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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 March 2005 (10.03.2005)

PCT

(10) International Publication Number
WO 2005/022458 A2

(51) International Patent Classification[7]: G06T

(21) International Application Number: PCT/US2004/025528

(22) International Filing Date: 5 August 2004 (05.08.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/497,045 22 August 2003 (22.08.2003) US
10/912,655 4 August 2004 (04.08.2004) US

(71) Applicant *(for all designated States except US)*: **ATRUA TECHNOLOGIES, INC.** [US/US]; 1696 Dell Avenue, Campell, CA 95008 (US).

(72) Inventor: **RUSSO, Anthony, P.**; 58 W 75th Street, #3A, New York, NY 10023 (US).

(74) Agent: **HAVERSTOCK, Thomas, B.**; Haverstock & Owens LLP, 162 North Wolfe Road, Sunnyvale, CA 94086 (US).

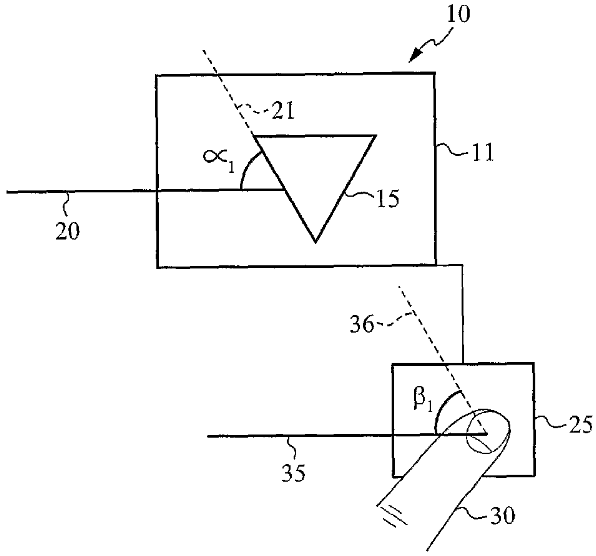
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR,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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SYSTEM FOR AND METHOD OF GENERATING ROTATIONAL INPUTS



(57) Abstract: A system for and method of obtaining rotation information is disclosed. The method comprises capturing a plurality of patterned images from a plurality of locations, correlating the plurality of patterned images to generate sets of linear differences, and using the sets of linear differences to generate the rotation information. Preferably, the plurality of locations comprise a first part of a fingerprint swipe sensor and a second part of the fingerprint swipe sensor, each part configured to capture a part of a fingerprint image. Each part captures a part of the fingerprint image at two positions and correlates the parts at the two positions to determine one set of linear differences. Together, the sets of linear differences are used to calculate the rotation information, which can be used to emulate a rotational device such as a steering wheel, a joystick, or a navigation bar.

WO 2005/022458 A2

SYSTEM FOR AND METHOD OF GENERATING ROTATIONAL INPUTS

RELATED APPLICATION

This application claims priority under 35 U.S.C. § 119(e) of the co-pending U.S. provisional application Serial Number 60/497,045, filed on August 22, 2003, and titled "ROTATIONAL INPUT METHOD PATENT." The provisional application Serial Number 60/497,045, filed on August 22, 2003, and titled "ROTATIONAL INPUT METHOD PATENT" is hereby incorporated by reference.

FIELD OF THE INVENTION

This invention relates to the field of biometric sensors. In particular, this invention relates to systems and methods that use fingerprint images to emulate electronic positioning devices.

BACKGROUND OF THE INVENTION

The emergence of portable electronic computing platforms allows functions and services to be enjoyed wherever necessary. Palmtop computers, personal digital assistants, mobile phones, portable game consoles, biometric/health monitors, remote controls, digital cameras, to name a few, are some daily-life examples of portable electronic computing platforms. The desire for portability has driven these computing platforms to become smaller and, consequently, to have longer battery life. A dilemma occurs when these ever-smaller devices require efficient ways to collect user input.

Portable electronic computing platforms need these user inputs for multiple purposes, including (a) navigating a cursor or a pointer to a certain location on a display, (b) selecting (e.g., choosing or not choosing) an item or an action, and (c) orientating (e.g., changing direction with or without visual feedback) an input device.

Concepts for user input from much larger personal computers have been borrowed. Micro joysticks, navigation bars, scroll wheels, touch pads, steering wheels and buttons have all been adopted, with limited success, in conventional devices. All these positioning devices consume substantial amounts of valuable surface real estate on a portable device. Mechanical positioning devices such as joysticks, navigation bars and scroll wheels can wear out and become unreliable. Their sizes and required movements often preclude optimal ergonomic placement on portable computing platforms.

Prior art methods calculate rotation by rotating one frame with respect to another and then applying standard correlation methods. These methods require the selection of a pivot point (e.g., the origin), followed by additional computations. These computations are not helpful for determining linear motion (*e.g.*, non-rotational movement in the x- and y-directions). Such a shortcoming makes prior art systems even more inefficient when used in portable devices, in which both rotational and linear movement are required, such as when emulating, respectively, a steering wheel and a pointing device.

SUMMARY OF THE INVENTION

The present invention discloses a system for and method of obtaining rotation information from a patterned image, such as a fingerprint image. Embodiments of the present invention thus require smaller footprints than those that use joy sticks, steering wheels, and other, larger devices that require additional power. Embodiments of the present invention use linear correlation methods that are easier to use than rotational and other methods such as those using trigonometric functions. Embodiments of the present invention thus use simpler algorithms that can be performed faster and more reliably.

In a first aspect of the present invention, a method of obtaining rotation information comprises capturing a plurality of patterned images from a plurality of locations, correlating the plurality of patterned images to generate sets of linear differences, and using the sets of linear differences to generate the rotation information. The plurality of locations comprise a first part of a sensor and a second part of the sensor. A first of the plurality of patterned images is captured in the first part of the sensor and a second of the plurality of patterned images is captured in the second part of the sensor.

Preferably, the sensor is a biometric image sensor, such as a finger image sensor. The first of the plurality of patterned images and the second of the plurality of patterned images together correspond to a fingerprint image in a first position on the sensor. A third of the plurality of patterned images is captured in the first part of the sensor and a fourth of the plurality of patterned images is captured in the second part of the sensor. The third of the plurality of patterned images and the fourth of the plurality of patterned images together correspond to the fingerprint image in a second position on the sensor. In one embodiment, the rotation information corresponds to an angular difference between the first position and the second position.

In one embodiment, correlating the plurality of patterned images comprises correlating the first patterned image with the third patterned image to generate a first set of linear differences from the sets of linear differences, correlating the second patterned image with the fourth patterned image to generate a second set of linear differences from the sets of linear differences, and correlating a first combination of the first patterned image and the second patterned image with a second combination of the third patterned image and the fourth patterned image to generate a third set of linear differences from the sets of linear differences. Correlating the first patterned image with the third patterned image, correlating the second patterned image with the fourth patterned image, and correlating the first combination with the second combination all comprise performing a cross correlation. In one embodiment, the cross correlation is either a normalized cross correlation or a standardized cross correlation.

In one embodiment, the first part of the sensor and the second part of the sensor are contiguous. Alternatively, the first part of the sensor and the second part of the sensor are not contiguous.

In one embodiment, the first part of the sensor comprises a first sub-frame of pixels and the second part of the sensor comprises a second sub-frame of pixels. In this embodiment, capturing the first patterned image comprises storing in the first sub-frame first data corresponding to the first patterned image, capturing the second patterned image comprises storing in the second sub-frame second data corresponding to the second patterned image, capturing the third patterned image comprises storing in the first sub-frame third data corresponding to the third patterned image, and capturing the fourth patterned image comprises storing in the second sub-frame fourth data corresponding to the fourth patterned image. Correlating the first patterned image with the third patterned image comprises correlating the first data with the third data to generate first and second linear differences from the first set of linear differences. Correlating the second patterned image with the fourth patterned image comprises correlating the second data with the fourth data to generate first and second linear differences from the second set of linear differences. And correlating the first combination with the second combination comprises correlating a combination of the first data and the second data with a combination of the third data and the fourth data to generate first and second linear differences from the third set of linear differences.

In another embodiment, correlating comprises determining a lag to correlate elements of one of the first and second sub- frames, the lag and a difference between the elements corresponding to first and second linear differences from one of the sets of linear differences. Each element corresponds to a row of one of the first and second sub-frames. Alternatively, each element corresponds to a column of one of the first and second sub- frames.

In another embodiment, the method further comprises filtering the first set of linear differences, the second set of linear differences, the third set of linear differences, and the rotation information. Filtering comprises multiplying by a scaling factor, performing a smoothing function, and performing a clipping function.

Preferably, the finger image sensor is a finger swipe sensor. Alternatively, the finger image sensor is a finger placement sensor.

In another embodiment, the method further comprises using the rotation information on a host platform having a display screen, the rotation information used to rotate an object on the display screen, thereby emulating a computer input device. The computer input device is selected from the group consisting of a steering wheel, a joystick, and a navigation bar. Emulating a computer input device comprises moving the object on the display screen at a rate related to the angular difference or the angular position.

In accordance with a second aspect of the invention, a system for obtaining rotation information comprises means for capturing a plurality of patterned images from a plurality of locations and means for correlating the plurality of patterned images to generate sets of linear differences and for using the sets of linear differences to generate the rotation information.

In accordance with a third aspect of the present invention, a method of emulating a rotational device using a pattern comprises capturing a first image of the pattern at a first orientation, capturing a second image of the pattern at a second orientation, correlating the first image with the second image to calculate linear differences between the first orientation and the second orientation, translating the linear difference into rotational data, and using the rotational data to emulate the movement of a rotational device.

In accordance with a fourth aspect of the present invention, a system for emulating a positional device comprises a sensor for capturing an image of a pattern and a processor coupled to the sensor. The processor is configured to calculate linear differences between a first position of the image of the pattern and a second position of the image of the pattern and to translate the linear differences into rotational data corresponding to a rotation of the image of the pattern.

In accordance with a fifth aspect of the present invention, a method of sensing rotation of an object on an image sensor comprises sensing a first image of the object, sensing a second image of the object, and comparing the first image with the second image to determine whether there is linear motion in each of at least two portions of an area containing the first image and the second image to determine whether the object remained stationary, moved in a linear manner, or rotated.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

Figures 1A and 1B show how an electronic image is rotated by rotating a finger on a finger image sensor, in accordance with the present invention.

Figure 2 shows a fingerprint image where ridges are shown in black and valleys are shown in white and indicating areas of bifurcation and ridge endings.

Figures 3A-D shows left- and right-hand sections of a fingerprint sensor with a portion of a fingerprint image placed upon it as the fingerprint image is linearly moved and rotated in accordance with the present invention.

Figure 4 is a flowchart of a frame acquisition and image correlation procedure in accordance with the present invention.

Figure 5A shows a finger image sensor in a horizontal orientation.

Figure 5B shows a finger image sensor in a vertical orientation.

Figure 6 shows pixel data from a frame (slice) from a fingerprint sensor.

Figure 7 shows pixel data from a fingerprint sensor during different iterations of reconstruction in accordance with the present invention.

Figures 8-12 show different partition configurations of a frame in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

The present invention is directed to systems for and methods of determining the rotational position and movement of an arbitrary patterned material imaged by an imaging sensor. Preferably, the arbitrary patterned material is a finger and the rotational position and movement of the image of the finger are determined.

Embodiments of the present invention advantageously determine and collect finger rotational information for use in a digital device and most preferably in personal computing devices. Unlike prior art rotational position correlators, which are non-linear, requiring trigonometric functions like sine, cosine, and tangent calculations, embodiments of the present invention use a linear correlation method that is easier to implement and more computationally efficient. Embodiments of the present invention allow for extremely efficient calculation of linear motion from the components used to determine the rotational motion, thereby reducing the complexity of systems that require one sensor to be used to gather both linear and rotational movement inputs.

A system in accordance with embodiments of the present invention reconstructs fingerprint images from swipe sensors, thereby efficiently providing rotational motion data along with data necessary to reconstruct the image. The system is particularly well suited for applications that do not require high precision rotational information. Methods of and systems for fingerprint sensing are described in detail in the U.S. Patent Application Serial Number 10/194,994, filed July 12, 2002, and titled "Method and System for Biometric Image Assembly from Multiple Partial Biometric Frame Scans," and in the U.S. Patent Application Serial Number 10/099,558, filed March 13, 2002, and titled "Fingerprint Biometric Capture Device and Method with Integrated On-Chip Data Buffering," both of which are hereby incorporated by reference in their entireties. In the preferred embodiment, the fingerprint sensor is an Atrua Wings ATW 100 capacitive swipe sensor by Atrua Technologies, Inc., at 1696 Dell Avenue, Campbell, California 95008.

A key aspect of the present invention is determining rotation from linear correlation rather than prior art methods that determine rotation by rotating one frame with respect to another, and then applying standard correlation methods. The prior art methods require choosing a pivot point (center of origin) and then performing additional computation. Furthermore, such computation is not helpful for determining linear motion (non-rotational movement in the x- and y-directions). These computations are even more inefficient in portable electronic devices where it may be important to calculate both kinds of movement, for instance, when emulating a pointing device and a steering wheel on one component.

Most of the prior art is concentrated on calculating exact rotational movement, and therefore requires the more precise steps outlined above. However, many applications do not require such precision, and it is these cases where the new invention is best suited. Embodiments of the present invention make use of the fact that, as a finger rotates clockwise, the left side of the image will appear to be moving upward, while the right half of the finger will appear to be moving downward. This is sometimes referred to as shear. The opposite is true of counterclockwise motion. Furthermore, in both cases, the left side will appear to be moving toward the right, and the right side will appear to be moving toward the left, as shown in Figures 2B and 2C. On a typical swipe type of finger imaging sensor, which is mounted horizontally so that its height is much smaller than its width, the upward and downward motion is easily observed and exploitable, but because the sensor is so small in height, observing the motion in the x-direction is more difficult. If a swipe sensor is mounted vertically, then it will be easier to observe motion in the x-direction than in the y-direction due to the device's small width in this case. The invention is equally applicable to both cases. It is also equally applicable on a large fingerprint placement sensor, where movement in both the x- and y-directions is easily observable and can be exploited.

Using the observations outlined above, embodiments of the present invention use the simpler linear correlation methods to determine rotational movement, which will occur when motion of the left side of a finger image is in an opposite direction to that of the right side. For instance, if the left half is moving upward and the right half downward, there is clockwise rotational movement as shown in Figure 2D, discussed below. If the linear movement detected at the left edge and right edge of the sensor are substantially equal but opposite, the center of rotation is at or near the center of the sensor. If the linear movement is not equal but opposite, the center of rotation can be calculated and will be closer to the end of the sensor with the smaller amount of linear movement. If the linear movement is not equal but opposite, the center of rotation can be calculated. If both sides are moving in the same direction, as in Figure 2B, then likely only overall linear movement is observed. It will be appreciated that the present invention can determine an angle of rotation even if the center of rotation is displaced, as when a finger slides along the sensor as it rotates.

Accordingly, the present invention offers a reliable and computationally-efficient solution to obtain high-resolution rotational information about a user's finger as it contacts a finger imaging sensor, so that such a sensor can, for example, emulate a steering wheel for use in gaming, or rotate the image of a map for easier viewing on a display.

Figure 1A shows a system 10 used to illustrate one embodiment of the present invention. The system 10 comprises a finger image sensor 20 coupled to a display device 11, displaying a triangular electronic image 15. The display device 11 can be a monitor used with a personal computer, a screen on a personal digital assistant (PDA) or game device, or any other kind of display device. Figure 1A also shows a finger 30 placed on the finger image sensor 25. It will be appreciated that the finger 30 has patterns on a surface contacting the finger image sensor 25 and that the finger image sensor 25 captures images of those patterns. A coordinate axis 20 makes an angle α_0 with an edge 21 of the triangular image 15. A coordinate axis 35 makes an angle β_0 with a line segment 36 associated with the finger 30. Figure 1B shows the system 10 after the finger 30 has been rotated so that the coordinate axis 35 makes an angle β_1 with the line segment 36. In accordance with the present invention, the triangular image 15 is rotated so that the coordinate axis 20 makes an angle α_1 with the edge 21. Thus, in accordance with one embodiment, rotating the finger 30 through an angle $\beta_1 - \beta_0$ ($\Delta\beta$) results in rotating the triangular electronic image through an angle $\alpha_1 - \alpha_0$ ($\Delta\alpha$).

It will be appreciated that $\Delta\alpha$ can correspond to $\Delta\beta$ in any number of ways. For example, $\Delta\alpha$ can equal $\Delta\beta$, $\Delta\alpha$ can be a multiple of $\Delta\beta$, $\Delta\alpha$ can be a fraction of $\Delta\beta$, $\Delta\alpha$ can be a multiple of $\Delta\beta$ plus some offset, etc. It will also be appreciated that in accordance with one embodiment, the finger 30 does not have to maintain a pivot point on the finger image sensor 25. The finger 30 can be moved horizontally or vertically on the finger image sensor 25 before, while, or after it is rotated so that the line segment 36 is displaced horizontally, or vertically, or both, and the angle $\Delta\beta$ still determined. It will also be appreciated that vertical and horizontal movements of the finger 30 can also captured to vertically and horizontally displace the triangular image 15 on the display device 11. It will also be appreciated that the triangular image 15 can be moved at a rate related to $\Delta\beta$ (called the rate mode) or at a rate related to β_1.

While the finger image sensor 25 is depicted as a placement sensor, it will be appreciated that other types of sensors can be used in accordance with the present invention. Preferably, the finger image sensor 25 is a swipe sensor, described in more detail below.

Figure 2 shows a typical fingerprint image, including physical ridges and valleys on the surface of the finger. The pattern of ridges and valleys has proven to be unique among very large populations of human beings, especially the ridge endings and bifurcations called "minutiae." Fingers also often have other measurable surface features such as pores, scars, and wrinkles. It is the overall pattern of features–not the unique individual features-- that are tracked to measure the distance, rotation, direction, or speed that a fingerprint has moved on the finger image sensor.

Many prior art electronic finger imaging sensors actively sense the entire surface of a fingerprint at the same time. Whether based on optical or electrical sensing methods, these sensors have a surface area at least as large as a typical person's fingertip pad (typically 15mm x 15mm). Using these sensors the user simply places his finger on the sensor until the image is captured. These sensors, now known as placement sensors, contain rows and columns and can capture large images, typically ranging from 250-500 rows and 200-500 columns depending on the sensor's capabilities and size. Such devices are capable of sensing rotational input, and can indeed be used with the new invention to collect rotational information, but they are larger than today's more miniaturized finger sensors.

The most promising of the miniaturized sensors is one that is fully sized in one direction (typically in width) but abbreviated in the other (typically height). This results in a sensor that only is capable of sensing a small rectangular portion of the fingerprint at any one time.

Such smaller sensors are better suited for use with the present invention, not only because they are more apropos for portable devices, but also because they produce smaller images. The smaller images have less data in them, making the computations less intense.

While it is possible to ignore or mask off data from a larger sensor to make it resemble a smaller one, such an approach is not ideal, because it does not guarantee that the finger of the user is even touching the area of interest. With a swipe sensor, this is not an issue.

Embodiments of the present invention can acquire rotational position data from any device capable of imaging the surface of a human finger or other patterned material and is therefore not limited to use with placement or swipe finger image sensors, which typically provide at least 250 dots per inch resolution. The present invention will also work with lower or higher resolution devices that may become available in the future.

Figure 3A shows the left half 205A and the right half 205B of a finger image sensor 205. The finger image sensor 205 has placed upon it a finger identified by the fingerprint image 203 having identifiable ridge portions 201 and 202. Figure 3A also shows an x-coordinate axis and a y-coordinate axis, with the arrow of each axis pointing in a direction of increasing values for the respective axis. Figure 3A shows the fingerprint image 203 in a first orientation on the finger image sensor 205 and thus in a first orientation with respect to the x-coordinate axis and the y-coordinate axis. Figure 3B shows the fingerprint image after it has been moved linearly in a vertical position, in a direction of decreasing values for the y-coordinate. Figure 3C shows the fingerprint image 203 after it has been rotated counter-clockwise on the finger image sensor 205. It is seen by comparing the orientations of the fingerprint image 203 from Figure 3B to Figure 3C that the y-coordinates of the identifiable portions 201 and 202 in the left half 205A have both decreased and that the y-coordinates in the right half 205B have both increased. Figure 3D shows the fingerprint image 203 after it has been rotated clockwise on the finger image sensor 205. It is seen by comparing the orientations of the fingerprint image 203 from Figure 3C to Figure 3D that the y-coordinates of the identifiable portions 201 and 202 in the left half 205A have both increased and that the y-coordinates in the right half 205B have both decreased.

Figure 4 is a flowchart for an algorithm 210 for determining rotational movement or placement in accordance with a preferred embodiment of the present invention. In the start step 211, a user is prompted by an operating system or application executing on a host (not shown). Next, in the step 212, a finger image sensor (not shown) in accordance with the present invention is initialized, readied for reading data. This step 212 comprises powering on the sensor ~~on~~ and making sure it is ready to capture a fingerprint image. Step 212 can also include setting contrast and brightness levels, setting the sensor to a desired data acquisition mode, calibrating the sensor, or otherwise initializing the sensor. It will be appreciated that step 212 is not required if the sensor has already been initialized.

Next, in the step 213, a frame is read by the sensor at a rate supported by it or by the hardware platform's computing power and bandwidth. In the step 215, the properties of the frame are estimated to determine whether it is useful. The metrics of the frame are analyzed in the step 220, to determine whether the frame is useful. If the frame is useful, it is kept and processing continues in the step 225; otherwise, the frame is disregarded, and processing continues in the step 255. As described in more detail below, in a preferred embodiment the usefulness of a frame is determined by measuring image statistics such as the average value and the variance of pixel data in the frame. The usefulness of a frame is directly related to whether or not a finger is present on the sensor. It will be appreciated that the step 215 can be eliminated if a less efficient implementation is acceptable, or when the sensor only generates frames when a finger is present on it.

In the step 225, the current frame (*e.g.*, the frame most recently read and currently being processed) is correlated with the last stored useful frame. On the very first iteration, since there is no "last useful frame," the current frame is copied to the last useful frame. In a preferred embodiment, the frame is divided into a left half and a right half. It will be appreciated, however, that a frame can be divided into any number of parts. Next, in the step 230, the linear movement of the left half of the frame and the linear movement of the right half of the frame are both calculated. In the step 235, using the linear movement of the left half of the frame and the linear movement of the right half of the frame, the overall linear movement of the frame is calculated. This calculation is described in more detail below. In the step 240, the calculations made in the step 235 are used to calculate the rotational movement of the fingerprint image.

Next, in the step 245 the process checks whether there was any movement, linear or rotational, of the fingerprint image. If there was movement, the process continues in the step 250, otherwise it continues in the step 255. In the step 250, the last frame useful frame is updated, and in the step 251, the last useful frame is stored. Processing then continues in the step 225.

In the step 255, the process checks whether more frames are to be acquired. If more frames are to be acquired, the process continues to the step 260, where a counter is incremented, and then continues on to the step 213. If no more frames are to be acquired, the process continues to the step 265, where it stops.

As described above, the pixels for the current frame are correlated to the pixels of the last useful frame to determine the amount of rotational or linear motion. If overall linear movement has been calculated in the step 235, data corresponding to the movement are sent to whatever downstream process needs it. For example, a program (e.g., an application program, a device driver, or an operating system) can use the corresponding data to linearly position a pointer on a display screen. If any overall rotational movement has been calculated in the step 240, data corresponding to the movement are sent to whatever downstream process needs it. For example, a program can use the corresponding data to rotate an image on the display screen. Once it is determined that movement has occurred, the last useful frame is replaced by the current frame and the algorithm continues by acquiring new image data from the sensor.

In a preferred embodiment, the system iterates in real time. Alternatively, the system stores all the frames in a memory buffer and calculates movement after multiple frames are acquired from the sensor. Preferably, the iteration halts when either the application or operating system tells it to stop. When the system is used as a pointing device for an operating system, the process can continue indefinitely.

The algorithm starts whenever there is a need for rotational feedback, such as power up of a device or start of a game. The algorithm terminates when rotational information is no longer needed.

In a preferred embodiment, the system executes on a computing device that is connected to a swipe fingerprint sensor 310 shown in Figure 5A. The swipe sensor 310 is mounted horizontally with respect to the x and y directions so that image frames as shown in Figure 6 are captured. The x direction is along the longest side of the sensor 310 while the y direction is perpendicular to the x direction. It will be appreciated that the sensor 310 can be mounted in any orientation. For consistency, the length of the sensor along the x-axis will always denote the length of the sensor. Figure 5B shows a sensor 315 having a second orientation used in accordance with the present invention. The sensor 315 is mounted vertically with respect to the x and y directions. In the preferred embodiment, the fingerprint sensor (*e.g.*, 310 or 315) provides a single frame of data to a program upon request. As described below, a single frame can be logically divided into any number of smaller sub-frames. Sensors that can provide more than one frame per request can also be used in accordance with the present invention. It will also be appreciated that fingerprint sensors in accordance with the present invention can be mounted at orientations other than those shown in Figures 5A and 5B.

Typically, swipe sensors are capable of delivering anywhere from 250 to 3000 frames per second (the "frame rate"), depending on the sensor's capabilities, the interface used and the speed of the host personal computer.

Figure 6 shows image data 400 captured by a finger image sensor in accordance with the present invention. The image data 400 comprises N rows by M columns of picture elements, or pixels, with each pixel typically represented by a single byte (8 bits) of data. M can be any positive value (100-300 is typical, depending on resolution) and N must be at least 2 (the typical number of rows in a frame is 8-32). Preferably, N=8 and M=192. A value of a pixel is a gray level, such that the image frame resembles a finger image when displayed on a computer monitor. Typically, this value ranges from 0 to 255, with 0 representing black and indicating the presence of a fingerprint ridge, and 255 representing white and indicating the presence of a valley. Other ranges of data and other representations of such data are possible without affecting the nature of the invention.

It is possible, in alternative embodiments, to use the system with little or no modification, with other similar types of sensors, such as optical document scanners. In one embodiment, the system of the present invention executes in specialized hardware or firmware instead of in software. In another embodiment, the algorithm executes on a general-purpose CPU and other portions execute solely in hardware.

Figure 7 shows a current frame and the last useful frame for several iterations of the algorithm 210 in Figure 3. Iteration 1, shown in column 401, shows a last useful frame 405 and a current frame 410. No rows of the frame 405 correspond to the frame 410, so no recognizable movement can be identified. Iteration 2, shown in column 402, shows a last useful frame 410 and a current frame 415. It will be appreciated that the last useful frame 410 corresponds to the current frame from the previous iteration, iteration 1. Row 0 of frame 410 (indicated by an arrow) corresponds to row 1 of frame 415 (also indicated by an arrow). Linear movement in a y direction is thus detected. Iteration 3, shown in column 403, shows a last useful frame 415 and a current frame 420. Frame 415 is identical to the frame 420, so zero motion is recognized. (E.g., the finger has not been moved on the finger image sensor.) In this example, the last useful frame does not have to be updated. Iteration 4, shown in column 404, shows a last useful frame 420 and a current frame 425. It will be appreciated that the last useful frame 420 corresponds to the current frame from iteration 3. Row 1 of frame 420 (indicated by a straight arrow) corresponds to row 2 of frame 425 (also indicated by a straight arrow). This vertical shift in rows (i.e., tracking images of fingerprint patterns as they and hence a finger moves from row 1 to row 2) indicates a downward movement of a finger on a finger image sensor, referred to as movement in a positive y direction. Furthermore, column 2 of frame 420 (indicated by a squiggly arrow) corresponds to column 1 of frame 425 (also indicated by a squiggly arrow). This horizontal shift in columns (i.e., tracking images of fingerprint patterns as they and hence a finger moves from column 2 to column 1) indicates a left movement of a finger on a finger image sensor, here labeled movement in a negative x-direction. In this example, because a positive y-movement and a negative x-movement has been detected, a clockwise rotation of the fingerprint image is recognized.

It will be appreciated that signs (positive or negative) assigned to a particular direction in an x-direction and a y-direction are arbitrarily chosen for the sake of explanation. The signs can be reversed.

The algorithm 210 in Figure 4 is now described in more detail. As described above, the algorithm 200 iterates, requesting new frames of data (step 210) and correlating them to a previous image frame (step 225) stored in a buffer for just this purpose. At the i^{th} iteration of the algorithm 210, one frame of data is requested from the sensing device (e.g., a finger image sensor). Once the frame is collected, it is analyzed to determine its usefulness. If the frame is deemed useful, it is saved in local memory for later use; if it is deemed useless, it is simply disregarded.

In the preferred embodiment, the usefulness of a frame is determined by ensuring the frame contains at least some finger image information. For instance, if a frame is collected when no finger is on the device, that frame likely will contain only noise or a blank image. This is done using rules based on measuring image statistics of the frame, namely the average value and the variance. Some sensors provide information on finger presence, and that can be used in systems where it is available, either by itself or in conjunction with the above statistics.

Mathematically, if the pixel in the n^{th} row and m^{th} column is given by *frame[n,m]*, then:

$$FrameAverage = \Phi = \frac{\sum_{n=1}^{N}\sum_{m=1}^{M} frame[n,m]}{NxM}$$

$$FrameVariance = \Psi = \frac{\sum_{n=1}^{N}\sum_{m=1}^{M} (frame[n,m]-\Phi)^2}{NxM}$$

For the purposes of efficiency, the Metrics (calculated in the step 215) can also be calculated just using portions of the frame (rather than the entire frame), where the portions are arbitrary sub-images or are obtained by skipping every p^{th} pixel in the calculation.

The frame is considered noise, and thereby disregarded, if:

$$\Phi \geq Noise\_average\_threshold\_high$$

or if

$$\Phi \leq Noise\_average\_threshold\_low$$

or if

$$\Psi \leq Variance\_average\_threshold$$

In other words, if the average is above or below a certain level (e.g., a threshold), or if the variance is less than expected for a normal finger, it may indicate — depending on the sensor used — that no finger exists on the device at that moment. In the preferred embodiment, typical values of these thresholds are:

$Noise\_average\_threshold\_high = 240$

$Noise\_average\_threshold\_low = 30$

$Variance\_average\_threshold = 10$

Note that other values can be used to tune the algorithm to the desired performance, and other more complicated combinations of the above statistics can also be used to determine the usefulness of a frame. Of course, other metrics, alone or in combination with the above, can also be used.

Once the current frame has been found useful, it is next correlated to the last useful frame (stored in the step 251) to determine the finger movement, if any, that occurred. Once it is determined that finger movement has occurred, the last useful frame (step 251) is replaced by the current frame and the algorithm continues by acquiring a new frame from the sensor.

In accordance with the present invention, a new frame ("*cF*") is correlated with an older one stored in memory ("*oF*"). Correlation is well known by any person skilled in the art, but it is described in more detail here to better explain the present invention.

Standard cross-correlation SCC of row R of the last useful frame with row S of the current frame is mathematically expressed as:

$$SCC\_whole(R,S,d) = \sum_{m=d+1}^{M-d} (oF[R,m] \times cF[S,m-d]) \qquad \text{[Equation 1]}$$

where d is referred to as the "lag" or "offset." The lag is related to the horizontal movement of the data in one frame with respect to the data in another, and can also be thought of as a velocity. Typically, the lag is $-L <= d <= +L$, where L is much less than M. All of the equations in this description are written assuming $d >= 0$ to keep the equations clear. It will be appreciated, however, that negative lag values can be processed by interchanging the column indices on oF and cF as shown below:

$$SCC\_whole(R,S,d) = \sum_{m=|d|+1}^{M-|d|} (oF[R,m-|d|] \times cF[S,m])$$

where $|d|$ is the absolute value of d.

This interchange method is valid for all correlation equations in this document, not just SCC but also normalized cross-correlation NCC, discussed below.

Though it is feasible to use standard correlation, the preferred embodiment uses a slightly modified version of the correlation called Normalized Cross Correlation NCC, defined in Equation 2 below, which is better suited to image registration tasks like fingerprint image reconstruction. Unlike standard correlation, NCC is invariant to changes in image intensity, has a range that is independent of the number of pixels used in the calculation, and is more accurate because it is less dependent on local properties of the image frames being correlated.

$$NCC\_whole(R,S,d) = \frac{\sum_{m=d+1}^{M-d} \left[\frac{\{oF[R,m] \times cF[S,m-d]\}}{(M-2d)} \right] - \frac{oFsum[R,d]}{(M-2d)} \times \frac{cFsum[S,d]}{(M-2d)} \}}{\sqrt{\left[\left\{ \sum_{m=d+1}^{M-d} \frac{\{oF[R,m]\}^2}{(M-2d)} \right\} - \left\{ \frac{oFsum[R,d]}{(M-2d)} \right\}^2 \right] \times \left[\left\{ \sum_{m=d+1}^{M-d} \frac{\{cF[S,m-d]\}^2}{(M-2d)} \right\} - \left\{ \frac{cFsum[S,d]}{(M-2d)} \right\}^2 \right]}}$$

[Equation 2]

where:

$$oFsum[R,d] = \sum_{m=d+1}^{M-d} \{oF[R,m]\}$$

is the sum along the row R from column $d+1$ through column $M-d$, and

$$cFsum[S,d] = \sum_{m=1}^{M-2d} \{cF[S,m]\}$$

is the sum along row S from column *1* through column $M-2d$.

The above equations are in terms of rows of each frame, but it is more general to think of "patches" of each frame, where a patch can be some subset or superset of a row. While the patches of each frame to be correlated can be any arbitrary set of pixels, the preferred embodiment uses a patch centered in the middle of each frame. While any row or subset of a row could be used, if the patch is too small, the statistical significance of the cross-correlation value will erode.

Since the lag, or offset, of the information in the current frame to that in the last frame corresponds to an unknown amount of movement in the x-direction, *NCC_whole*(R,S,d) must typically be calculated for multiple values of *d* to find the one that corresponds to the best fit. Therefore, in one embodiment:

$$PeakNCC_{whole}(R,S,L) = MAX\{\ NCC\_whole(R,S,d)\ \}\ for\ d = -L\ to\ d = L. \qquad \text{[Equation 3]}$$

$d_{peakwhole}(R,S,L)$ = *the value of d at which the above equation is satisfied.*

In the preferred embodiment, L = 8, but L should be chosen so that it is as large as the maximum x-velocity that can occur from frame to frame. A smaller L is more computationally efficient, but will produce inaccurate results if the finger shifts more than ±L from one frame to the next. In an alternative embodiment, L can be a function of the $d_{peakwhole}$ from the previous iteration *i-1*. For example,

L(at iteration i) = $d_{peakwhole}$ *(at iteration i-1) + e, where e is typically equal to 1 or 2.*

In yet another embodiment, L can be a function of the row number in the frame (i.e. R and/or S). Also note that it is possible to use scaled versions of the NCC equations so that floating-point operations can be avoided, and that for computing purposes it is also possible to use NCC-squared to avoid an expensive square-root operation.

The $PeakNCC_{whole}$ corresponds to the correlation coefficient of the best fit, while $d_{peakwhole}$ corresponds to the amount of movement in the x direction. A method to obtain the amount of motion in the y direction is described below.

The NCC calculation in Equation 2 below can be restated, for last frame oF and current frame cF,

$$NCC\_whole(R,S,d)=\frac{\sum_{m=d+1}^{M-d}\left[\frac{\{oF[R,m]\times cF[S,m-d]\}}{(M-2d)}\right]-\frac{oFsum[R,d]}{(M-2d)}\times\frac{cFsum[S,d]}{(M-2d)}}{\sqrt{\left[\left\{\sum_{m=d+1}^{M-d}\frac{\{oF[R,m]\}^2}{(M-2d)}\right\}-\left\{\frac{oFsum[R,d]}{(M-2d)}\right\}^2\right]\times\left[\left\{\sum_{m=d+1}^{M-d}\frac{\{cF[S,m-d]\}^2}{(M-2d)}\right\}-\left\{\frac{cFsum[S,d]}{(M-2d)}\right\}^2\right]}}$$

[Equation 2]

as:

$$NCC\_whole(R,S,d)=\frac{\left[(M-2d)\sum_{m=d+1}^{M-d}[\{oF[R,m]\times cF[S,m-d]\}]\right]-oFsum[R,d]\times cFsum[S,d]}{\sqrt{\left[\left\{(M-2d)\sum_{m=d+1}^{M-d}\{oF[R,m]\}^2\right\}-\{oFsum[R,d]\}^2\right]\times\left[\left\{(M-2d)\sum_{m=d+1}^{M-d}\{cF[S,m-d]\}^2\right\}-\{cFsum[S,d]\}^2\right]}}$$

where the numerator and denominator have both been multiplied by $(M-2d)^2$ to make it simpler to compute and understand.

This can be separated into left and right halves of each row as:

$$NCC\_whole\ (R,\ S,\ d) = (A-B)\ /\ (C^{1/2} \times D)$$

[Equation 4]

where A =

$$\left[(\mathrm{M}-2\mathrm{d})\left\{\sum_{m=d+1}^{M/2}\left[\{oF[R,m]\times cF[S,m-d]\}\right]+\sum_{m=M/2+1}^{M-d}\left[\{oF[R,m]\times cF[S,m-d]\}\right]\right\}\right]$$

where B =

$$\left[oFsum_{left}[R,d]+oFsum_{right}[R,d]\right]\times\left[cFsum_{left}[S,d]+cFsum_{right}[S,d]\right]$$

where C =

$$\left[(M-2d)\left\{\sum_{m=d+1}^{M/2}\{oF[R,m]\}^2+\sum_{m=M/2+1}^{M-d}\{oF[R,m]\}^2\right\}-\left\{oFsum_{left}[R,d]+oFsum_{right}[R,d]\right\}^2\right]$$

and, where D =

$$(M-2d)\left\{\sum_{m=d+1}^{M/2}\{cF[S,m-d]\}^2+\sum_{m=M/2+1}^{M-d}\{cF[S,m-d]\}^2\right\}-\left\{cFsum_{left}[S,d]+cFsum_{right}[S,d]\right\}^2$$

where:

$$cFsum_{right}=\sum_{m=M/2-d+1}^{M-2d}\{cF[S,m]\}$$

is the sum along row R from column *M/2+1* through column *M-d*,

$$cFsum_{left}=\sum_{m=1}^{M/2-d}\{cF[S,m]\}$$

is the sum along row R from column *d +1* through column *M/2*,

$$oFsum_{right} = \sum_{m=M/2+1}^{M-d} \{oF[R,m]\}$$

is the sum along row S from column *M/2-d+1* through column *M-2d*, and

$$oFsum_{left} = \sum_{m=d+1}^{M/2} \{oF[R,m]\}$$

is the sum along row S from column *1* through column *M/2-d.*

Furthermore, the *NCC* for the left and right halves of each row can be determined using:

$$NCC\_left(R,S,d) = \frac{\left[(M-2d)\left\{\sum_{m=d+1}^{M/2}\left[\{oF[R,m]\times cF[S,m-d]\}\right]\right\}\right] - \left[oFsum_{left}[R,d]\right]\times\left[cFsum_{left}[S,d]\right]}{\sqrt{\left[(M-2d)\left\{\sum_{m=d+1}^{M/2}\{oF[R,m]\}^2\right\} - \left\{oFsum_{left}[R,d]\right\}^2\right]\times\left[(M-2d)\left\{\sum_{m=d+1}^{M/2}\{cF[S,m-d]\}^2\right\} - \left\{cFsum_{left}[S,d]\right\}^2\right]}}$$

[Equation 5a]

$$NCC\_right(R,S,d) = \frac{\left[(M-2d)\left\{\sum_{m=M/2+1}^{M-d}\left[\{oF[R,m]\times cF[S,m-d]\}\right]\right\}\right] - \left[oFsum_{right}[R,d]\right]\times\left[cFsum_{right}[S,d]\right]}{\sqrt{\left[(M-2d)\left\{\sum_{m=M/2+1}^{M-d}\{oF[R,m]\}^2\right\} - \left\{oFsum_{right}[R,d]\right\}^2\right]\times\left[(M-2d)\left\{\sum_{m=M/2+1}^{M-d}\{cF[S,m-d]\}^2\right\} - \left\{cFsum_{right}[S,d]\right\}^2\right]}}$$

[Equation 5b]

In addition, the *PeakNCC* for the left and right sides is given by:

$PeakNCC_{left}(R,S,L) = MAX\{ NCC\_left(R,S,d) \}$ for d = -L to d = L. [Equation 6a]

$d_{peakleft}(R,S,L)$ = *the value of d at which the above equation is satisfied.*

$PeakNCC_{right}(R,S,L) = MAX\{ NCC\_right(R,S,d) \}$ *for d = -L to d = L.* [Equation 6b]

$d_{peakright}(R,S,L)$ = *the value of d at which the above equation is satisfied.*

These equations allow the left and right sides of the sensor array to be treated separately, and efficiently determine the rotational movement as described below.

Once all the NCC terms for left and right sides are calculated for the left and right sides in Equations 5a and 5b, only a few addition and division operations are required to calculate *NCC_whole* for the entire sensing array using Equation 4. Then, overall linear motion can be calculated using Eq. 3 as before, while rotational movement is calculated using the linear motion for the left and right sides of the sensing array.

Using *PeakNCC(R,S,L)* defined above in Equation 3 or Equations 6a and 6b, the calculation of exact x and y motion is straightforward. The last useful frame at iteration *i* has rows numbered 1 through N, as shown in Figure 4, where N is the number of rows supplied by the sensor. Similarly, the current frame to be processed has rows 1 through N.

For a given row R in the last frame, *PeakNCC* and d_{peak} are calculated as in Equations 3, 6a and 6b with respect to rows 1 through N of the current frame, and take the d_{peak} that corresponds to the maximum *PeakNCC*. Preferably, this is done for two values of R: R=1 and R=N. In this way, both upward and downward motion in the y direction can be determined while maximizing the speed at which a user can move his finger. It is also possible to choose only one R, at R=N/2 (or very near the middle row). However that is sub-optimal. It is also possible to choose values other than 1 or N, such as R=2 and R=N-1, which may be advantageous for accuracy reasons since they are not on the edge of the sensor array.

Table 1 shows the pseudo-code for performing a single frame iteration for a given value of R. Although the calculations are carried out separately for the left side, right side, and whole row, only the generic case is described by the pseudo-code in Table 1.

```
Step 0: START
Step 1: Set R=1 ///// start with R=1 and then do R=N
Step 2: Set L=8 ///// lag value fixed at 8 columns
Step 3: Set bestS = 0 ///// initialize
Step 4: set MaxPeakNCC = 0 ///// initialize
Step 5: set d_peakMax = 0 /////initialize

Step 6: Loop from currentRow = 1 through N //loop over N rows of current
frame
{
      Step 6.1: set tmp = PeakNCC(R,currentRow,L)
      Step 6.2: if tmp greater than MaxPeakNCC then
                  {
                  Step 6.3: set MaxPeakNCC = tmp //keep largest
                  Step 6.4: set d_peakMax = d_peak(R,currentRow,L)
                  Step 6.5: set bestS = currentRow
                  Step 6.6: set bestR = R
                  }
}
Step 7: if R equals 1 then
      {
            Step 7.1: set R = N
            Step 7.2: Go to Step 2
      }
Step 8: STOP
```

Table 1

The pseudo-code in Table 1 can be summarized as:

$MaxPeakNCC = NCC\_whole(bestR, bestS, d_{peak}(bestR,bestS,L))$ [Equation 6c]

$d_{peak}Max = dpeak(bestR,bests,L)$ [Equation 6d]

Thus, after the above calculations, the following information is obtained:

1. *MaxPeakNCC*, the value of the correlation of the best pair of matching rows in the current frame and the last frame
2. *bestR*, the row of the last frame that results in *MaxPeakNCC*
3. *bestS*, the row of the current frame that results in *MaxPeakNCC*
4. d_{peak}Max, the correlation lag where *MaxPeakNCC* is reached

bestR and *bestS* are the pair of rows that yield the highest correlation value (*MaxPeakNCC*).

Typically, *MaxPeakNCC* will be close to 1.0 (the closer to 1.0, the stronger the correlation), but if the finger being analyzed is moved too quickly, it is possible that the current frame does not have any rows in common with the last frame (i.e. a non-overlapping case). Therefore, *MaxPeakNCC* must be checked to ensure that it is large enough.

Using the above information, the following calculations are performed:

if *MaxPeakNCC* > *corr_threshold*:

1. $\Delta x(i) = d_{peak}Max$, which is the x-velocity at iteration *i*
2. $\Delta y(i) =$ *bestS-bestR*, which is the y-velocity at iteration *i*

otherwise

3. $\Delta x(i) = 0$
4. $\Delta y(i) = 0$

where *corr_threshold* is used to make sure the correlation is high enough to indicate an overlap at all. Preferably, *corr_threshold* = 0.75, but other values can be used to tune the performance of the algorithm. If the correlation is below a threshold, it is impossible to determine actual values for the x- and y-velocities, so the algorithm simply outputs no motion vector. However, in accordance with alternative embodiments other values can be output as the default, such as maximum movements of N rows and M columns, or combinations thereof.

In any case, after x and y motion have been calculated for the left side, right side, and whole rows for the current iteration *i*--denoted by $\Delta x_{left}(i)$ and $\Delta y_{left}(i)$; $\Delta x_{right}(i)$ and $\Delta y_{right}(i)$; $\Delta x_{whole}(i)$ and $\Delta y_{whole}(i)$, respectively–are calculated, the rotational movement Δtheta(i) can now be determined. The $\Delta x_{whole}(i)$ and $\Delta y_{whole}(i)$ are made available to the host requiring the rotational information, and represent the overall linear x- and y-motion.

Table 2 shows the pseudo code for determining rotational movement. The pseudo code continues iterating until told to stop by the application program or operating system using the rotational data.

```
Step 0: START
Step 1: set cumulativeDelY_left(0) = 0
Step 2: set cumulativeDelY_right(0) = 0
Step 3: Loop from iteration i = 1 through infinity
      {
      Step 3.1: Acquire a frame and calculate Δy_left(i) and Δy_right(i)
      Step 3.2: Set cumulativeDelY_left(i)=cumulativeDelY_left(i-1)+Δy_left(i)
      Step 3.3: Set cumulativeDelY_right(i)=cumulativeDelY_right(i-1)+Δy_right(i)
      Step 3.4: If absolute value (cumulativeDelY_left(i) -
                  cumulativeDelY_right(i)) >= THRESH,
                  then
                  {
                        Step 3.5: Set Δtheta(i)= [cumulativeDelY_left(i) -
                                          cumulativeDelY_right(i)]
                        Step 3.6: Set cumulativeDelY_left(i) = 0
                        Step 3.7: Set cumulativeDelY_right(i) = 0

                  }
                  Else:
                        Step 3.8: Set Δtheta(i)= 0
      }

where THRESH can be any value >= 1. Preferably, THRESH=1.
```

Table 2

It will be appreciated that there are alternative ways to compute the rotational delta, including arbitrary functions of the $\Delta y_{left}(i)$ and $\Delta y_{right}(i)$. For alternative mountings of the sensor, where the x and y directions are transposed, $\Delta x_{left}(i)$ and $\Delta x_{right}(i)$ are used in the pseudo code in Table 2. On full size placement sensors, more accuracy can be achieved using both $\Delta x_{left}(i)$, $\Delta x_{right}(i)$ and $\Delta y_{left}(i)$, $\Delta y_{right}(i)$. This is achieved in one embodiment by calculating a Δtheta(i) using the pseudo code in Table 2 using Δx values and again using Δy values. The resulting two estimated values can be averaged together or otherwise combined to form the final Δtheta(i). The Δtheta(i) are made available to the host application and/or operating system.

In other embodiments a standard correlation is used instead of normalized cross correlation. Standard cross correlation given in Equation 1 could be used instead of Normalized Cross Correlation. It is straightforward to split Equation 1 into terms from the left and right sides of each row.

$$SCC\_whole(R,S,d) = \sum_{m=d+1}^{M/2}(oF[R,m]\times cF[S,m-d]) + \sum_{m=M/2+1}^{M-d}(oF[R,m]\times cF[S,m-d])$$

[Equation 7]

In this case, which is much simpler than the NCC case in the preferred embodiment, the SCC value for the entire row is simply the sum of the correlation values of each half.

It will also be appreciated that the maximum standard cross-correlation between a row S of a frame and a row R of the last useful frame can be given by other expressions. For example, weighted or voting schemes can be used. In one embodiment,

$PeakSCC_{whole}(R,S,L)$ = Weighted_MAX{SCC_Whole(R,S,d)} for d=-L to d=L

[Equation 8]

where Weighted_MAX is a function that assigns a plurality of predetermined weights to its elements before generating a value, $d_{peakwhole}(R,S,L)$ is the value of d at which Equation 8 is satisfied, and L is approximately equal to the maximum horizontal speed from the last useful frame to the current frame.

While the preferred embodiment splits each row into a left and right side of equal length, alternative embodiments use any arbitrary division of each row, including more than 2 equal parts instead of 2, and also using divisions that are of differing length. It is also not necessary to have each division touch the next. Figures 8-12 show some of the possibilities, each with advantages over the other.

For example, Figure 8 shows a fingerprint sensor having a left half (sub-frame) 605 contiguous with a right half 610. Figure 9 shows a fingerprint sensor having a first section (sub-frame) 615, a second section 620, a third section 625, and a fourth section 630. Section 615 is contiguous with section 620, section 620 is contiguous with 625, and section 625 is contiguous with section 630. Figure 10 shows a finger image sensor having two sections 631 and 632 that are not contiguous. Figure 11 shows a finger image sensor having four sections 635, 640, 645, and 650, none of which are contiguous. And Figure 12 shows a finger image sensor having a section 655 contiguous with a section 660, and a section 665 contiguous with a section 670. It will be appreciated that other configurations are also possible, each preferably having more than one single division and using linear correlation to determine rotational movement. The number and configurations of the sections can be chosen based on production cost, surface area of the finger image sensor, computational algorithms, processor configuration, speed required, and other criteria.

In other embodiments, it is desirable to modify the raw values $\Delta x_{whole}(i)$, $\Delta y_{whole}(i)$, and Δtheta(i) before sending it to the host. These types of modifications involve three different mathematical transformations, generically called filtering, where the transformed output is noted by the ' notation:

1. Scaling: applying a linear or non-linear scaling factor to multiply the original movement values to obtain scaled versions more appropriate to the displayed coordinate system of the host. An example is to multiply all values by a factor of 2. Another example is to multiply x movement by a factor of 2 and y movement by a 0.5.
2. Smoothing: applying a smoothing (low-pass) filter to successive movement values in order to make the values less jagged over time. Examples are:
   Linear Average = $\Delta x'_{whole}(i)=[\Delta x_{whole}(i)+\Delta x_{whole}(i-1)+\Delta x_{whole}(i-2)]/3$
   Exponential average = $\Delta theta'(i)=[\Delta theta'(i-1)+\Delta theta(i)]/2$
3. Clipping: limiting values of the movement to arbitrary values. An example is:
   If $\Delta y(i)>T$, then $\Delta y'(i)=T$. Otherwise $\Delta y'(i)=\Delta y(i)$.

These operations can be combined in many ways in accordance with the present invention. Here, "filtering" is used to describe the function of processing an input in a well-defined way to produce an output.

Those skilled in the art will recognize that correlation is computationally intensive. Accordingly, in one embodiment, the calculation of Δtheta(i) and/or the $\Delta x(i)$, $\Delta y(i)$ for the left, right, and whole array are performed on a separate processor or dedicated hardware. In this embodiment, the hardware can be integrated into the silicon fingerprint sensor itself. The hardware performing the correlation must have access to the current frame and the last useful frame, both of which can be stored in memory on the device. If, for example, this is integrated into the finger image sensor, such a device would obviously have access to the current frame (since the device itself created it), and it could save the last useful frame in volatile memory registers. In such a case the device would also need to determine whether a frame is useful or not, using the method described here in the preferred embodiment. In such an embodiment the host computing device is not necessary. Obviously, such a hardware implementation could also be used to reconstruct fingerprint images since doing so only requires the $\Delta x(i)$ and $\Delta y(i)$ for the whole array.

It will be readily apparent to one skilled in the art that various modifications may be made to the embodiments without departing from the spirit and scope of the invention as defined by the appended claims.

CLAIMS:

I claim:

1. A method of obtaining rotation information, the method comprising:
   a. capturing a plurality of patterned images from a plurality of locations;
   b. correlating the plurality of patterned images to generate sets of linear differences; and
   c. using the sets of linear differences to generate the rotation information.

2. The method of claim 1, wherein the plurality of locations comprise a first part of a sensor and a second part of the sensor.

3. The method of claim 2, wherein a first of the plurality of patterned images is captured in the first part of the sensor and a second of the plurality of patterned images is captured in the second part of the sensor.

4. The method of claim 3, wherein the sensor is a biometric image sensor.

5. The method of claim 4, wherein the biometric image sensor is a finger image sensor.

6. The method of claim 5, wherein the first of the plurality of patterned images and the second of the plurality of patterned images together correspond to a fingerprint image in a first position on the sensor.

7. The method of claim 6, wherein a third of the plurality of patterned images is captured in the first part of the sensor and a fourth of the plurality of patterned images is captured in the second part of the sensor.

8. The method of claim 7, wherein the third of the plurality of patterned images and the fourth of the plurality of patterned images together correspond to the fingerprint image in a second position on the sensor.

9. The method of claim 8, wherein the rotation information corresponds to an angular difference between the first position and the second position.

10. The method of claim 9, wherein correlating the plurality of patterned images comprises:
    a. correlating the first patterned image with the third patterned image to generate a first set of linear differences from the sets of linear differences;
    b. correlating the second patterned image with the fourth patterned image to generate a second set of linear differences from the sets of linear differences; and
    c. correlating a first combination of the first patterned image and the second patterned image with a second combination of the third patterned image and the fourth patterned image to generate a third set of linear differences from the sets of linear differences.

11. The method of claim 10, wherein correlating the first patterned image with the third patterned image, correlating the second patterned image with the fourth patterned image, and correlating the first combination with the second combination all comprise performing a cross correlation.

12. The method of claim 11, wherein the cross correlation is a normalized cross correlation.

13. The method of claim 11, wherein the cross correlation is a standardized cross correlation.

14. The method of claim 2, wherein the first part of the sensor and the second part of the sensor are contiguous.

15. The method of claim 2, wherein the first part of the sensor and the second part of the sensor are not contiguous.

16. The method of claim 11, wherein the first part of the sensor comprises a first sub-frame of pixels and the second part of the sensor comprises a second sub-frame of pixels.

17. The method of claim 16, wherein capturing the first patterned image comprises

storing in the first sub-frame first data corresponding to the first patterned image, capturing the second patterned image comprises storing in the second sub-frame second data corresponding to the second patterned image, capturing the third patterned image comprises storing in the first sub-frame third data corresponding to the third patterned image, and capturing the fourth patterned image comprises storing in the second sub-frame fourth data corresponding to the fourth patterned image.

18. The method of claim 17, wherein correlating the first patterned image with the third patterned image comprises correlating the first data with the third data to generate first and second linear differences from the first set of linear differences.

19. The method of claim 18, wherein correlating the second patterned image with the fourth patterned image comprises correlating the second data with the fourth data to generate first and second linear differences from the second set of linear differences.

20. The method of claim 19, wherein correlating the first combination with the second combination comprises correlating a combination of the first data and the second data with a combination of the third data and the fourth data to generate first and second linear differences from the third set of linear differences.

21. The method of claim 20, wherein correlating comprises determining a lag to correlate elements of one of the first and the second sub-frames, the lag and a difference between the elements corresponding to first and second linear differences from one of the sets of linear differences.

22. The method of claim 21, wherein each element corresponds to a row of one of the first and the second sub-frames.

23. The method of claim 21, wherein each element corresponds to a column of one of the first and the second sub-frames.

24. The method of claim 11, further comprising filtering the first set of linear differences, the second set of linear differences, the third set of linear differences, and the rotation information.

25. The method of claim 24, wherein filtering comprises multiplying by a scaling factor.

26. The method of claim 25, wherein filtering further comprises performing a smoothing function.

27. The method of claim 26, wherein filtering further comprises performing a clipping function.

28. The method of claim 10, wherein the finger image sensor is a finger placement sensor.

29. The method of claim 10, wherein the finger image sensor is a finger swipe sensor.

30. The method of claim 9, further comprising using the rotation information on a host platform having a display screen, the rotation information used to rotate an object on the display screen, thereby emulating a computer input device.

31. The method of claim 30, wherein the computer input device is selected from the group consisting of a steering wheel, a joystick, and a navigation bar.

32. The method of claim 30, wherein emulating a computer input device comprises moving the object on the display screen at a rate related to the angular difference.

33. A system for obtaining rotation information, the system comprising:
   a. means for capturing a plurality of patterned images from a plurality of locations; and
   b. means for correlating the plurality of patterned images to generate sets of linear differences and for using the sets of linear differences to generate the rotation information.

34. The system of claim 33, wherein the means for capturing comprises a sensor having a first part and a second part.

35. The system of claim 34, wherein the sensor is a biometric image sensor.

36. The system of claim 35, wherein the biometric image sensor is a finger image sensor.

37. The system of claim 36, wherein the first part of the sensor is configured to capture a first of the plurality of patterned images and the second part of the sensor is configured to capture a second of the plurality of patterned images.

38. The system of claim 37 wherein the first of the plurality of patterned images and the second of the plurality of patterned images together correspond to a fingerprint image in a first position.

39. The system of claim 38, wherein the first part of the sensor is further configured to capture a third of the plurality of patterned images and the second part of the sensor is further configured to capture a fourth of the plurality of patterned images.

40. The system of claim 39, wherein the third of the plurality of patterned images and the fourth of the plurality of patterned images together correspond to the fingerprint image in a second position.

41. The system of claim 40, wherein the rotation information corresponds to an angular difference between the first position and the second position.

42. The system of claim 40, wherein correlating the plurality of patterned images comprises:

a. correlating the first patterned image with the third patterned image to generate a first set of linear differences from the sets of linear differences;

b. correlating the second patterned image with the fourth patterned image to generate a second set of linear differences from the sets of linear differences; and

c. correlating a first combination of the first patterned image and the second patterned image with a second combination of the third patterned image and the fourth patterned image to generate a third set of linear differences from the sets of linear differences.

43. The system of claim 42, wherein correlating the first patterned image with the third patterned image, correlating the second patterned image with the fourth patterned image, and correlating the first combination with the second combination all comprise performing a cross correlation.

44. The system of claim 43, wherein the cross correlation is a normalized cross correlation.

45. The system of claim 43, wherein the cross correlation is a standardized cross correlation.

46. The system of claim 34, wherein the first part of the sensor and the second part of the sensor are contiguous.

47. The system of claim 34, wherein the first part of the sensor and the second part of the sensor are not contiguous.

48. The system of claim 43, wherein the first part of the sensor comprises a first sub-frame of pixels and the second part of the sensor comprises a second sub-frame of pixels.

49. The system of claim 48, wherein capturing a patterned image comprises storing in the first sub-frame first data corresponding to the first patterned image, capturing the second patterned image comprises storing in the second sub-frame second data corresponding to the second patterned image, capturing the third patterned image comprises storing in the first sub-frame third data corresponding to the third patterned image, and capturing the fourth patterned image comprises storing in the fourth sub-frame data corresponding to the fourth patterned image.

50. The system of claim 49, wherein correlating the first patterned image with the third patterned image comprises correlating the first data with the third data to generate first and second linear differences from the first set of linear differences.

51. The system of claim 50, wherein correlating the second patterned image with the fourth patterned image comprises correlating the second data with the third data to generate first and second linear differences from the second set of linear differences.

52. The system of claim 51, wherein correlating the first combination with the second combination comprises correlating a combination of the first data and the second data with a combination of the third data and the fourth data to generate first and second linear differences from the third set of linear differences.

53. The system of claim 52, wherein correlating comprises determining a lag to correlate elements of one of the first and second sub-frames, the lag and a difference between the elements corresponding to the first and second linear differences from one of the sets of linear differences.

54. The system of claim 53, wherein each element corresponds to a row of one of the first and second sub-frames.

55. The system of claim 53, wherein each element corresponds to a column of one of the first and second sub-frames.

56. The system of claim 33, wherein the means for correlating is further configured for filtering the first set of linear differences, the second set of linear differences, the third set of linear differences, and the rotation information.

57. The system of claim 56, wherein filtering comprises multiplying by a scaling factor.

58. The system of claim 57, wherein filtering further comprises performing a smoothing function.

59. The system of claim 58, wherein filtering further comprises performing a clipping function.

60. The system of claim 33, wherein the means for capturing comprises a finger placement sensor.

61. The system of claim 33, wherein the means for capturing comprises a finger swipe sensor.

62. The system of claim 41, further comprising a host device coupled to the means for correlating, the host device having a display screen and configured to receive the rotation information and use the rotation information to control an object on the display screen, thereby emulating a computer input device.

63. The system of claim 62, wherein the computer input device is selected from the group consisting of a steering wheel, a joystick, and a navigation bar.

64. The system of claim 63, wherein the host device is a portable device.

65. The system of claim 64, wherein the portable device is a device selected from the group consisting of a personal computer, a portable telephone, a portable electronic game device, and a digital camera.

66. The system of claim 62, wherein emulating a computer input device comprises moving the object on the display screen at a rate proportional to the angular difference.

67. The system of claim 41, further comprising a host device integral with at least one of the means for capturing and the means for correlating.

68. A method of emulating a rotational device using a pattern, the method comprising:
a. capturing a first image of the pattern at a first orientation;
b. capturing a second image of the pattern at a second orientation;
c. correlating the first image with the second image to calculate linear differences between the first orientation and the second orientation;
d. translating the linear difference into rotational data; and
f. using the rotational data to emulate the movement of a rotational device.

69. The method of claim 68, wherein the image comprises a fingerprint image.

70. The method of claim 69, wherein the rotational device is further configured to emulate a linear positioning device.

71. A system for emulating a positional device, the system comprising:
a. a sensor for capturing an image of a pattern; and
b. a processor coupled to the sensor, the processor configured to calculate linear differences between a first position of the image of the pattern and a second position of the image of the pattern and to translate the linear differences into rotational data corresponding to a rotation of the image of the pattern.

72. A method of sensing rotation of an object on an image sensor comprising:
a. sensing a first image of the object;
b. sensing a second image of the object; and
c. comparing the first image with the second image to determine whether there is linear motion in each of at least two portions of an area containing the first image and the second image to determine whether the object remained stationary, moved in a linear manner, or rotated.

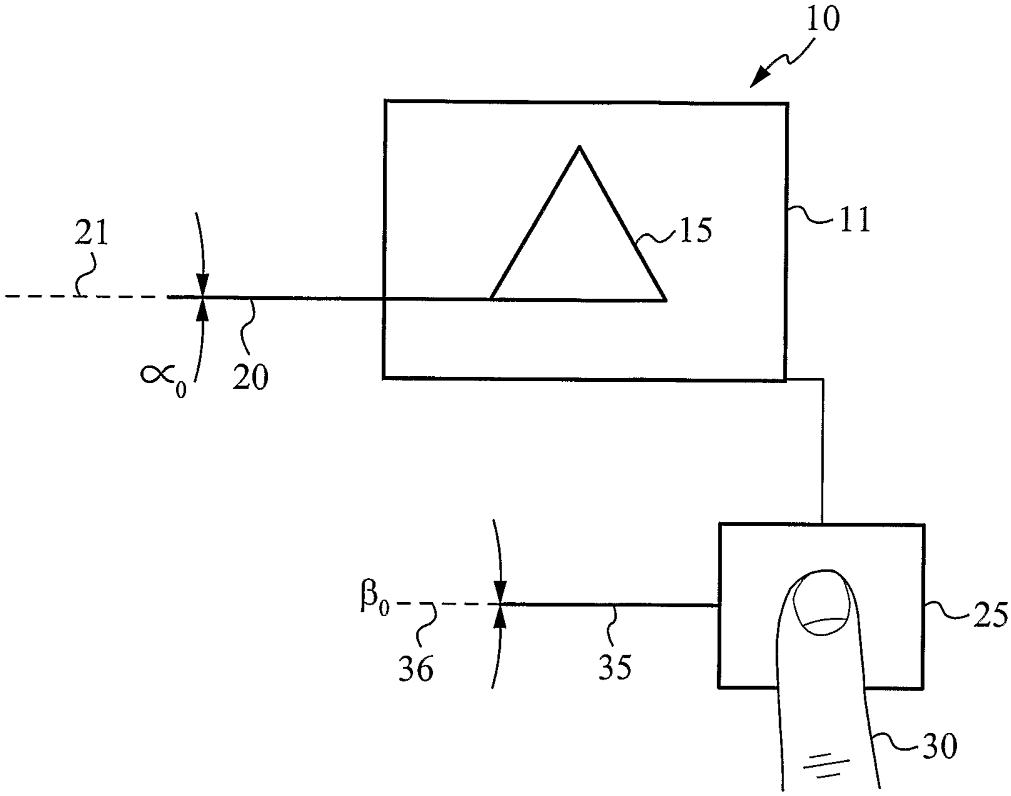

10
11
15
21
20
α_0
β_0
36
35
25
30

Fig. 1A

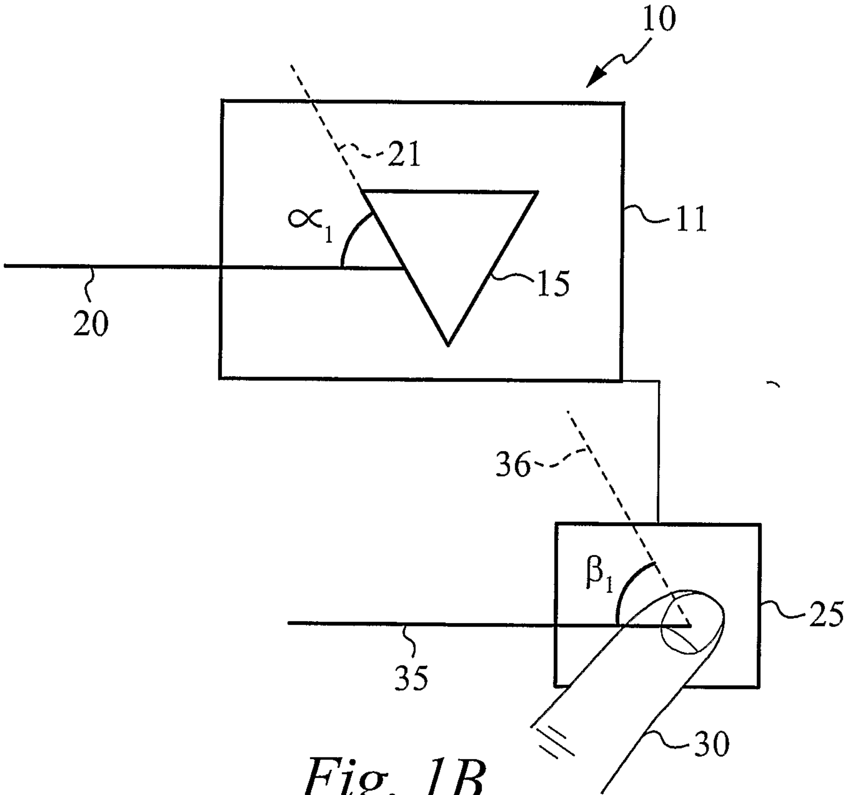

10
21
11
α_1
15
20
36
β_1
25
35
30

Fig. 1B

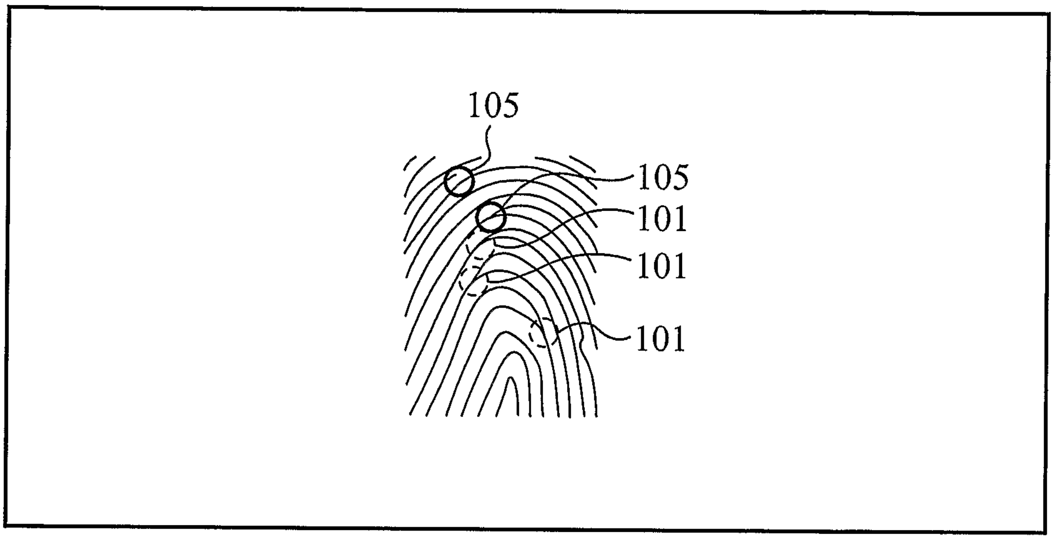


Fig. 2

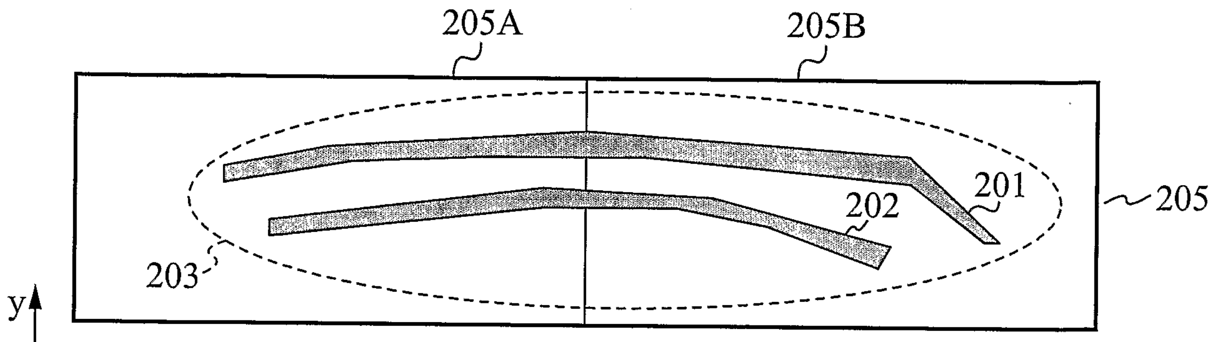


Fig. 3A

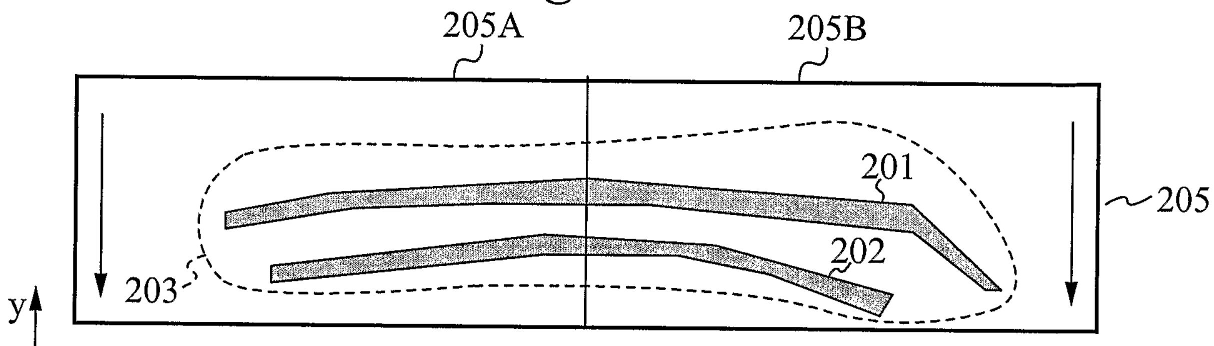


Fig. 3B

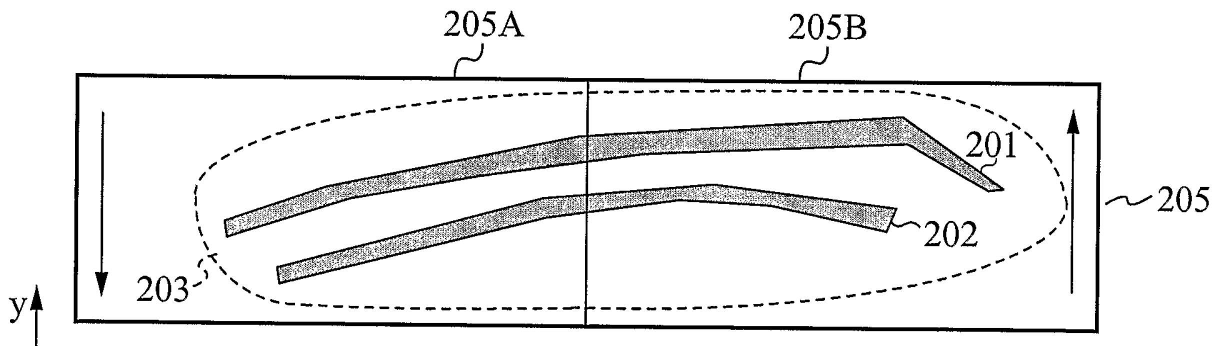


Fig. 3C

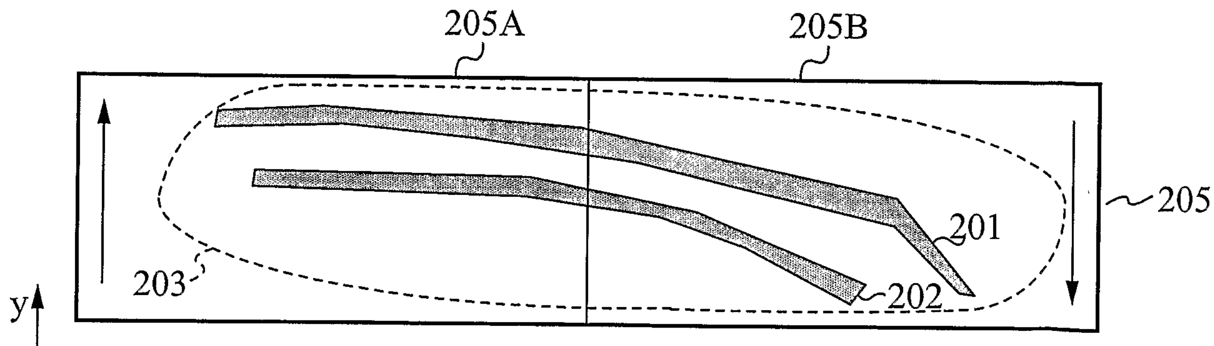


Fig. 3D

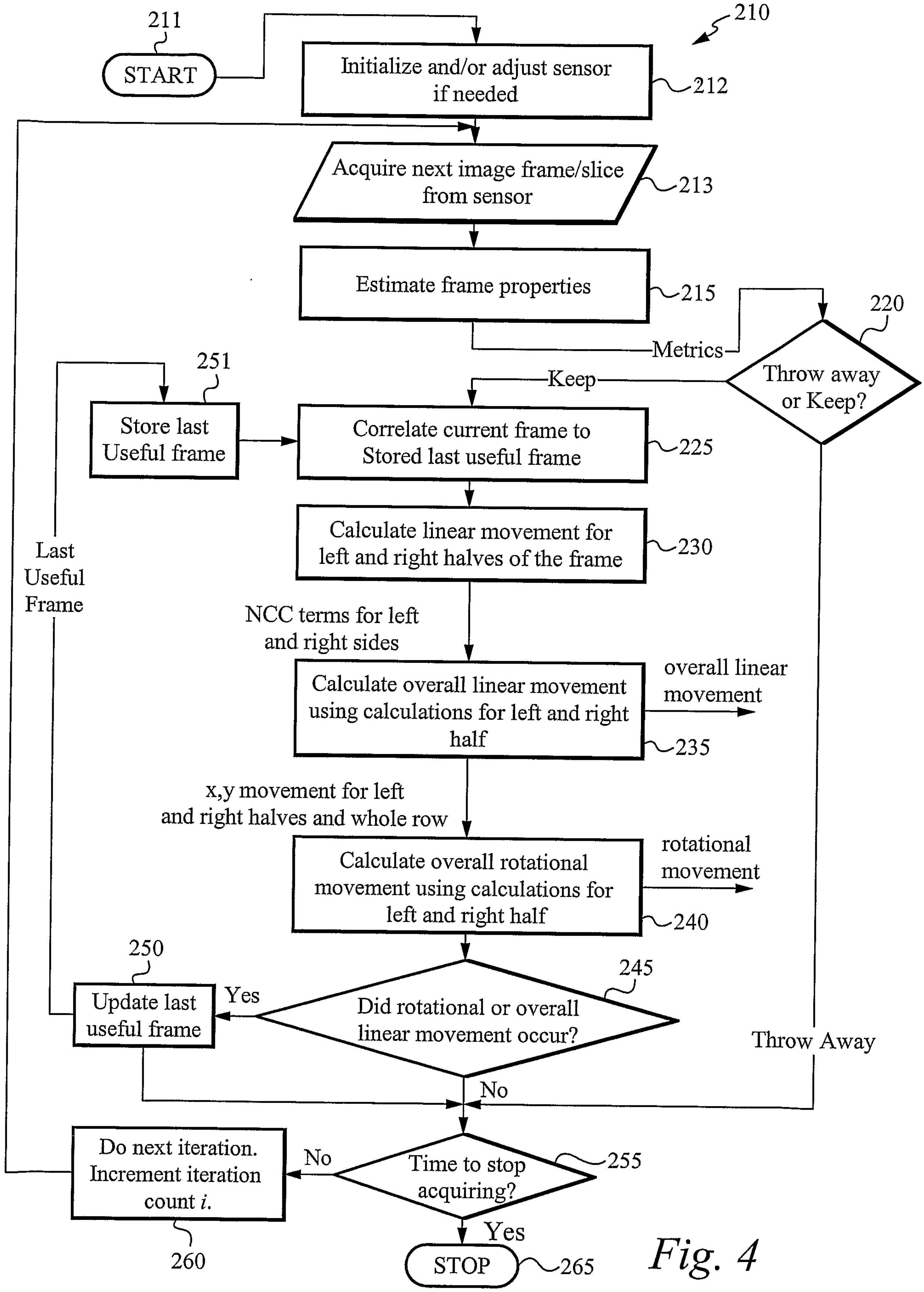

210
211
START
Initialize and/or adjust sensor if needed
212
Acquire next image frame/slice from sensor
213
Estimate frame properties
215
Metrics
220
Throw away or Keep?
Keep
251
Store last Useful frame
Correlate current frame to Stored last useful frame
225
Calculate linear movement for left and right halves of the frame
230
Last Useful Frame
NCC terms for left and right sides
Calculate overall linear movement using calculations for left and right half
overall linear movement
235
x,y movement for left and right halves and whole row
Calculate overall rotational movement using calculations for left and right half
rotational movement
240
250
245
Update last useful frame
Yes
Did rotational or overall linear movement occur?
Throw Away
No
Do next iteration. Increment iteration count i.
No
Time to stop acquiring?
255
Yes
260
STOP
265

Fig. 4

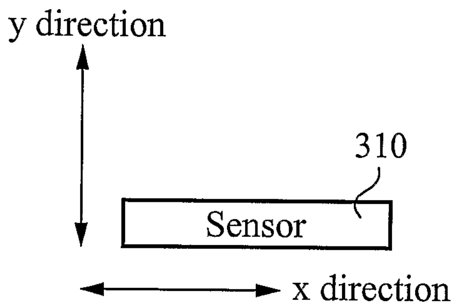

y direction
310
Sensor
x direction

Fig. 5A

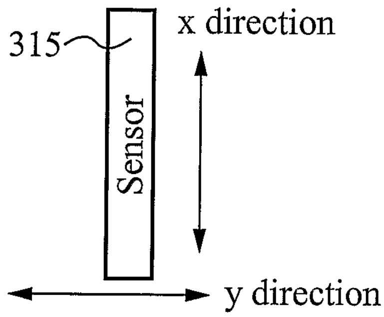

315
Sensor
x direction
y direction

Fig. 5B

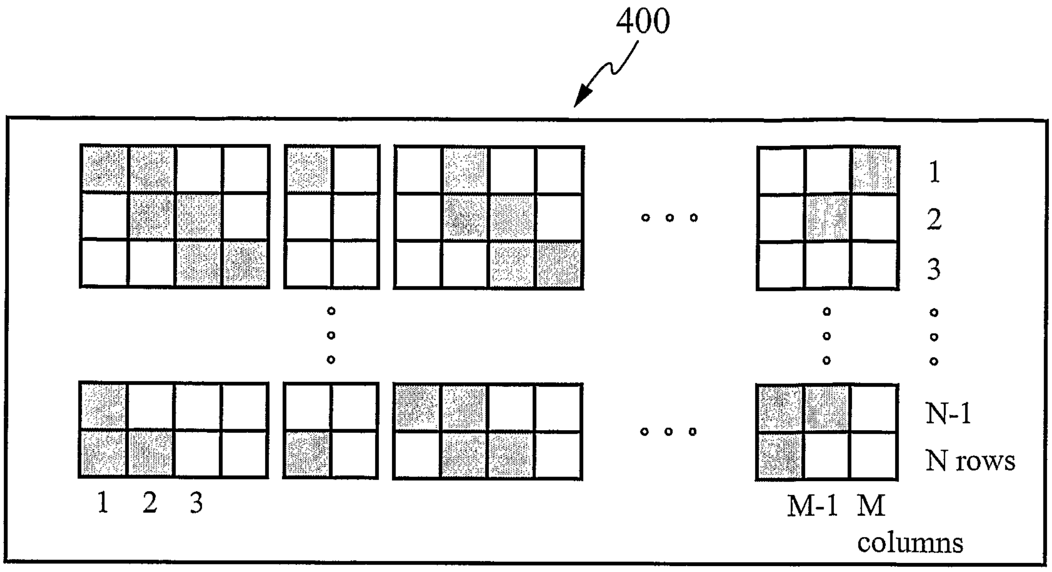

400
1
2
3
N-1
N rows
1 2 3
M-1 M
columns

Fig. 6

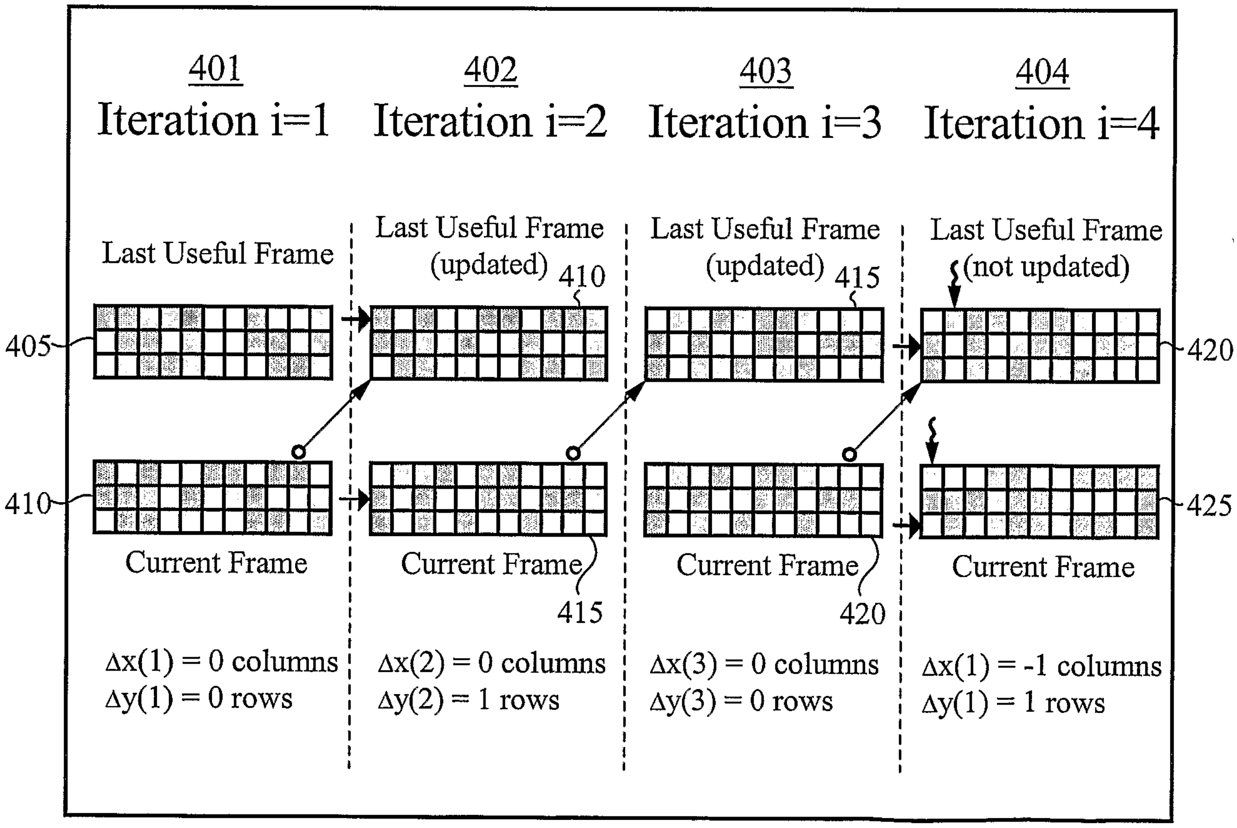

401
Iteration i=1
402
Iteration i=2
403
Iteration i=3
404
Iteration i=4
Last Useful Frame
Last Useful Frame (updated)
410
Last Useful Frame (updated)
415
Last Useful Frame (not updated)
405
420
410
425
Current Frame
Current Frame
415
Current Frame
420
Current Frame
Δx(1) = 0 columns
Δy(1) = 0 rows
Δx(2) = 0 columns
Δy(2) = 1 rows
Δx(3) = 0 columns
Δy(3) = 0 rows
Δx(1) = -1 columns
Δy(1) = 1 rows

Fig. 7

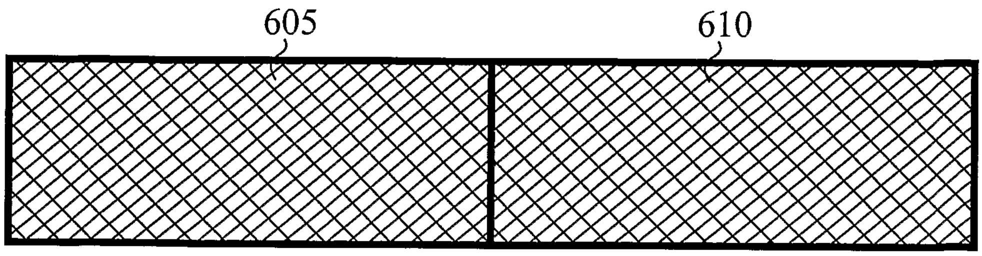


Fig. 8

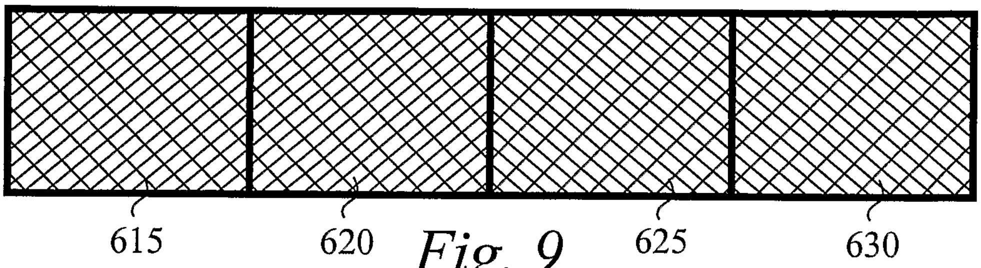


Fig. 9

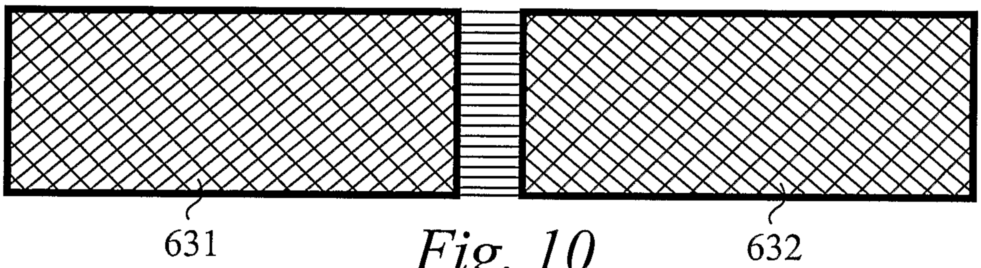


Fig. 10

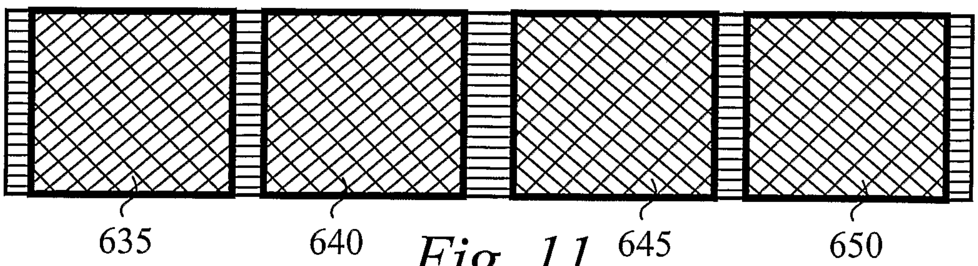


Fig. 11

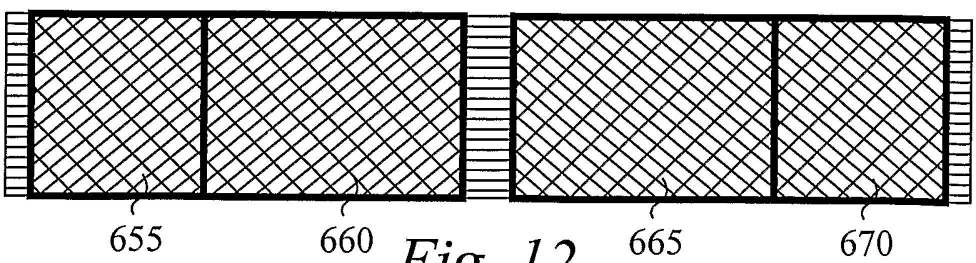


Fig. 12