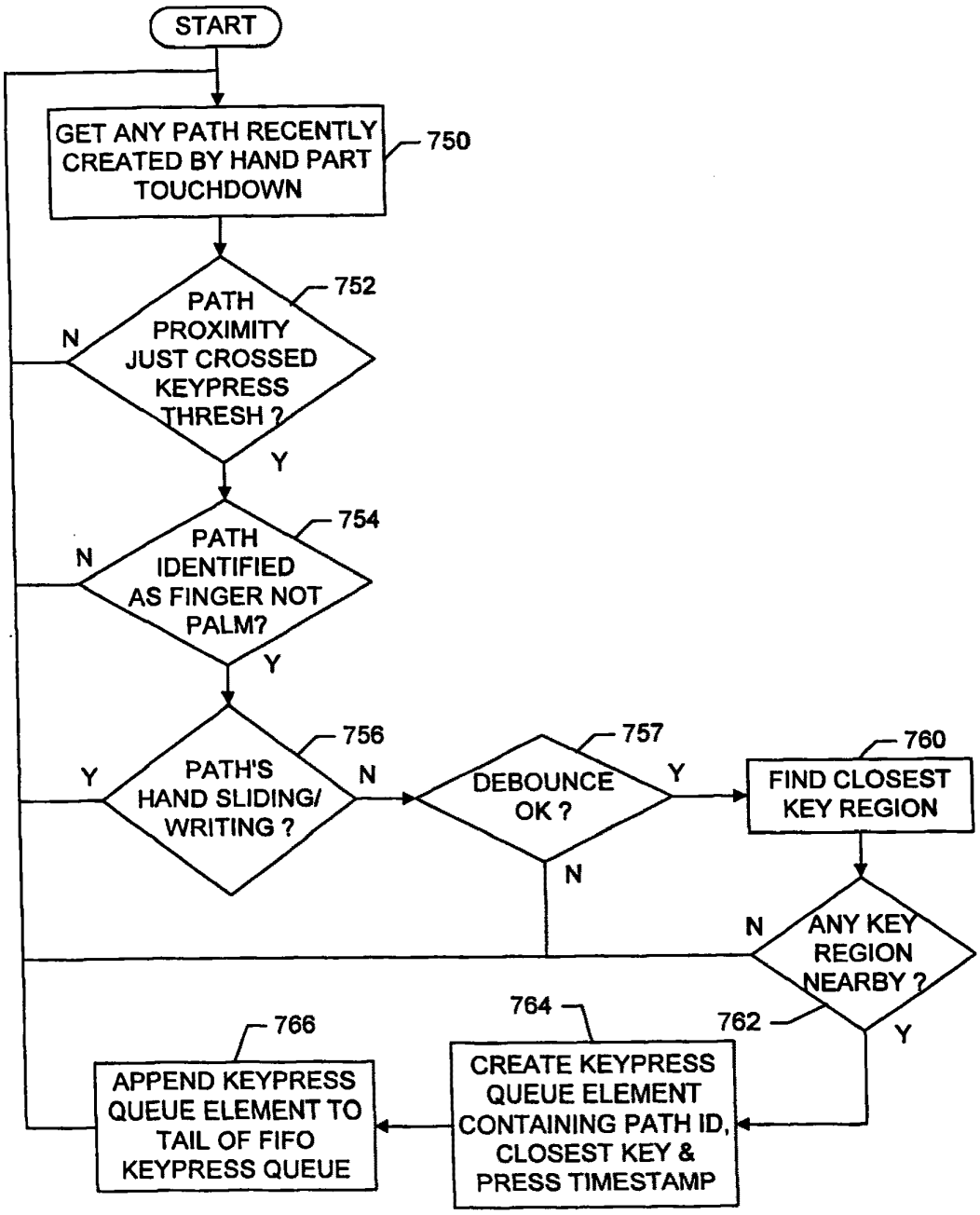

FIG. 42

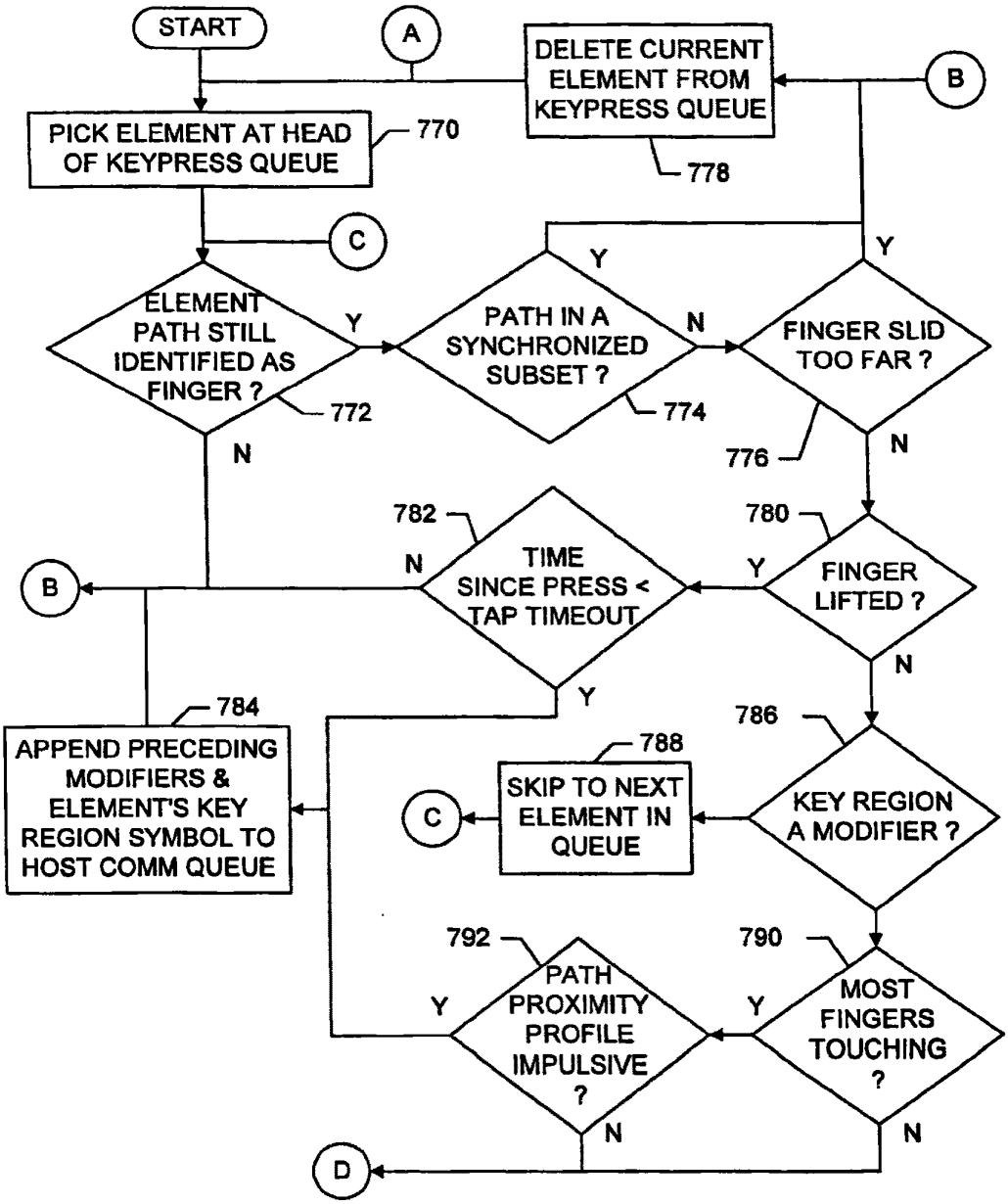

FIG. 43A

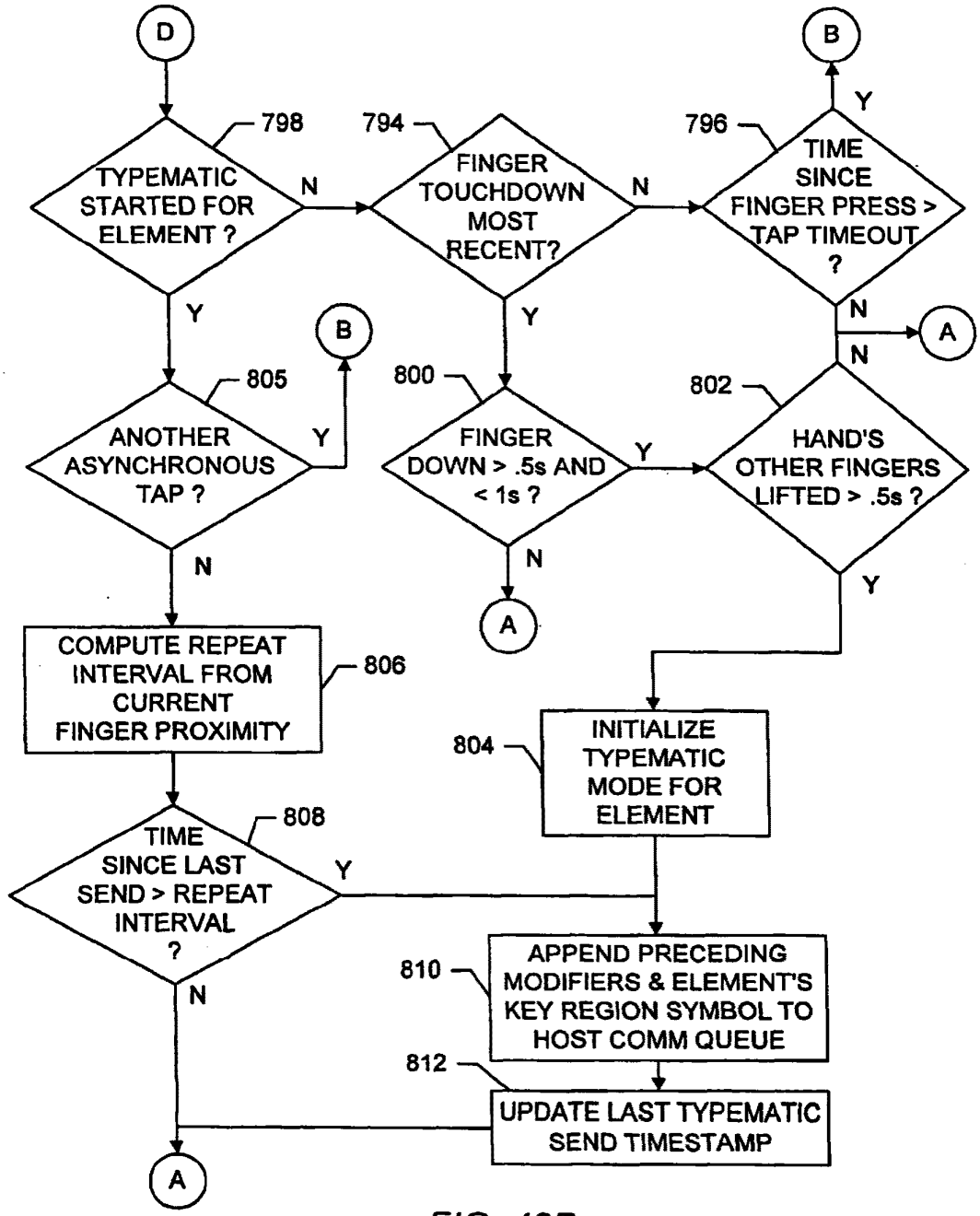

FIG. 43B

ELLIPSE FITTING FOR MULTI-TOUCH SURFACES

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of 11/015,434, entitled "Method and Apparatus for Integrating Manual Input," filed Dec. 17, 2004 now U.S. Pat. No. 7,339,580, which is a continuation of 09/236,513 (now Pat. No. 6,323,846) filed Jan. 25, 1999 which claims the benefit of provisional application 60/072,509, filed Jan. 26, 1998, each of which is hereby incorporated by reference in its entirety. This application is also related to Application Ser. No. 11/428,501, entitled "Capacitive Sensing Arrangement," 11/428,503, entitled "Touch Surface," 11/428,506, entitled "User Interface Gestures," 11/428,515, entitled "User Interface Gestures", 11/428,522, entitled "Identifying Contacts on a Touch Surface," 11/428,521, entitled "Identifying Contacts on a Touch Surface", 11/559,736, entitled "Multi-Touch Contact Tracking Algorithm", 11/559,763, "Multi-Touch Contact Motion Extraction," 11/559,799, entitled "Multi-Touch Contact Motion Extraction," 11/559,822, entitled "Multi-Touch Contact Motion Extraction," 11/559,833, entitled Multi-Touch Hand Position Offset Computation, each of which is hereby incorporated by reference in its entirety.

BACKGROUND OF THE INVENTION

A. Field of the Invention

The present invention relates generally to methods and apparatus for data input, and, more particularly, to a method and apparatus for integrating manual input.

B. Description of the Related Art

Many methods for manual input of data and commands to computers are in use today, but each is most efficient and easy to use for particular types of data input. For example, drawing tablets with pens or pucks excel at drafting, sketching, and quick command gestures. Handwriting with a stylus is convenient for filling out forms which require signatures, special symbols, or small amounts of text, but handwriting is slow compared to typing and voice input for long documents. Mice, finger-sticks and touchpads excel at cursor pointing and graphical object manipulations such as drag and drop. Rollers, thumbwheels and trackballs excel at panning and scrolling. The diversity of tasks that many computer users encounter in a single day call for all of these techniques, but few users will pay for a multitude of input devices, and the separate devices are often incompatible in a usability and an ergonomic sense. For instance, drawing tablets are a must for graphics professionals, but switching between drawing and typing is inconvenient because the pen must be put down or held awkwardly between the fingers while typing. Thus, there is a long-felt need in the art for a manual input device which is cheap yet offers convenient integration of common manual input techniques.

Speech recognition is an exciting new technology which promises to relieve some of the input burden on user hands. However, voice is not appropriate for inputting all types of data either. Currently, voice input is best-suited for dictation of long text documents. Until natural language recognition matures sufficiently that very high level voice commands can be understood by the computer, voice will have little advantage over keyboard hot-keys and mouse menus for command and control. Furthermore, precise pointing, drawing, and manipulation of graphical objects is difficult with voice commands, no matter how well speech is understood. Thus, there will always be a need in the art for multi-function manual input devices which supplement voice input.

A generic manual input device which combines the typing, pointing, scrolling, and handwriting capabilities of the standard input device collection must have ergonomic, economic, and productivity advantages which outweigh the unavoidable sacrifices of abandoning device specialization. The generic device must tightly integrate yet clearly distinguish the different types of input. It should therefore appear modeless to the user in the sense that the user should not need to provide explicit mode switch signals such as buttonpresses, arm relocations, or stylus pickups before switching from one input activity to another. Epidemiological studies suggest that repetition and force multiply in causing repetitive strain injuries. Awkward postures, device activation force, wasted motion, and repetition should be minimized to improve ergonomics. Furthermore, the workload should be spread evenly over all available muscle groups to avoid repetitive strain.

Repetition can be minimized by allocating to several graphical manipulation channels those tasks which require complex mouse pointer motion sequences. Common graphical user interface operations such as finding and manipulating a scroll bar or slider control are much less efficient than specialized finger motions which cause scrolling directly, without the step of repositioning the cursor over an on-screen control. Preferably the graphical manipulation channels should be distributed amongst many finger and hand motion combinations to spread the workload. Touchpads and mice with auxilliary scrolling controls such as the Cirque®™ Smartcat touchpad with edge scrolling, the IBM®™ ScrollPoint™ mouse with embedded pointing stick, and the Roller Mouse described in U.S. Pat. No. 5,530,455 to Gillick et al. represent small improvements in this area, but still do not provide enough direct manipulation channels to eliminate many often-used cursor motion sequences. Furthermore, as S. Zhai et al. found in "Dual Stream Input for Pointing and Scrolling," Proceedings of CHI '97 Extended Abstracts (1997), manipulation of more than two degrees of freedom at a time is very difficult with these devices, preventing simultaneous panning, zooming and rotating.

Another common method for reducing excess motion and repetition is to automatically continue pointing or scrolling movement signals once the user has stopped moving or lifts the finger. Related art methods can be distinguished by the conditions under which such motion continuation is enabled. In U.S. Pat. No. 4,734,685, Watanabe continues image panning when the distance and velocity of pointing device movement exceed thresholds. Automatic panning is, stopped by moving the pointing device back in the opposite direction, so stopping requires additional precise movements. In U.S. Pat. No. 5,543,591 to Gillespie et al., motion continuation occurs when the finger enters an edge border region around a small touchpad. Continued motion speed is fixed and the direction corresponds to the direction from the center of the touchpad to the finger at the edge. Continuation mode ends when the finger leaves the border region or lifts off the pad. Disadvantageously, users sometimes pause at the edge of the pad without intending for cursor motion to continue, and the unexpected motion continuation becomes annoying. U.S. Pat. No. 5,327,161 to Logan et al. describes motion continuation when the finger enters a border area as well, but in an alternative trackball emulation mode, motion continuation can be a function solely of lateral finger velocity and direction at liftoff. Motion continuation decays due to a friction factor or can be stopped by a subsequent touchdown on the surface. Disadvantageously, touch velocity at liftoff is not a reliable indicator of the user's desire for motion continuation since

when approaching a large target on a display at high speeds the user may not stop the pointer completely before liftoff. Thus it would be an advance in the art to provide a motion continuation method which does not become activated unexpectedly when the user really intended to stop pointer movement at a target but happens to be on a border or happens to be moving at significant speed during liftoff.

Many attempts have been made to embed pointing devices in a keyboard so the hands do not have to leave typing position to access the pointing device. These include the integrated pointing key described in U.S. Pat. No. 5,189,403 to Franz et al., the integrated pointing stick disclosed by J. Rutledge and T. Selker in "Force-to-Motion Functions for Pointing," Human-Computer Interaction—INTERACT '90, pp. 701-06 (1990), and the position sensing keys described in U.S. Pat. No. 5,675,361 to Santilli. Nevertheless, the limited movement range and resolution of these devices, leads to poorer pointing speed and accuracy than a mouse, and they add mechanical complexity to keyboard construction. Thus there exists a need in the art for pointing methods with higher resolution, larger movement range, and more degrees of freedom yet which are easily accessible from typing hand positions.

Touch screens and touchpads often distinguish pointing motions from emulated button clicks or keypresses by assuming very little lateral fingertip motion will occur during taps on the touch surface which are intended as clicks. Inherent in these methods is the assumption that tapping will usually be straight down from the suspended finger position, minimizing those components of finger motion tangential to the surface. This is a valid assumption if the surface is not finely divided into distinct key areas or if the user does a slow, "hunt and peck" visual search for each key before striking. For example, in U.S. Pat. No. 5,543,591 to Gillespie et al., a touchpad sends all lateral motions to the host computer as cursor movements. However, if the finger is lifted soon enough after touchdown to count as a tap and if the accumulated lateral motions are not excessive, any sent motions are undone and a mouse button click is sent instead. This method only works for mouse commands such as pointing which can safely be undone, not for dragging or other manipulations. In U.S. Pat. No. 5,666,113 to Logan, taps with less than about 1/16" lateral motion activate keys on a small keypad while lateral motion in excess of 1/16" activates cursor control mode. In both patents cursor mode is invoked by default when a finger stays on the surface a long time.

However, fast touch typing on a surface divided into a large array of key regions tends to produce more tangential motions along the surface than related art filtering techniques can tolerate. Such an array contains keys in multiple rows and columns which may not be directly under the fingers, so the user must reach with the hand or flex or extend fingers to touch many of the key regions. Quick reaching and extending imparts significant lateral finger motion while the finger is in the air which may still be present when the finger contacts the surface. Glancing taps with as much as 1/4" lateral motion measured at the surface can easily result. Attempting to filter or suppress this much motion would make the cursor seem sluggish and unresponsive. Furthermore, it may be desirable to enter a typematic or automatic key repeat mode instead of pointing mode when the finger is held in one place on the surface. Any lateral shifting by the fingertip during a prolonged finger press would also be picked up as cursor jitter without heavy filtering. Thus, there is a need in the art for a method to distinguish keying from pointing on the same surface via more robust hand configuration cues than lateral motion of a single finger.

An ergonomic typing system should require minimal key tapping force, easily distinguish finger taps from resting hands, and cushion the fingers from the jarring force of surface impact. Mechanical and membrane keyboards rely on the spring force in the keyswitches to prevent activation when the hands are resting on the keys. This causes an irreconcilable tradeoff between the ergonomic desires to reduce the fatigue from key activating force and to relax the full weight of the hands onto the keys during rest periods. Force minimization on touch surfaces is possible with capacitive or active optical sensing, which do not rely on finger pressure, rather than resistive-membrane or surface-acoustic-wave sensing techniques. The related art touch devices discussed below will become confused if a whole hand including its four fingertips a thumb and possibly palm heels, rests on the surface. Thus, there exists a long felt need in the art for a multi-touch surface typing system based on zero-force capacitive sensing which can tolerate resting hands and a surface cushion.

An ergonomic typing system should also adapt to individual hand sizes tolerate variations in typing style, and support a range of healthy hand postures. Though many ergonomic keyboards have been proposed, mechanical keyswitches can only be repositioned at great cost. For example, the keyboard with concave keywells described by Hargreaves et al. in U.S. Pat. No. 5,689,253 fits most hands well but also tends to lock the arms in a single position. A touch surface key layout could easily be morphed, translated, or arbitrarily reconfigured as long as the changes did not confuse the user. However, touch surfaces may not provide as much laterally orienting tactile feedback as the edges of mechanical keyswitches. Thus, there exists a need in the art for a surface typing recognizer which can adapt a key layout to fit individual hand postures and which can sustain typing accuracy if the hands drift due to limited tactile feedback.

Handwriting on smooth touch surfaces using a stylus is well-known in the art, but it typically does not integrate well with typing and pointing because the stylus must be put down somewhere or held awkwardly during other input activities. Also, it may be difficult to distinguish the handwriting activity of the stylus from pointing motions of a fingertip. Thus there exists a need in the art for a method to capture coarse handwriting gestures without a stylus and without confusing them with pointing motions.

Many of the input differentiation needs cited above could be met with a touch sensing technology which distinguishes a variety of hand configurations and motions such as sliding finger chords and grips. Many mechanical chord keyboards have been designed to detect simultaneous downward activity from multiple fingers, but they do not detect lateral finger motion over a large range. Related art shows several examples of capacitive touchpads which emulate a mouse or keyboard by tracking a single finger. These typically measure the capacitance of or between elongated wires which are laid out in rows and columns. A thin dielectric is interposed between the row and column layers. Presence of a finger perturbs the self or mutual capacitance for nearby electrodes. Since most of these technologies use projective row and column sensors which integrate on one electrode the proximity of all objects in a particular row or column, they cannot uniquely determine the positions of two or more objects as discussed in S. Lee, "A Fast Multiple-Touch-Sensitive Input Device," University of Toronto Masters Thesis (1984). The best they can do is count fingertips which happen to lie in a straight row, and even that will fail if a thumb or palm is introduced in the same column as a fingertip.

In U.S. Pat. Nos. 5,565,658 and 5,305,017, Gerpheide et al. measure the mutual capacitance between row and column electrodes by driving one set of electrodes at some clock frequency and sensing how much of that frequency is coupled onto a second electrode set. Such synchronous measurements are very prone to noise at the driving frequency, so to increase signal-to-noise ratio they form virtual electrodes comprised of multiple rows or multiple columns, instead of a single row and column, and scan through electrode combinations until the various mutual capacitances are nulled or balanced. The coupled signal increases with the product of the rows and columns in each virtual electrodes, but the noise only increases with the sum, giving a net gain in signal-to-noise ratio for virtual electrodes consisting of more than two rows and two columns. However, to uniquely distinguish multiple objects, virtual electrode sizes would have to be reduced so the intersection of the row and column virtual electrodes would be no larger than a finger tip, i.e., about two rows and two columns, which will degrade the signal-to-noise ratio. Also, the signal-to-noise ratio drops as row and column lengths increase to cover a large area.

In U.S. Pat. Nos. 5,543,591, 5,543,590, and 5,495,077, Gillespie et al measure the electrode-finger self-capacitance for row and column electrodes independently. Total electrode capacitance is estimated by measuring the electrode voltage change caused by injecting or removing a known amount of charge in a known time. All electrodes can be measured simultaneously if each electrode has its own drive/sense circuit. The centroid calculated from all row and column electrode signals establishes an interpolated vertical and horizontal position for a single object. This method may in general have higher signal-to-noise ratio than synchronous methods, but the signal-to-noise ratio is still degraded as row and column lengths increase. Signal-to-noise ratio is especially important for accurately locating objects which are floating a few millimeters above the pad. Though this method can detect such objects, it tends to report their position as being near the middle of the pad, or simply does not detect floating objects near the edges.

Thus there exists a need in the art for a capacitance-sensing apparatus which does not suffer from poor signal-to-noise ratio and the multiple finger indistinguishability problems of touchpads with long row and column electrodes.

U.S. Pat. No. 5,463,388 to Boie et al. has a capacitive sensing system applicable to either keyboard or mouse input, but does not consider the problem of integrating both types of input simultaneously. Though they mention independent detection of arrayed unit-cell electrodes, their capacitance transduction circuitry appears too complex to be economically reproduced at each electrode. Thus the long lead wires connecting electrodes to remote signal conditioning circuitry can pickup noise and will have significant capacitance compared to the finger-electrode self-capacitance, again limiting signal-to-noise ratio. Also, they do not recognize the importance of independent electrodes for multiple finger tracking, or mention how to track multiple fingers on an independent electrode array.

Lee built an early multi-touch electrode array, with 7 mm by 4 mm metal electrodes arranged in 32 rows and 64 columns. The "Fast Multiple-Touch-Sensitive Input Device (FMTSID)" total active area measured 12" by 16", with a 0.075 mm Mylar dielectric to insulate fingers from electrodes. Each electrode had one diode connected to a row charging line and a second diode connected to a column discharging line. Electrode capacitance changes were measured singly or in rectangular groups by raising the voltage on one or more row lines, selectively charging the electrodes in those rows, and then timing the discharge of selected columns to ground through a discharge resistor. Lee's design required only two diodes per electrode, but the principal disadvantage of Lee's design is that the column diode reverse bias capacitances allowed interference between electrodes in the same column.

All of the related capacitance sensing art cited above utilize interpolation between electrodes to achieve high pointing resolution with economical electrode density. Both Boie et al. and Gillespie et al. discuss compultattion of a centroid from all row and column electrode readings. However, for multiple finger detection, centroid calculation must be carefully limited around local maxima to include only one finger at a time. Lee utilizes a bisective search technique to find local maxima and then interpolates only on the eight nearest neighbor electrodes of each local maximum electrode. This may work fine for small fingertips, but thumb and palm contacts may cover more than nine electrodes. Thus there exists a need in the art for improved means to group exactly those electrodes which are covered by each distinguishable hand contact and to compute a centroid from such potentially irregular groups.

To take maximum advantage of multi-touch surface sensing, complex proximity image processing is necessary to track and identify the parts of the hand contacting the surface at any one time. Compared to passive optical, images, proximity images provide clear indications of where the body contacts the surface, uncluttered by luminosity variation and extraneous objects in the background. Thus proximity image filtering and segmentation stages can be simpler and more reliable than in computer vision approaches to free-space hand tracking such as S. Alimad, "A Usable Real-Time 3D Hand Tracker," Proceedings of the 28th Asilomar Conference on Signals, Systems, and Computers—Part 2, vol. 2, IEEE (1994) or Y. Cui and J. Wang, "Hand Segmentation Using Learning-Based Prediction and Verification for Hand Sign Recognition," Proceedings of the 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 88-93 (1996). However, parts of the hand such as intermediate finger joints and the center of the palms do not show up in capacitive proximity images at all if the hand is not flattened on the surface. Without these intermediate linkages between fingertips and palms the overall hand structure can only be guessed at, making hand contact identification very difficult. Hence the optical flow and contour tracking techniques which have been applied to free-space hand sign language recognition as in F. Quek, "Unencumbered Gestural Interaction," IEEE Multimedia, vol. 3, pp. 36-47 (1996), do not address the special challenges of proximity image tracking.

Synaptics Corp. has successfully fabricated their electrode array on flexible mylar film rather than stiff circuit board. This is suitable for conforming to the contours of special products, but does not provide significant finger cushioning for large surfaces. Even if a cushion was placed under the film, the lack of stretchability in the film, leads, and electrodes would limit the compliance afforded by the compressible material. Boie et al suggests that placing comressihle insulators on top of the electrode array cushions finger impact. However, an insulator more than about one millimeter thick would seriously attenuate the measured finger-electrode capacitances. Thus there exists a need in the art for a method to transfer finger capacitance influences through an arbitrarily thick cushion.

SUMMARY OF THE INVENTION

It is a primary object of the present invention to provide a system and method for integrating different types of manual

input such as typing, multiple degree-of-freedom manipulation, and handwriting on a multi-touch surface.

It is also an object of the present invention to provide a system and method for distinguishing different types of manual input such as typing, multiple degree-of-freedom manipulation, and handwriting on a multi-touch surface, via different hand configurations which are easy for the user to learn and easy for the system to recognize.

It is a further object of the present invention to provide an improved capacitance-transducing apparatus that is cheaply implemented near each electrode so that two-dimensional sensor arrays of arbitrary size and resolution can be built without degradation in signal to noise.

It is a further object of the present invention to provide an electronic system which minimizes the number of sensing electrodes necessary to obtain proximity images with such resolution that a variety of hand configurations can be distinguished.

Yet another object of the present invention is to provide a multi-touch surface apparatus which is compliant and contoured to be comfortable and ergonomic under extended use.

Yet another object of the present invention is to provide tactile key or hand position feedback without impeding hand resting on the surface or smooth, accurate sliding across the surface.

It is a further object of the present invention to provide an electronic system which can provide images of flesh proximity to an array of sensors with such resolution that a variety of hand configurations can be distinguished.

It is another object of the present invention to provide an improved method for invoking cursor motion continuation only when the user wants it by not invoking it when significant deceleration is detected.

Another object of the present invention is to identify different hand parts as they contact the surface so that a variety of hand configurations can be recognized and used to distinguish different kinds of input activity.

Yet another object of the present invention is to reliably extract rotation and scaling as well as translation degrees of freedom from the motion of two or more hand contacts to aid in navigation and manipulation of two-dimensional electronic documents.

It is a further object of the present invention to reliably extract tilt and roll degrees of freedom from hand pressure differences to aid in navigation and manipulation of three-dimensional environments.

Additional objects and advantages of the invention will be set forth in part in the description which follows, and in part will be obvious from the description, or may be learned by practice of the invention. The objects and advantages of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims.

To achieve the objects and in accordance with the purpose of the invention, as embodied and broadly described herein, the invention comprises a sensing device that is sensitive to changes in self-capacitance brought about by changes in proximity of a touch device to the sensing device, the sensing device comprising: two electrical switching means connected together in series having a common node, an input node, and an output node; a dielectric-covered sensing electrode connected to the common node between the two switching means; a power supply providing an approximately constant voltage connected to the input node of the series-connected switching means; an integrating capacitor to accumulate charge transferred during multiple consecutive switchings of the series connected switching means; another switching means connected in parallel across the integrating capacitor to deplete its residual charge; and a voltage-to-voltage translation device connected to the output node of the series-connected switching means which produces a voltage representing the magnitude of the self-capacitance of the sensing device. Alternatively, the sensing device comprises: two electrical switching means connected together in series having a common node, an input node, and an output node; a dielectric-covered sensing electrode connected to the common node between the two switching means; a power supply providing an approximately constant voltage connected to the input node of the series-connected switching means; and an integrating current-to-voltage translation device connected to the output node of the series connected switching means, the current-to-voltage translation device producing a voltage representing the magnitude of the self-capacitance of the sensing device.

To further achieve the objects, the present invention comprises a multi-touch surface apparatus for detecting a spatial arrangement of multiple touch devices on or near the surface of the multi-touch apparatus, comprising: one of a rigid or flexible surface; a plurality of two-dimensional arrays of one of the sensing devices (recited in the previous paragraph) arranged on the surface in groups wherein the sensing devices within a group have their output nodes connected together and share the same integrating capacitor, charge depletion switch, and voltage-to-voltage translation circuitry; control circuitry for enabling a single sensor device from each two-dimensional array; means for selecting the sensor voltage data from each two-dimensional array; voltage measurement circuitry to convert sensor voltage data to a digital code; and circuitry for communicating the digital code to another electronic device. The sensor voltage data selecting means comprises one of a multiplexing circuitry and a plurality of voltage measurement circuits.

To still further achieve the objects, the present invention comprises a multi-touch surface apparatus for sensing diverse configurations and activities of touch devices and generating integrated manual input to one of an electronic or electromechanical device, the apparatus comprising: an array of one of the proximity sensing devices described above; a dielectric cover having symbols printed thereon that represent action-to-be-taken when engaged by the touch devices; scanning means for forming digital proximity images from the array of sensing devices; calibrating means for removing background offsets from the proximity images; recognition means for interpreting the configurations and activities of the touch devices that make up the proximity images; processing means for generating input signals in response to particular touch device configurations and motions; and communication means for sending the input signals to the electronic or electromechanical device.

To even further achieve the objects, the present invention comprises a multi-touch surface apparatus for sensing diverse configurations and activities of fingers and palms of one or more hands near the surface and generating integrated manual input to one of an electronic or electromechanical device, the apparatus comprising: an array of proximity sensing means embedded in the surface; scanning means for forming digital proximity images from the proximities measured by the sensing means; image segmentation means for collecting into groups those proximity image pixels intensified by contact of the same distinguishable part of a hand; contact tracking means for parameterizing hand contact features and trajectories as the contacts move across successive proximity images, contact identification means for determining which hand and which part of the hand is causing each surface contact; syn-

chronization detection means for identifying subsets of identified contacts which touchdown or liftoff the surface at approximately the same time, and for generating command signals in response to synchronous taps of multiple fingers on the surface; typing recognition means for generating intended key symbols from asynchronous finger taps; motion component extraction means for compressing multiple degrees of freedom of multiple fingers into degrees of freedom common in two and three dimensional graphical manipulation; chord motion recognition means for generating one of command and cursor manipulation signals in response to motion in one or more extracted degrees of freedom by a selected combination of fingers; pen grip detection means for recognizing contact arrangements which resemble the configuration of the hand when gripping a pen, generating inking signals from motions of the inner fingers, and generating cursor manipulation signals from motions of the palms while the inner fingers are lifted; and communication means for sending the sensed configurations and activities of finger and palms to one of the electronic and electromechanical device.

To further achieve the objects, the present invention comprises a method for tracking and identifying hand contacts in a sequence of proximity images in order to support interpretation of hand configurations and activities related to typing, multiple degree-of-freedom manipulation via chords, and handwriting, the method comprising the steps of: segmenting each proximity image into groups of electrodes which indicate significant proximity, each group representing proximity of a distinguishable hand part or other touch device; extracting total proximity, position, shape, size, and orientation parameters from each group of electrodes; tracking group paths through successive proximity images including detection of path endpoints at contact touchdown and liftoff; computing velocity and filtered position vectors along each path; assigning a hand and finger identity to each contact path by incorporating relative path positions and velocities, individual contact features, and previous estimates of hand and finger positions; and maintaining estimates of hand and finger positions from trajectories of paths currently assigned to the fingers, wherein the estimates provide high level feedback to bias segmentations and identifications in future images.

To still further achieve the objects, the present invention comprises a method for integrally extracting multiple degrees of freedom of hand motion from sliding motions of two or more fingers of a hand across a multi-touch surface, one of the fingers preferably being the opposable thumb, the method comprising the steps of: tracking across successive scans of the proximity sensor array the trajectories of individual hand parts on the surface; finding an innermost and an outermost finger contact from contacts identified as fingers on the given hand; computing a scaling velocity component from a change in a distance between the innermost and outermost finger contacts; computing a rotational velocity component from a change in a vector angle between the innermost and outermost finger contacts; computing a translation weighting for each contacting finger; computing translational velocity components in two dimensions from a translation weighted average of the finger velocities tangential to surface; suppressively filtering components whose speeds are consistently lower than the fastest components; transmitting the filtered velocity components as control signals to an electronic or electromechanical device.

To even further achieve the objects, the present invention comprises a manual input integration method for supporting diverse hand input activities such as resting the hands, typing, multiple degree-of-freedom manipulation, command gesturing and handwriting on a multi-touch surface, the method enabling users to instantaneously switch between the input activities by placing their hands in different configurations comprising distinguishable combinations of relative hand contact timing, proximity, shape, size, position, motion and/or identity across a succession of surface proximity images, the method comprising the steps of: tracking each touching hand part across successive proximity images; measuring the times when each hand part touches down and lifts off the surface; detecting when hand parts touch down or lift off simultaneously; producing discrete key symbols when the user asynchronously taps, holds, or slides a finger on key regions defined on the surface; producing discrete mouse button click commands, key commands, or no signals when the user synchronously taps two or more fingers from the same hand on the surface; producing gesture commands or multiple degree-of-freedom manipulation signals when the user slides two or more fingers across the surface; and sending the produced symbols, commands and manipulation signals as input to an electronic or an electro-mechanical device.

To still even further achieve the objects, the present invention comprises a method for choosing what kinds of input signals will be generated and sent to an electronic or electromechanical device in response to tapping or sliding of fingers on a multi-touch surface, the method comprising the following steps: identifying each contact on the surface as either a thumb, fingertip or palm; measuring the times when each hand part touches down and lifts off the surface; forming a set of those fingers which touch down from the all finger floating state before any one of the fingers lifts back off the surface; choosing the kinds of input signals to be generated by further distinctive motion of the fingers from the combination of finger identities in the set; generating input signals of this kind when further distinctive motions of the fingers occur; forming a subset any two or more fingers which touch down synchronously after at least one finger has lifted back off the surface; choosing a new kinds of input signals to be generated by further distinctive motion of the fingers from the combination of finger identities in the subset; generating input signals of this new kind when further distinctive motions of the fingers occur; and continuing to form new subsets, choose and generate new kinds of input signals in response to liftoff and synchronous touchdowns until all fingers lift off the surface.

To further achieve the objects, the present invention comprises a method for continuing generation of cursor movement or scrolling signals from a tangential motion of a touch device over a touch-sensitive input device surface after touch device liftoff from the surface if the touch device operator indicates that cursor movement continuation is desired by accelerating or failing to decelerate the tangential motion of the touch device before the touch device is lifted, the method comprising the following steps: measuring, storing and transmitting to a computing device two or more representative tangential velocities during touch device manipulation; computing and storing a liftoff velocity from touch device positions immediately prior to the touch device liftoff; comparing the liftoff velocity with the representative tangential velocities, and entering a mode for continuously moving the cursor if a tangential liftoff direction approximately equals the representative tangential directions and a tangential liftoff speed is greater than a predetermined fractional multiple of representative tangential speeds; continuously transmitting cursor movement signals after liftoff to a computing device such that the cursor movement velocity corresponds to one of the representative tangential velocities; and ceasing transmission of the cursor movement signals when the touch device engages the surface again, if comparing means detects significant

deceleration before liftoff, or if the computing device replies that the cursor can move no farther or a window can scroll no farther.

It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention as claimed.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate several embodiments of the invention and together with the description, serve to explain the principles of the invention. In the drawings:

FIG. **1** is a block diagram of the integrated manual input apparatus;

FIG. **2** is a schematic drawing of the proximity sensor with voltage amplifier;

FIG. **3** is a schematic drawing of the proximity sensor with integrating current amplifier;

FIG. **4** is a schematic drawing of the proximity sensor implemented with field effect transistors;

FIG. **5** is a schematic drawing of the proximity sensor as used to implement 2D arrays of proximity sensors;

FIG. **6** is a block diagram showing a typical architecture for a 2D array of proximity sensors where all sensors share the same amplifier;

FIG. **7** is a block diagram of circuitry used to convert proximity sensor output to a digital code;

FIG. **8** is a block diagram showing a typical architecture for a 2D array of proximity sensors where sensors within a row share the same amplifier;

FIG. **9** is a schematic of a circuit useful for enabling the output gates of all proximity sensors within a group (arranged in columns);

FIG. **10** is a side view of a 2D proximity sensor array that is sensitive to the pressure exerted by non-conducting touch objects;

FIG. **11** is a, side view of a 2D proximity sensor array that provides a compliant surface without loss of spatial sensitivity;

FIG. **12** is a side view of a 2D proximity sensor array that is sensitive to both the proximity of conducting touch objects and to the pressure exerted by non-conducting touch objects;

FIG. **13** is an example proximity image of a hand flattened onto the surface with fingers outstretched;

FIG. **14** is an example proximity image of a hand partially closed with fingertips normal to surface;

FIG. **15** is an example proximity image of a hand in the pen grip configuration with thumb and index fingers pinched;

FIG. **16** is a data flow diagram of the hand tracking and contact identification system;

FIG. **17** is a flow chart of hand position estimation:

FIG. **18** is a data flow diagram of proximity image segmentation;

FIG. **19** is a diagram of the boundary search pattern during construction of an electrode group;

FIG. **20**A is a diagram of the segmentation strictness regions with both hands in their neutral, default position on surface;

FIG. **20**B is a diagram of the segmentation strictness regions when the hands are in asymmetric positions on surface;

FIG. **20**C is a diagram of the segmentation strictness regions when the right hand crosses to the left half of the surface and the left hand is off the surface;

FIG. **21** is a flow chart of segmentation edge testing;

FIG. **22** is a flow chart of persistent path tracking;

FIG. **23** is a flow chart of the hand part identification algorithm;

FIG. **24** is a Voronoi cell diagram constructed around hand part attractor points;

FIG. **25**A is a plot of orientation weighting factor for right thumb, right inner palm, and left outer palm versus contact orientation;

FIG. **25**B is a plot of thumb size factor versus contact size;

FIG. **25**C is a plot of palm size factor versus ratio of total contact proximity to contact eccentricity;

FIG. **25**D is a plot of palm separation factor versus distance between a contact and it nearest neighbor contact;

FIG. **26** is a flow chart of the thumb presence verification algorithm;

FIG. **27** is a flow chart of an alternative hand part identification algorithm;

FIG. **28** is a flow chart of the pen grip detection process:

FIG. **29** is a flow chart of the hand identification algorithm:

FIGS. **30**A-C show three different hand partition hypotheses for a fixed arrangement of surface contacts;

FIG. **31**A is a plot of the hand clutching direction factor versus horizontal hand velocity;

FIG. **31**B is a plot of the handedness factor versus vertical position of outermost finger relative to next outermost;

FIG. **31**C is a plot of the palm cohesion factor versus maximum horizontal separation between palm contacts within a hand;

FIG. **32** is a plot of the inner finger angle factor versus the angle between the innermost and next innermost finger contacts;

FIG. **33** is a plot of the inter-hand separation factor versus the estimated distance between the right thumb and left thumb;

FIG. **34** is a flow chart of hand motion component extraction;

FIG. **35** is a diagram of typical finger trajectories when hand is contracting;

FIG. **36** is a flow chart of radial and angular hand velocity extraction;

FIG. **37** is a flow chart showing extraction of translational hand velocity components;

FIG. **38** is a flow chart of differential hand pressure extraction;

FIG. **39**A is a flow chart of the finger synchronization detection loop;

FIG. **39**B is a flow chart of chord tap detection;

FIG. **40**A is a flow chart of the chord motion recognition loop;

FIG. **40**B is a flow chart of chord motion event generation;

FIG. **41** is a flow chart of key layout morphing;

FIG. **42** is a flow chart of the keypress detection loop;

FIG. **43**A is a flow chart of the keypress acceptance and transmission loop; and

FIG. **43**B is a flow chart of typematic emulation.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Reference will now be made in detail to the present preferred embodiments of the invention, examples of which are illustrated in the accompanying drawings. Wherever possible the same reference numbers will be used throughout the drawings to refer to the same or like parts.

FIG. **1** is a system block diagram of the entire, integrated manual input apparatus. Sensor embedded in the multi-touch

surface **2** detect proximity of entire flattened hands **4**, fingertips thumbs, palms, and other conductive touch devices to the surface **2**. In a preferred embodiment, the surface is large enough to comfortably accommodate both hands **4** and is arched to reduce forearm pronation.

In alternative embodiments the multi-touch surface **2** may be large enough to accommodate motion of one hand, but may be flexible so it can be fitted to an armrest or clothing.

Electronic scanning hardware **6** controls and reads from each proximity sensor of a sensor array. A calibration module **8** constructs a raw proximity image from a complete scan of the sensor array and subtracts off any background sensor offsets. The background sensor offsets can simply be a proximity image taken when nothing is touching the surface.

The offset-corrected proximity image is then passed on to the contact tracking and identification module **10**, which segments the image into distinguishable hand-surface contacts, tracks and identifies them as they move through successive images.

The paths of identified contacts are passed on to a typing recognizer module **12**, finger synchronization detection module **14**, motion component extraction module **16**, and pen grip detection module **17**, which contain software algorithms to distinguish hand configurations and respond to detected hand motions.

The typing recognizer module **12** responds to quick presses and releases of fingers which are largely asynchronous with respect to the activity of other fingers on the same hand. It attempts to find the key region nearest to the location of each finger tap and forwards the key symbols or commands associated with the nearest key region to the communication interface module **20**.

The finger synchronization detector **14** checks the finger activity within a hand for simultaneous presses or releases of a subset of fingers. When such simultaneous activity is detected it signals the typing recognizer to ignore or cancel keystroke processing for fingers contained in the synchronous subset. It also passes on the combination of finger identities in the synchronous subset to the chord motion recognizer **18**.

The motion component extraction module **16** computes multiple degrees of freedom of control from individual finger motions during easily performable hand manipulations on the surface **2**, such as hand translations, hand rotation about the wrist, hand scaling by grasping with the fingers, and differential hand tilting.

The chord motion recognizer produces chord tap or motion events dependent upon both the synchronized finger subset identified by the synchronization detector **14** and on the direction and speed of motion extracted in **16**. These events are then posted to the host communication interface **20**.

The pen grip detection module **17** checks for specific arrangements of identified hand contacts which indicate the hand is configured as if gripping a pen. If such an arrangement is, detected, it forwards the movements of the gripping fingers as inking events to the host communication interface **20**. These inking events can either lay digital ink on the host computer display for drawing or signature capture purposes, or they can be further interpreted by handwriting recognition software which is well known in the art. The detailed steps within each of the above modules will be further described later.

The host communication interface keeps events from both the typing recognizer **12** and chord motion recognizer **18** in a single temporally ordered queue and dispatches them to the host computer system **22**. The method of communication between the interface **20** and host computer system **22** can vary widely depending on the function and processing power of the host computer. In a preferred embodiment, the communication would take place over computer cables via industry standard protocols such as Apple Desktop Bus, PS/2 keyboard and mouse protocol for PCs, or Universal Serial Bus (USB). In alternative embodiments the software processing of modules **10-18** would be performed within the host computer **22**. The multi-touch surface apparatus would only contain enough hardware to scan the proximity sensor array **6**, form proximity images **8**, and compress and send them to the host computer over a wireless network. The host communication interface **20** would then play the role of device driver on the host computer, conveying results of the proximity image recognition process as input to other applications residing on the host computer system **22**.

In a preferred embodiment the host computer system outputs to a visual display device **24** so that the hands and fingers **4** can manipulate graphical objects on the display screen. However, in alternative embodiments the host computer might output to an audio display or control a machine such as a robot.

The term "proximity" will only be used in reference to the distance or pressure between a touch device such as a finger and the surface **2**, not in reference to the distance between adjacent fingers. "Horizontal" and "vertical" refer to x and y directional axes within the surface plane. Proximity measurements are then interpreted as pressure in a z axis normal to the surface. The direction "inner" means toward the thumb of a given hand, and the direction "outer" means towards the pinky finger of a given hand. For the purposes of this description, the thumb is considered a finger unless otherwise noted, but it does not count as a fingertip. "Contact" is used as a general term for a hand part when it touches the surface and appears in the current proximity image, and for the group and path data structures which represent it.

FIG. **2** is a schematic diagram of a device that outputs a voltage **58** dependent on the proximity of a touch device **38** to a conductive sense electrode **33**. The proximity sensing device includes two electrical switching means **30** and **31** connected together in series having a common node **48**, an input node **46**, and an output node **45**. A thin dielectric material **32** covers the sensing electrode **33** that is electrically connected to the common node **48**. A power supply **34** providing an approximately constant voltage is connected between reference ground and the input node **46**. The two electrical switches **30** and **31** gate the flow of charge from the power supply **34** to an integrating capacitor **37**. The voltage across the integrating capacitor **37** is translated to another voltage **58** by a high-impedance voltage amplifier **35**. The plates of the integrating capacitor **37** can be discharged by closing electrical switch **36** until the voltage across the integrating capacitor **37** is near zero. The electrical switches **30** and **31** are opened and closed in sequence but are never closed at the same time, although they may be opened at the same time as shown in FIG. **2**. Electrical switch **30** is referred to as the input switch; electrical switch **31** is referred to as the output switch; and, electrical switch **36** is referred to as the shorting switch.

The proximity sensing device shown in FIG. **2** is operated by closing and opening the electrical switches **30**, **31**, and **36** in a particular sequence after which the voltage output from the amplifier **58**, which is dependent on the proximity of a touch device **38**, is recorded. Sensor operation begins with all switches in the open state as shown in FIG. **2**. The shorting switch **36** is then closed for a sufficiently long time to reduce the charge residing on the integrating capacitor **37** to a low level. The shorting switch **37** is then opened. The input switch

30 is then closed thus allowing charge to flow between the power supply and the common node 48 until the voltage across the input switch 30 becomes zero. Charge Q will accumulate on the sensing electrode 33 according to

$$Q=V(e*A)/D \quad (1)$$

where V is the voltage of the power supply 34, e is the permittivity of the dielectric sensing electrode cover 32 and the air gap between the cover and the touch device 38, D is the thickness of this dielectric region, and A is the overlap area of the touch device 38 and the sensing electrode 33. Therefore the amount of charge accumulating on the sensing electrode 33 will depend, among other things, on the area of overlap of the touch device 38 and the sensing electrode 33 and the distance between the touch device 38 and the sensing electrode 33. The input switch 30 is opened after the voltage across it has become zero, or nearly so. Soon after input switch 30 is opened the output switch 31 is closed until the voltage across it is nearly zero. Closing the output switch 31 allows charge to flow between the sensing electrode 33 and the integrating capacitor 37 resulting in a voltage change across the integrating capacitor 37 according to:

$$\text{delta}V=(V-Vc)/(1+C*D/e*A) \quad (2)$$

where Vc is the voltage across the integrating capacitor 37 before the output switch 31 was closed, C is the capacitance of the integrating capacitor 37, and A and D are equal to their values when input switch 30 was closed as shown in Equation 1. Multiple switchings of the input 30 and output 31 switches as described above produce a voltage on the integrating capacitor 37 that reflects the proximity of a touch device 38 to the sensing electrode 33.

FIG. 3A is a schematic diagram of the proximity sensor in which the shorting transistor 36 and the voltage-to-voltage translation device 35 are replaced by a resistor 40 and a current-to-voltage translation device 41, respectively. The integrating function of capacitor 37 shown in FIG. 2 is, in this variation of the proximity sensor, carried out by the capacitor 39 shown in FIG. 3A. Those skilled in the art will see that this variation of the proximity sensor produces a more linear output 58 from multiple switchings of the input and output switches, depending on the relative value of the resistor 40. Alternatively, the resistor 40 can be replaced by a shorting switch 69 (cf. FIG. 3B) to improve linearity. Although, the circuits shown in FIG. 3 provide a more linear output than the circuit shown in FIG. 2 the circuits of FIG. 3 generally require dual power supplies while the circuit of FIG. 2 requires only one.

The electrical switches shown in, FIG. 2 can be implemented with various transistor technologies: discrete, integrated, thin film, thick film, polymer, optical, etc. One such implementation is shown in FIG. 4A where field effect transistors (FETs) are used as the input 30, output 31, and shorting 36 switches. The FETs are switched on and off by voltages applied to their gate terminals (43, 44, and 55). For the purpose of this description we will assume the FET is switched on when its gate voltage is logic 1 and switched off when its gate voltage is logic 0. A controller 42 is used to apply gate voltages as a function of time as shown in FIG. 4B. In this example, a sequence of three pairs of pulses (43 and 44) are applied to the input and output transistor gates. Each pair of pulses 43 and 44 produces a voltage change across the integrating capacitor 37 as shown in Equation 2. The number of pulse pairs applied to input 43 and output 44 gates depends on the desired voltage across integrating capacitor 37. In typical applications the number is between one and several hundred pulse-pairs.

FIG. 5 shows the proximity sensor circuitry appropriate for use in a system comprising an array of proximity sensors 47 as in a multi-touch surface system. The proximity sensor 47 consists of the input transistor 30, the output transistor 31, the sensing electrode 33, the dielectric cover 32 for the sensing electrode 33, and conductive traces 43, 44, 45, and 46. The conductive traces are arranged so as to allow the proximity sensors 47 comprising a 2D array to be closely packed and to share the same conductive traces, thus reducing the number of wires needed in a system. FIG. 6 shows an example of such a system where the input nodes 46 of all proximity sensors are connected together and connected to a power supply 34. The output nodes 45 of all proximity sensors are connected together and connected to a single integrating capacitor 37, a single shorting transistor 36, and a single voltage-to-voltage amplifier 35. In this implementation, a single proximity sensor 47 is enabled at a time by applying a logic 1 signal first to its input gate 43 and then to its output gate 44. This gating of a single proximity sensor 47 one at a time is done by input gate controller 50 and output gate controller 51. For example, to enable the proximity sensor 47 in the lower right corner the input gate controller 50 would output a logic one pulse on conductive trace 43*a*. This is followed by a logic one pulse on conductive trace 44*h* produced by output gate controller 51. Repetition of this pulse as shown in FIG. 4B would cause charge to build up on integrating capacitor 37 and a corresponding voltage to appear at the output of the amplifier 58. The entire array of proximity sensors 47 is thus scanned by enabling a single sensor at a time and recording its output.

FIG. 7A is a schematic of typical circuitry useful for converting the proximity sensor output 58 to a digital code appropriate for processing by computer. The proximity sensor output 58 is typically non-zero even when there is no touch device (e.g., ref. no. 38 in FIG. 2) nearby. This non-zero signal is due to parasitic or stray capacitance present at the common node 48 of the proximity sensor and is of relatively constant value. It is desirable to remove this non-zero background signal before converting the sensor output 58 to a digital code. This is done by using a differential amplifier 64 to subtract a stored record of the background signal 68 from the sensor output 58. The resulting difference signal 65 is then converted to a digital code by an ADC (analog to digital converter) 60 producing a K-bit code 66. The stored background signal is first recorded by sampling the array of proximity sensors 47 (FIG. 6) with no touch devices nearby and storing a digital code specific for each proximity sensor 47 in a memory device 63. The particular code corresponding to the background signal of each proximity sensor is selected by an M-bit address input 70 to the memory device 63 and applied 69 to a DAC (digital to analog converter) 61.

The 2D array of proximity sensors 47 shown in FIG. 6 can be connected in groups so as to improve the rate at which the entire array is scanned. This is illustrated in FIG. 8 where the groups are arranged as columns of proximity sensors. In this approach, the input nodes of the proximity sensors are connected together and connected to a power supply 34, as in FIG. 6. The output gates 44 are also connected in the same way. However, the input gates 43 are now all connected together and the output nodes 45 are connected to only those proximity sensors 47 within a row and to a dedicated voltage amplifier 35. With this connection method, all of the proximity sensors in a column are enabled at a time, thus reducing the time to scan the array by a factor N, where N is the number of proximity sensors in a group. The outputs 58*a-h* could con-

nect to dedicated converter circuitry as shown in FIG. **7A** or alternatively each output **58***a-h* could be converted one at a time using the circuitry shown in FIG. **7B**. In this figure, the output signals from each group **58***a-h* are selected one at a time by multiplexer **62** and applied to the positive input of the differential amplifier **64**. With this later approach, it is assumed that the ADC **60** conversion time is much faster than the sensor enable time, thus providing the suggested speed up in sensor array scanning.

FIG. **9** shows a typical circuit useful for the control of the proximity sensor's output gate **44**. It consists of three input signals **75, 76, 78** and two output signals **44, 77**. The output gate signal **44** is logic 1 when both inputs to AND gate **79** are logic 1. The AND input signal **77** becomes logic 1 if input signal **76** is logic 1 when input signal **78** transitions from logic 0 to logic 1, otherwise it remains logic 0. A linear array of these circuits **81** can be connected end-to-end to enable the output gates of a single group of proximity sensors at a time as shown in FIG. **8**.

FIG. **10** shows a cover for the multi-touch surface **89** that permits the system to be sensitive to pressure exerted by non-conducting touch objects (e.g., gloved fingers) contacting the multi-touch surface. This cover comprises a deformable dielectric touch layer **85**, a deformable conducting layer **86**, and a compliant dielectric layer **87**. The touch surface **85** would have a symbol set printed on it appropriate for a specific application, and this surface could be removed and replaced with another one having a different symbol set. The conducting layer **86** is electrically connected **88** to the reference ground of the proximity sensor's power supply **34**. When a touch object presses on the top surface **85** it causes the conducting surface **86** under the touch device to move closer to the sensing electrode **33** of the proximity sensor. This results in a change in the amount of charge stored on the sensing electrode **33** and thus the presence of the touch object can be detected. The amount of charge stored will depend on the pressure exerted by the touch object. More pressure results in more charge stored as indicated in Equation 1.

To obtain a softer touch surface on the multi-touch device a thicker and more, compliant dielectric cover could be used. However, as the dielectric thickness increases the effect of the touch device on the sensing electrodes **33** spreads out thus lowering spatial resolution. A compliant anisotropically-conducting material can be used to counter this negative effect while also providing a soft touch surface. FIG. **11** shows a cover in which a compliant anisotropically-conducting material **90** is set between a thin dielectric cover **85** and the sensing electrodes **33**. If the conductivity of the compliant material **90** is oriented mostly in the vertical direction, the image formed by a touch device on the surface **85** will be translated without significant spreading to the sensing electrodes **33**, thus preserving spatial resolution while providing a compliant touch surface.

FIG. **12** shows a cross section of a multi-touch surface that senses both the proximity and pressure of a touch device. The touch layer **85** is a thin dielectric that separates touch devices from the sensing electrodes **33**. Proximity sensing is relative to this surface. The electrodes **33** and associated switches and conductors are fabricated on a compliant material **89** which is attached to a rigid metal base **92**. The metal base **92** is electrically connected **88** to the reference ground of the proximity sensor's power supply **34**. When a touch device presses on the touch surface **85** it causes the sensing electrodes **33** directly below to move closer to the rigid metal base **92**. The distance moved depends on the pressure applied and thus the pressure exerted by a touch device can be detected as described before.

To illustrate typical properties of hand contacts as they appear in proximity images, FIGS. **13-15** contain sample images captured by a prototype array of parallelogram-shaped electrodes. Shading of each electrode darkens to indicate heightened proximity signals as flesh gets closer to the surface, compresses against the surface due to hand pressure, and overlaps the parallelogram more completely. Note that the resolution of these images is in no way intended to limit the scope of the invention, since certain applications such as handwriting recognition will clearly require finer electrode arrays than indicated by the electrode size in these sample images. In the discussion that follows, the proximity data measured at one electrode during a particular scan cycle constitutes one "pixel" of the proximity image captured in that scan cycle.

FIG. **13** shows a right hand flattened against the surface with fingers outstretched. At the far left is the oblong thumb **201** which tends to point off at about 120-degrees. The columnar blobs arranged in an arc across the top of the image are the index finger **202**, middle finger **203**, ring finger **204** and pinky finger **205**. Flesh from the proximal finger joint, or proximal phalanges **209**, will appear below each fingertip if the fingers are fully extended. The inner **207** and outer **206** palm heels cause the pair of very large contacts across the bottom of the image. Forepalm calluses **213** are visible at the center of the hand if the palm is fully flattened. This image shows that all the hand contacts are roughly oval-shaped, but they differ in pressure, size, orientation, eccentricity and spacing relative to one another. This image includes all of the hand parts which can touch the surface from the bottom of one hand but in many instances only a few of these parts will be touching the surface, and the fingertips may roam widely in relation to the palms as fingers are flexed and extended.

FIG. **14** shows another extreme in which the hand is partially closed. The thumb **201** is adducted toward the fingertips **202-208** and the fingers are flexed so the fingertips come down normal instead of tangential to the surface. The height and intensity of fingertip contacts is lessened somewhat because the boney tip rather than fleshy pulp pad is actually touching the surface, but fingertip width remains the same. Adjacent fingertips **202-205** and thumb **201** are so close together as to be distinguishable only by slight proximity valleys **210** between them. The proximal phalange finger joints are suspended well above the surface and do not appear in the image, nor do the forepalm calluses. The palm heels **206, 207** are somewhat shorter since only the rear of the palm can touch the surface when fingers are flexed, but the separation between them is unchanged. Notice that the proximity images are uncluttered by background objects. Unlike optical images, only conductive objects within a few millimeters of the surface show up at all.

FIG. **15** is a proximity image of a right hand in a pen grip configuration. The thumb **201** and index fingertip **202** are pinched together as if they were holding a pen but in this case they are touching the surface instead. Actually the thumb and index finger appear the same here as in FIG. **14**. However, the middle **203**, ring **204**, and pinky **205** fingers are curled under as if making a fist, so the knuckles from the top of the fingers actually touch the surface instead of the finger tips. The curling under of the knuckles actually places them behind the pinched thumb **201** and index fingertip **202** very close to the palm heels **206, 207**. The knuckles also appear larger than the curled fingertips of FIG. **14** but the same size as the flattened fingertips in FIG. **13**. These differences in size and arrangement will be measured by the pen grip detector **17** to distinguish this pen grip configuration from the closed and flattened hand configurations.

FIG. **16** represents the data flow within the contact tracking and identification module **10**. The image segmentation process **241** takes the most recently scanned proximity image data **240** and segments it into groups of electrodes **242** corresponding to the distinguishable hand parts of FIG. **13**. The filtering and segmentation rules applied in particular regions of the image are partially determined by feedback of the estimated hand offset data **252**. The image segmentation process **241** outputs a set of electrode group data structures **242** which are parameterized by fitting an ellipse to the positions and proximity measurements of the electrodes within each group.

The path tracking process **245** matches up the parameterized electrode groups **242** with the predicted continuations of contact path data structures **243** extracted from previous images. Such path tracking ensures continuity of contact representation across proximity images. This makes it possible to measure the velocity of individual hand contacts and determine when a hand part lifts off the surface, disappearing from future images. The path tracking process **245** updates the path positions, velocities, and contact geometry features from the parameters of the current groups **242** and passes them on to the contact identification processes **247** and **248**. For notational purposes, groups and unidentified paths will be referred to by data structure names of the form Gi and Pi respectively, where the indices i are arbitrary except for the null group G**0** and null path P**0**. Particular group and path parameters will be denoted by subscripts to these structure names and image scan cycles will be denoted by bracketed indices, so that, for example, $P2_x$, [n] represents the horizontal position of path **2** in the current proximity image, and $P2_x$, [n−1] represents the position in the previous proximity image. The contact identification system is hierarchically split into a hand identification process **247** and within-hand finger and palm identification process **248**. Given a hand identification for each contact, the finger and palm identification process **248** utilizes combinatorial optimization and fuzzy pattern recognition techniques to identify the part of the hand causing each surface contact. Feedback of the estimated hand offset helps identify hand contacts when so few contacts appear in the image that the overall hand structure is not apparent.

The hand identification process **247** utilizes a separate combinatorial optimization algorithm to find the assignment of left or right hand identity to surface contacts which results in the most biomechanically consistent within-hand identifications. It also receives feedback of the estimated hand and finger offsets **252**, primarily for the purpose of temporarily storing the last measured hand position after fingers in a hand lift off the surface. Then if the fingers soon touch back down in the same region they will more likely receive their previous hand identifications.

The output of the identification processes **247** and **248** is the set of contact paths with non-zero hand and finger indices attached. For notational purposes identified paths will be referred to as F**0** for the unidentified or null finger, F**1** for the thumb **201**, F**2** for the index finger **202**, F**3** for the middle finger **203**, F**4** for the ring finger **204**, F**5** for the pinky finger **205**, F**6** the outer palm heel **206**. F**7** for the inner palm heel **207**, and F**8** for the forepalm calluses **208**. To denote a particular hand identity this notation can be prefixed with an L for left hand or R for right hand, so that, for example, RF**2** denotes the right index finger path. When referring to a particular hand as a whole. LH denotes the left hand and RH denotes the right hand. In the actual algorithms left hand identity is represented by a −1 and right hand by +1, so it is easy to reverse the handedness of measurements taken across the vertical axis of symmetry.

It is also convenient to maintain for each hand a set of bitfield data registers for which each bit represents touchdown, continued contact or liftoff of a particular finger. Bit positions within each bit field correspond to the hand part indices above. Such registers can quickly be tested with a bit mask to determine whether a particular subset of fingers has touched down. Alternatively, they can be fed into a lookup table to find the input events associated with a particular finger chord (combination of fingers). Such finger identity bitfields are needed primarily by the synchronization detector **14** and chord motion recognizer **18**.

The last process within the tracking and identification subsystem is the hand position estimator **251**, which as described above provides biasing feedback to the identification and segmentation processes. The hand position estimator is intended to provide a conservative guess **252** of lateral hand position under all conditions including when the hand is floating above the surface without touching. In this case the estimate represents a best guess of where the hand will touch down again. When parts of a hand are touching the surface, the estimate combines the current position measurements of currently identified hand parts with past estimates which may have been made from more or less reliable identifications.

The simplest but inferior method of obtaining a hand position measurement would be to average the positions of all the hand's contacts regardless of identity. If hand parts **201-207** were all touching the surface as in FIG. **13** the resulting centroid would be a decent estimate, lying somewhere under the center of the palm since the fingers and palm heels typically form a ring around the center of the palm. However, consider when only one hand contact is available for the average. The estimate would assume the hand center is at the position of this lone contact, but if the contact is from the right thumb the hand center would actually be 4-8 cm to the right, or if the contact is from a palm heel the hand center is actually 4-6 cm higher, or if the lone contact is from the middle finger the hand center should actually be actually 4-6 cm lower.

FIG. **17** shows the detailed steps within the hand position estimator **251**. The steps must be repeated for each hand separately. In a preferred embodiment, the process utilizes the within-hand contact identifications (**250**) to compute (step **254**) for each contact an offset between the measured contact position (Fi_x[n],Fi_y[n]) and the default position of the particular finger or palm heel (Fi_{defx},Fi_{defy}) with hand part identity i. The default positions preferably correspond to finger and palm positions when the hand is in a neutral posture with fingers partially closed, as when resting on home row of a keyboard. Step **255** averages the individual contact offsets to obtain a measured hand offset, (H_{mox}[n],H_{moy}[n]):

$$H_{mox}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n](Fi_x[n] - Fi_{defx})}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (3)$$

$$H_{moy}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n](Fi_y[n] - Fi_{defy})}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (4)$$

Preferably the weighting Fi_{mow}[n] of each finger and palm heel is approximately its measured total proximity, i.e., Fi_{mow}[n]=Fi_z[n]. This ensures that lifted fingers, whose proximity is zero, have no influence on the average, and that contacts with

lower than normal proximity, whose measured positions and identities are less accurate, have low influence. Furthermore, if palm heels are touching, their large total proximities will dominate the average. This is beneficial because the palm heels, being immobile relative to the hand center compared to the highly flexible fingers, supply a more reliable indication of overall hand position. When a hand is not touching the surface, i.e., when all proximities are zero, the measured offsets are set to zero. This will cause the filtered hand position estimate below to decay toward the default hand position.

As long as the contact identifications are correct, this hand position measurement method eliminates the large errors caused by assuming lone contacts originate from the center of the hand. Flexing of fingers from their default positions will not perturb the measured centroid more than a couple centimeters. However, this scheme is susceptible to contact misidentification, which can cause centroid measurement errors of up to 8 cm if only one hand part is touching. Therefore, the current measured offsets are not used directly, but are averaged with previous offset estimates ($H_{eox}[n-1],H_{eoy}[n-1]$) using a simple first-order autoregressive filter, forming current offset estimates ($H_{eox}[n],H_{eoy}[n]$).

Step **256** adjusts the filter pole $H_{oa}[n]$ according to confidence in the current contact identifications. Since finger identifications accumulate reliability as more parts of the hand contact the surface one simple measure of identification confidence: is the number of fingers which have touched down from the hand since the hand last left the surface. Contacts with large total proximities also improve identification reliability because they have strong disambiguating features such as size and orientation. Therefore $H_{oa}[n]$ is set roughly proportional to the maximum finger count plus the sum of contact proximities for the hand. $H_{oa}[n]$ must of course be normalized to be between zero and one or the filter will be unstable. Thus when confidence in contact identifications is high, i.e., when many parts of the hand firmly touch the surface, the autoregressive filter favors the current offset measurements. However, when only one or two contacts have reappeared since hand liftoff, the filter emphasizes previous offset estimates in the hope that they were based upon more reliable identifications.

The filtered offsets must also maintain a conservative estimate of hand position while the hand is floating above the surface for optimal segmentation and identification as the hand touches back down. If a hand lifts off the surface in the middle of a complex sequence of operations and must, quickly touch down again, it will probably touch down close to where it lifted off. However, if the operation sequence has ended, the hand is likely to eventually return to the neutral posture, or default position, to rest. Therefore, while a hand is not touching the surface, $H_{oa}[n]$ is made small enough that the estimated offsets gradually decay to zero at about the same rate as a hand lazily returns to default position.

When $H_{oa}[n]$ is made small due to low identification confidence, the filter tracking delay becomes large enough to lag behind a pair of quickly moving fingers by several centimeters. The purpose of the filter is to react slowly to questionable changes in contact identity, not to smooth contact motion. This motion tracking delay can be safely eliminated by adding the contact motion measured between images to the old offset estimate. Step **257** obtains motion from the average, ($H_{mvx}[n],H_{mvy}[n]$) of the current contact velocities:

$$H_{mvx}[n]=\frac{\sum_{i=1}^{i=7} Fi_{mow}[n]Fi_{vx}[n]}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (5)$$

$$H_{mvy}[n]=\frac{\sum_{i=1}^{i=7} Fi_{mow}[n]Fi_{vy}[n]}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (6)$$

The current contact velocities. ($Fi_{vx}[n],F_{vy}[n]$), are retrieved from the path tracking process **245**, which measures them independent of finger identity. Step **258** updates the estimated hand offsets ($H_{eox}[n],H_{eoy}[n]$) using the complete filter equations:

$$H_{eox}[n]=H_{oa}[n]H_{mox}[n]+(1-H_{oa}[n])(H_{eox}[n-1]+H_{mox}[n]\Delta t) \quad (7)$$

$$H_{eoy}[n]=H_{oa}[n]H_{moy}[n]+(1-H_{oa}[n])(H_{eoy}[n-1]+H_{moy}[n]\Delta t) \quad (8)$$

Finally, to provide a similarly conservative estimate of the positions of particular fingers step **259** computes individual finger offsets ($Fi_{eox}[n],Fi_{eoy}[n]$) from the distance between identified contacts and their corresponding default finger positions less the estimated hand offsets. For each identifiable contact i, the offsets are computed as:

$$Fi_{eox}[n]=H_{oa}[n](H_{mox}[n]+Fi_x[n]-Fi_{defx})+(1-H_{oa}[n])(Fi_{eox}[n-1]+Fi_{vx}[n]\Delta t) \quad (9)$$

$$FI_{eoy}[n]=H_{oa}[n](H_{moy}[n]+Fi_y[n]-Fi_{defy})+(1-H_{oa}[n])(Fi_{eoy}[n-1]+Fi_{vy}[n]\Delta t) \quad (10)$$

These finger offsets reflect deviations of finger flexion and extension from the neutral posture. If the user places the fingers in an extreme configuration such as the flattened hand configuration, the collective magnitudes of these finger offsets can be used as an indication of user hand size and finger length compared to the average adult.

The parameters ($H_{eox}[n],H_{eoy}[n]$) and ($Fi_{eox}[n],Fi_{eoy}[n]$) for each hand and finger constitute the estimated hand and finger offset data **252**, which is fed back to the segmentation and identification processes during analysis of the next proximity image. If the other processes need the estimate in absolute coordinates, they can simply add (step **260**) the supplied offsets to the default finger positions, but in many cases the relative offset representation is actually more convenient.

It should be clear to those skilled in the art that many improvements can be made to the above hand position estimation procedure which remain well within the scope of this invention, especially in the manner of guessing the position of lifted hands. One improvement is to make the estimated hand offsets decay toward zero at a constant speed when a hand is lifted rather than decay exponentially. Also, the offset computations for each hand have been independent as described so far. It is actually advantageous to impose a minimum horizontal separation between the estimated left hand position and estimated right hand position such that when a hand such as the right hand slides to the opposite side of the board while the other hand is lifted, the estimated position of the other hand is displaced. In this case the estimated position of the lifted left hand would be forced from default to the far left of the surface, possibly off the surface completely. If the right hand is lifted and the left is not, an equation like the following can be applied to force the estimated right hand position out of the way:

$$Rh_{eox}[n]:=\min(RH_{eox}[n],(LF1_{defx}-RF1_{defx})+Lh_{eox}[n]+\text{min\_hard\_sep}) \quad (11)$$

where $(LF1_{defx}-RF1_{defx})$ is the default separation between left and right thumbs, is the minimum horizontal separation to be imposed, and $LH_{eox}[n]$ is the current estimated offset of the left hand.

FIG. **18** represents the data flow within the proximity image segmentation process **241**. Step **262** makes a spatially smoothed copy **263** of the current proximity image **240** by passing a two-dimensional diffusion operator or Gaussian kernel over it. Step **264** searches the smoothed image **263** for local maximum pixels **265** whose filtered proximity exceeds a significance threshold and exceeds the filtered proximities of nearest neighbor pixels. The smoothing reduces the chance that an isolated noise spike on a single electrode will result in a local maximum-which exceeds the significance threshold, and consolidates local maxima to about one per distinguishable fleshy contact.

Process **268** then constructs a group of electrodes or pixels which register significant proximity around each local maximum pixel by searching outward from each local maximum for contact edges. Each electrode encountered before reaching a contact boundary is added to the local maximum's group. FIG. **19** shows the basic boundary electrode search pattern for an example contact boundary **274**. In this diagram, an electrode or image pixel lies at the tip of each arrow. The search starts at the local maximum pixel **276**, proceeds to the left pixels **277** until the boundary **274** is detected. The last pixel before the boundary **278** is marked as an edge pixel, and the search resumes to the right **279** of the local maximum pixel **276**. Once the left and right edges of the local maximum's row have been found, the search recurses to the rows above and below, always starting **281** in the column of the pixel in the previous row which had the greatest proximity. As the example illustrates, the resulting set of pixels or electrodes is connected in the mathematical sense but need not be rectangular. This allows groups to closely fit the typical oval-shape of flesh contacts without leaving electrodes out or including those from adjacent contacts.

If contacts were small and always well separated, edges could simply be established wherever proximity readings fell to the background level. But sometimes fingertips are only separated by a slight valley or shallow saddle point **210**. To segment adjacent fingertips the partial minima of these valleys must be detected and used as group boundaries. Large palm heel contacts, on the other hand, may exhibit partial minima due to minor nonuniformities in flesh proximity across the contact. If all electrodes under the contact are to be collected in a single group, such partial minima must be ignored. Given a hand position estimate the segmentation system can apply strict edge detection rules in regions of the image where fingertips and thumb are expected to appear but apply sloppy edge detection rules in regions of the image where palms are expected to, appear. This ensures that adjacent fingertips are not joined into a single group and that each palm heel is not broken into multiple groups.

Step **266** of FIG. **18** defines the positions of these segmentation regions using the hand position estimates **252** derived from analyses of previous images. FIG. **20**A shows the extent of the strict and sloppy segmentation regions while the hands are in their default positions, making estimated offsets for both hands zero. Plus signs in the diagram **252** indicate the estimated position of each finger and palm heel in each hand. Rectangular outlines in the lower corners represent the left **284** and right **286** sloppy segmentation regions where partial minima are largely ignored. The T-shaped region remaining is the strict segmentation region **282**, where proximity saddle points must serve as contact boundaries. As a preferred embodiment the sloppy regions are rectangular, their inner boundaries **285** are placed just inside of the columns where the index fingers **202** are expected to lie, and the upper boundaries **287** are placed at the estimated vertical levels of their respective thumbs **201**. The outer and lower boundaries of the sloppy regions are determined by the outside edges of the surface. Due to the decay in estimated hand offsets after hands leave the surface, the sloppy segmentation regions return to the positions shown after the hands have stayed off the surface a few seconds, regardless of hand position at liftoff. FIG. **20**B shows how the sloppy regions follow the estimated hand positions **252** as the right hand moves toward the upper left and the left hand moves toward the lower left. This ensures that the palms and only the palms fall in the sloppy regions as long as the hand position estimates are correct.

FIG. **20**C shows that the left sloppy region **284** is moved left off the surface entirely when the left hand is lifted off the surface and the right hand slides to the left side of the surface. This prevents the fingers of one hand from entering the sloppy segmentation region of the opposite hand. This effect is implemented by imposing a minimum horizontal separation between the sloppy regions and, should the regions get too close to one another, letting the hand with the most surface contacts override the estimated position of the hand with fewer contacts. FIG. **21** is a detailed flow chart of the edge tests which are applied at each searched electrode depending on whether the electrode is in a strict or sloppy segmentation region. Decision diamond **290** checks whether the unsmoothed proximity of the electrode is greater than the background proximity levels. If not, the electrode is labeled an edge electrode in step **304** regardless of the segmentation region or search direction, and in step **305** the search returns to the row maximum to recurse in another direction. If the unsmoothed proximity is significant farther tests are applied to the smoothed proximity of neighboring electrodes depending on whether decision diamond **292** decides the search electrode is in a sloppy or strict region.

If a strict region search is advancing horizontally within a row, decision diamond **306** passes to decision diamond **308** which tests whether the electrode lies in a horizontal or diagonal partial minimum with respect to its nearest neighbor electrodes. If so, a proximity valley between adjacent fingers has probably been detected, the electrode is labeled as an edge **314** and search resumes in other directions **305**. If not, the search continues on the next electrode in the row **302**. If a strict region search is advancing vertically to the next row, decision diamond **306** passes to decision diamond **310** which tests whether the electrode lies in a vertical partial minimum with respect to the smoothed proximity of its nearest neighbor electrodes. If so, a proximity valley between a finger and the thumb has probably been detected, the electrode is labeled as an edge **312** and search resumes in other directions **305**. If not, the search continues into the next row **302**. If decision diamond **294** determines that a sloppy region search is advancing horizontally within a row, stringent horizontal minimum tests are performed to check for the crease or proximity valley between the inner and outer palm heels. To qualify, the electrode must be more than about 2 cm horizontal distance from the originating local maximum, as checked by decision diamond **296**. Also the electrode must be part of a tall valley or partial horizontal minimum which extends to the rows above and below and the next-nearest neighbors within the row, as checked by decision diamond **298**. If so, the electrode is labeled as an edge **300** and search recurses in other directions **305**. All other partial minima within the sloppy regions are

ignored, so the search continues **302** until a background level edge is reached on an upcoming electrode.

In sloppy segmentation regions it is possible for groups to overlap significantly because partial minima between local maxima do not act as boundaries. Typically when this happens the overlapping groups are part of a large fleshy contact such as a palm which, even after smoothing, has multiple local maxima. Two groups are defined to be overlapping if the search originating local maximum electrode of one group is also an element of the other group. In the interest of presenting only one group per distinguishable fleshy contact to the rest of the system, step **270** of FIG. **18** combines overlapping groups into single supergroups before parameter extraction. Those skilled in the art will realize that feedback from high-level analysis of previous images can be applied in various alternative ways to improve the segmentation process and still lie well within the scope of this invention. For example, additional image smoothing in sloppy segmentation regions could consolidate each palm heel contact into a single local maximum which would pass strict segmentation region boundary tests. Care must be taken with this approach however, because too much smoothing can cause finger pairs which unexpectedly enter sloppy palm regions to be joined into one group. Once a finger pair is joined the finger identification process **248** has no way to tell that the fingertips are actually not a single palm heel, so the finger identification process will be unable to correct the hand position estimate or adjust the sloppy regions for proper segmentation of future images.

More detailed forms of feedback than the hand position estimate can be utilized as well. For example, the proximal phalanges(**209** in FIG. **13**) are actually part of the finger but tend to be segmented into separate groups than the fingertips by the vertical minimum test **310**. The vertical minimum test is necessary to separate the thumb group from index fingertip group in the partially closed FIG. **14** and pen grip FIG. **15** hand configurations. However, the proximal phalanges of flattened fingers can be distinguished from a thumb behind a curled fingertip by the fact that it is very difficult to flatten one long finger without flattening the other long fingers. To take, advantage of this constraint, a flattened finger flag **267** is set whenever two or more of the contacts identified as index through pinky in previous images are larger than normal, reliably indicating that fingertips are flattening. Then decision diamond **310** is modified during processing of the current image to ignore the first vertical minimum encountered during search of rows below the originating local minimum **276**. This allows the proximal phalanges to be included in the fingertip group but prevents fingertip groups from merging with thumbs or forepalms. The last step **272** of the segmentation process is to extract shape, size, and position parameters from each electrode group. Group position reflects hand contact position and is necessary to determine finger velocity. The total group proximity, eccentricity, and orientation are used by higher level modules to help distinguish finger, palm, and thumb contacts.

Provided G_E is the set of electrodes in group G, e_z is the unsmoothed proximity of an electrode or pixel e, and e_x and e_y are the coordinates on the surface of the electrode center in centimeters, to give a basic indicator of group position, the proximity-weighted center, or centroid, is computed from positions and proximities of the group's electrodes:

$$G_z = \sum_{e \in G_E} e_z \tag{12}$$

$$G_x = \sum_{e \in G_E} \frac{e_z e_x}{G_z} \tag{13}$$

$$G_y = \sum_{e \in G_E} \frac{e_z e_y}{G_z} \tag{14}$$

Note that since the total group proximity G_z integrates proximity over each pixel in the group, it depends upon both of the size of a hand part, since large hand parts tend to cause groups with more pixels, and of the proximity to or pressure on the surface of a hand part.

Since most groups are convex, their shape is well approximated by ellipse parameters. The ellipse fitting procedure requires a unitary transformation of the group covariance matrix G_{eov} of second moments Q_{xx}, Q_{xy}, G_{yy}:

$$G_{cov} = \begin{bmatrix} G_{xx} & G_{xy} \\ G_{yx} & G_{yy} \end{bmatrix} \tag{15}$$

$$G_{xx} = \sum_{e \in G_E} e_z (G_x - e_x)^2 \tag{16}$$

$$G_{yx} = G_{xy} = \sum_{e \in G_E} e_z (G_x - e_x)(G_y - e_y) \tag{17}$$

$$G_{yy} = \sum_{e \in G_E} e_x (G_y - e_y)^2 \tag{18}$$

The eigenvalues λ_0 and λ_1 of the covariance matrix G_{eov} determine the ellipse axis lengths and orientation G_θ:

$$G_{major} = \sqrt{\lambda_0} \tag{19}$$

$$G_{minor} = \sqrt{\lambda_1} \tag{20}$$

$$G_\theta = \arctan\left(\frac{\lambda_0 - G_{xx}}{G_{xy}}\right) \tag{21}$$

where G_θ is uniquely wrapped into the range (0,180°).

For convenience while distinguishing fingertips from palms at higher system levels, the major and minor axis lengths are converted via their ratio into an eccentricity $G_\in$:

$$G_\varepsilon = \frac{G_{major}}{G_{minor}} \tag{22}$$

Note that since the major axis length is always greater than or equal to the minor axis length, the eccentricity will always be greater than or equal to one. Finally, the total group proximity is empirically renormalized so that the typical curled fingertip will have a total proximity around one:

$$G_z := \frac{G_z}{Z_{averageFingertip}} \tag{23}$$

On low resolution electrode arrays, the total group proximity G_z is a more reliable indicator of contact size as well as finger pressure than the fitted ellipse parameters. Therefore, if proximity images have low resolution, the orientation and eccentricity of small contacts are set to default values rather than their measured values, and total group proximity G_z is used as the primary measure of contact size instead of major and minor axis lengths.

FIG. **22** shows the steps of the path tracking process, which chains together those groups from successive proximity images which correspond to the same physical hand contact. To determine where each hand part has moved since the last proximity image, the tracking process must decide which current groups should be matched with which existing contact paths. As a general rule, a group and path arising from the same contact will be closer to one another than to other groups and paths. Also, biomechanical constraints on lateral finger velocity and acceleration limit how far a finger can travel between images. Therefore a group and path should not be matched unless they are within a distance known as the tracking radius of one another. Since the typical lateral separation between fingers is greater than the tracking radius for reasonable image scan rates touchdown and liftoff are easily detected by the fact that touchdown usually causes a new group to appear outside the tracking radii of existing paths, and liftoff will leave an active path without a group within its tracking radius. To prevent improper breaking of paths at high finger speeds each path's tracking radius P_{rtruck} can be made dependent on its existing speed and proximity.

The first step **320** predicts the current locations of surface contacts along existing trajectories using path positions and velocities measured from previous images. Applying previous velocity to the location prediction improves the prediction except when a finger suddenly starts or stops or changes direction. Since such high acceleration events occur less often than zero acceleration events, the benefits of velocity-based prediction outweigh the potentially bad predictions during finger acceleration. Letting $P_x[n-1],P_y[n-1]$ be the position of path P from time step n−1 and $P_{vx}[n-1]$. $P_{vy}[n-1]$ the last known velocity, the velocity-predicted path continuation is then:

$$P_{predx}[n]=P_x[n-1]+\Delta t P_{vx}[n-1] \quad (24)$$

$$P_{predy}[n]=P_y[n-1]+\Delta t P_{vy}[n-1] \quad (25)$$

Letting the set of paths active in the previous image be PA, and let the set electrode groups constructed in the current image be G, step **322** finds for each group Gk the closest active path and records the distance to it:

??????? (26)

Gk closestPdist 2=min Pl PA d 2(Gk,Pl) Gk G (27)

where the squared Euclidean distance is an easily computed distance metric:

$$a^2(Gk,Pl)=(Gk_x-Pl_{predx})^2+(Gk_y-Pl_{predy})^2 \quad (28)$$

Step **324** then finds for each active path Pl, the closest active group and records the distance to it:

9 Pl closestG=arg min Gk Gd 2(Gk,Pl) Pl PA (29)

Pl closestGdist 2=min Gk Gd 2(Gk,Pl) Pl PA (30)

In step **326**, an active group Gk and path Pl are only paired with one another if they are closest to one another, i.e., $Gk_{closestP}$ and $Pl_{closestG}$ refer to one another, and the distance between them is less than the tracking radius. All of the following conditions must hold:

$$Gk_{closestP}=Pl \quad (31)$$

$$Pl_{closestG}=Gk \quad (32)$$

$$Pl_{closestGdist2}<Pl_{mack2} \quad (33)$$

To aid in detection of repetitive taps of the same finger, it may be useful to preserve continuity of path assignment between taps over the same location. This is accomplished in step Via USPTO EFS **334** by repeating steps **322-326** using only groups which were left unpaired above and paths which were deactivated within the last second or so due to finger liftoff.

In step **336**, any group which has still not be paired with an active or recently deactivated path is allocated a new path, representing touchdown of a new finger onto the surface. In step **344**, any active path which cannot be so paired with a group is deactivated, representing hand part liftoff from the surface.

Step **346** incorporates the extracted parameters of each group into its assigned path via standard filtering techniques. The equations shown below apply simple autoregressive filters to update the path position ($P_x[n],P_y[n],P_z[n]$), velocity ($P_x[n],P_y[n]$), and shape ($P_\theta[n]$, $P_\in[n]$) parameters from corresponding group parameters, but Kalman or finite impulse response filters would also be appropriate.

If a path P has just been started by group G at time step n, i.e., a hand part has just touched down, its parameters are initialized as follows:

$$P_x[n]=G_x \quad (34)$$

$$P_y[n]=G_y \quad (35)$$

$$P_z[n]=G_z \quad (36)$$

$$P_\theta[n]=G_\theta \quad (37)$$

$$P_\in[n]=G_\in \quad (38)$$

$$P_{vx}[n]=0 \quad (39)$$

$$P_{vy}[n]=0 \quad (40)$$

$$P_{vz}[n]=G_z/\Delta t \quad (41)$$

else if group G is a continuation of active path P[n−1] to time step n:

$$P_x[n]=G_\alpha G_x+(1-G_\alpha)(P_{predx}[n-1]) \quad (42)$$

$$P_y[n]=G_\alpha G_y+(1-G_\alpha)(P_{predy}[n-1]) \quad (43)$$

$$P_z[n]=G_\alpha G_z+(1-G_\alpha)(P_{predz}[n-1]) \quad (44)$$

$$P_\theta[n]=G_\alpha G_\theta+(1-G_\alpha)(P_\theta[n-1]) \quad (45)$$

$$P_\in[n]=G_\alpha G_\in+(1-G_\alpha)(P_\in[n-1]) \quad (46)$$

$$P_{vx}[n]=(P_x[n]-P_x[n-1])/\Delta t \quad (47)$$

$$P_{vy}[n]=(P_y[n]-P_y[n-1])/\Delta t \quad (48)$$

$$P_{vz}[n]=(P_z[n]-P_z[n-1])/\Delta t \quad (49)$$

It is also useful to compute the magnitude P_{speed} and angle P_{dir} from the velocity vector (P_{vx}, P_{vz}). Since the reliability of position measurements increases considerably with total proximity P_z, the low-pass filter pole G_α is decreased for groups with total proximities lower than normal. Thus when

signals are weak, the system relies heavily on the previously established path velocity, but when the finger firmly touches the surface causing a strong, reliable signal, the system relies entirely on the current group centroid measurement.

The next process within the tracking module is contact identification. On surfaces large enough for multiple hands, the contacts of each hand tend to form a circular cluster, and the clusters tend to remain separate because users like to avoid entangling the fingers of opposite hands. Because the arrangement of fingers within a hand cluster is independent of the location of and arrangement within the other hand's cluster, the contact identification system is hierarchically split. The hand identification process **247** first decides to which cluster each contact belongs. Then a within-cluster identification process **248** analyzes for each hand the arrangement of contacts within the hand's cluster, independent of the other hand's cluster. Because within-cluster or finger identification works the same for each hand regardless of how many hands can fit on the surface, it will be described first. The description below is for identification within the right hand. Mirror symmetry must be applied to some parameters before identifying left hand contacts.

FIG. **23** shows the preferred embodiment of the finger identification process **248**. For the contacts assigned to each hand this embodiment attempts to match contacts to a template of hand part attractor points, each attractor point having an identity which corresponds to a particular finger or palm heel. This matching between contact paths and attractors should be basically one to one but in the case that some hand parts are not touching the surface, some attractors will be left unfilled, i.e., assigned to the null path or dummy paths.

Step **350** initializes the locations of the attractor points to the approximate positions of the corresponding fingers and palms when the hand is in a neutral posture with fingers partially curled. Preferably these are the same default finger locations (Fi_{defx},Fi_{defy}) employed in hand offset estimation. Setting the distances and angles between attractor points from a half-closed hand posture allows the matching algorithm to perform well for a wide variety of finger flexions and extensions.

The resulting attractor points tend to lie in a ring as displayed by the crosses in FIG. **24**. The identities of attractor points **371-377** correspond to the identities of hand parts **201-207**. If the given hand is a left hand, the attractor ring must be mirrored about the vertical axis from that shown. FIG. **24** also includes line segments **380** forming the Voronoi cell around each attractor point. Every point within an attractor's Voronoi cell is closer to that attractor than any other attractor. When there is only one contact in the cluster and its features are not distinguishing, the assignment algorithm effectively assigns the contact to the attractor point of the Voronoi cell which the contact lies within. When there are multiple surface contacts in a hand cluster, they could all lie in the same Voronoi cell, so the assignment algorithm must perform a global optimization which takes into account all of the contact positions at once.

Alternative embodiments can include additional attractors for other hand part or alternative attractor arrangements for atypical hand configurations. For example, attractors for forepalm contacts can be placed at the center of the ring, but since the forepalms typically do not touch the surface unless the rest of the hand is flattened onto the surface as well, forepalm attractors should be weighted such that contacts are assigned to them only when no regular attractors are left unassigned.

For optimal matching accuracy the ring should be kept roughly centered on the hand cluster. Therefore step **352** translates all of the attractor points for a given hand by the hand's estimated position offset. The final attractor positions ($Aj_x[n]$,$Aj_y[n]$) are therefore given by:

$$Aj_x[n]=H_{eox}[n]+Fj_{defx} \quad (50)$$

$$Aj_y[n]=H_{eoy}[n]+Fj_{defy} \quad (51)$$

In alternative embodiments the attractor ring can also be rotated or scaled by estimates of hand rotation and size such as the estimated finger offsets, but care must be taken that wrong finger offset estimates and identification errors do not reinforce one another by severely warping the attractor ring.

Once the attractor template is in place, step **354** constructs a square matrix $[d_{ij}]$ of the distances in the surface plane from each active contact path Pi to each attractor point Aj. If there are fewer surface contacts than attractors, the null path **P0**, which has zero distance to each attractor, takes place of the missing contacts. Though any distance metric can be used, the squared Euclidean distance,

$$d_{ij}=(Aj_x[n]-Pi_x[n])^2+(Aj_y[n]-Pi_y[n]) \quad (52)$$

is preferred because it specially favors assignments wherein the angle between any pair of contacts is close to the angle between the pair of attractors assigned to those contacts. This corresponds to the biomechanical constraint that fingertips avoid crossing over one another, especially while touching a surface.

In step **356**, the distances from each contact to selected attractors are weighted according to whether the geometrical features of the given contact match those expected from the hand part that the attractor represents. Since the thumb and palm heels exhibit the most distinguishing geometrical features, weighting functions are computed for the thumb and palm heel attractors, and distances to fingertip attractors are unchanged. In a preferred embodiment, each weighting function is composed of several factor versus feature relationships such as those plotted approximately in FIG. **25**. Each factor is designed to take on a default value of 1 when its feature measurement provides no distinguishing information, take on larger values if the measured contact feature uniquely resembles the given thumb or palm hand part, and take on smaller values if the measured feature is inconsistent with the given attractor's hand part. The factor relationships can be variously stored and computed as lookup tables, piecewise linear functions, polynomials, trigonometric functions, rational functions, or any combination of these. Since assignment between a contact and an attractor whose features match is favored as the weighted distance between becomes smaller, the distances are actually weighted (multiplied) with the reciprocals of the factor relationships shown.

FIG. **25**A shows the right thumb and right inner palm heel orientation factor versus orientation of a contact's fitted ellipse. Orientation of these hand parts tends to be about 120°, whereas fingertip and outer palm heel contacts are usually very close to vertical (90°), and orientation of the left thumb and left inner palm heel averages 60°. The right orientation factor therefore approaches a maximum at 120°. It approaches the default value of 1 at 0°, 90°, and 180° where orientation is inconclusive of identity, and reaches a minimum at 60°, the favored orientation of the opposite thumb or palm heel. The corresponding relationship for the left thumb and inner palm heel orientation factor is flipped about 90°.

FIG. **25**B approximately plots the thumb size factor. Since thumb size as indicated by total proximity tends to peak at two or three times the size of the typical curled fingertip, the thumb size factor peaks at these sizes. Unlike palm heels, thumb contacts can not be much larger than two or three times the default fingertip size, so the thumb factor drops back down

for larger sizes. Since any hand part can appear small when touching the surface very lightly or just starting to touchdown, small size is not distinguishing, so the size factor defaults to **1** for very small contacts.

FIG. **25**C approximately plots the palm heel size factor. As more pressure is applied to the palms, the palm heel contacts can grow quite large, remaining fairly round as they do so. Thus the palm heel size factor is much like the thumb size factor except the palm factor is free to increase indefinitely. However, fingertip contacts can grow by becoming taller as the fingers are flattened. But since finger width is constant, the eccentricity of an ellipse fitted to a growing fingertip contact increases in proportion to the height. To prevent flattened fingers from having a large palm factor, has little effect for palms, whose eccentricity remains near **1**, but cancels the high proximities of flattened fingertips. Though directly using fitted ellipse width would be less accurate for low resolution electrode arrays, the above ratio basically captures contact width.

Another important distinguishing feature of the palm heels is that wrist anatomy keeps the centroids of their contacts separated from one other and from the fingers by several centimeters. This is not true of the thumb and fingertips, which can be moved within a centimeter of one another via flexible joints. The inter-palm separation feature is measured by searching for the nearest neighbor contact of a given contact and measuring the distance to the neighbor. As plotted approximately in FIG. **25**D, the palm separation factor quickly decreases as the separation between the contact and its nearest neighbor falls below a few centimeters, indicating that the given contact (and its nearest neighbor) are not palm heels. Unlike the size and orientation factors which only become reliable as the weight of the hands fully compresses the palms, the palm separation factor is especially helpful in distinguishing the palm heels from pairs of adjacent fingertips because it applies equally well to light, small contacts.

Once the thumb and palm weightings have been applied to the distance matrix, step **358** finds the one-to-one assignment between attractors and contacts which minimizes the sum of weighted; distances between each attractor and it's assigned contact. For notational purposes, let a new matrix $[c_{ij}]$ hold the weighted distances:

$$c_{ij} = \begin{cases} d_{ij} / (Pi_{thumb\_size\_fact} Pi_{orient\_fact}) & \text{if } j = 1 \\ d_{ij} & \text{if } 2 \le j \le 5 \\ d_{ij} / (Pi_{palm\_size\_fact} Pi_{palm\_sep\_fact}) & \text{if } j = 6 \\ d_{ij} / (Pi_{palm\_size\_fact} Pi_{palm\_sep\_fact}) & \text{if } j = 7 \end{cases} \quad (53)$$

Mathematically the optimization can then be stated as finding the permutation $\{\pi_1, \ldots, \pi_7\}$ of integer hand part identities $\{1, \ldots, 7\}$ which minimizes:

$$\sum_{i=1}^{7} c_{i\pi_i} \quad (54)$$

where c_{ij} is the weighted distance from contact i to attractor j, and contact i and attractor j are considered assigned to one another when π_i=j. This combinatorial optimization problem, known more specifically in mathematics as an assignment problem, can be efficiently solved by a variety of well-known mathematical techniques, such as branch and bound, localized combinatorial search, the Hungarian method, or network flow solvers. Those skilled in the art will recognize that this type of combinatorial optimization problem has a mathematically equivalent dual representation in which the optimization is reformulated as a maximization of a sum of dual parameters. Such reformulation of the above hand part identification method as the dual of attractor-contact distance minimization remains well within the scope of this invention.

To avoid unnecessary computation, decision diamond **360** ends the finger identification process at this stage if the hand assignment of the given contact cluster is only a tentative hypothesis being evaluated by the hand identification module **247**. However, if the given hand assignments are the final preferred hypothesis, further processes verify finger identities and compile identity statistics such as finger counts.

The identifications produced by this attractor assignment method are highly reliable when all five fingers are touching the surface or when thumb and palm features are unambiguous. Checking that the horizontal coordinates for identified fingertip contacts are in increasing order easily verifies that fingertip identities are not erroneously swapped. However, when-only two to four fingers are touching, yet no finger strongly exhibits thumb size or orientation features, the assignment of the innermost finger contact may wrongly indicate whether the contact is the thumb. In this case, decision diamond **362** employs a thumb verification process **368** to take further measurements between the innermost finger contact and the other fingers. If these further measurements strongly suggest the innermost finger contact identity is wrong, the thumb verification process changes the assignment of the innermost finger contact. Once the finger assignments are verified, step **364** compiles statistics about the assignments within each hand such as the number of touching fingertips and bitfields of touching finger identities. These statistics provide convenient summaries of identification results for other modules.

FIG. **26** shows the steps within the thumb verification module. The first **400** is to compute several velocity, separation, and angle factors for the innermost contact identified as a finger relative to the other contacts identified as fingers. Since these inter-path measurements presuppose a contact identity ordering, they could not have easily been included as attractor distance weightings because contact identities are not known until the attractor distance minimization is complete. For the factor descriptions below, let FI be the innermost finger contact, FN be the next innermost finger contact, FO be the outermost finger contact.

The separation between thumb and index finger is often larger than the separations between fingertips, but all separations tend to grow as the fingers are outstretched. Therefore an inner separation factor inner_separation_fact is defined as the ratio of the distance between the innermost and next innermost finger contacts to the average of the distances between other adjacent fingertip contacts, avg_separation: 12 innerseparationfact min

$$innerseparationfact \approx \min\left(1, \frac{\sqrt{(FI_x - FN_x)^2 + (FI_y - FN_y)^2}}{avgseparation}\right) \quad (55)$$

The factor is clipped to be greater than one since an innermost separation less than the average can occur regardless of whether thumb or index finger is the innermost touching finger. In case there are only two finger contacts, a default average separation of 2-3 cm is used. The factor tends to