining if there may be at least one previous touch location; determining if the next worst weight value for the current location may be less than the next worst weight value for the assigned location, wherein if yes then setting the temporary location to the next worst location and returning to the step of determining if there may be at least one more current touch location, and if no then setting the assigned location to the next worst weight value, selecting a moved assignment location and returning to the step of determining if there may be at least one previous touch location.

According to yet another embodiment, a method for caching mutual touch values of a plurality of touch columns may comprise the steps of: receiving a mutual scan location request; determining if a cache memory contains scan data of the requested mutual scan location, wherein if yes then continue to the step of determining if the scan data may be valid, and if no then continue to next step; determining if the requested mutual scan location may be beyond a right edge of the cache memory, wherein if yes then de-allocate the scan data in a left-most column of the cache memory, allocate the de-allocated scan data to a right edge of the cache memory and invalidate values thereof, and if no then de-allocate the scan data in a right-most column of the cache memory, allocate the de-allocated scan data to a left edge of the cache memory and invalidate values thereof; determining if the scan data may be valid, wherein if yes then return the requested the scan data for further processing thereto, and if no

then perform a mutual scan at the requested location, place the resulting scan data in the cache memory and return the requested scan data for further processing thereto.

According to still another embodiment, a system for decoding multiple touches according to the methods claimed herein may comprise: a first plurality of electrodes arranged in a parallel orientation having a first axis, wherein each of the first plurality of touch electrodes may have a self capacitance; a second plurality of electrodes arranged in a parallel orientation having a second axis substantially perpendicular to the first axis, the first plurality of electrodes may be located over the second plurality of electrodes to form a touch matrix wherein each of overlapping intersection of the first and second plurality of electrodes may have a mutual capacitance; the self capacitance may be measured for each of the first plurality of electrodes to produce respective self values; the mutual capacitance may be measured for each of the overlapping intersections of the first and second plurality of electrodes to produce respective mutual values; the self and mutual capacitances may be measured by an analog front end of a microcontroller; the self and mutual values may be stored in a memory of the microcontroller; and a digital processor in the microcontroller uses the self and mutual values in determining at least one location of at least one touch per touch acquisition frame, and tracks changing locations of the at least one touch in subsequent touch acquisition frames.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete understanding of the present disclosure thereof may be acquired by referring to the following description taken in conjunction with the accompanying drawings wherein:

Figure 1 illustrates a schematic block diagram of an electronic system having a capacitive touch sensor, a capacitive touch analog front end and a digital processor, according to the teachings of this disclosure;

Figures 1A to 1D illustrate schematic plan views of touch sensors having various capacitive touch sensor configurations, according to the teachings of this disclosure;

Figures 1E and 1F illustrate schematic plan views of self and mutual capacitive touch detection of a single touch to a touch sensor, according to the teachings of this disclosure;

Figures 1G to 1K illustrate schematic plan views of self and mutual capacitive touch detection of two touches to a touch sensor, according to the teachings of this disclosure;

Figure 2 illustrates a schematic process flow diagram for multi-touch decoding of a touch sensor as shown in Figure 1, according to specific example embodiments of this disclosure;

Figure 3 illustrates a graph of single touch peak detection data, according to specific example embodiments of this disclosure;

Figure 4 illustrates a schematic plan diagram of potential touch and mutual touch locations of a touch sensor, according to specific example embodiments of this disclosure;

Figure 5 illustrates a schematic plan view diagram of a touch sensor showing a cache data window thereof, according to specific example embodiments of this disclosure;

Figure 6 illustrates a graph of self scan values and a table of mutual scan values for two touch peak detection data, according to specific example embodiments of this disclosure;

Figures 7 and 8 illustrate schematic diagrams of historic and current point locations used for a point weighting example, according to the teachings of this disclosure;

Figure 9 illustrates schematic drawings of a normal finger touch and a flat finger touch, according to the teachings of this disclosure; and

Figures 10 to 19 illustrate schematic process flow diagrams for touch decoding, according to specific example embodiments of this disclosure.

While the present disclosure is susceptible to various modifications and alternative forms, specific example embodiments thereof have been shown in the drawings and are herein described in detail. It should be understood, however, that the description herein of specific example embodiments is not intended to limit the disclosure to the particular forms disclosed herein, but on the contrary, this disclosure is to cover all modifications and equivalents as defined by the appended claims.

DETAILED DESCRIPTION

According to various embodiments, a series of optimized processes may be provided that scan touch sensors comprising a plurality of (electrically) conductive columns and rows arranged in a matrix on a surface, *e.g.*, touch sensor, and which identify and track a plurality of touches thereto. These processes may be further optimized for operation with a low cost 8-bit microcontroller, according to specific embodiments of this disclosure.

According to various embodiments, these processes utilize both self and mutual scans to perform an optimized scan of the plurality of conductive columns and rows used for touch sensing. Using that as the basis, the proposed processes may use a subset of the data from the plurality of conductive columns and rows in order to do all necessary processing for touch location identification and tracking. The various embodiments specifically focus on a low-resource requirement solution for achieving touch location identification and tracking.

According to various embodiments, self capacitances of either the conductive columns or rows may be measured first then mutual capacitances of only those conductive columns or rows may be measured in combination with the other axis of conductive rows or columns. The various embodiments disclosed herein overcome the problem of transforming these self and mutual capacitance measurements into one or more touches and tracking these one or more touches through multiple frames of the capacitance measurements of the conductive columns or rows as described hereinabove.

According to various embodiments, at least one process may scan a plurality of conductive columns and rows arranged in a matrix, detect and track up to N touches, using various unique techniques disclosed and claimed herein. A process for peak detection examines slope ratios to accurately and quickly determine peak measurements. According to various embodiments, the challenge of tracking multiple touch locations may be solved through time on associated ones of the plurality of conductive columns or rows.

The various embodiments may allow for N touches to compensate for touches of different finger positions, *e.g.*, such as a flat finger, that prevents missed touches and substantially eliminates incorrect touches.

According to various embodiments, a process is provided for quickly identifying accurate touches instead of only looking at true peaks, wherein a "virtual" peak may be found by examining slope ratios using various techniques disclosed herein for touch identification. A combination of unique processes, according to the teachings of this disclosure, may be used to achieve better accuracy and speed improvements for multi-touch decoding. For example, a peak detection process may be implemented as a "fuzzy" peak detection process that examines slope relationships, not just signs of the slopes between the conductive columns measured. Furthermore, a so-called "nudge technique" may be used that "nudges" a potential touch location to a best location by examining adjacent values thereto.

"Windowed" data cache may be used to accelerate processing in a low capacity RAM environment, *e.g.*, 8-bit microcontroller. Interpolation may be used to increase the touch location resolution based upon measured values adjacent thereto. Multi-touch tracking may be used to identify N touches through time. Multi-touch tracking may be used to track N touches through time. Weighted matching may be used in a weighting method to best match touch points over time. "Area" detection may use a process that allows easy area and/or pressure detection based upon the sum of the nudged values for a given touch location.

Significant accuracy and speed of decoding improvements may use a combination of novel techniques for use in a low memory capacity and low cost digital processor, *e.g.*, microcontroller, microprocessor, digital signal processor (DSP), application specific integrated circuit (ASIC), programmable logic array (PLA), *etc.* Various embodiments may track eight or more touches on, for example but not limited to, a 3.5 inch touch sensor capacitive sensor array. For example when using a Microchip PIC18F46K22 (64K ROM, <4K RAM) microcontroller.

Conventional capacitive touch decoding neither uses the techniques described more fully hereinafter, nor exhibits these performance results.

Referring now to the drawings, the details of example embodiments are schematically illustrated. Like elements in the drawings will be represented by like numbers, and similar elements will be represented by like numbers with a different lower case letter suffix.

Referring to Figure 1, depicted is a schematic block diagram of an electronic system having a capacitive touch X-Y sensor, a capacitive touch analog front end and a digital processor, according to the teachings of this disclosure. A microcontroller integrated circuit device 112 may comprise a digital processor and memory 106, an analog-to-digital converter (ADC) controller 108, and a capacitive touch analog front end (AFE) 110. The microcontroller integrated circuit device 112 may be coupled to a touch sensor 102 comprised of a plurality of conductive columns 104 and rows 105 arranged in a matrix. It is contemplated and within the scope of this disclosure that the conductive rows 105 and/or conductive columns 104 may be printed circuit board conductors, wires, ITO coatings on a clear substrate, *e.g.*, display/touch screen, *etc.*, or any combinations thereof.

Referring to Figures 1A to 1D, depicted are schematic plan views of touch sensors having various capacitive touch sensor configurations, according to the teachings of this

disclosure. Figure 1A shows conductive columns 104 and conductive rows 105. Each of the conductive columns 104 has a "self capacitance" that may be individually measured when in a quiescent state, or all of the conductive rows 105 may be actively excited while each one of the conductive columns 104 has self capacitance measurements made thereof. Active excitation of all of the conductive rows 105 may provide a stronger measurement signal for individual capacitive measurements of the conductive columns 104.

For example, if there is a touch detected on one of the conductive columns 104 during a self capacitance scan, then only that conductive column 104 having the touch detected thereon need be measured further during a mutual capacitance scan thereof. The self capacitance scan may only determine which one of the conductive columns 104 has been touched, but not at what location along the axis of that conductive column 104 where it was touched. The mutual capacitance scan may determine the touch location along the axis of that conductive column 104 by individually exciting (driving) one at a time the conductive rows 105 and measuring a mutual capacitance value for each one of the locations on that conductive column 104 that intersects (crosses over) the conductive rows 105. There may be an insulating non-conductive dielectric (not shown) between and separating the conductive columns 104 and the conductive rows 105. Where the conductive columns 104 intersect with (crossover) the conductive rows 105, mutual capacitors 120 are thereby formed. During the self capacitance scan above, all of the conductive rows 105 may be either grounded or driven with a logic signal, thereby forming individual column capacitors associated with each one of the conductive columns 104.

Figures 1B and 1C show interleaving of diamond shaped patterns of the conductive columns 104 and the conductive rows 105. This configuration may maximize exposure of each axis conductive column and/or row to a touch (*e.g.*, better sensitivity) with a smaller overlap between the conductive columns 104 and the conductive rows 105. Figure 1D shows receiver (top) conductive rows (*e.g.*, electrodes) 105a and transmitter (bottom) conductive columns 104a comprising comb like meshing fingers. The conductive columns 104a and conductive rows 105a are shown in a side-by-side plan view, but normally the top conductive rows 105a would be over the bottom conductive columns 104a. Self and mutual capacitive touch detection is more fully described in Technical Bulletin TB3064, entitled "mTouch™ Projected Capacitive Touch Screen Sensing Theory of Operation" by Todd O'Connor, available at www.microchip.com; and commonly owned United States Patent Application

Publication No. US 2012/0113047, entitled "Capacitive Touch System Using Both Self and Mutual Capacitance" by Jerry Hanauer; wherein both are hereby incorporated by reference herein for all purposes.

Referring back to Figure 1, microcontrollers 112 now include peripherals that enhance the detection and evaluation of such capacitive value changes. More detailed descriptions of various capacitive touch system applications are more fully disclosed in Microchip Technology Incorporated application notes AN1298, AN1325 and AN1334, available at www.microchip.com, and all are hereby incorporated by reference herein for all purposes. One such application utilizes the capacitive voltage divider (CVD) method to determine a capacitance value and/or evaluate whether the capacitive value has changed. The CVD method is more fully described in Application Note AN1208, available at www.microchip.com; and a more detailed explanation of the CVD method is presented in commonly owned United States Patent Application Publication No. US 2010/0181180, entitled "Capacitive Touch Sensing using an Internal Capacitor of an Analog-To-Digital Converter (ADC) and a Voltage Reference," by Dieter Peter; wherein both are hereby incorporated by reference herein for all purposes.

A Charge Time Measurement Unit (CTMU) may be used for very accurate capacitance measurements. The CTMU is more fully described in Microchip application notes AN1250 and AN1375, available at www.microchip.com, and commonly owned U.S. Patent Nos. US 7,460,441 B2, entitled "Measuring a long time period;" and US 7,764,213 B2, entitled "Current-time digital-to-analog converter," both by James E. Bartling; wherein all of which are hereby incorporated by reference herein for all purposes.

It is contemplated and within the scope of this disclosure that any type of capacitance measurement circuit having the necessary resolution may be used in determining the capacitance values of the plurality of conductive columns 104, and that a person having ordinary skill in the art of electronics and having the benefit of this disclosure could implement such a capacitance measurement circuit.

Referring to Figures 1E and 1F, depicted are schematic plan views of self and mutual capacitive touch detection of a single touch to a touch sensor, according to the teachings of this disclosure. In Figure 1E a touch, represented by a picture of a part of a finger, is at approximately the coordinates of X05, Y07. During self capacitive touch detection each one

of the rows Y01 to Y09 may be measured to the determine the capacitance values thereof. Note that baseline capacitance values with no touches thereto for each one of the rows Y01 to Y09 have been taken and stored in a memory (*e.g.*, memory 106 - Figure 1). Any significant capacitance change to the baseline capacitance values of the rows Y01 to Y09 will be obvious and taken as a finger touch. In the example shown in Figure 1E the finger is touching row Y07 and the capacitance value of that row will change, indicating a touch thereto. However it is still unknown from the self capacitance measurements where on this row that the touch has occurred.

Once the touched row (Y07) has been determined using the self capacitance change thereof, mutual capacitive detection may be used in determining where on the touched row (Y07) the touch has occurred. This may be accomplished by exciting, *e.g.*, putting a voltage pulse on, each of the columns X01 to X12 one at a time while measuring the capacitance value of row Y07 when each of the columns X01 to X12 is individually excited. The column (X05) excitation that causes the largest change in the capacitance value of row Y07 will be the location on that row which corresponds to the intersection of column X05 with row Y07, thus the single touch is at point or node X05, Y07. Using self and mutual capacitance touch detection significantly reduces the number of row and column scans to obtain the X,Y touch coordinate on the touch sensor 102. In this example, nine (9) rows were scanned during self capacitive touch detection and twelve (12) columns were scanned during mutual capacitive touch detection for a total number of 9+12 = 21 scans. If individual x-y capacitive touch sensors for each node (location) were used then 9 x 12 = 108 scans would be necessary to find this one touch, a significant difference. It is contemplated and within the scope of this disclosure that the self capacitances of the columns X01 to X21 may be determined first then mutual capacitances determined of a selected column(s) by exciting each row Y01 to Y09 to find the touch location on the selected column(s).

Referring to Figures 1G to 1K, depicted are schematic plan views of self and mutual capacitive touch detection of two touches to a touch sensor, according to the teachings of this disclosure. In Figure 1G two touches, represented by a picture of parts of two fingers, are at approximately the coordinates of X05, Y07 for touch #1 and X02, Y03 for touch #2. During self capacitive touch detection each one of the rows Y01 to Y09 may be measured to the determine the capacitance values thereof. Note that baseline capacitance values with no touches thereto for each one of the rows Y01 to Y09 have been taken and stored in a memory

(*e.g.*, memory 106 - Figure 1). Any significant capacitance changes to the baseline capacitance values of the rows Y01 to Y09 will be obvious and taken as finger touches. In the example shown in Figure 1H the first finger is touching row Y07 and the second finger is touching row Y03, wherein the capacitance values of those two rows will change, indicating touches thereto. However it is still unknown from the self capacitance measurements where on these two row that the touches have occurred.

Once the touched rows (Y07 and Y03) have been determined using the self capacitance changes thereof, mutual capacitive detection may be used in determining where on these two touched rows (Y07 and Y03) the touches have occurred. Referring to Figure 1I, this may be accomplished by exciting, *e.g.*, putting a voltage pulse on, each of the columns X01 to X12 one at a time while measuring the capacitance value of row Y07 when each of the columns X01 to X12 is individually excited. The column (X05) excitation that causes the largest change in the capacitance value of row Y07 will be the location on that row that corresponds to the intersection of column X05 with row Y07. Referring to Figure 1J, likewise measuring the capacitance value of row Y03 when each of the columns X01 to X12 is individually excited determines where on column Y03 the touch #2 has occurred. Referring to Figure 1K, the two touches are at points or nodes (X05, Y07) and (X02, Y03). It is contemplated and within the scope of this disclosure that if the capacitances of more then one of the selected rows, *e.g.*, Y07 and Y03, can be measured simultaneously, then only one set of individual column X01 to X12 excitations is needed in determining the two touches to the touch sensor 102.

Referring to Figure 2, depicted is a schematic process flow diagram for multi-touch decoding of a touch sensor as shown in Figure 1, according to specific example embodiments of this disclosure. A process for multi-touch decoding may comprise the steps of Data Acquisition 202, Touch Identification 204, Touch Tracking 206 and Data Output 208. The step of Touch Identification 204 may further comprise the steps of Peak Detection 210, Nudge 212 and Interpolation 214, more fully described hereinafter.

Data Acquisition

Data Acquisition 202 is the process of taking self and mutual capacitance measurements of the plurality of conductive columns 104 or conductive rows 105 to acquire touch identification data. The touch identification data may be further processed to locate

potential touches on the touch sensor 102 using the process of Touch Identification 204 as more described fully hereinafter.

Touch Identification

Touch Identification 204 is the process of using the touch identification data acquired during the process of Data Acquisition 202 to locate potential touches on the touch sensor 102. The following are a sequence of process steps to determine which ones of the plurality of conductive columns 104 or conductive rows 105 to select that have a touch(es) thereto using self capacitance measurements thereof, and where on the selected conductive columns 104 or conductive rows 105 the touch(es) may have occurred using mutual capacitance measurements thereof

Peak Detection

Peak detection 210 is the process of identifying where potential touch locations may be on the touch sensor 102. However according to the teachings of this disclosure, instead of only looking at actual detected "peaks," peak detection may purposely be made "fuzzy," *e.g.*, identifying potential peaks by looking for ratios of differences of slope values as well as slope "signs," not just a low-high-low value sequence. A "virtual" peak may be detected by examining slope ratios, *e.g.*, 2:1 slope ratio, wherein a change in slope may be identified as a potential peak. This may be repeated until no additional peaks are detected.

Nudge

Nudge 212 is the process of examining each adjacent location of a potential touch location once it has been identified. If the adjacent location(s) has a greater value than the existing touch potential location then eliminate the current potential touch location and identify the adjacent location having the greater value as the potential touch location (see Figure 5 and the description thereof hereinafter).

Interpolation

Once a touch location has been identified, Interpolation 214 is the process that examines the adjacent values to generate a higher resolution location.

Touch Tracking

Touch Tracking 206 is the process of comparing time sequential "frames" of touch identification data and then determining which touches are associated between sequential

frames. A combination of weighting and "best guess" matching may be used to track touches through multiple frames during the process of Data Acquisition 202 described hereinabove. This is repeated for every peak detected and every touch that was identified on the previous frame. A "frame" is the set of self and mutual capacitive measurements of the plurality of capacitive touch sensors 104 in order to capture a single set of touches at a specific time. Each full set of scans (a "frame") of the self and mutual capacitance measurements of the plurality of conductive columns 104 or conductive rows 105 to acquire touch identification data of the touch sensor 102 at a given time associated with that frame.

Touch Tracking 206 associates a given touch in one frame with a given touch in a subsequent frame. Touch tracking may create a history of touch frames and may associate the touch locations of a current frame with the touch locations of a previous frame or frames. In order to associate a previous touch location to a current potential touch location a "weighting" function may be used. The weight values ("weight" and "weight values" will be used interchangeably herein) between time sequential touch locations (of different frames) represent the likelihood that time sequential touch locations (of different frames) are associated with each other. Distance calculations may be used to assign weight values between these associated touch locations. A "true" but complex and processor intensive calculation for determining weight value between touch locations is:

$$\text{Weight value} = \text{SQRT}\left[(X_{previous} - X_{current})^2 + (Y_{previous} - Y_{current})^2\right] \qquad \text{Eq. (1)}$$

A simplified distance (weight value) calculation may be used that measures ΔX and ΔY and then sums them together:

$$\text{Weight value}' = \text{ABS}\,(X_{previous} - X_{current}) + \text{ABS}\,(Y_{previous} - Y_{current}) \qquad \text{Eq. (2)}$$

The above simplified weight value calculation, Eq. (2), creates a diamond shaped pattern for a given weight value instead of a circular pattern of the more complex weight value calculation, Eq. (1). Use of Eq. (2) may be optimized for speed of the weight value calculations in a simple processing system, distance may be calculated based upon the sum of the change of the X-distances and the change in the Y-distances, *e.g.*, Eq. (2) herein above. A better weight value may be defined as a smaller distance between sequential touch locations.

For each new touch location a weight value may be calculated for all touch locations from the previous frame. The new touch location is then associated with the previous touch

location having the best weight value therebetween. If the previous touch location already has an associated touch location from a previous frame, a secondary second-best weight value for each touch location may be examined. The touch location with the lower-cost second-best weight value may then be shifted to its second best location, and the other touch location may be kept as the best touch location. This process is repeated until all touch locations have been associated with previous frame touch locations, or have been identified as "new touches" having new locations with no touch locations from the previous frame being close to the new touch location(s).

An alternative to the aforementioned weighting process, may be a vector-based process utilizing a vector created from the previous two locations to create the most likely next location. This vector-based weighting process may use the same distance calculations as the aforementioned weighting process, running it from multiple points and modifying the weight values based upon from which point the measurement was taken.

By looking at the previous two locations of a touch, the next "most likely" location of that touch may be predicted. Once the extrapolated location has been determined that location may be used as the basis for a weighting value. To improve matching on the extrapolated location an "acceleration model" may be used to add weighting points along the vector to the extrapolated locations and past the extrapolated locations.. These additional points assist in detecting changes in speed of the touch movement, but may not be ideal for determining direction of the touch motion.

Referring to Figures 7 and 8, depicted are schematic diagrams of historic and current point locations used for a point weighting example, according to the teachings of this disclosure. Once weights have been generated, the best combination of weight values and associated touches may be generated. Certain touch scenarios may cause nearly identical weight values, in which case the second best weight values should be compared and associations appropriately shifted. Depending upon the order of operations, points A and D may be associated first. As the weight values for B are generated BD is a better match then BC. In this case look at secondary weight values. Is it less costly to shift A to be associated with C or to shift B to be associated with C?

By extending this sequence of operations, all points can have associations shifted for the best overall match, not just the best local match. Some caution may be needed to prevent

infinite loops of re-weighing. This may be accomplished by limiting the number of shifts to a finite number. Referring now to Figure 8, points A and B are existing points, and points 1 and 2 are "new" points that need to be associated.

Step 1) Calculate weight values between touch locations:

$A \leftrightarrow 1$ weight = 5 $((\Delta X = 2) + (\Delta Y = 3) = 5)$
$A \leftrightarrow 2$ weight = 4
$B \leftrightarrow 1$ weight = 10
$B \leftrightarrow 2$ weight = 5

Step 2) Select the "best" pair (lowest weight) for each existing touch location:

$A \leftrightarrow 2$ weight = 4 and $B \leftrightarrow 2$ weight = 5

Step 3) If more than one existing touch location pairs with a given new touch location, then look at the second-best touch locations for each and the difference in weight values from the best to the second best pair (the "cost").

$A \leftrightarrow 1$ (weight: 5) Cost = 1: ($A \leftrightarrow 1$ weight) - ($A \leftrightarrow 2$ weight 4)
$B \leftrightarrow 1$ (weight: 10) Cost = 5: ($B \leftrightarrow 1$ weight) - ($B \leftrightarrow 2$ weight 5)

Step 4) Shift the pairing to the lowest cost pair thereby allowing the other touch location to maintain the original pairing.

$A \leftrightarrow 1$
$B \leftrightarrow 2$

Step 5) Repeat steps 2) through 4) until all pairing are 1:1. If there are more touch locations than existing touch locations then start tracking a new touch location. If fewer new touch locations than existing "worst match" touch locations then these worst match touch locations may be lost and no longer tracked.

Flat Finger Identification

Referring to Figure 9, depicted are schematic drawings of a normal finger touch and a flat finger touch, according to the teachings of this disclosure. One challenge of identifying a touch is the "flat finger" scenario. This is when the side or flat part of a finger 1020, rather then the finger tip 1022, is placed on the touch sensor 102. Note that a flat finger 1020 may generate two or more potential touch locations 1024 and 1026. It is possible using the teaching of this disclosure to detect a flat finger 1020 by accumulating the sum of the values of all nodes nudged to each peak. If the sum of these values surpasses a threshold then it is likely caused by a flat finger touch. If a flat finger touch is detected then other touches that are near the flat finger peak(s) may be suppressed.

Data Output

Referring back to Figure 2, Data Output 208 is the process of providing determined touch location coordinates in a data packet(s) to a host system for handling thereof.

Touch Determination

Given an array of touch data, examine the differences between the values thereof and flag certain key scenarios as potential peaks for further examination. All touch data values below a threshold value may be ignored when determining touch locations.

Key Scenario 1: True Peak

Referring to Figure 3, identify the transition from a positive to a negative slope as a potential peak. This would be the point circled in column 7 of the example data values shown in Figure 3.

Key Scenario 2: Slope Ratio Beyond Threshold ("Fuzzy" Peak Detection)

A key threshold of slope ratios may be used to flag additional peaks. The threshold value used may be, for example but is not limited to, 2:1; so instances where there is a change of slope greater than 2:1 may be identified as potential peaks. This applies to positive and negative slopes. This would be the point circled in column 6 of the example data values shown in Figure 3.

Why not just look at the slope signs?

Since the self scan is only one axis of a two-axis sensor array (*e.g.*, conductive rows 105 and conductive columns 104 of touch sensor 102, Figure 1), it is possible for two touches that are off by a single "bar" (*e.g.*, column) to only show a single peak. With the example data, there could be two touches, one at 6,6 and another at 7,7 (see Figures 3 and 6). Without the additional peak detection, the touch at 6,3 may not be detected.

Nudge Location Refinement

Once a potential touch location is identified, each adjacent touch location may be examined to determine if they have a greater value. If a greater value is present, eliminate the current potential touch location and identify the touch location of the greater value as a potential touch location. This process is repeated until a local peak has been identified.

Referring to Figure 3, depicted is a graph of single touch peak detection data, according to specific example embodiments of this disclosure. An example graph of data values for one column (*e.g.*, column 7) of the touch sensor 102 is shown wherein a maximum

data value determined from the self and mutual capacitance measurements of column 7 occurs at the capacitive touch sensor 104 area located a row 7, column 7. All data values that are below a threshold value may be ignored, *e.g.*, below about 12 in the graphical representation shown in Figure 3. Therefore only data values taken at row 6 (data value = 30) and at row 7 (data value = 40) need be processed in determining the location of a touch to the touch sensor 102. Slope may be determined by subtracting a sequence of adjacent row data values in a column to produce either a positive or negative slope value. When the slope value is positive the data values are increasing, and when the slope value is negative the data values are decreasing A true peak may be identified as a transition from a positive to a negative slope as a potential peak. A transition from a positive slope to a negative slope is indicated at data value 422 of the graph shown in Figure 3.

However another touch may have occurred at column 6 and was not directly measured in the column 7 scan, but shows up as data value 420 during the column 7 scan. Without another test besides the slope sign transition, the potential touch at column 6 may be missed. Therefore a threshold of slope ratios may further be used to flag additional potential peaks. Slope is the difference between two data values of adjacent conductive columns 104. This threshold of slope ratios may be, for example but is not limited to, 2:1 so instances where there is a change of slope greater than 2:1 may be identified as another potential peak. This may apply to both positive and negative slopes. For example, the data value 420, taken at row 6, has a left slope of 23:1 (30 - 7) and a right slope of 10:1 (40 - 30). The data value 422, taken at row 7, has a left slope of 10:1 (40 - 30) and right slope of -30:1 (10 - 40). The slope ratio for row 6 of 23:10, exceeds the example 2:1 threshold and would be labeled for further processing. All other data values are below the data value threshold and may be ignored.

Referring to Figure 4, depicted is a schematic plan diagram of potential touch and mutual touch locations of a touch sensor, according to specific example embodiments of this disclosure. Once a potential touch location is identified, each adjacent location thereto may be examined to determine whether any one of them may have a greater data value than the current potential touch location (labeled "C" in Figure 4(a) & (b)). If a greater data value is found, then the current potential touch location may be eliminated and the touch location having the greater value may be identified as a potential touch location. This is referred to herein as a Nudge 212 process and may be repeated until a data peak has been identified.

During a data acquisition scan of a column of rows, only tier one nodes (labeled "1" in Figure 4(a) and 4(b) - adjacent locations to the current potential touch location) are examined. If any of these tier one nodes has a larger data value than the data value of the current potential touch location, a new current touch location is shifted ("nudged") to that node having the highest data value and the Nudge process 212 is repeated. If a tier one node is already associated with a different potential peak, then no further searching is necessary and the current data peak may be ignored. Tier two nodes (labeled "2" in Figure 4(a) & (b) - adjacent locations to the tier one nodes) are examined when there is a potential of a large area activation of the touch sensor 102.

After one conductive column 104 has been scanned for mutual capacitance values, the Nudge process 212 may be speeded up by storing the mutual capacitance data values of that one column in a cache memory, then doing the Nudge process 212 first on the tier one nodes, and then on the tier two nodes of that one column from the mutual capacitance data values stored in the cache memory. Then only after there are no further nudges to do in that one column will the Nudge process 212 examine the tier one and tier two nodes from the mutual capacitance measurement scans of the two each adjacent columns on either side of the column having the Nudge process 212 performed thereon.

Interpolation of the potential touch location may be performed by using the peak data value node (touch location) as well as each adjacent node thereto (*e.g.*, tier one nodes from a prior Nudge process 212) to create sub-steps between each node. For example, but not limited to, 128 steps may be created between each node. Referring to Figure 4(c), node A is the potential touch location and nodes B, C, D and E are tier one nodes adjacent thereto. The interpolated X, Y location may be found using the following equations:

$$Location_x = (D_{Value} - B_{Value})/A_{Value}*64$$

$$Location_y = (E_{Value} - C_{Value})/A_{Value}*64$$

It is contemplated and within the scope of this disclosure that variations of the above equations may be used based upon the ratio of values and the signs of the numerator of the division.

Referring to Figure 5, depicted is a schematic plan view diagram of a touch sensor showing a cache data window thereof, according to specific example embodiments of this disclosure. The conductive columns 104 of the touch sensor 102 may be scanned column by

column for self capacitance values until all conductive columns 104 have been scanned. Each conductive column 104 indicating a potential touch from the self capacitance data may be sequentially scanned for determining mutual capacitive values thereof (touch data) and when peaks are discovered they may be processed contemporaneously with the column scan. Furthermore, touch data may be stored in a cache memory for further processing. Since the Nudge process 212 looks at the first tier nodes then the second tier nodes, if necessary, not all of the touch data from all of the conductive columns 104 need be stored at one time. This allows a simple caching system using a minimum amount of random access memory (RAM). For example, storing five columns of touch data in a cache. The five columns are contiguous and a cache window may move across the columns 104 of the touch sensor 102 one column 104 at a time. It is contemplated and within the scope of this disclosure that more or fewer than five columns of touch data may be stored in a cache memory and processed therefrom, and/or self capacitance scanning by rows instead of columns may be used instead. All descriptions herein may be equally applicable to self capacitance scanning of rows then mutual capacitance scanning by columns of those row(s) selected from the self capacitance scan data.

Whenever a Mutual Scan of a first or second tier node (capacitive sensor 104) is requested, it may be first called from the cache memory. If the requested node touch data is present in the cache memory, the cache memory returns the requested touch data of that first or second tier node. However, if the requested touch data is not present in the cache memory then the following may occur: 1) If the column of the requested touch data is in the range of the cache window then perform the mutual scan of that column and add the touch data to the cache memory, or 2) If the column of the requested touch data is not in the range of the present cache window then shift the cache window range and perform the mutual scan of the new column and add the resulting touch data from the new cache window to the cache memory.

Referring to Figure 6, depicted are a graph of self scan values and a table of mutual scan values for two touch peak detection data, according to specific example embodiments of this disclosure. Since a self scan is performed in only one axis (*e.g.*, one column), it is possible for two touches that are off by a single column to only show a single peak. For the example data values shown in Figure 6, two touches may have occurred, one at self scan data value 422 and the other indicated at self scan data value 420. Without being aware of change

of slopes greater than 2:1, the potential touch represented by self scan data value 420 may have been missed. A first touch may cause data value 422 and a second touch may cause data value 420. Peak Detection and nudging, as described hereinabove, may further define these multiple touches as described herein.

Referring to Figures 10 to 19, depicted are schematic process flow diagrams for touch decoding, according to specific example embodiments of this disclosure. Figure 10 shows a general overview of possible processes for multi-touch decoding in a touch sensor 102 enabled device. It is contemplated and within the scope of this disclosure that more, fewer and/or some different processes may be utilized with a touch sensor 102 enabled device and still be within the scope, intent and spirit of this disclosure. In step 1050 a device is started, actuated, *etc.*, when in step 1052 power is applied to the device. In step 1054 the device may be initialized, and thereafter in step 1056 the process of touch identification may begin. In step 1058 touch tracking may be performed on those touches identified in step 1056. In step 1060 the touch data may be further processed if necessary, otherwise it may be transmitted to the processing and control logic of the device for display and/or control of the device's intended purpose(s) in step 1062.

In the descriptions of the following process steps references to "top" or "north" channel or node will mean the channel or node above another channel or node, "bottom" or "south" channel or node will mean the channel or node below another channel or node, "left" or "west" channel or node will mean the channel or node to the left of another channel or node, and "right" or "east" channel or node will mean the channel or node to the right of another channel or node. Referring to Figure 11, a flow diagram of a touch identification process 204 is shown and described hereinafter. In step 1102 the touch identification process 204 (Figure 2) begins. In step 1104 a self scan of all channels on one axis may be performed, *e.g.*, either all columns or all rows. In step 1106 the first self scan value may be examined. In step 1108 the (first or subsequent) self scan value may be compared to a self touch threshold value.

A self peak detection process 1100 may comprise steps 1110 to 1118, and is part of the overall Peak Detection process 210 (Figure 2). If the self scan value is less than the self touch threshold value as determined in step 1108, then step 1238 (Figure 12) may determine whether there are any additional self scan values to be examined. However, if the self scan

value is equal to or greater than the self touch threshold value as determined in step 1108, then step 1110 may calculate a left slope between the self scan value and a self scan value of the channel to the left of the present channel. Then step 1112 may calculate a right slope between the self scan value and a self scan value of the channel to the right of the present channel.

Step 1114 determines whether the left slope may be greater than zero (positive slope) and the right slope may be less than zero (negative slope), identifying a peak. If a yes result in step 1114, then step 1120 may perform mutual scan measurements on each node of the channel selected from the self scan data. If a no result in step 1114, then step 1116 determines whether the left slope may be greater than zero (positive slope) and greater than the right slope may be, for example but is not limited to, two times (twice) greater than the right slope. If a yes result in step 1116, then in step 1120 mutual scan measurements may be performed on each node of the selected self scan channel. If a no result in step 1116, then step 1118 determines whether the left slope may be, for example but is not limited to, less than zero (negative slope) and greater than a percentage of the right slope, *e.g.*, fifty (50) percent. If a yes result in step 1116, then step 1120 may perform mutual scan measurements on each node of the channel selected from the self scan data. If a no result in step 1116, then step 1238 (Figure 12) may determine whether there are any additional columns to be examined based upon the self scan values thereof. Step 1122 may examine a first mutual scan value.

Referring to Figure 12, a mutual peak detection process 1244 may comprise steps 1226 to 1234, and is part of the overall Peak Detection process 210 (Figure 2). Step 1224 may compare the (first or subsequent) mutual scan value to a mutual touch threshold value, wherein if the mutual scan value is less than the mutual touch threshold value then step 1236 may determine whether there are any additional mutual scan values to be examined. However, if the mutual scan value is equal to or greater than the mutual touch threshold value then step 1226 may calculate a slope to the next mutual scan value node, then step 1228 may calculate a slope to the previous mutual scan value node.

Step 1230 determines whether the next slope may be less than zero (negative slope) and the previous slope may be greater than zero (positive slope). If a yes result in step 1230, then step 1350 (Figure 13) may start the nudge process 212 and/or the interpolation process

214 (Figure 2). If a no result in step 1230, then step 1232 determines whether the next slope may be, for example but is not limited to, greater than zero (positive slope) and less than a percentage of the previous slope. If a yes result in step 1232, then step 1350 (Figure 13) may start the nudge process 212 and/or the interpolation process 214 (Figure 2). If a no result in step 1232, then step 1234 determines whether the next slope may be, for example but is not limited to, less than zero (negative slope) and greater than the previous slope. If a yes result in step 1234, then step 1350 (Figure 13) may start the nudge process 212 and/or the interpolation process 214 (Figure 2). If a no result in step 1234, then step 1236 determines whether there may be any additional mutual values to be examined. If a yes result in step 1236, then step 1242 may examine a next mutual value. If a no result in step 1236, then step 1238 determines whether there may be any additional self scan values to be examined. If a yes result in step 1238, then step 1240 examines a next self scan value that may be returned to step 1108 (Figure 11) for further processing thereof. If a no result in step 1238, then in step 1244 a touch detection frame may be complete.

Referring to Figures 13-15, flow diagrams of a nudge process 212 and an interpolation process 214 (Figure 2) are shown and described hereinafter. Step 1350 may start the nudge process 212 and/or the interpolation process 214 by using a peak location from the touch identification process 204 (Figure 2) and may comprise the following process steps: Step 1352 determines whether there may be a valid node to the north. If a no result in step 1352, then continue to step 1360. If a yes result in step 1352, then step 1354 may make a mutual scan measurement of the node to the north. Step 1356 determines whether the mutual scan data of the north node may be greater than the current node. If a no result in step 1356, then continue to step 1360. If a yes result in step 1356, then in step 1358 the north node may become the current node, and then continue to step 1486 (Figure 14).

Step 1360 determines whether there may be a valid node to the south. If a no result in step 1360, then continue to step 1470 (Figure 14). If a yes result in step 1360, then step 1362 may make a mutual scan measurement of the node to the south. Step 1364 determines whether the mutual scan data of the south node may be greater than the current node. If a no result in step 1364, then continue to step 1470 (Figure 14). If a yes result in step 1364, then in step 1366 the south node may become the current node, and then continue to step 1486 (Figure 14).

Referring to Figure 14, step 1470 determines whether there may be a valid node to the east. If a no result in step 1470, then continue to step 1478. If a yes result in step 1470, then step 1472 may make a mutual scan measurement of the node to the east. Step 1474 determines whether the mutual scan data of the east node may be greater than the current node. If a no result in step 1474, then continue to step 1478. If a yes result in step 1474, then in step 1476 the east node may become the current node, and then continue to step 1486.

Step 1478 determines whether there may be a valid node to the west. If a no result in step 1478, then continue to step 1502 (Figure 15). If a yes result in step 1478, then step 1480 may make a mutual measurement of the node to the west. Step 1482 determines whether the mutual scan data of the west node may be greater than the current node. If a no result in step 1482, then continue to step 1502 (Figure 15). If a yes result in step 1482, then in step 1484 the west node may become the current node. Step 1486 determines whether a touch point may already exist at the selected node. If a no result in step 1486, then continue to step 1352 (Figure 13). If a yes result in step 1486, then step 1488 may eliminate the current peak, and then continue to step 1236 (Figure 12).

Referring to Figure 15, an interpolation process 214 may comprise steps 1502 to 1518. Step 1502 determines whether there may be a valid node to the left. If a no result in step 1502, then continue to step 1510 wherein the left node value may be defined as a center value minus a right value then continue to step 1506. If a yes result in step 1502, then step 1504 may perform a mutual scan measurement on the node to the left. Then step 1506 determines whether there may be a valid node to the right. If a no result in step 1506, then continue to step 1512 wherein the right node value may be defined as a center value minus a left value then continue to step 1516. If a yes result in step 1506, then step 1508 may perform a mutual scan measurement on the node to the right. Step 1516 may determine a fine position by subtracting the left value from the right value, dividing the difference thereof by the center value, and then multiplying the result by, for example but is not limited to, the number 64. It is contemplated and within the scope and spirit of this disclosure that many ways of determining valid peaks and nodes may be used as one having ordinary skill in the art of touch detection and tracking could readily implement by having knowledge based upon the teachings of this disclosure

After step 1516 has completed the aforementioned calculations, step 1514 determines whether an interpolation may have been performed for each axis. If a no result in step 1514, then step 1518 may interpolate another axis, thereafter steps 1502 to 1516 may be repeated, with "above" replacing "left" and "below" replacing "right" in each step. If a yes result in step 1514, then step 1520 may add this touch point to a list of all detected touch points. Then step 1522 may return to step 1236 (figure 12) for any additional mutual scan values to be examined.

Referring to Figures 16, 17 and 18, flow diagrams of a touch tracking process 206 are shown and described hereinafter. In step 1602 the touch tracking process 206 may start by using the previously found and current touch locations. Step 1604 determines whether there may be any current touch locations. If a yes result in step 1604, then step 1606 may select the first of the current touch locations, and thereafter may continue to step 1722 (Figure 17). If a no result in step 1604, then step 1610 determines whether there may be any previous touch location(s). If a yes result in step 1610, then step 1612 may select the first previous touch location. If a no result in step 1610, then at step 1611 tracking is complete.

Step 1614 determines whether the previous touch location may be associated with a current touch location. If a no result in step 1614, then step 1608 may assert an output of "touch no longer present at previous touch location, stop tracking," and then return to step 1616. If a yes result in step 1614, then step 1616 determines whether there may be any more previous touch locations. If a no result in step 1616, then at step 1620 tracking touch locations is complete and the touch location data may be transmitted as data output 208 (Figure 2) for further processing by the microcontroller 112 (Figure 1). If a yes result in step 1616, then step 1618 may select the next previous touch location, and thereafter return to step 1614.

Referring to Figure 17, step 1722 determines whether there may be any previous touch locations. If a no result in step 1722, then continue to step 1868 (Figure 18) wherein a "New Touch to track is identified" at current location, and thereafter continue to step 1856 (Figure 18). If a yes result in step 1722, then step 1724 may set a temporary weight value to a maximum weight value. Step 1726 may select the first of the previous touch locations. Then step 1728 may measure a distance between the selected current touch location and the selected previous touch location to determine a current distance (weight value) therebetween.

Step 1730 determines whether the current weight value may be less than the temporary weight value. If a yes result in step 1730, then step 1732 may set the temporary weight value to the current weight value and thereafter may record the selected previous touch location as a temporary location and continue to step 1734. If a no result in step 1730, then step 1734 determines whether there may be more previous touch locations. If a yes result in step 1734, then step 1736 may select the next previous touch location, and thereafter return to step 1728. If a no result in step 1734, then step 1738 determines whether the temporary location may have already been assigned to a different current location. If a yes result in step 1738, then step 1740 may calculate a next worst weight value for the current location and for an assigned current location, and thereafter continue to step 1860 (Figure 18). If a no result in step 1738, then continue to step 1850 (Figure 18).

Referring to Figure 18, step 1850 determines whether the weight value may be below a maximum association threshold. If a no result in step 1850, then step 1854 may identify a new touch location for tracking. If a yes result in step 1850, then step 1852 may assign a new temporary location to the current location and then continue to step 1856. Step 1860 determines whether the next worst weight value for the current location may be less than the next worst weight value for the assigned location. If a yes result in step 1860, then step 1862 may set the temporary location to the next worst location and thereafter continue to step 1856. If a no result in step 1860, then step 1864 may set the assigned location to the next worst weight value. Step 1866 may select a moved assignment location and thereafter return to step 1722 (Figure 17). Step 1856 determines whether there may be more current touch locations. If a yes result in step 1856, then step 1858 may select the next current touch location and thereafter return to step 1722 (Figure 17).

Referring to Figure 19, depicted is a column cache process flow diagram, according to specific example embodiments of this disclosure. Step 1902 may received a mutual scan location request. Step 1904 determines whether the mutual scan area location requested may be stored in the cache memory. If a yes result in step 1904, then step 1920 determines whether the mutual scan data stored in the cache memory may be valid. If a yes result in step 1920, then step 1922 may return mutual scan data to the cache memory. If a no result in step 1920, then step 1918 may perform a mutual scan at the requested location, wherein step 1916 may write the mutual scan data to a location in the cache memory and then return back to step 1922.

If a no result in step 1904, then step 1906 determines whether the requested touch location may be beyond the right edge of the cache. If a yes result in step 1906, then step 1908 may de-allocate the left-most column of mutual scan data from the cache memory. In step 1910 the de-allocated mutual scan data may be allocated to the right edge of the cache memory so as to move the edge values thereof, and thereafter return to step 1904. If a no result in step 1906, then step 1914 may de-allocate the right-most column of data from the cache memory. In step 1912 the de-allocated mutual scan data may be allocated to the left edge of the cache memory so as to move the edge values thereof, and thereafter return to step 1904.

While embodiments of this disclosure have been depicted, described, and are defined by reference to example embodiments of the disclosure, such references do not imply a limitation on the disclosure, and no such limitation is to be inferred. The subject matter disclosed is capable of considerable modification, alteration, and equivalents in form and function, as will occur to those ordinarily skilled in the pertinent art and having the benefit of this disclosure. The depicted and described embodiments of this disclosure are examples only, and are not exhaustive of the scope of the disclosure.

CLAIMS

What is claimed is:

1. A method for decoding multiple touches on a touch sensing surface, said method comprising the steps of:

scanning a plurality of channels aligned on an axis for determining self values of the channels;

comparing the at least one self value to determine which one of the channels is a local maximum self value;

scanning a plurality of nodes of the at least one channel that has the local maximum self value for determining mutual values of the nodes; and

comparing the mutual values to determine which one of the nodes has the largest mutual value, wherein the node having the largest mutual value on the local maximum self value channel is a potential touch location.

2. The method according to claim 1, further comprising the steps of:

determining if at least one of the self values is greater than a self touch threshold, wherein

if yes then continue to the step of scanning a plurality of nodes of the at least one channel having the largest self value, and

if no then end a touch detection frame as completed.

3. The method according to claim 1, further comprising the step of:

determining left and right slope values for the at least one self value, wherein:

the left slope value is equal to the at least one self value minus a self value of a channel to the left of the at least one channel, and

the right slope value is equal to the at least one self value minus a self value of a channel to the right of the at least one channel.

4. The method according to claim 3, further comprising the steps of:

determining if the left slope value is greater than zero (0) and the right slope value is less than zero (0), wherein

if yes then return to the step of scanning the plurality of nodes of the at least one channel, and

if no then continue to next step;

determining if the left slope value is greater than zero (0) and greater than the right slope value, wherein

if yes then return to the step of scanning the plurality of nodes of the at least one channel, and

if no then continue to next step;

determining if the left slope value is less than zero (0) and greater than a percentage of the right slope value, wherein

if yes then return to the step of scanning the plurality of nodes of the at least one channel, and

if no then continue to next step;

determining if there is another self value, wherein

if yes then return to the step of determining if at least one of the self values is greater than the self touch threshold value using the another self value, and

if no then end a touch detection frame as completed.

5. The method according to claim 2, further comprising the steps of:

determining if at least one of the mutual values is greater than a mutual touch threshold, wherein

if yes then continue to the step of scanning a plurality of nodes of the at least one channel having the largest self value, and

if no then end the touch detection frame as completed.

6. The method according to claim 5, further comprising the steps of:

determining a next slope value, wherein the next slope value is equal to a current mutual value minus a next mutual value of a next node; and

determining a previous slope value, wherein the previous slope value is equal to the current mutual value minus a previous mutual value of a previous node.

7. The method according to claim 6, further comprising the steps of:

determining if the next slope value is less than zero (0) and the previous slope value is greater than zero (0), wherein

if yes then begin the step of validating the node, and

if no then continue to next step;

determining if the next slope value is greater than zero (0) and less than a percentage of the previous slope value, wherein

if yes then begin the step of validating the node, and

if no then continue to next step;

determining if the next slope value is less than zero (0) and greater than the previous slope value, wherein

if yes then begin the step of validating the node, and

if no then continue to next step;

determining if there is another mutual value, wherein

if yes then return to the step of determining if at least one of the mutual values is greater than the mutual touch threshold, and

if no then continue to the next step; and

determining if there is another self value, wherein

if yes then examine another self value and return to the step of determining if at least one of the self values is greater than a self touch threshold, and

if no then end the touch detection frame as completed.

8. The method according to claim 7, wherein the step of validating the node comprises the steps of:

identifying the node having a local maximum mutual value as a current node;

determining if there is a valid node north of the current node, wherein

if no then continue to the step of determining if there is a valid node south of the current node, and

if yes then perform a mutual measurement on the north node and continue to the next step;

determining if the north node is greater then the current node,

if yes then make the north node the current node and continue to the step of determining whether a touch point already exists at this node, and

if no then continue to the next step;

determining if there is a valid node south of the current node, wherein

if no then continue to the step of determining if there is a valid node east of the current node, and

if yes then perform a mutual measurement on the south node and continue to the next step;

determining if the south node is greater then the current node, wherein

if yes then make the south node the current node and continue to the step of determining whether a touch point already exists at this node, and

if no then continue to the next step;

determining if there is a valid node east of the current node, wherein

if no then continue to the step of determining if there is a valid node west of the current node, and

if yes then perform a mutual measurement on the east node and continue to the next step;

determining if the east node is greater then the current node,

if yes then make the east node the current node and continue to the step of determining whether a touch point already exists at this node, and

if no then continue to the next step;

determining if there is a valid node west of the current node, wherein

if no then continue to the step of determining if there is a valid node left of the current node, and

if yes then perform a mutual measurement on the west node and continue to the next step;

determining if the west node is greater then the current node,

if yes then make the west node the current node and continue to the step of determining whether a touch point already exists at this node, and

if no then continue to the next step;

determining if there is a valid node left of the current node, wherein

if no then define a left mutual value as a center mutual value minus a right mutual value and continue to the step of determining a fine position for the node, and

if yes then perform a mutual measurement on the left node and continue to the next step;

determining if there is a valid node right of the current node, wherein

if no then define the mutual value as the center mutual value minus the left mutual value and continue to the step of determining the fine position for the node, and

if yes then perform a mutual measurement on the right node and continue to the next step;

defining a fine position of the node by subtracting the left value from the right value, dividing this difference by the center value and multiplying the result thereof by 64 and continue to the next step; and

determining whether interpolation was performed for each axis, wherein

if yes, then add another touch point to a list of all detected touch points and return to the step of determining if there are additional mutual values, and

if no, then interpolate an other axis by using left and right nodes of the other axis for starting again at the step of determining if there is a valid node left of the current node.

9. A method for tracking previously found and current touch locations on a touch sensing surface, said method comprising the steps of:

determining if there is at least one current touch location, wherein

if yes then select one of the current touch locations, and

if no then continue to next step;

determining if there is at least one previous touch location, wherein

if no then end tracking, and

if yes then select one of the previous touch locations and continue to next step;

determining if the previous touch location is associated with the current touch location, wherein

if no then a touch is no longer present at the previous touch location, stop tracking that previous touch location and continue to the step of determining if there is at least one more previous touch location, and

if yes continue to the next step; and

determining if there is at least one more previous touch location, wherein

if yes then select a next previous touch location and continue to the step of determining if the previous touch location is associated with the current touch location using the next previous touch location for the previous touch location, and

if no then output touch locations tracked.

10. The method according to claim 9, wherein the step of selecting one of the current touch locations further comprises the steps of:

determining if there is at least one previous touch location, wherein

if no then new touch to track at current touch location and continue to the step of determining if there is at least one more current touch location, and

if yes then set a temporary weight value to a maximum weight value, select a previous touch location and continue to next step;

measuring a distance between the selected current touch location and the selected previous touch location, use this distance as a current weight value for determining pairing of the selected current touch location and the previous touch location, and continue to next step;

determining if the current weight value is less than the temporary weight value, wherein

if no then continue to the step of determining if there is at least one more previous touch location, and

if yes then set the temporary weight value to the current weight value, record the selected previous touch location as a temporary touch location and continue to next step;

determining if there is at least one more previous touch location, wherein

if yes then select the next previous touch location and return to the step of measuring the distance between the selected current touch location and the selected previous touch location, and

if no then continue to next step;

determining if the temporary location is already assigned to a different current location, wherein,

if yes then calculate a next worst weight value for the current location and for an assigned current location then continue to the step of determining if the next worst weight value for the current location is less than the next worst weight value for the assigned location, and

if no then continue to next step;

determining if the weight value is below a maximum association threshold, wherein

if yes then assigning the temporary location to the current location and continue to the step of determining if there is at least one more current touch location, and

if no then a new touch location is identified for tracking thereof and continue to next step;

determining if there is at least one more current touch location, wherein

if no then return to the step of determining if there is at least one other previous touch locations, and

if yes then select a next current touch location and return to the step of determining if there is at least one previous touch location;

determining if the next worst weight value for the current location is less than the next worst weight value for the assigned location, wherein

if yes then setting the temporary location to the next worst location and returning to the step of determining if there is at least one more current touch location, and

if no then setting the assigned location to the next worst weight value, selecting a moved assignment location and returning to the step of determining if there is at least one previous touch location.

11. A method for caching mutual touch values of a plurality of touch columns, said method comprising the steps of:

receiving a mutual scan location request;

determining if a cache memory contains scan data of the requested mutual scan location, wherein

if yes then continue to the step of determining if the scan data is valid, and

if no then continue to next step;

determining if the requested mutual scan location is beyond a right edge of the cache memory, wherein

if yes then de-allocate the scan data in a left-most column of the cache memory, allocate the de-allocated scan data to a right edge of the cache memory and invalidate values thereof, and

if no then de-allocate the scan data in a right-most column of the cache memory, allocate the de-allocated scan data to a left edge of the cache memory and invalidate values thereof;

determining if the scan data is valid, wherein

if yes then return the requested the scan data for further processing thereto, and

if no then perform a mutual scan at the requested location, place the resulting scan data in the cache memory and return the requested scan data for further processing thereto.

12. A system for decoding multiple touches using the method according to claim 1, said system comprising:

a first plurality of electrodes arranged in a parallel orientation having a first axis, wherein each of the first plurality of touch electrodes has a self capacitance;

a second plurality of electrodes arranged in a parallel orientation having a second axis substantially perpendicular to the first axis, the first plurality of electrodes are located over the second plurality of electrodes to form a touch matrix wherein each of overlapping intersection of the first and second plurality of electrodes has a mutual capacitance;

the self capacitance is measured for each of the first plurality of electrodes to produce respective self values;

the mutual capacitance is measured for each of the overlapping intersections of the first and second plurality of electrodes to produce respective mutual values;

the self and mutual capacitances are measured by an analog front end of a microcontroller;

the self and mutual values are stored in a memory of the microcontroller; and

a digital processor in the microcontroller uses the self and mutual values in determining at least one location of at least one touch per touch acquisition frame, and tracks changing locations of the at least one touch in subsequent touch acquisition frames.

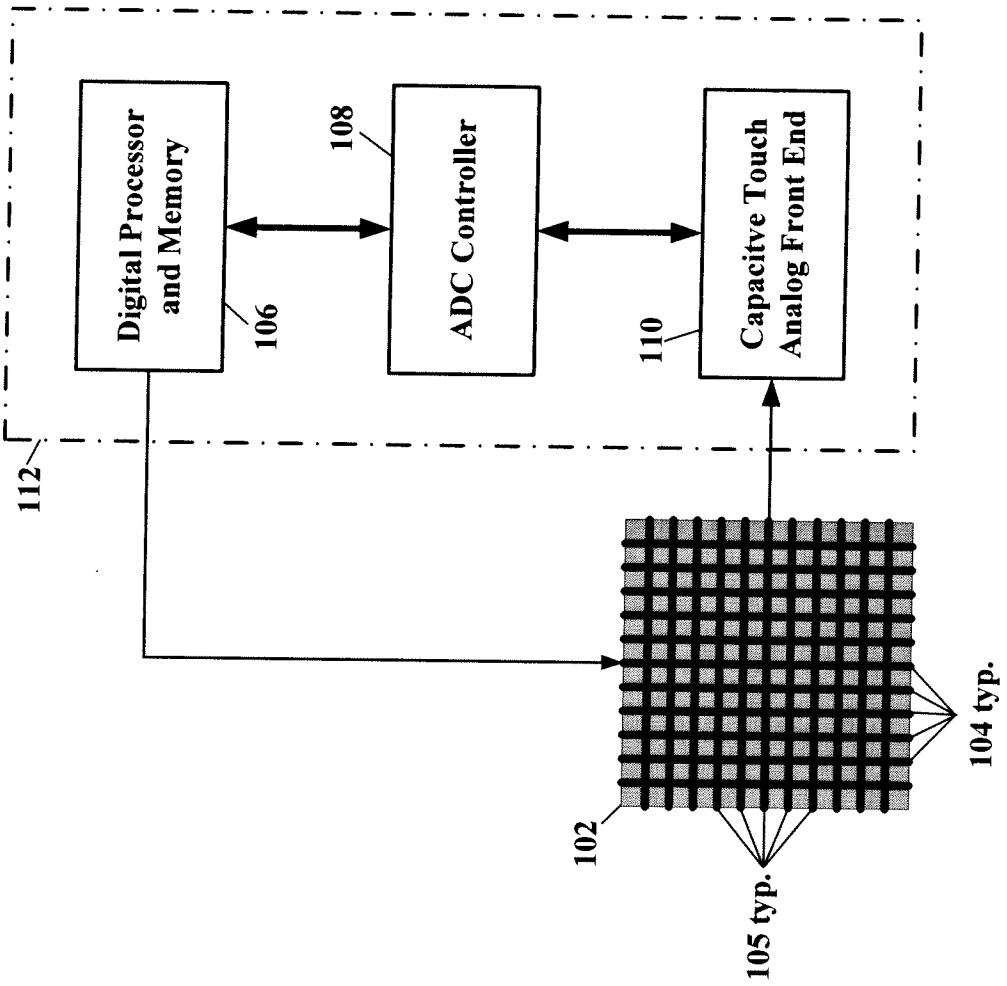
Digital Processor and Memory
106
108
ADC Controller
110
Capacitve Touch Analog Front End
112
102
104 typ.
105 typ.

Figure 1

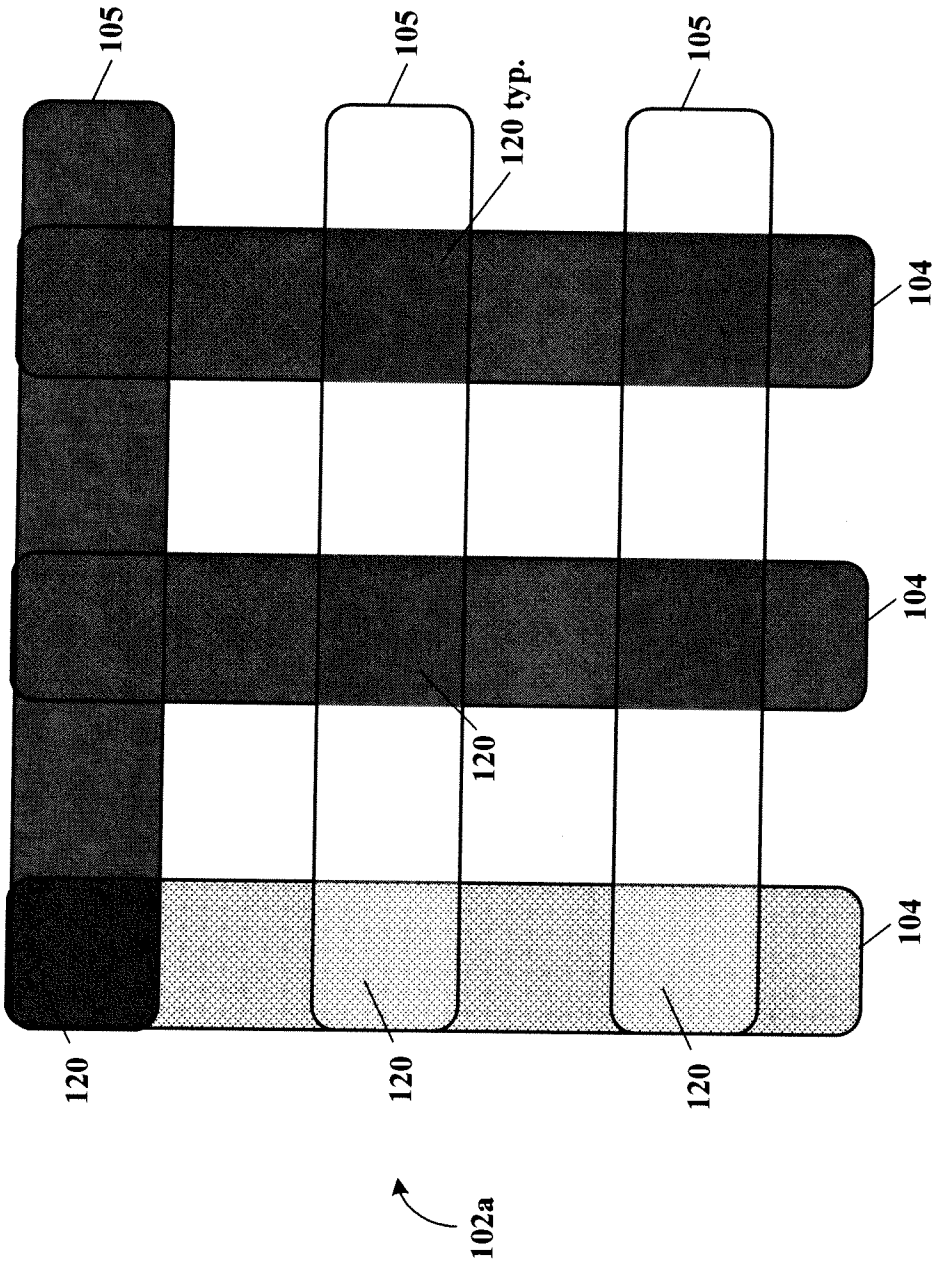

Figure 1A

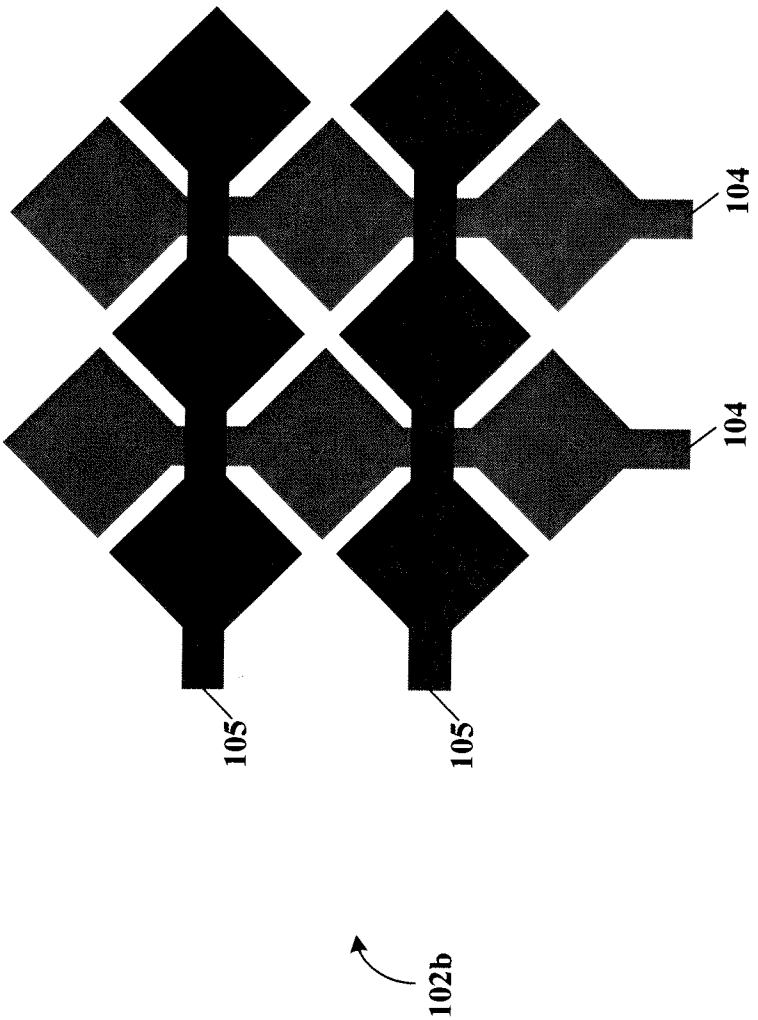

Figure 1B

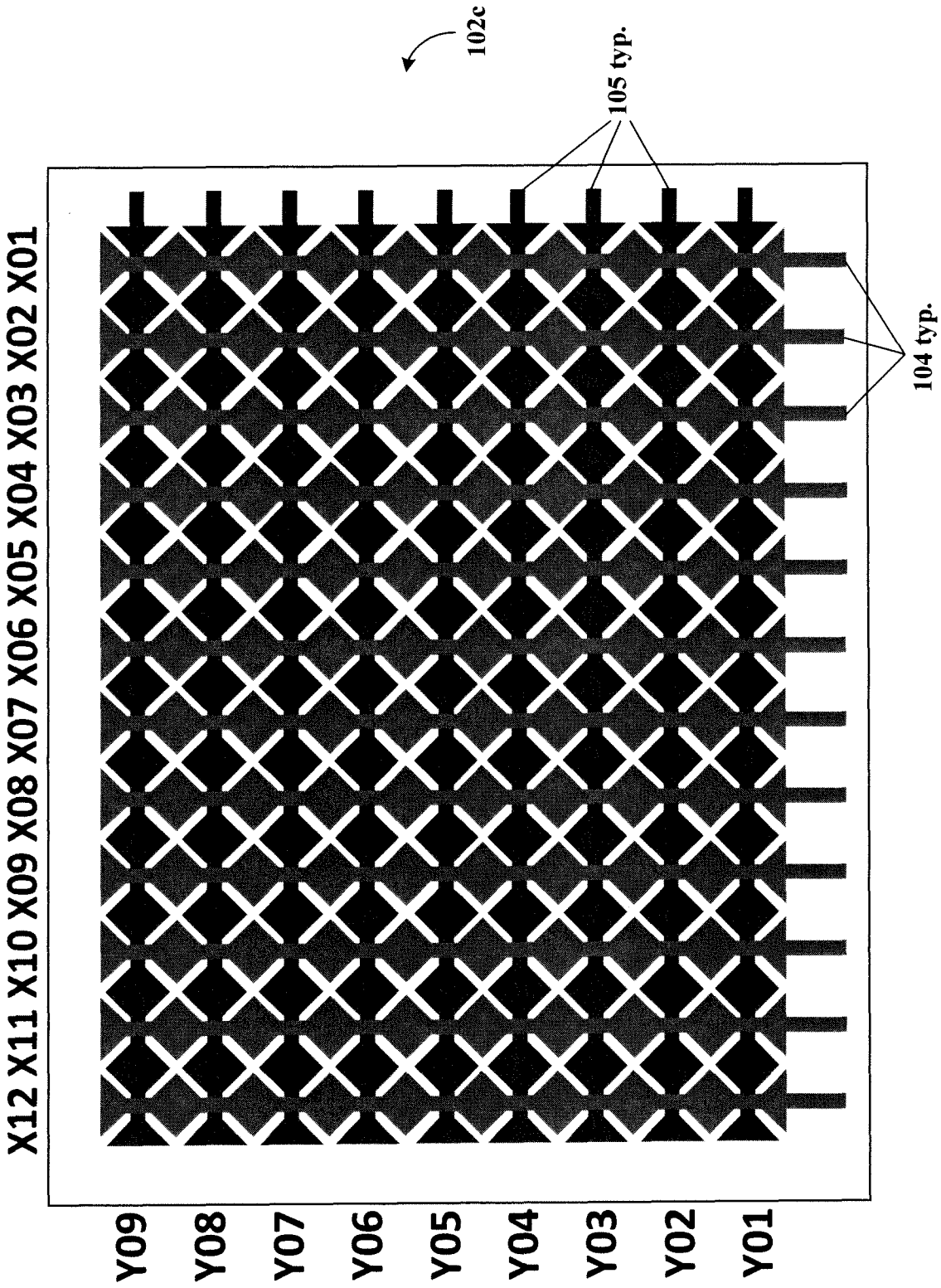
102c
105 typ.
104 typ.
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01
Y09 Y08 Y07 Y06 Y05 Y04 Y03 Y02 Y01

Figure 1C

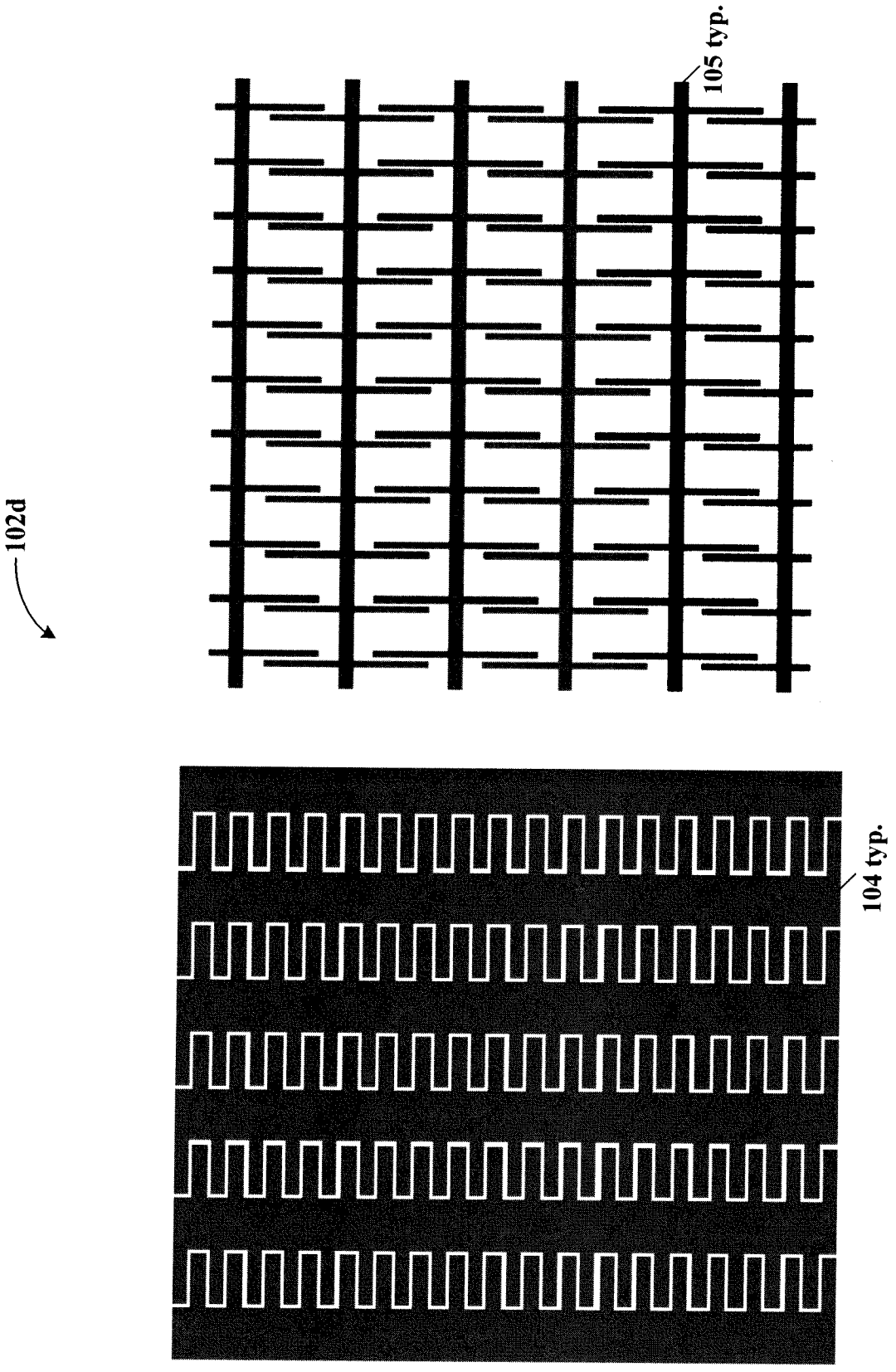

Figure 1D

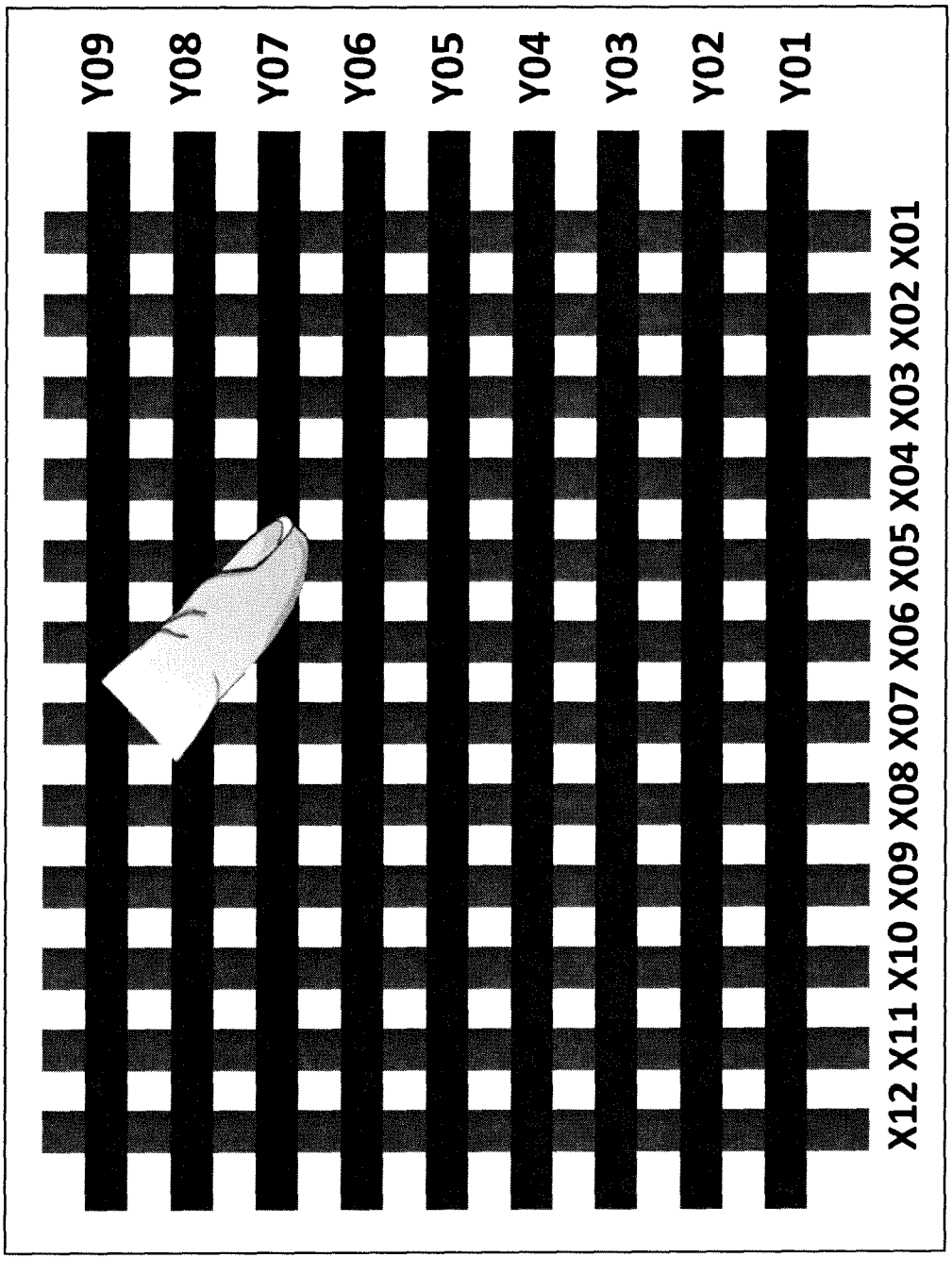
Y09
Y08
Y07
Y06
Y05
Y04
Y03
Y02
Y01
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01

Figure 1E

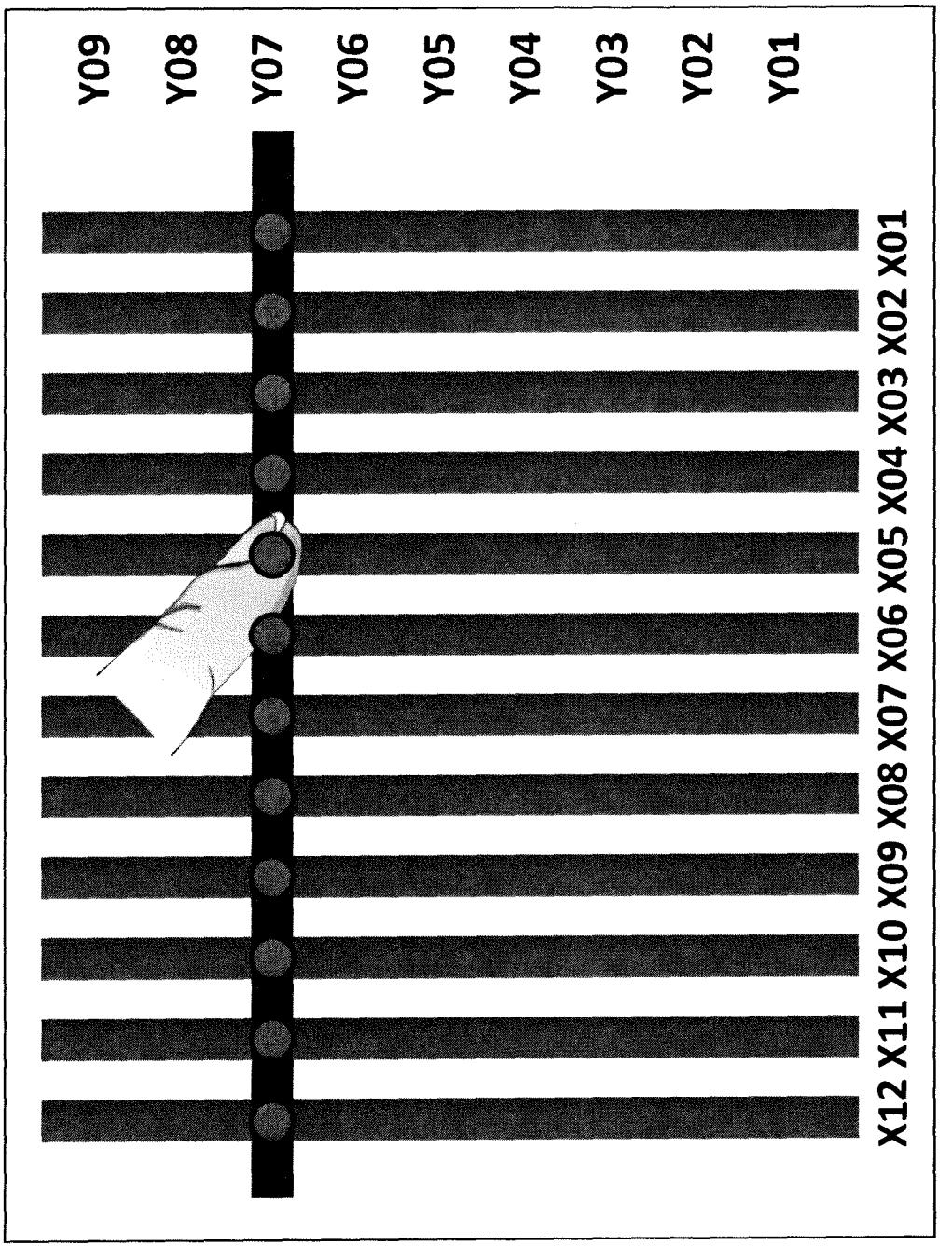
Y09
Y08
Y07
Y06
Y05
Y04
Y03
Y02
Y01
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01

Figure 1F

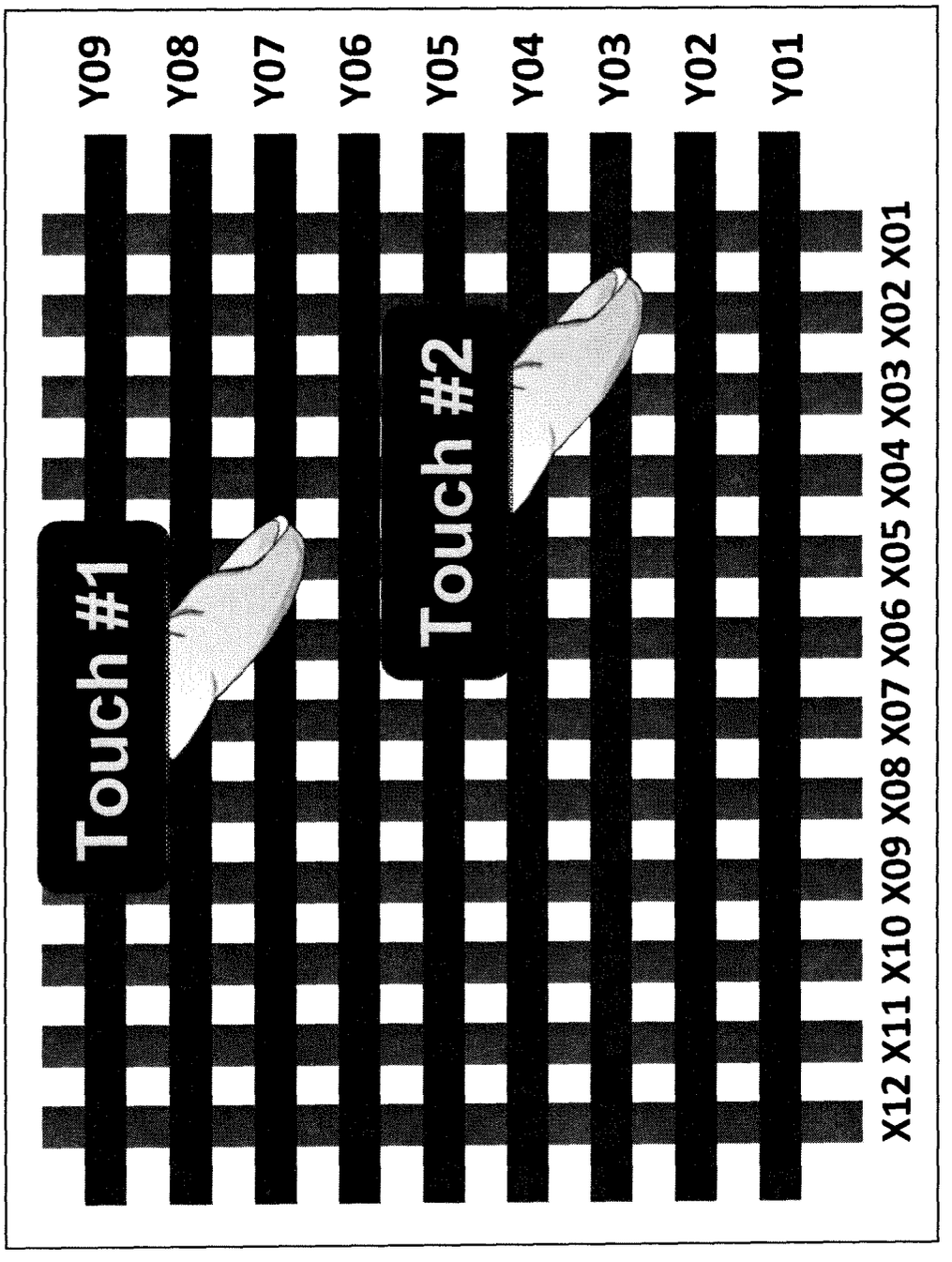
Y09
Y08
Y07
Y06
Y05
Y04
Y03
Y02
Y01
Touch #1
Touch #2
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01

Figure 1G

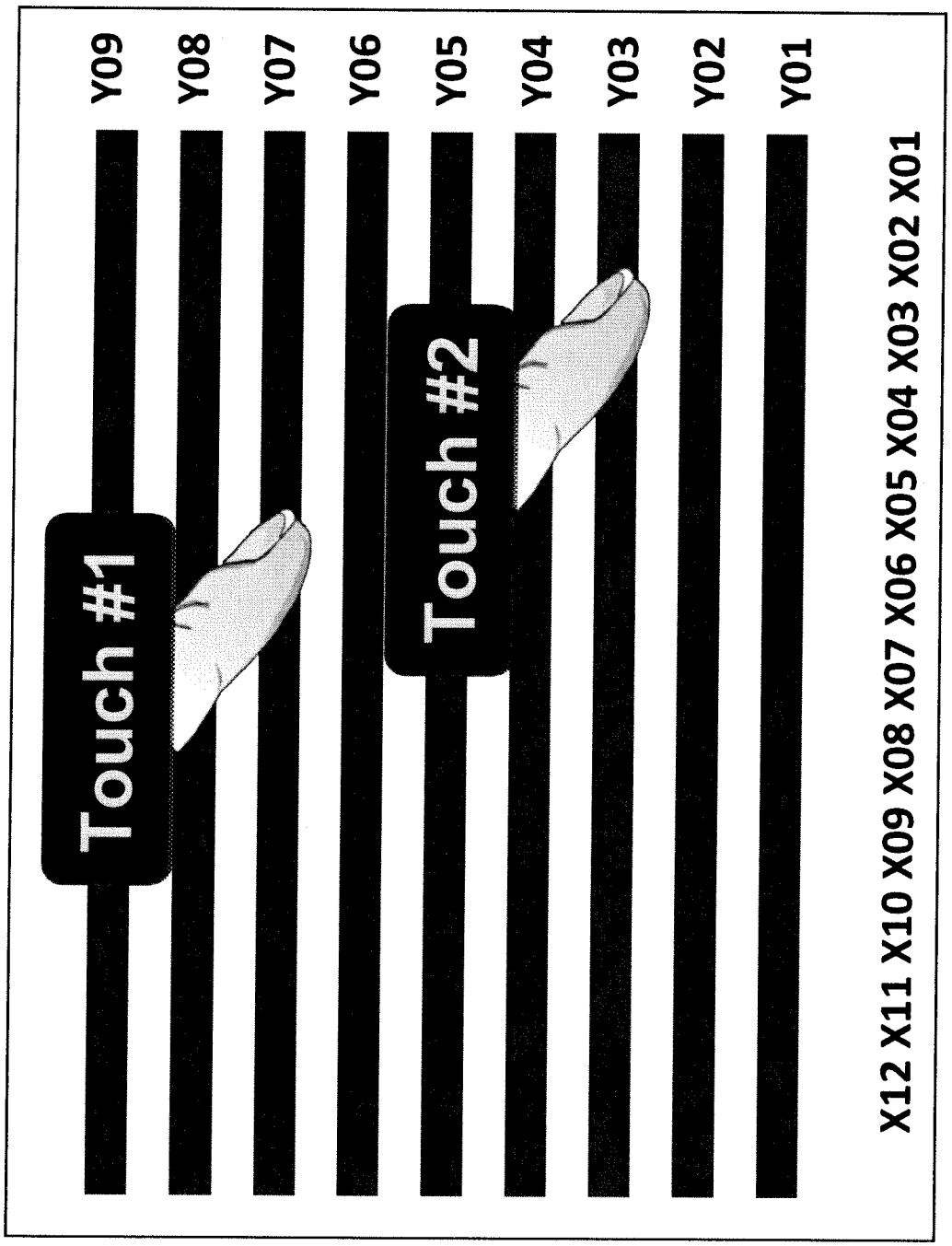
Y09
Y08
Y07
Y06
Y05
Y04
Y03
Y02
Y01
Touch #1
Touch #2
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01

Figure 1H

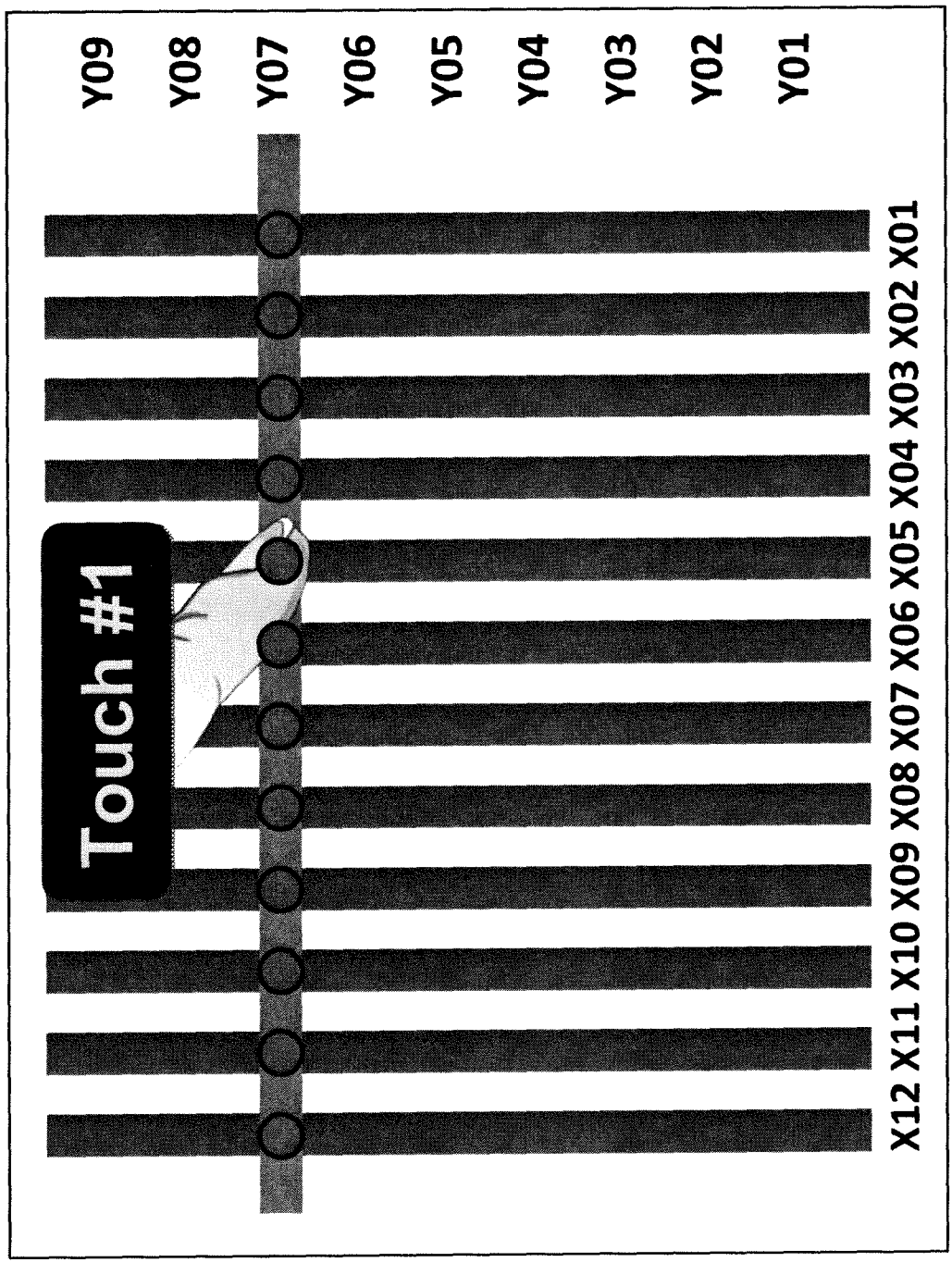
Y09
Y08
Y07
Y06
Y05
Y04
Y03
Y02
Y01
Touch #1
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01

Figure 1I

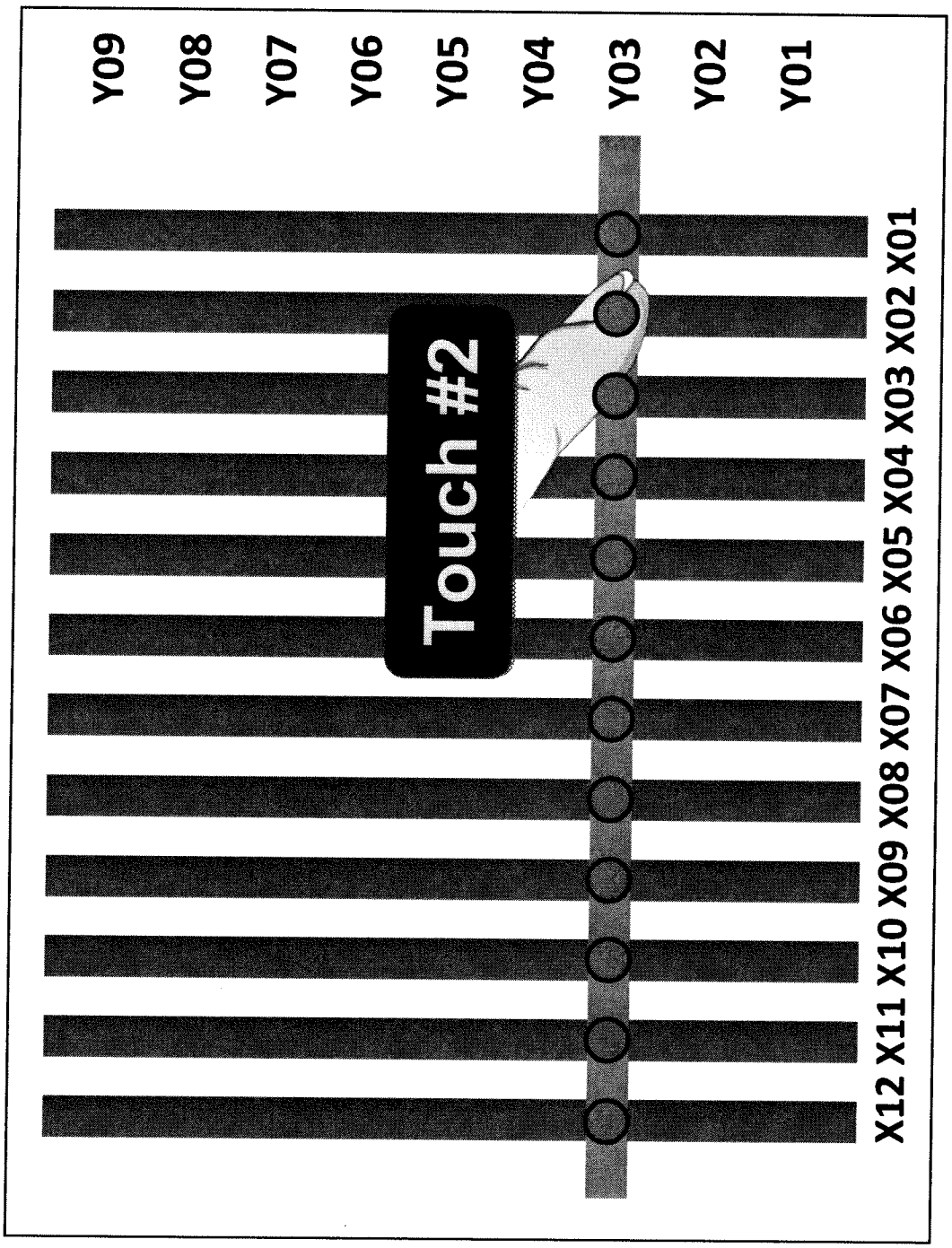
Y09
Y08
Y07
Y06
Y05
Y04
Y03
Y02
Y01
Touch #2
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01

Figure 1J

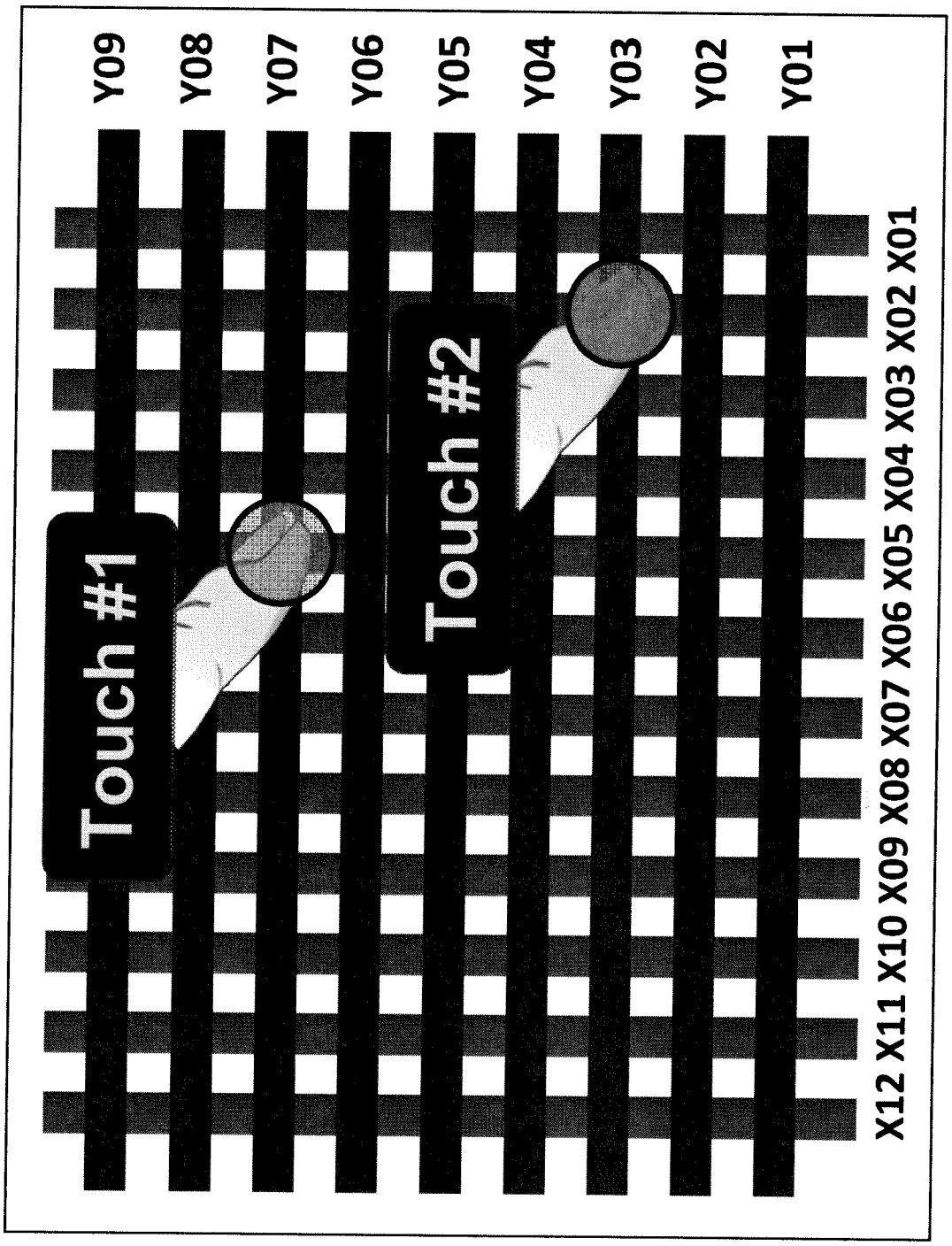
Y09
Y08
Y07
Y06
Y05
Y04
Y03
Y02
Y01
Touch #1
Touch #2
X12 X11 X10 X09 X08 X07 X06 X05 X04 X03 X02 X01

Figure 1K

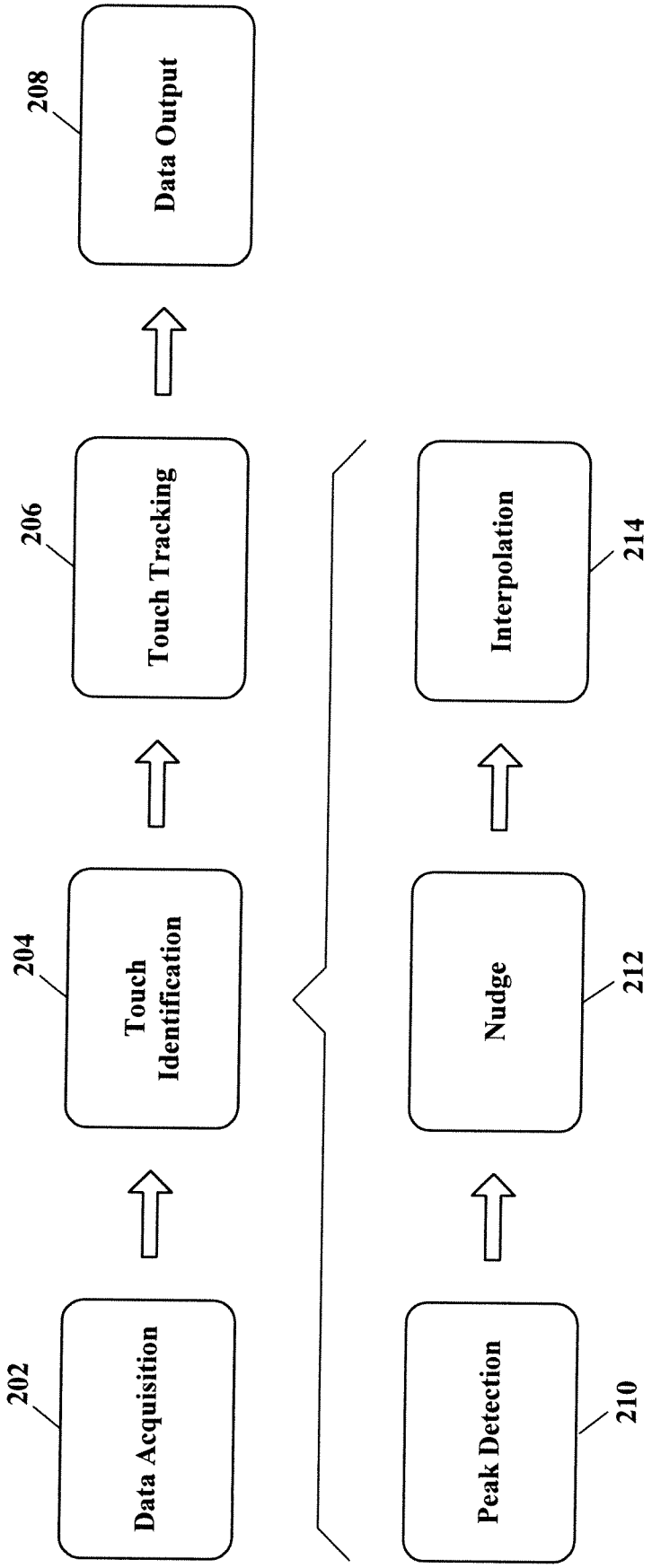
202
Data Acquisition
204
Touch Identification
206
Touch Tracking
208
Data Output
210
Peak Detection
212
Nudge
214
Interpolation

Figure 2

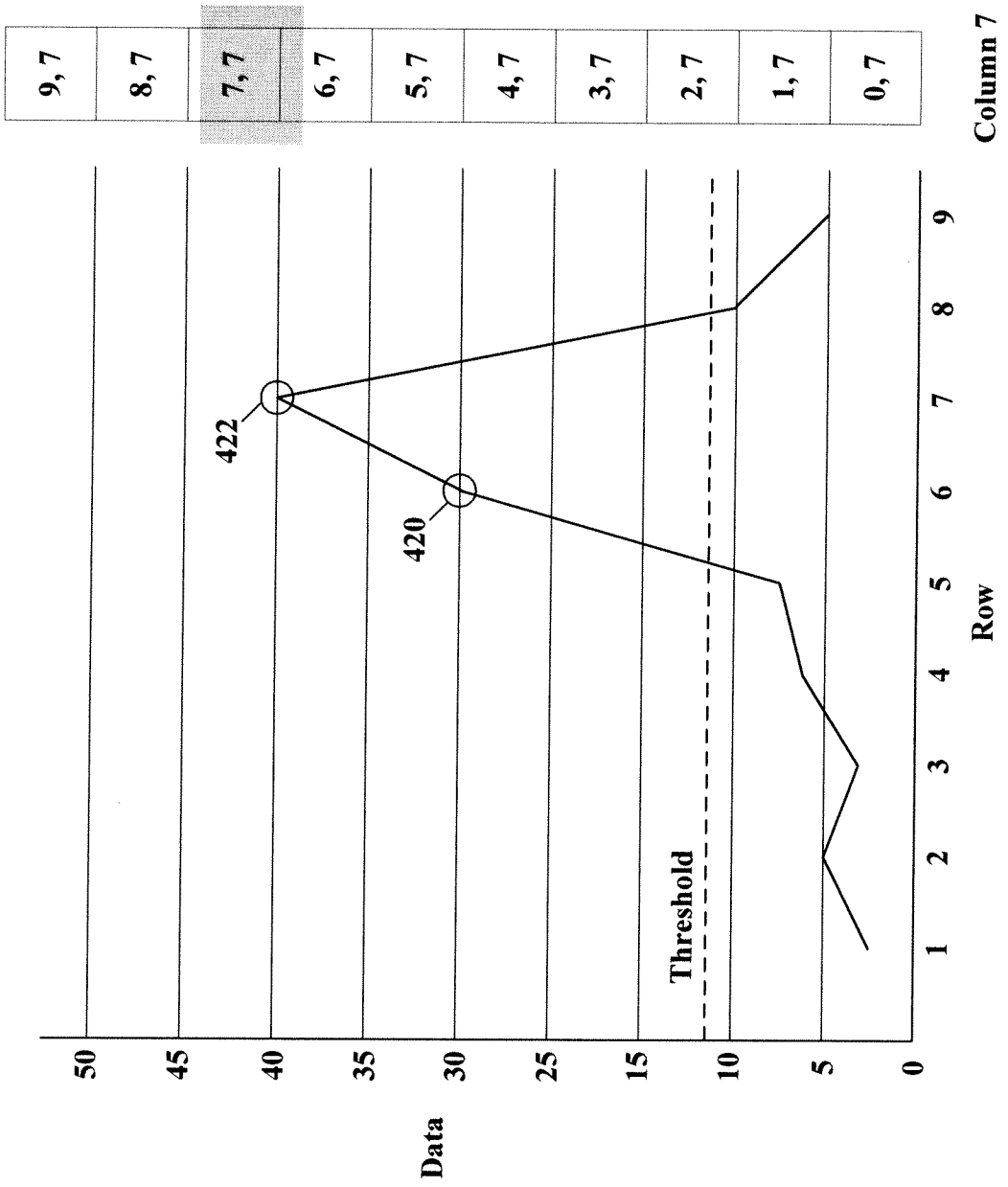
9, 7
8, 7
7, 7
6, 7
5, 7
4, 7
3, 7
2, 7
1, 7
0, 7
Column 7
422
420
Threshold
Data
Row
50
45
40
35
30
25
15
10
5
0
1
2
3
4
5
6
7
8
9

Figure 3

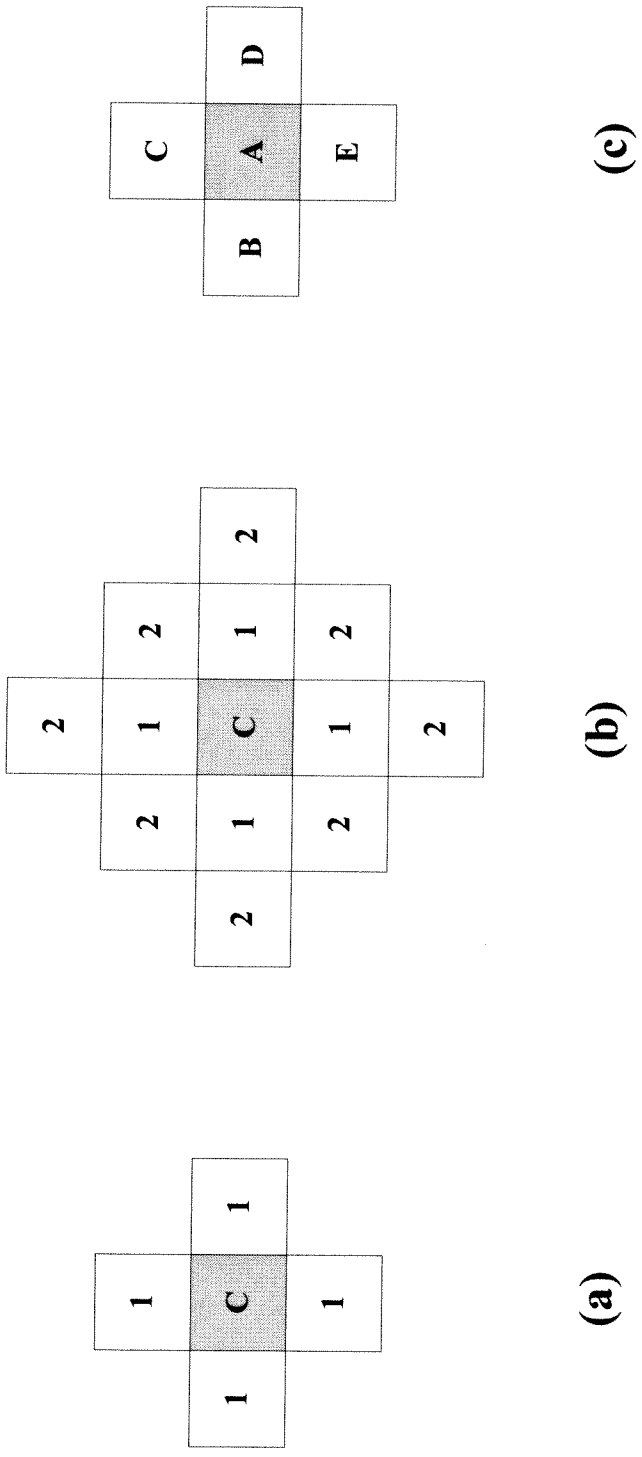

Figure 4

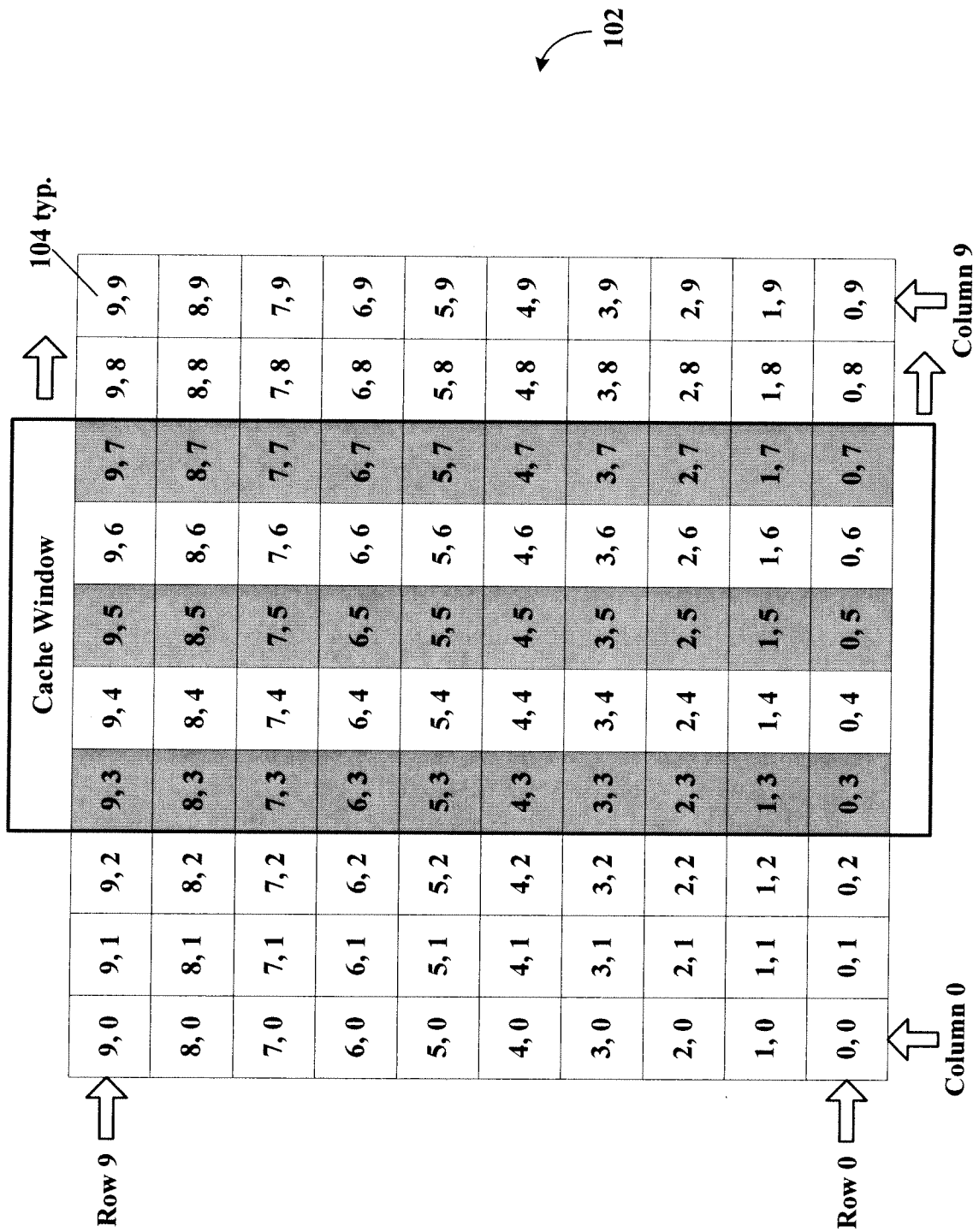
102
104 typ.
Cache Window
Row 9
Row 0
Column 0
Column 9
9,0 9,1 9,2 9,3 9,4 9,5 9,6 9,7 9,8 9,9
8,0 8,1 8,2 8,3 8,4 8,5 8,6 8,7 8,8 8,9
7,0 7,1 7,2 7,3 7,4 7,5 7,6 7,7 7,8 7,9
6,0 6,1 6,2 6,3 6,4 6,5 6,6 6,7 6,8 6,9
5,0 5,1 5,2 5,3 5,4 5,5 5,6 5,7 5,8 5,9
4,0 4,1 4,2 4,3 4,4 4,5 4,6 4,7 4,8 4,9
3,0 3,1 3,2 3,3 3,4 3,5 3,6 3,7 3,8 3,9
2,0 2,1 2,2 2,3 2,4 2,5 2,6 2,7 2,8 2,9
1,0 1,1 1,2 1,3 1,4 1,5 1,6 1,7 1,8 1,9
0,0 0,1 0,2 0,3 0,4 0,5 0,6 0,7 0,8 0,9

Figure 5

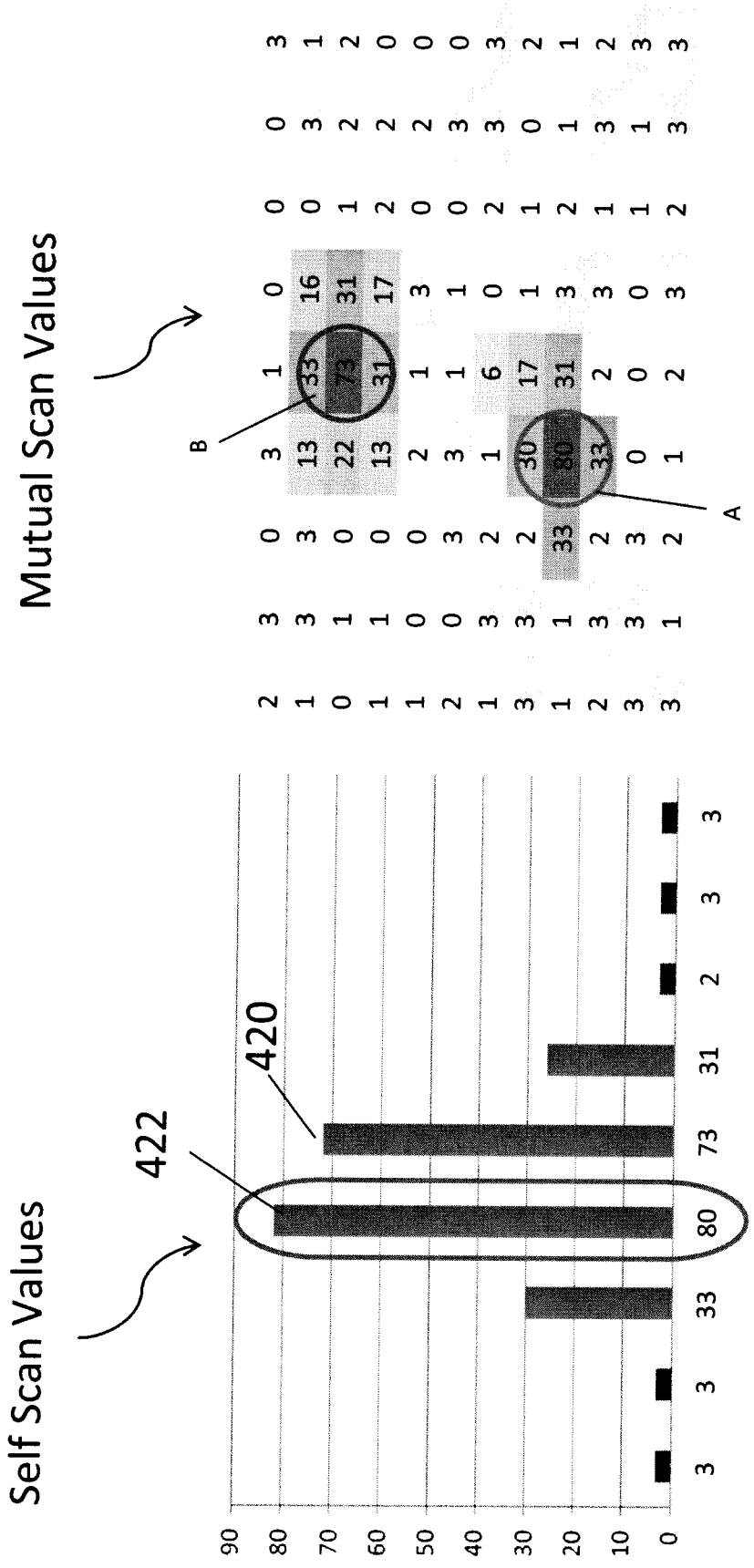
Mutual Scan Values
B
A
Self Scan Values
422
420
90
80
70
60
50
40
30
20
10
0

Figure 6

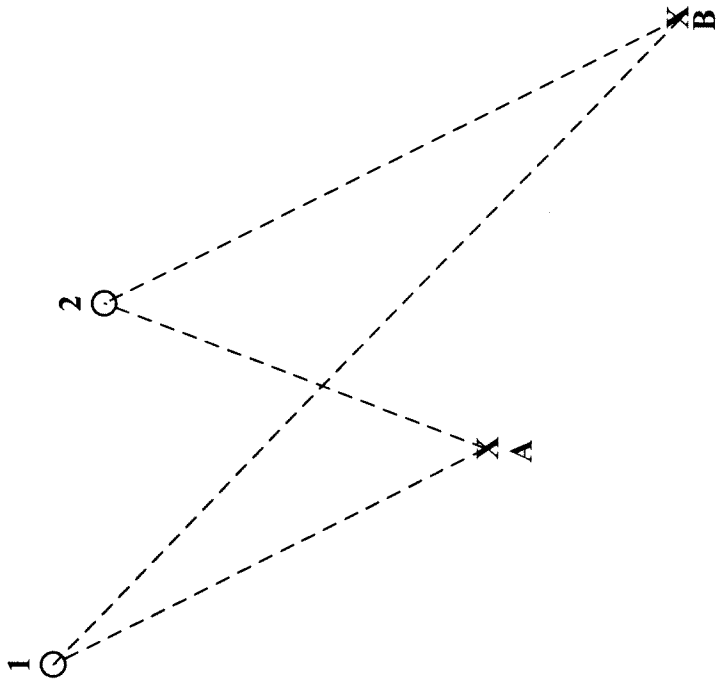

Figure 8

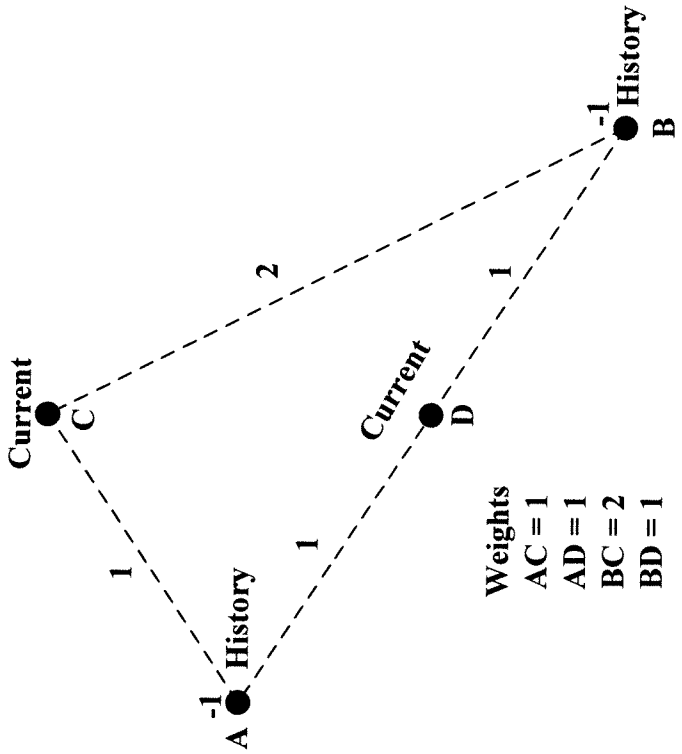

Figure 7

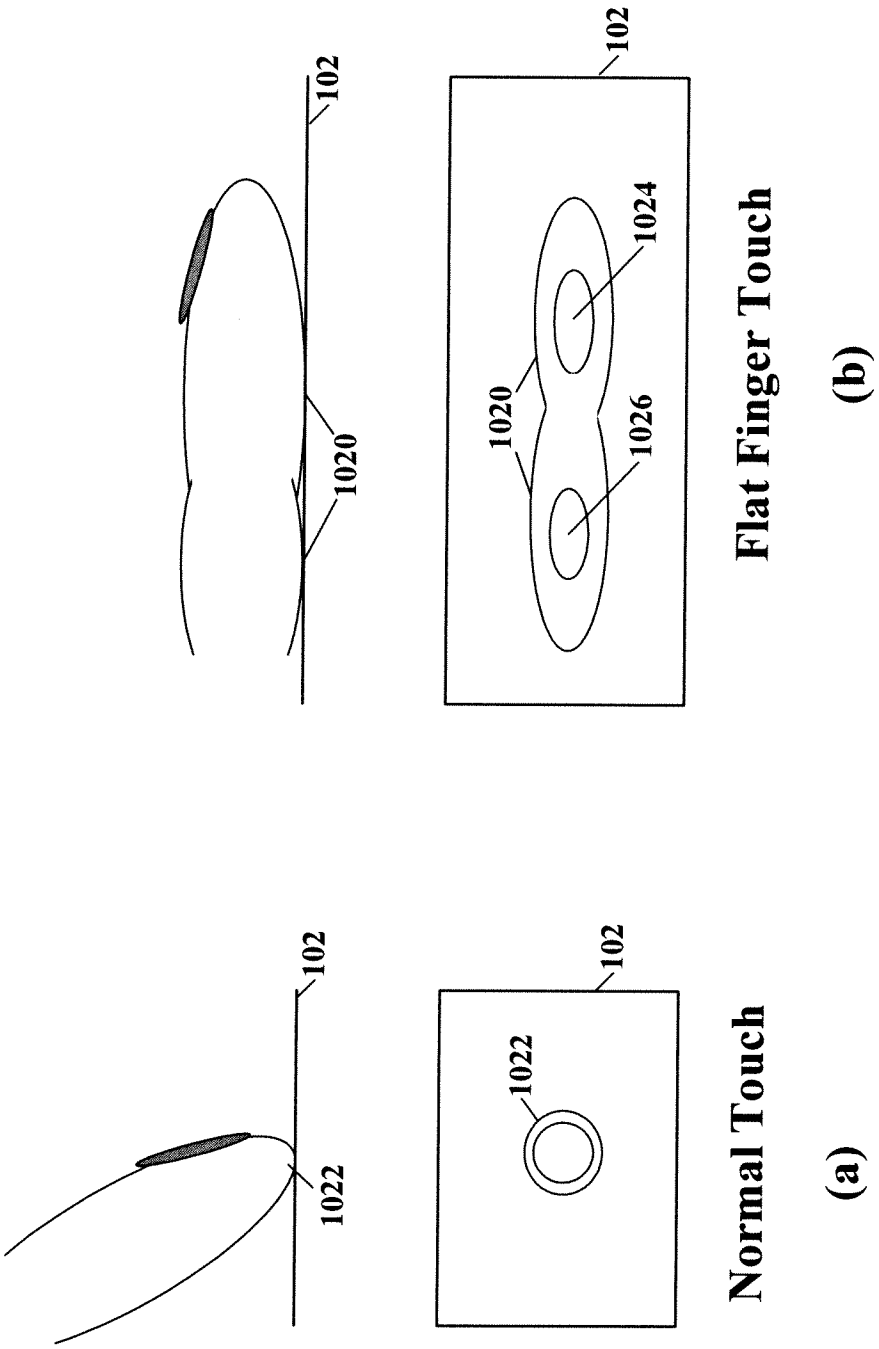
102
1020
102
1024
1020
1026
Flat Finger Touch
(b)
102
1022
102
1022
Normal Touch
(a)

Figure 9

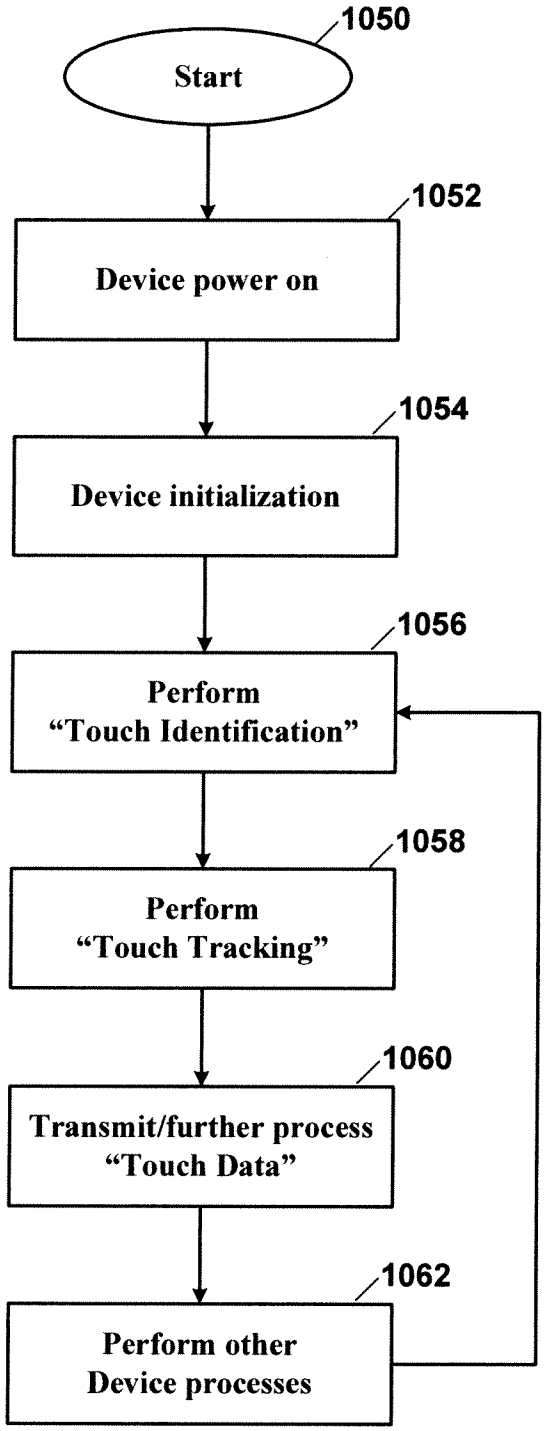
1050
Start
1052
Device power on
1054
Device initialization
1056
Perform
"Touch Identification"
1058
Perform
"Touch Tracking"
1060
Transmit/further process
"Touch Data"
1062
Perform other
Device processes

Figure 10

Touch Identification

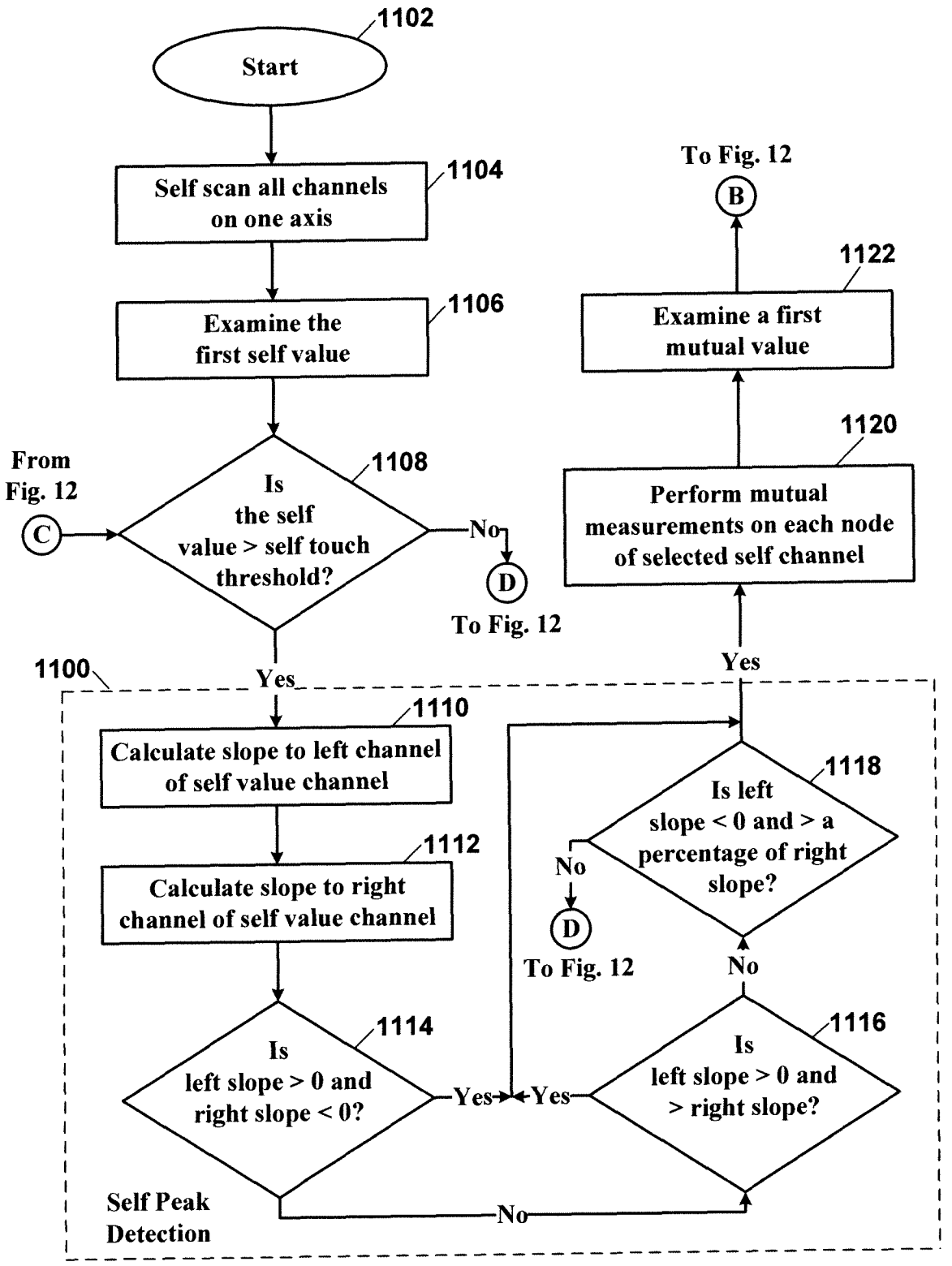

Figure 11

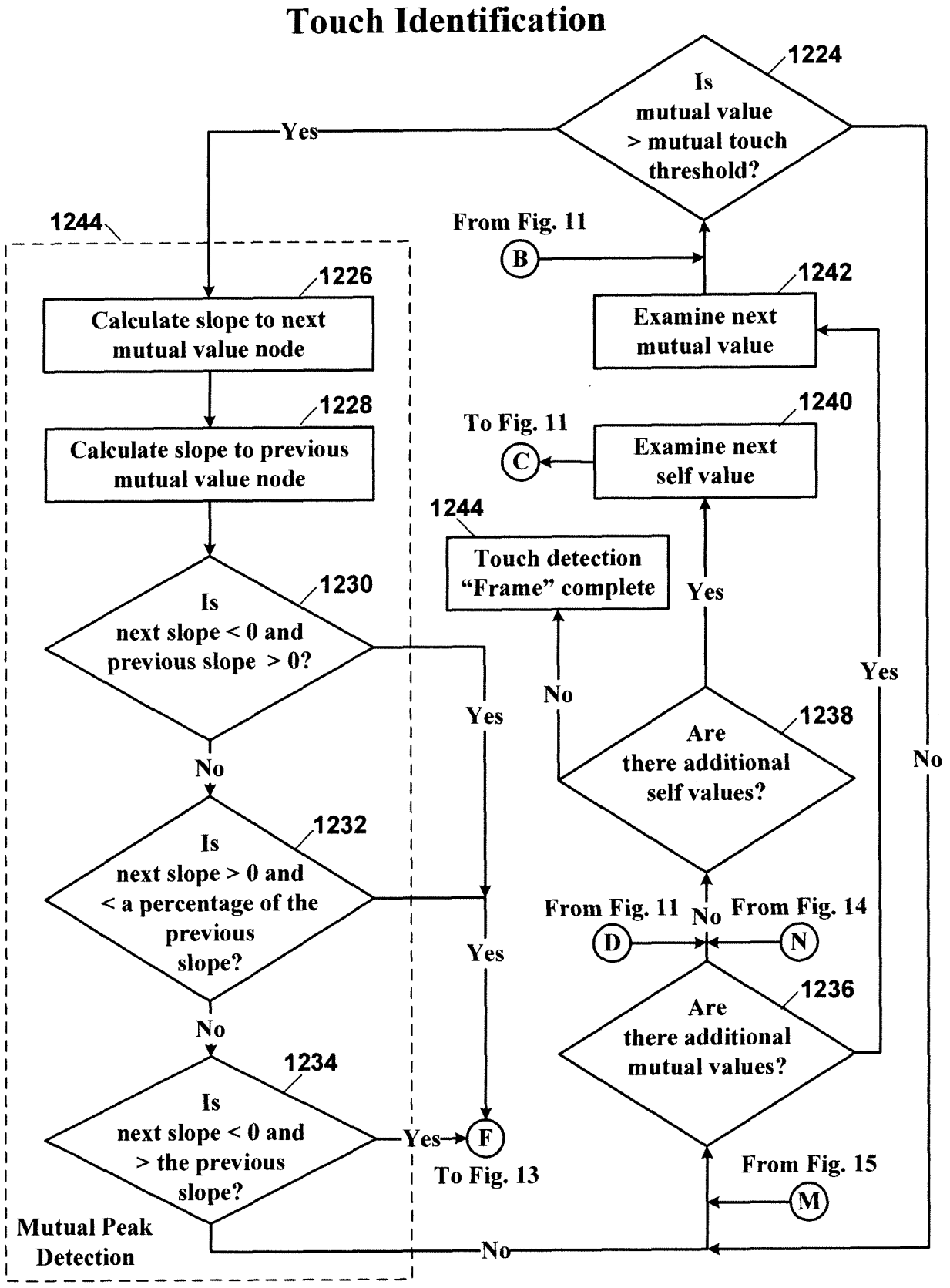
Touch Identification
1224
Is mutual value > mutual touch threshold?
Yes
No
1244
From Fig. 11
B
1226
Calculate slope to next mutual value node
1228
Calculate slope to previous mutual value node
1230
Is next slope < 0 and previous slope > 0?
Yes
No
1232
Is next slope > 0 and < a percentage of the previous slope?
Yes
No
1234
Is next slope < 0 and > the previous slope?
Yes
F
To Fig. 13
Mutual Peak Detection
1242
Examine next mutual value
1240
To Fig. 11
C
Examine next self value
1244
Touch detection "Frame" complete
Yes
No
1238
Are there additional self values?
From Fig. 11
D
No
From Fig. 14
N
1236
Are there additional mutual values?
Yes
No
From Fig. 15
M
No

Figure 12

Nudge & Interpolation

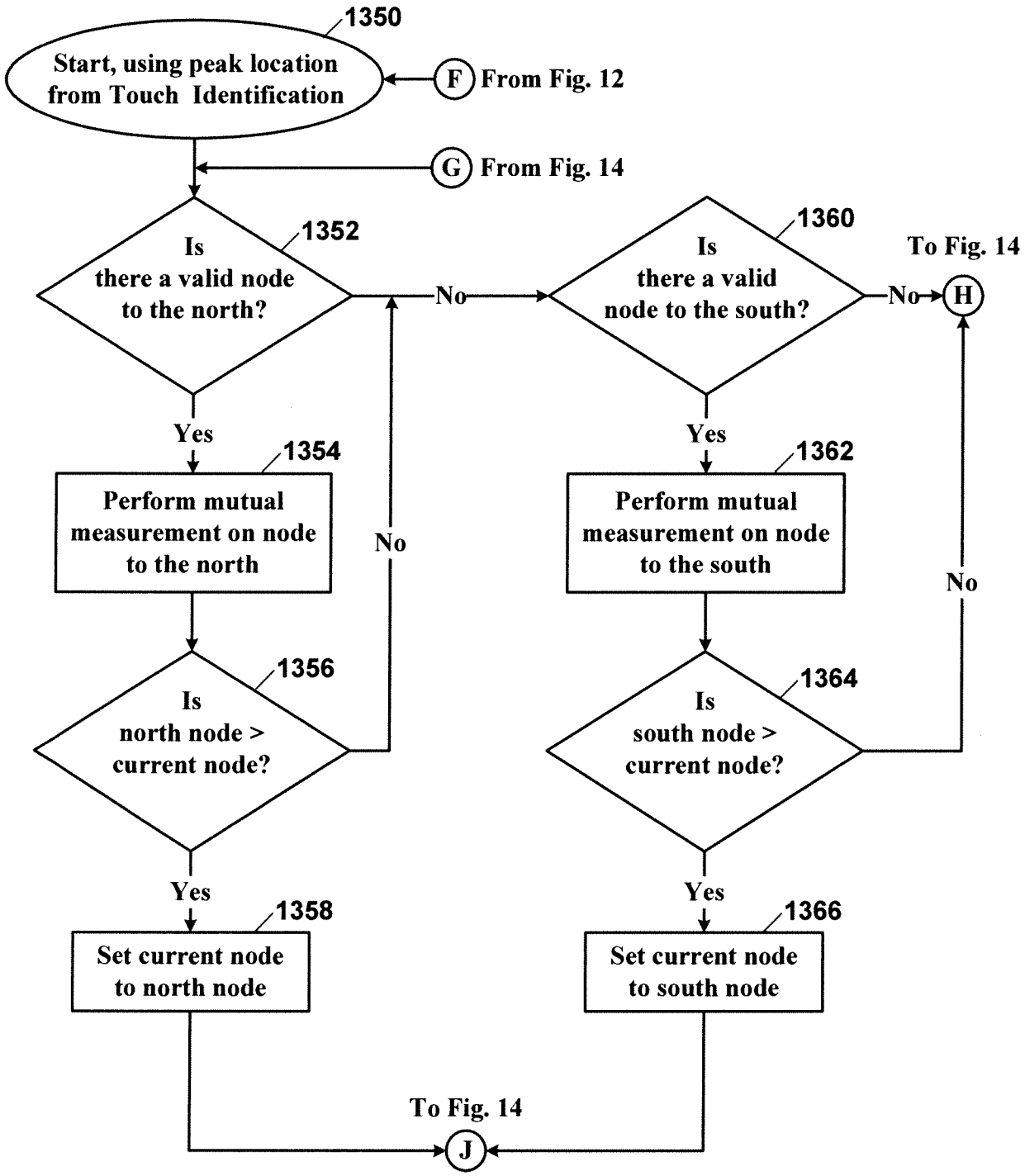

Figure 13

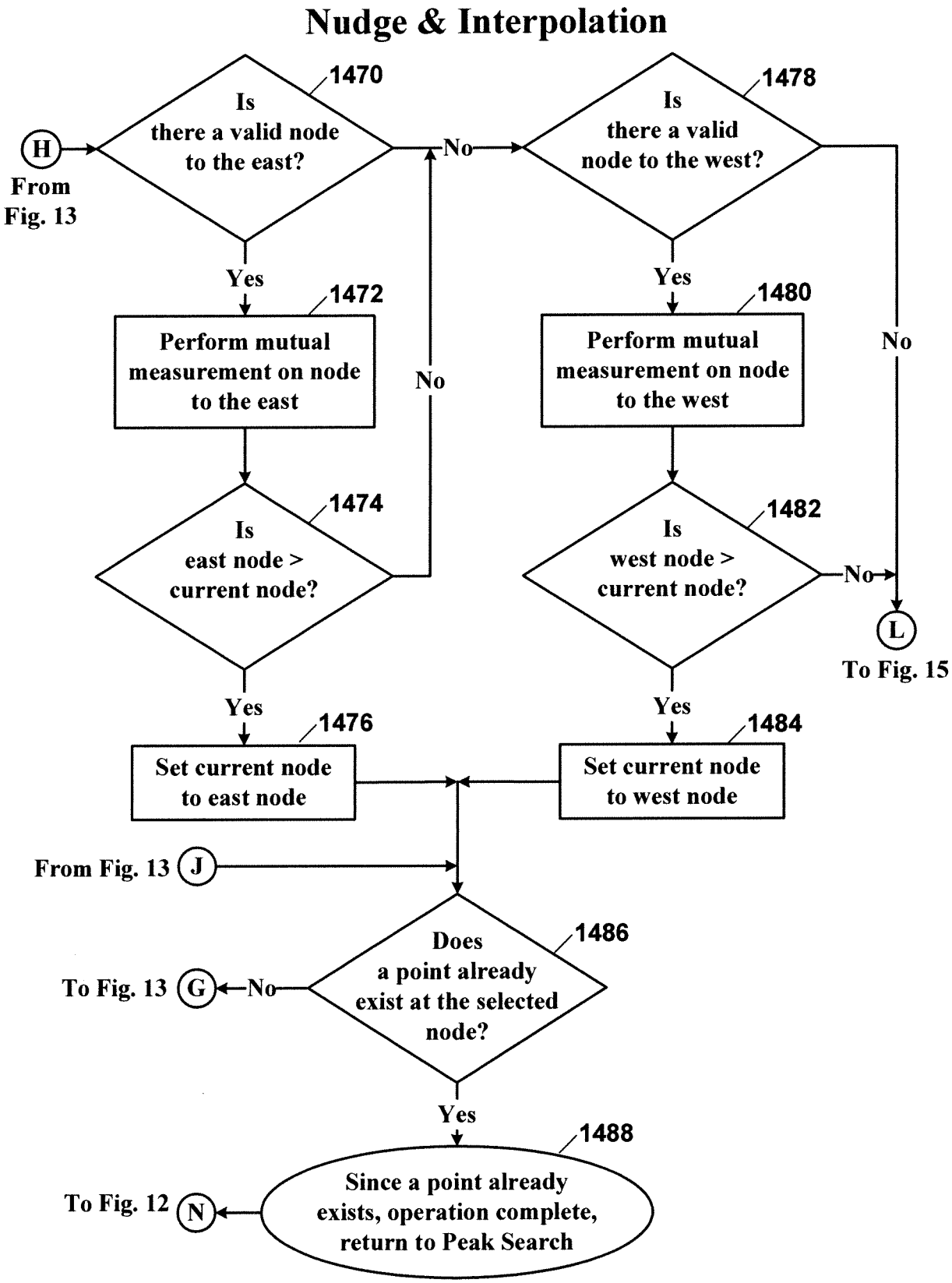
Nudge & Interpolation
1470
Is there a valid node to the east?
H
From Fig. 13
No
1478
Is there a valid node to the west?
Yes
1472
Perform mutual measurement on node to the east
No
1480
Perform mutual measurement on node to the west
No
1474
Is east node > current node?
No
1482
Is west node > current node?
No
L
To Fig. 15
Yes
1476
Set current node to east node
Yes
1484
Set current node to west node
From Fig. 13 J
1486
Does a point already exist at the selected node?
To Fig. 13 G
No
Yes
1488
Since a point already exists, operation complete, return to Peak Search
To Fig. 12 N

Figure 14

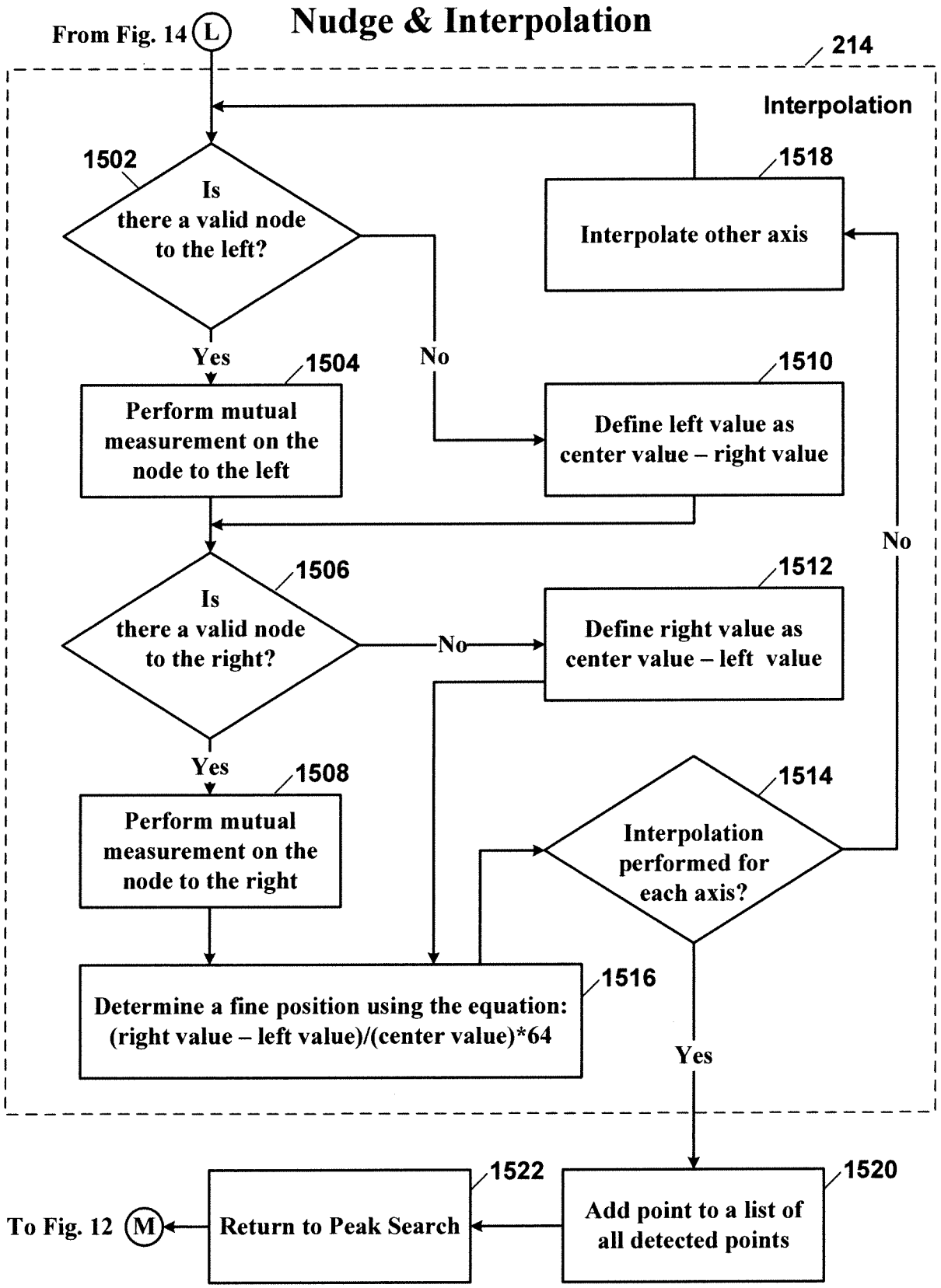
From Fig. 14
L
Nudge & Interpolation
214
Interpolation
1502
Is there a valid node to the left?
Yes
No
1504
Perform mutual measurement on the node to the left
1510
Define left value as center value – right value
1518
Interpolate other axis
No
1506
Is there a valid node to the right?
No
1512
Define right value as center value – left value
Yes
1508
Perform mutual measurement on the node to the right
1514
Interpolation performed for each axis?
1516
Determine a fine position using the equation: (right value – left value)/(center value)*64
Yes
1522
Return to Peak Search
1520
Add point to a list of all detected points
To Fig. 12
M

Figure 15

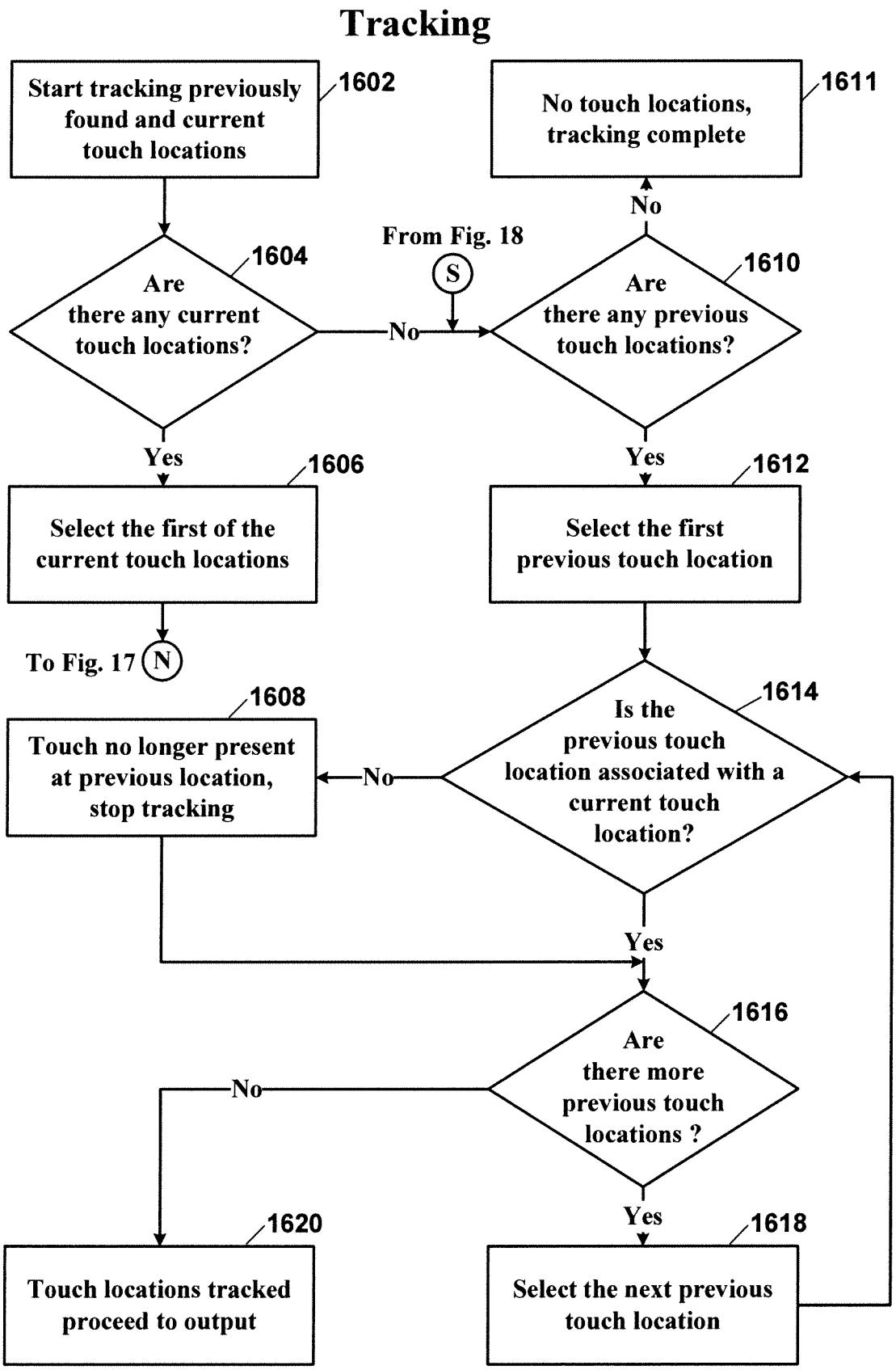
Tracking
Start tracking previously found and current touch locations
1602
Are there any current touch locations?
1604
Yes
Select the first of the current touch locations
1606
To Fig. 17
N
No
From Fig. 18
S
Are there any previous touch locations?
1610
No
No touch locations, tracking complete
1611
Yes
Select the first previous touch location
1612
Is the previous touch location associated with a current touch location?
1614
No
Touch no longer present at previous location, stop tracking
1608
Yes
Are there more previous touch locations ?
1616
No
Touch locations tracked proceed to output
1620
Yes
Select the next previous touch location
1618

Figure 16

Tracking

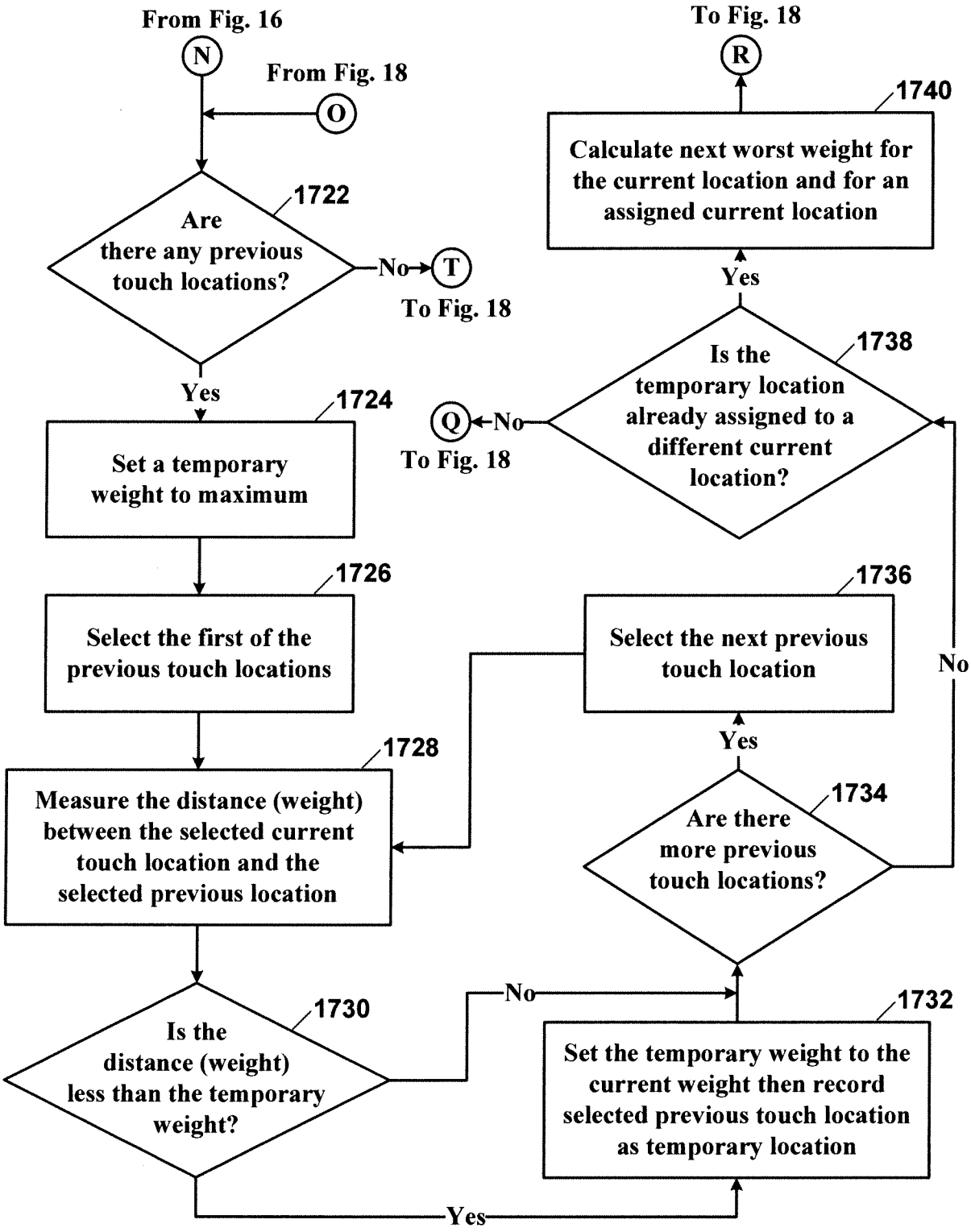

Figure 17

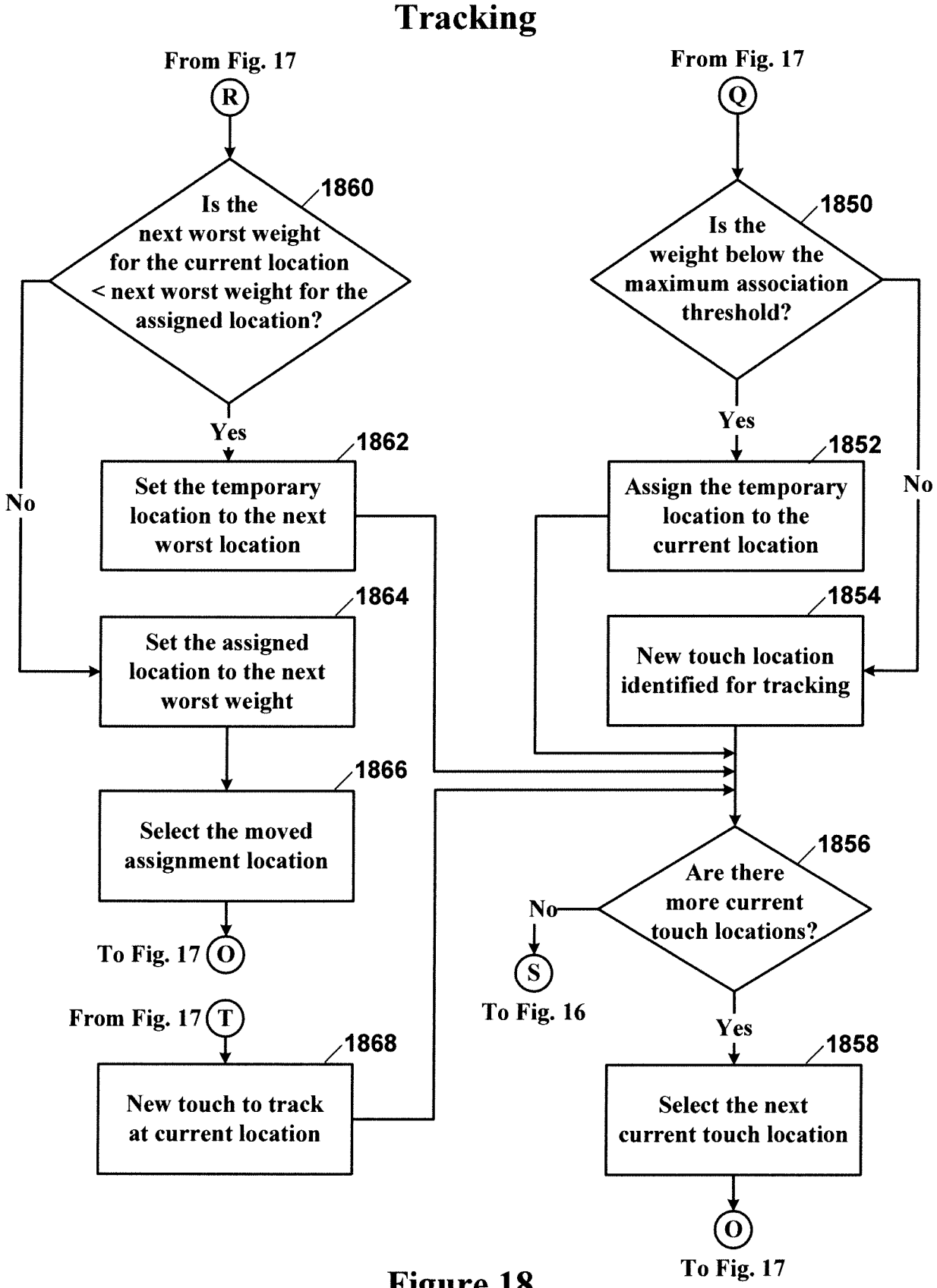
Tracking
From Fig. 17
R
Is the next worst weight for the current location < next worst weight for the assigned location?
1860
Yes
No
Set the temporary location to the next worst location
1862
Set the assigned location to the next worst weight
1864
Select the moved assignment location
1866
To Fig. 17
O
From Fig. 17
T
New touch to track at current location
1868
From Fig. 17
Q
Is the weight below the maximum association threshold?
1850
Yes
No
Assign the temporary location to the current location
1852
New touch location identified for tracking
1854
Are there more current touch locations?
1856
No
S
To Fig. 16
Yes
Select the next current touch location
1858
O
To Fig. 17

Figure 18

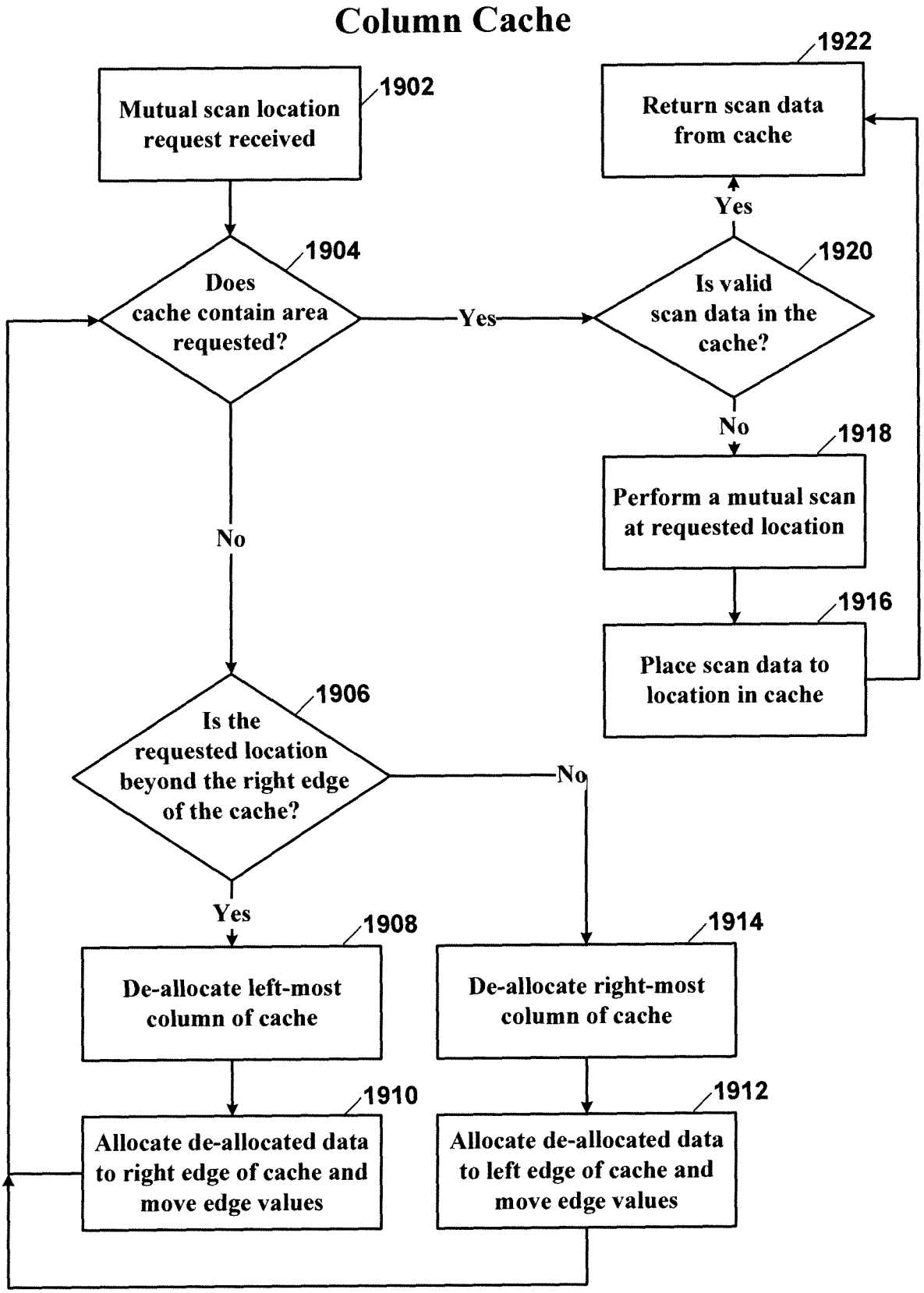
Column Cache
Mutual scan location request received
1902
Does cache contain area requested?
1904
Yes
No
Is valid scan data in the cache?
1920
Yes
No
Return scan data from cache
1922
Perform a mutual scan at requested location
1918
Place scan data to location in cache
1916
Is the requested location beyond the right edge of the cache?
1906
Yes
No
De-allocate left-most column of cache
1908
Allocate de-allocated data to right edge of cache and move edge values
1910
De-allocate right-most column of cache
1914
Allocate de-allocated data to left edge of cache and move edge values
1912

Figure 19