# EXHIBIT 4.08

leave the surface within a release setup time after holding finger touched down;

periodically issuing additional keypress signals every repeat time interval subsequent to the second keypress signal as long as the holding finger continues touching the desired key region; and

ceasing repetitive issuance of the additional keypress signals when the holding finger lifts off the surface.

106. The method of claim 105, wherein touchdown, resting or liftoff of hand contacts identified as palms on either hand does not affect the typematic state.

107. The method of claim 105, wherein the cycle of keypress signal generation continues irrespective of whether other fingers touch down and rest on the surface subsequent to issuing the first keypress signal.

108. The method of claim 105, wherein the repeat time interval is continuously adjusted to be inversely proportional to current measurements of holding finger proximity or pressure.

109. A method for choosing what kinds of input signals will be generated and sent to an electronic or electro-mechanical 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.

110. The method of claim 109, wherein all sets or subsets which contain the same number of fingertips choose the same kinds of input signals, such that sets or subsets are uniquely distinguished by the number of fingertips they contain and whether they contain the thumb.

111. The method of claim 109, wherein all sets or subsets which contain the same combination of thumb, index, middle, ring, and pinky fingers choose the same kinds of input signals.

112. The method of claim 109, wherein the method is applied to contacts identified as left hand parts independently from contacts identified as right hand parts.

113. The method of claim 109, wherein no input signals are generated after a set or subset is formed without further distinctive finger motions, to support resting of fingers on the surface.

114. The method of claim 109, wherein one of the distinctive finger motions is synchronized liftoff of a finger set or subset quickly following synchronized touchdown, and wherein each such motion generates a tap signal of the selected kind.

115. The method of claim 109, wherein one of the distinctive finger motions is sliding of the finger set or subset across the surface, and wherein such motion continuously generates slide signals of the selected kind which include measurements of the sliding motion.

116. The method of claim 109, wherein asynchronous touchdown quickly followed by liftoff of a finger forms a new subset of one finger and generates a tap event dependent on the location on the surface of the touchdown.

117. The method of claim 109, wherein generation of input signals is accompanied by generation of activation signals to a light or sound generating feedback device, and wherein the activation signals depend upon the kinds of input signals currently selected.

118. The method of claim 109, wherein a new subset of fingers is formed upon simultaneous finger release from the all fingers resting state, wherein this new subset consists of those fingers which remain on the surface, and wherein this new subset chooses a new kinds of input signals which can be generated in response to further distinctive finger motions.

119. 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.

120. The method of claim 119, wherein one of the representative tangential velocities is

a weighted average of several instantaneous velocities.

121. The method of claim 119, wherein the touch surface is a multi-touch surface, the touch devices are fingers, the cursor movement velocity is the hand translation, rotation, or scaling velocity extracted from the touching fingers, and the mode for continuously moving the cursor is entered when the velocity of the dominant hand motion component passes the deceleration test as the last fingers are lifted.

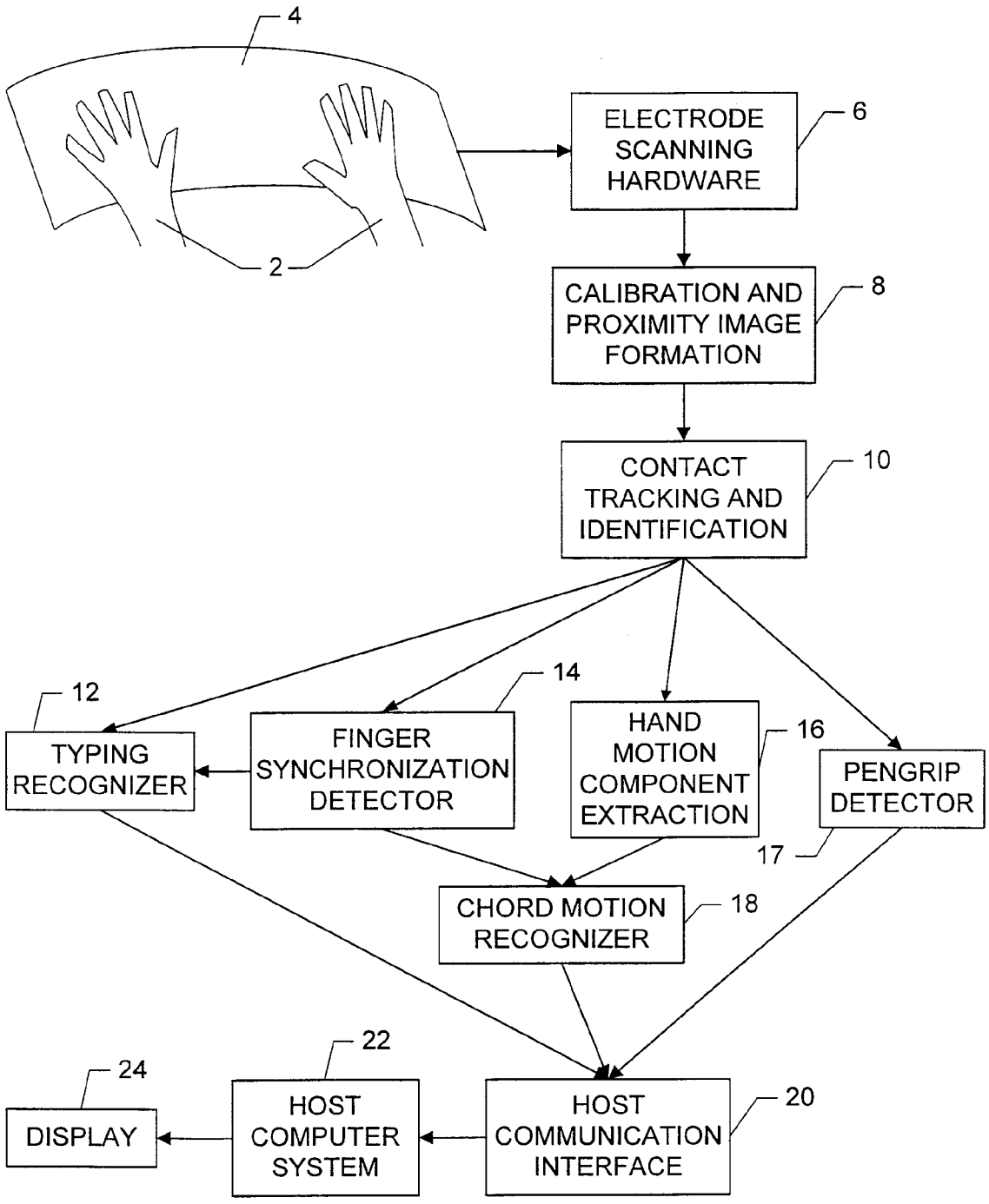


*FIG. 1*

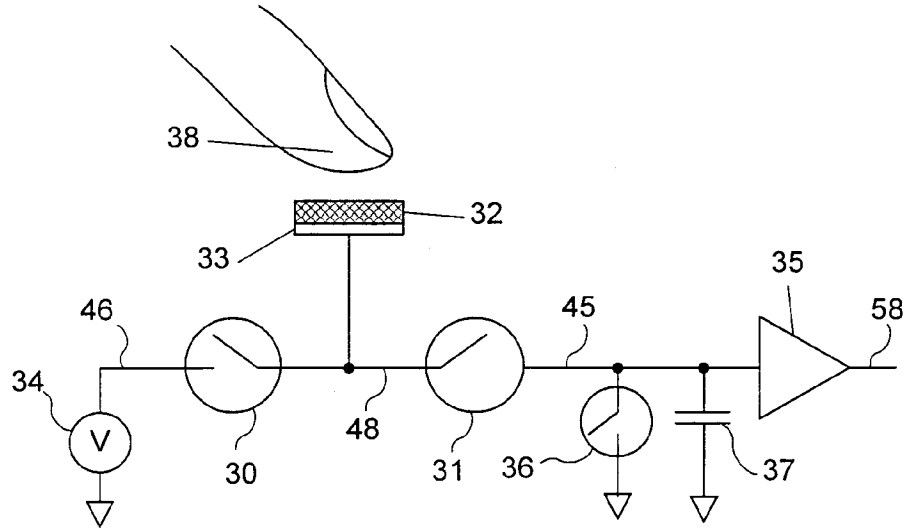


FIG. 2

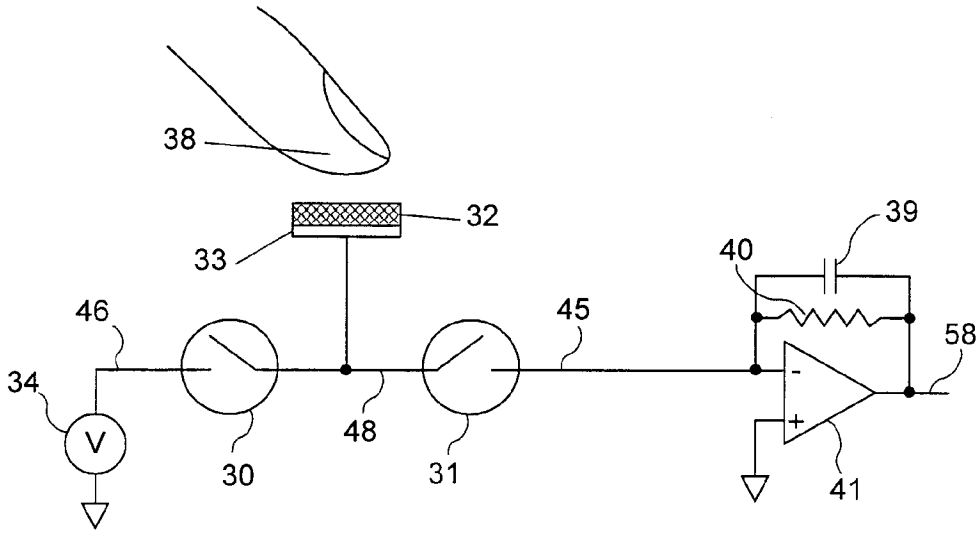


FIG. 3A

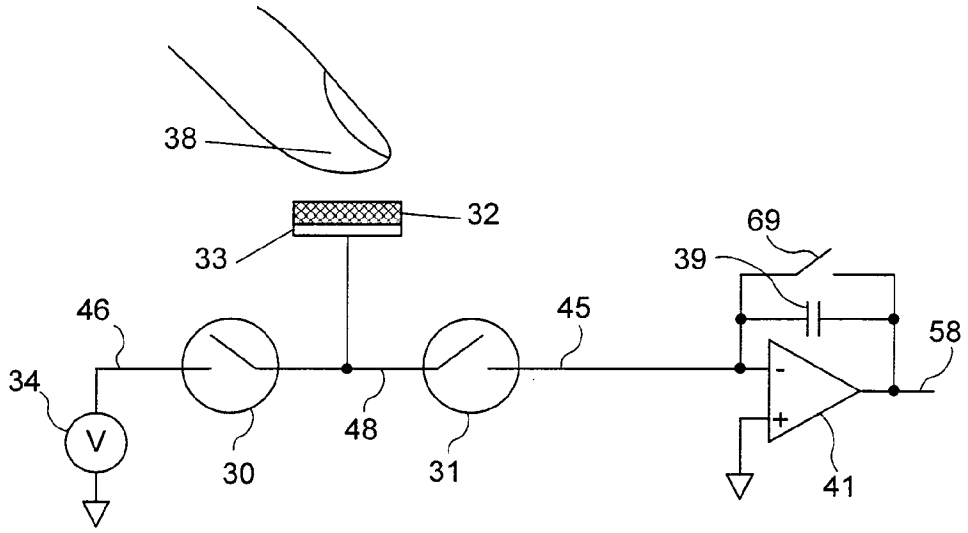


FIG. 3B

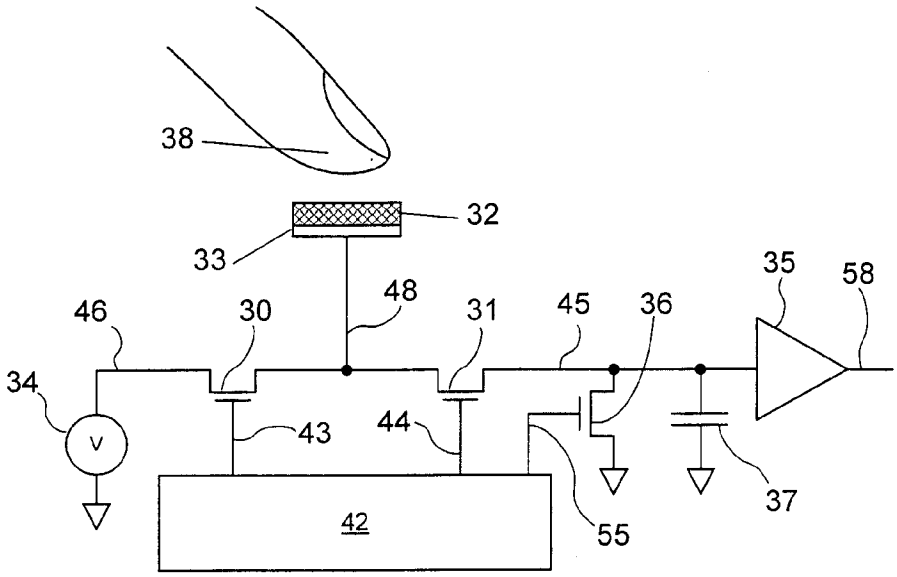


FIG. 4A

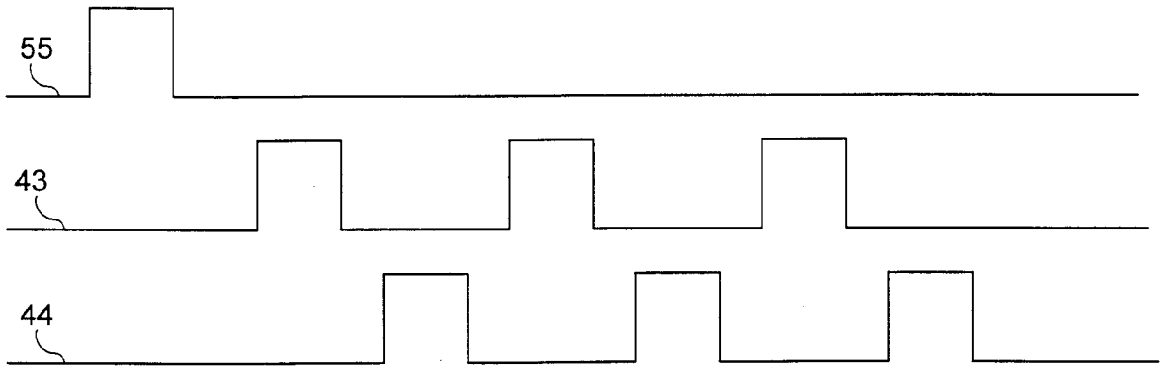


FIG. 4B

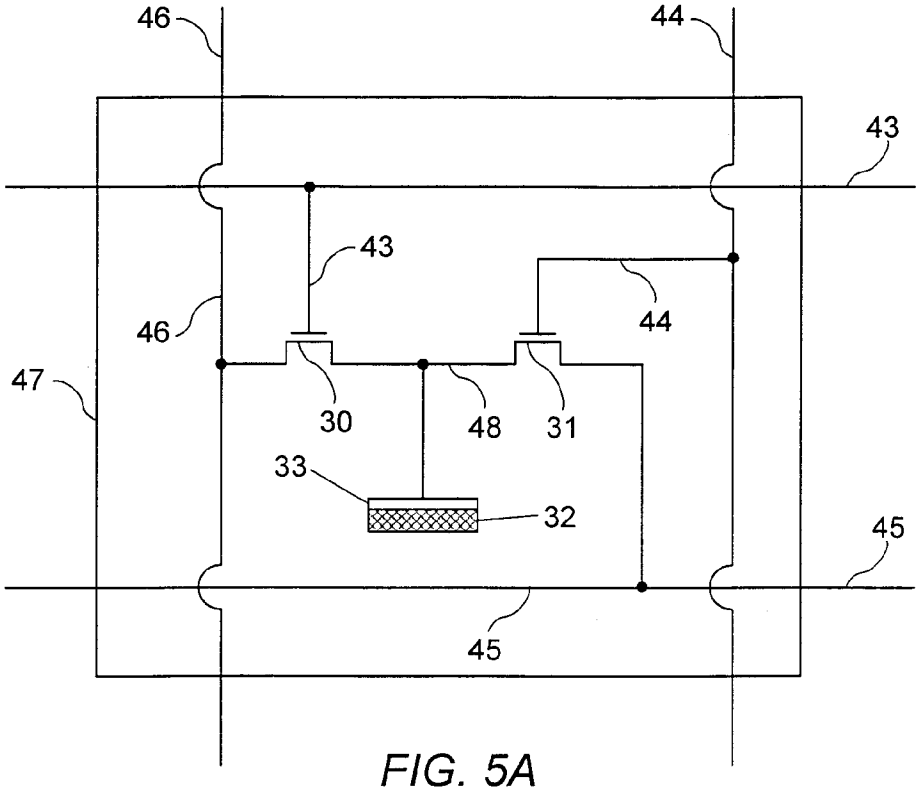


FIG. 5A

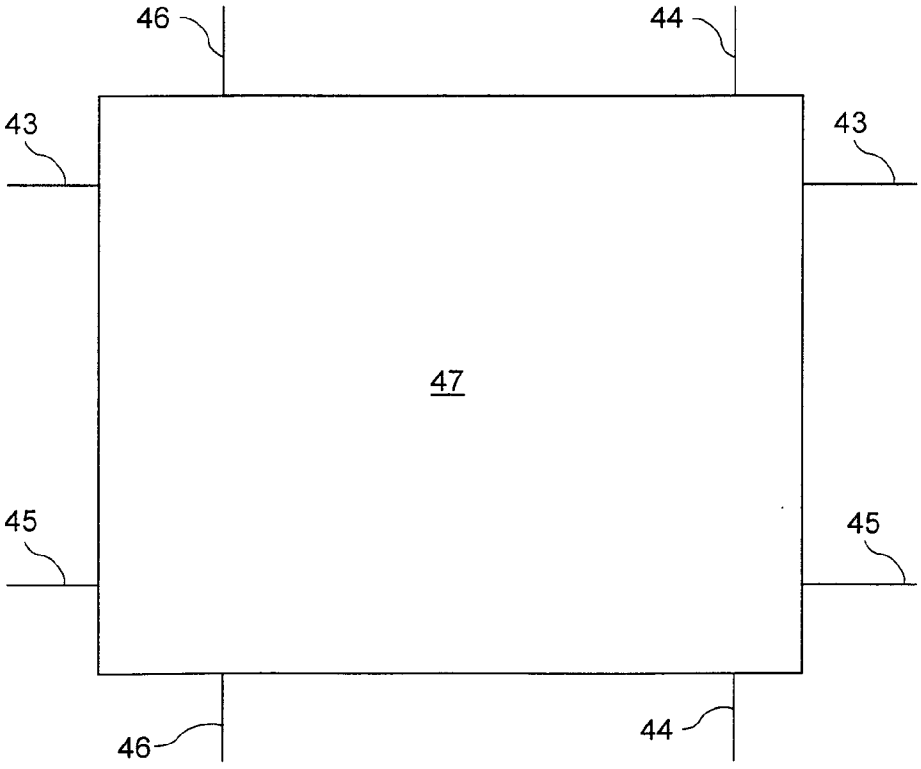


FIG. 5B

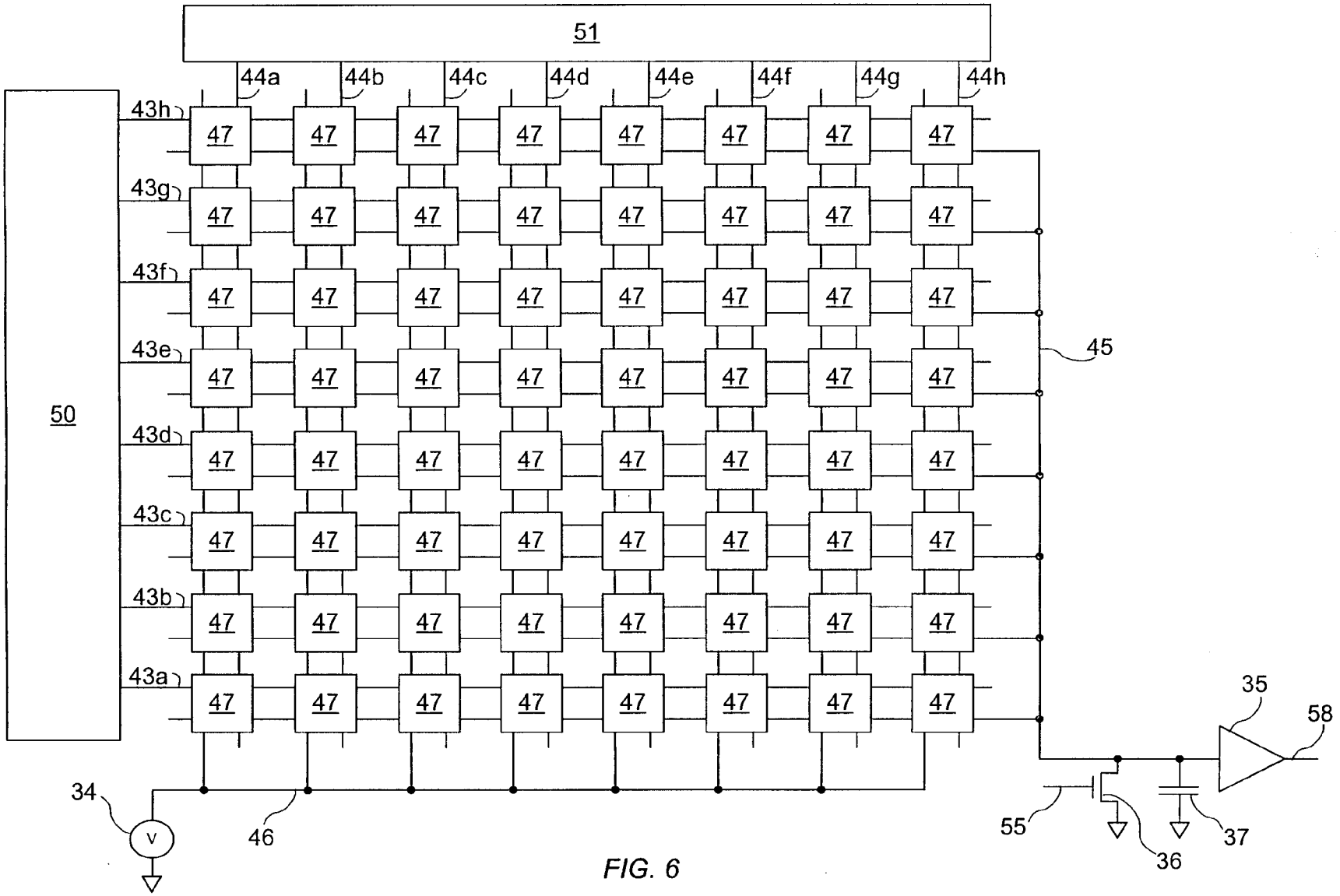


FIG. 6

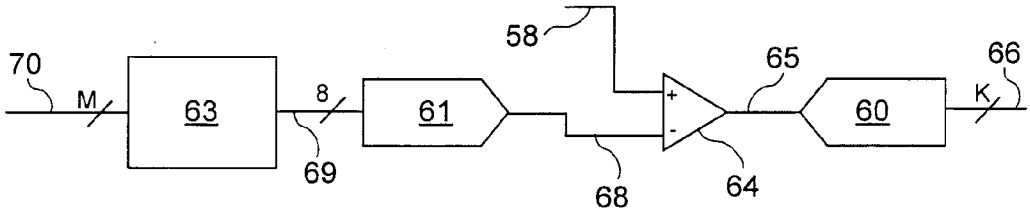


FIG. 7A

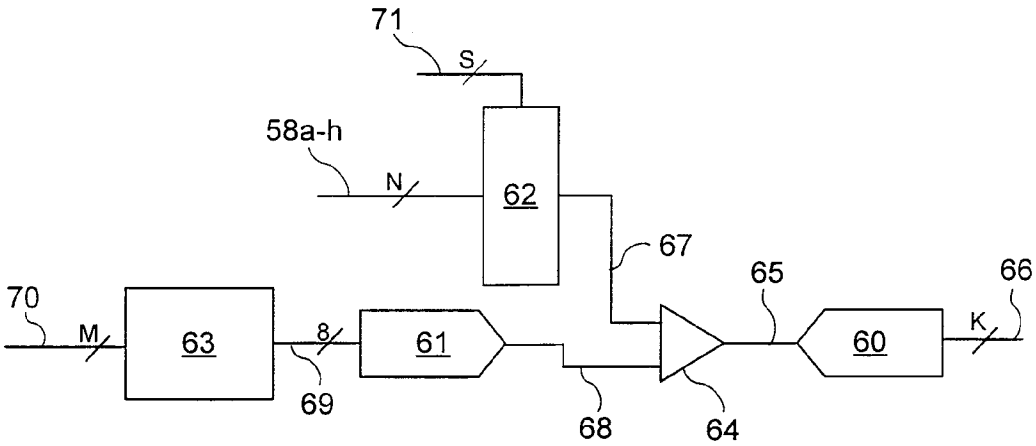


FIG. 7B

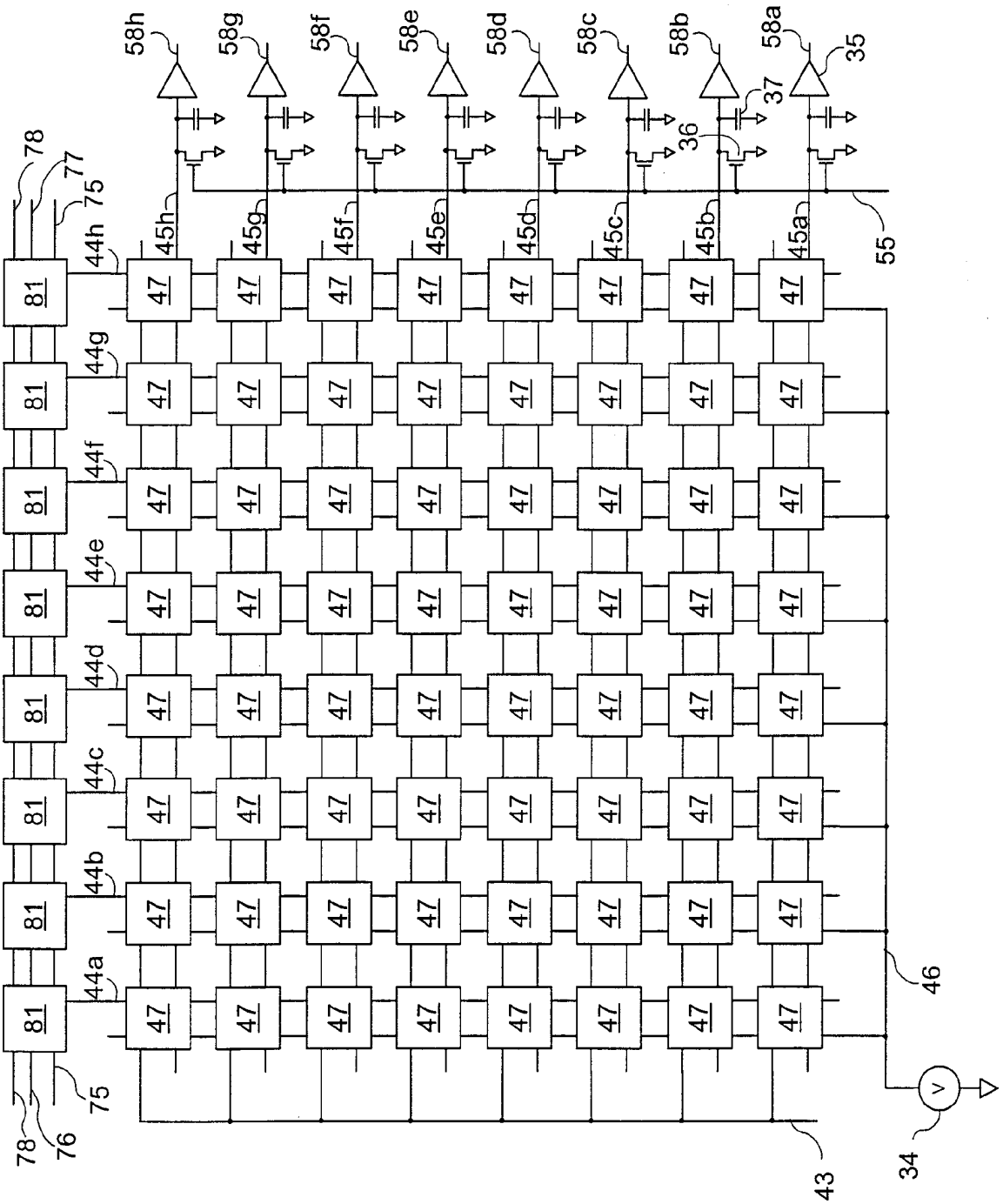


FIG. 8

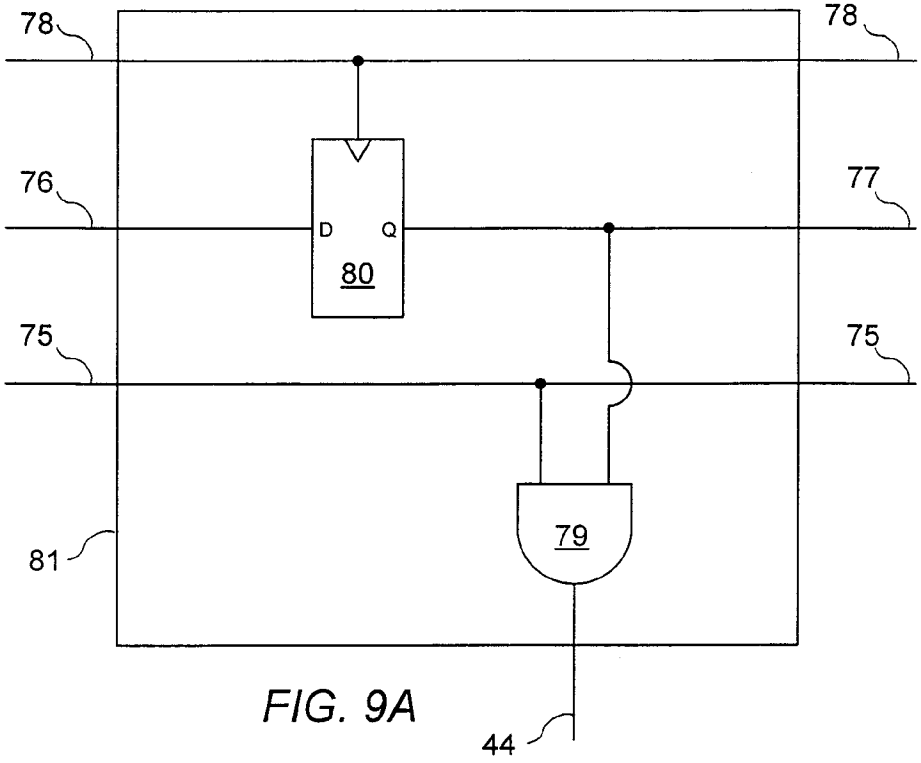


FIG. 9A

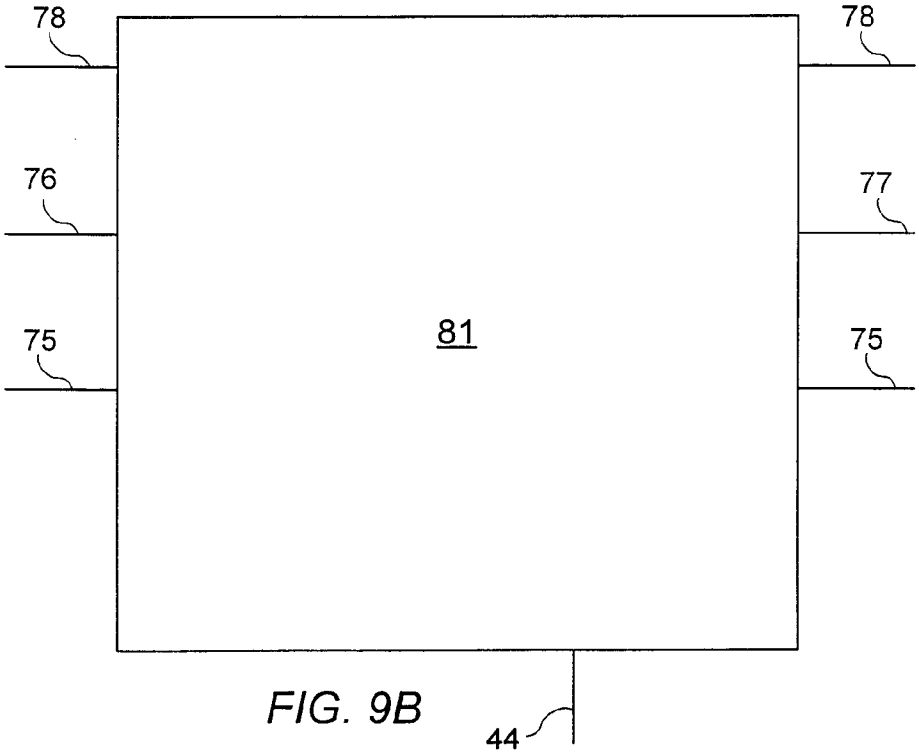


FIG. 9B

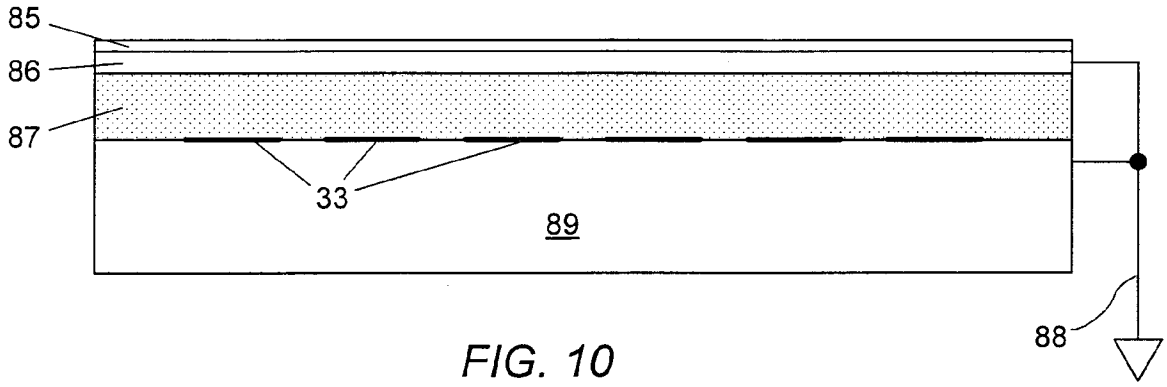


FIG. 10

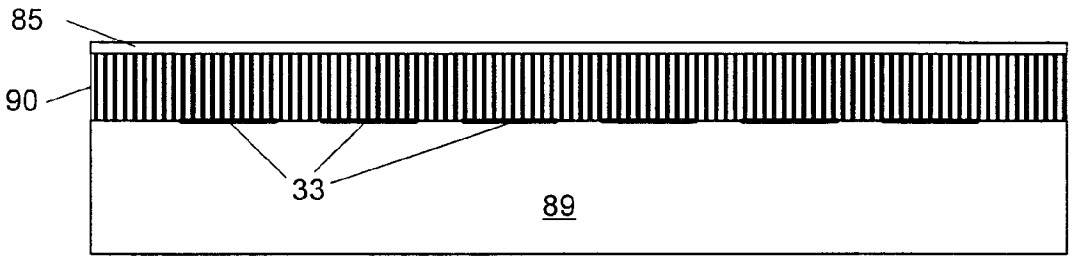


FIG. 11

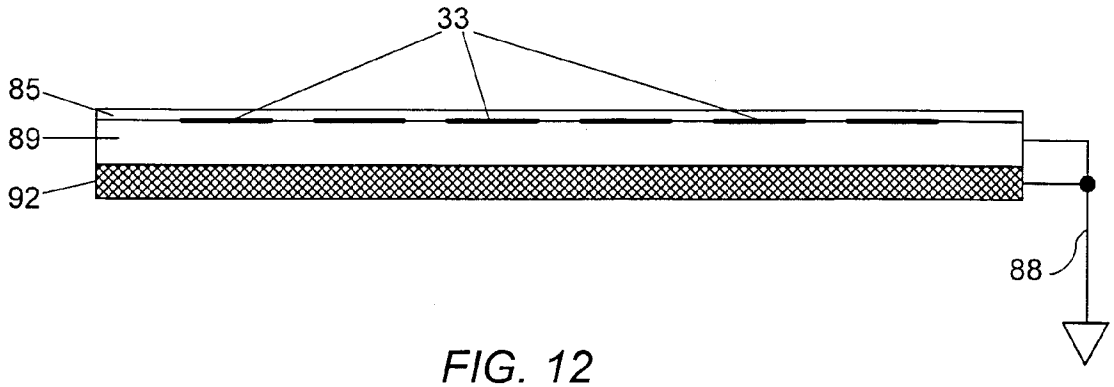


FIG. 12

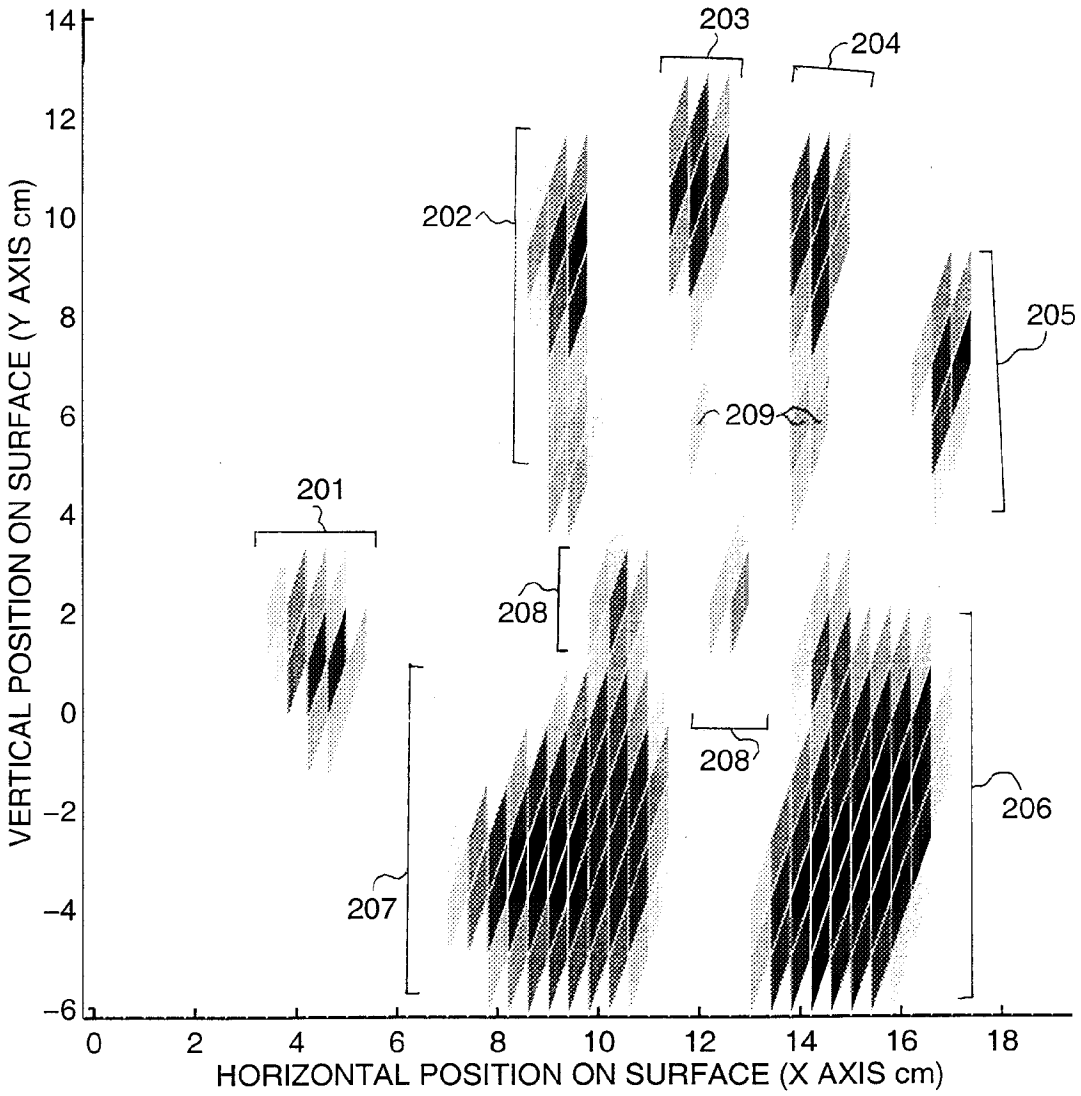


*FIG. 13*

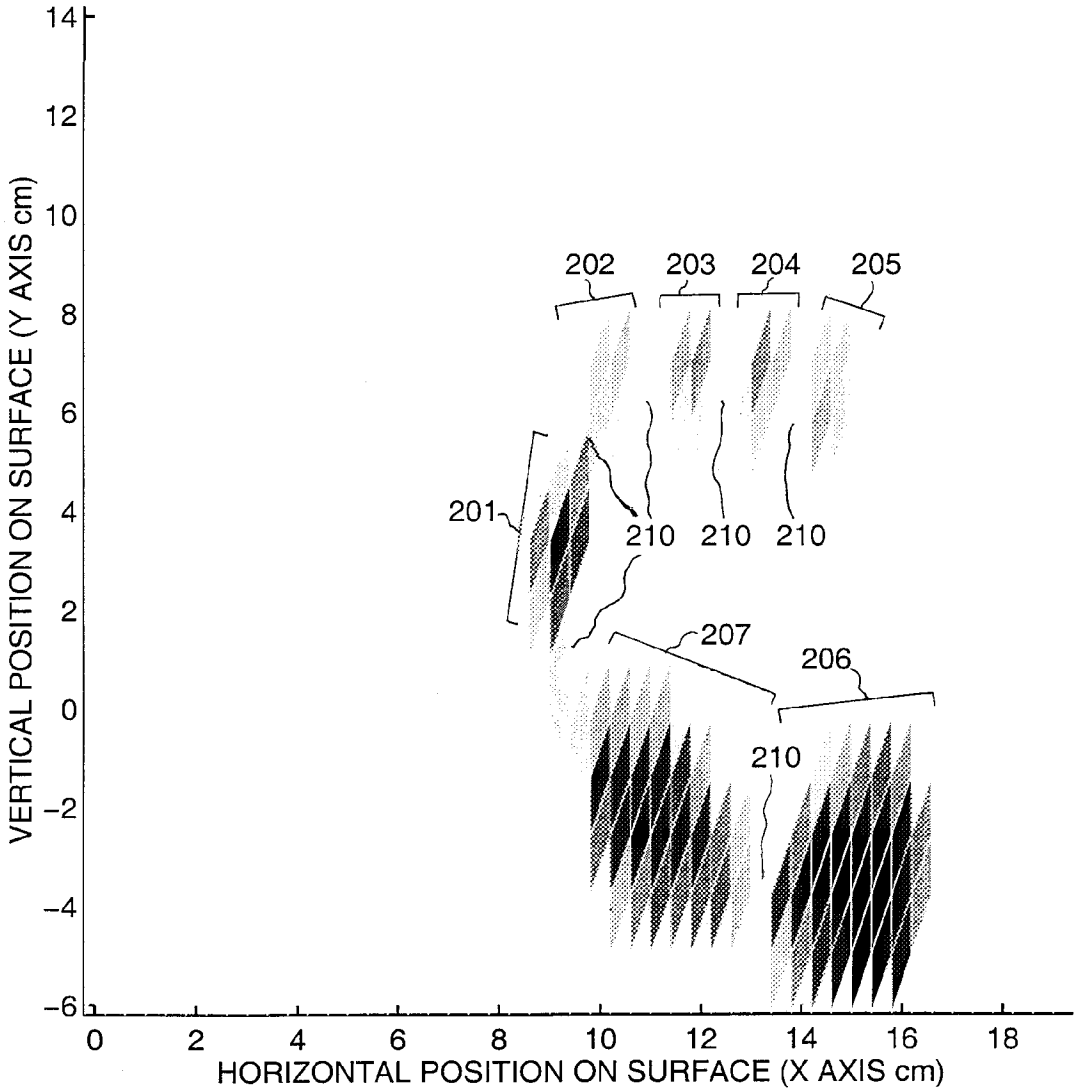


*FIG. 14*

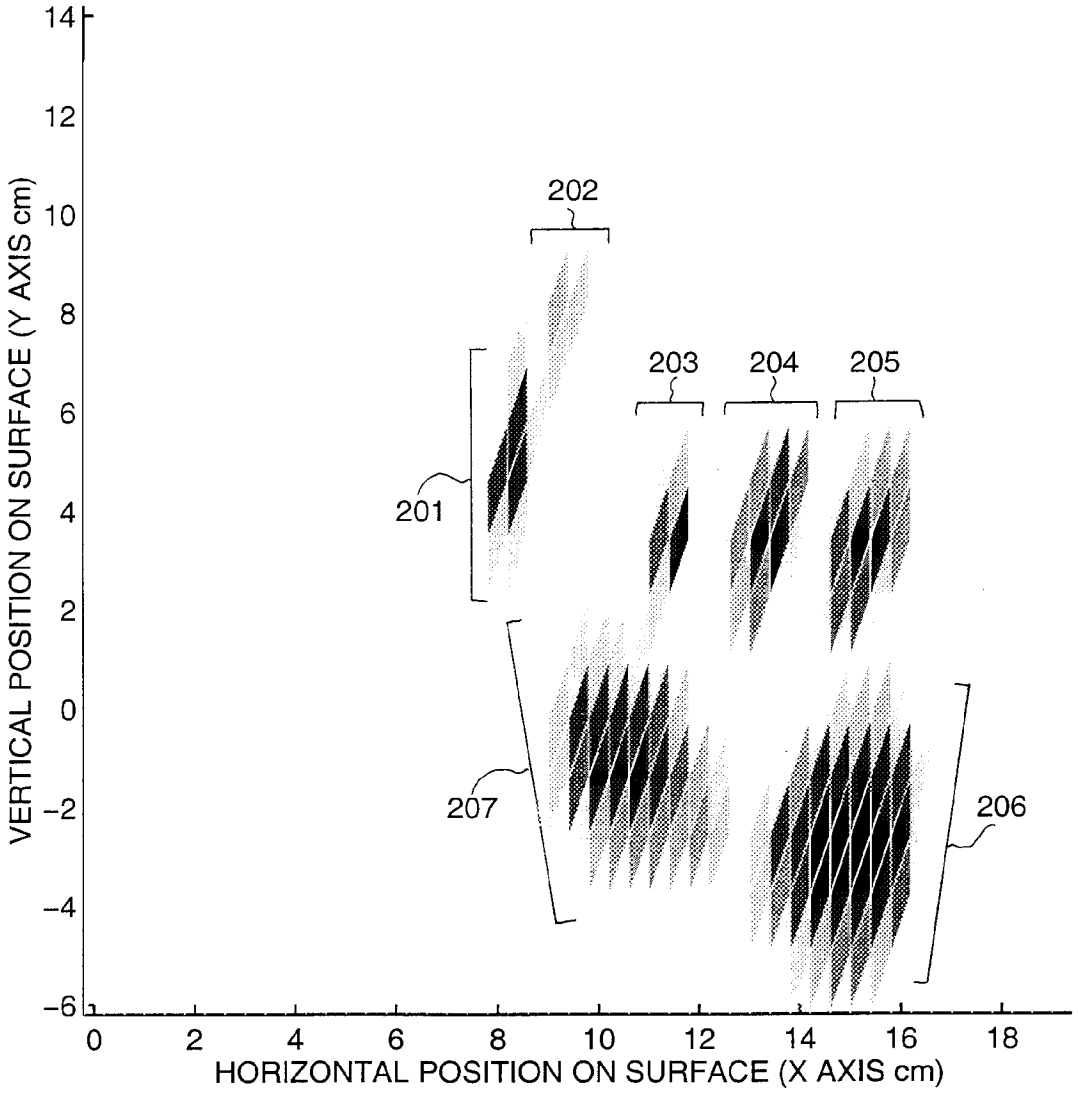


*FIG. 15*

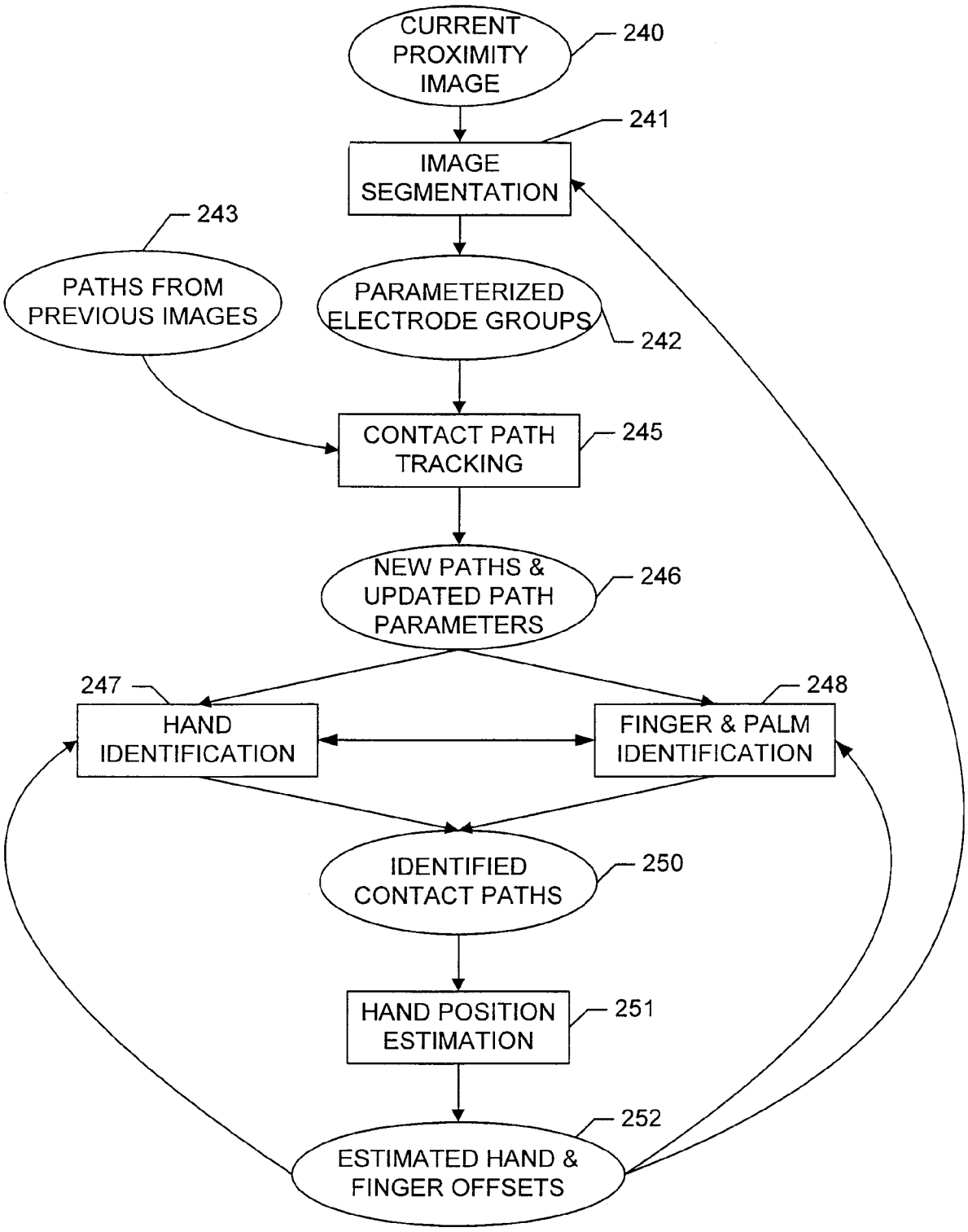
CURRENT PROXIMITY IMAGE
240
IMAGE SEGMENTATION
241
PATHS FROM PREVIOUS IMAGES
243
PARAMETERIZED ELECTRODE GROUPS
242
CONTACT PATH TRACKING
245
NEW PATHS & UPDATED PATH PARAMETERS
246
247
HAND IDENTIFICATION
248
FINGER & PALM IDENTIFICATION
IDENTIFIED CONTACT PATHS
250
HAND POSITION ESTIMATION
251
252
ESTIMATED HAND & FINGER OFFSETS

*FIG. 16*

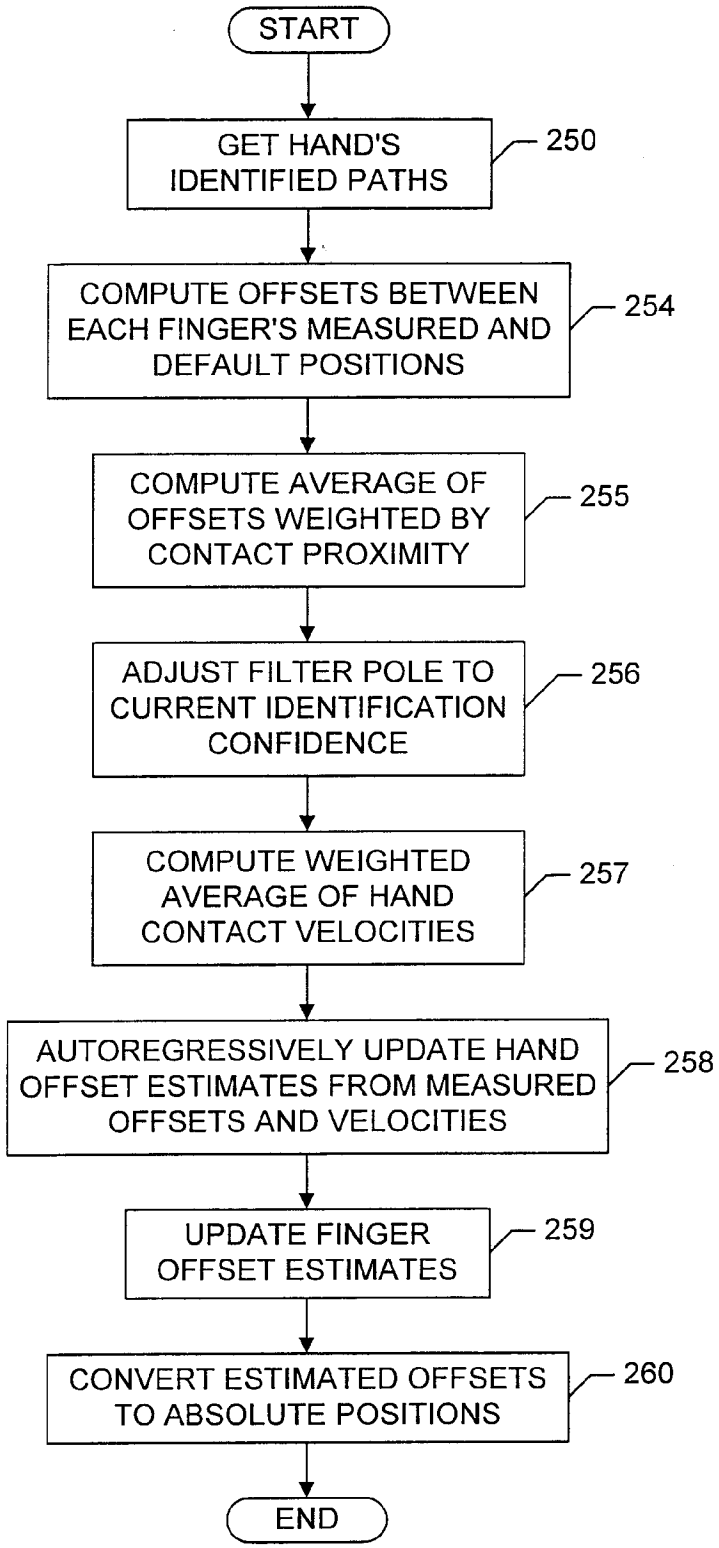


*FIG. 17*

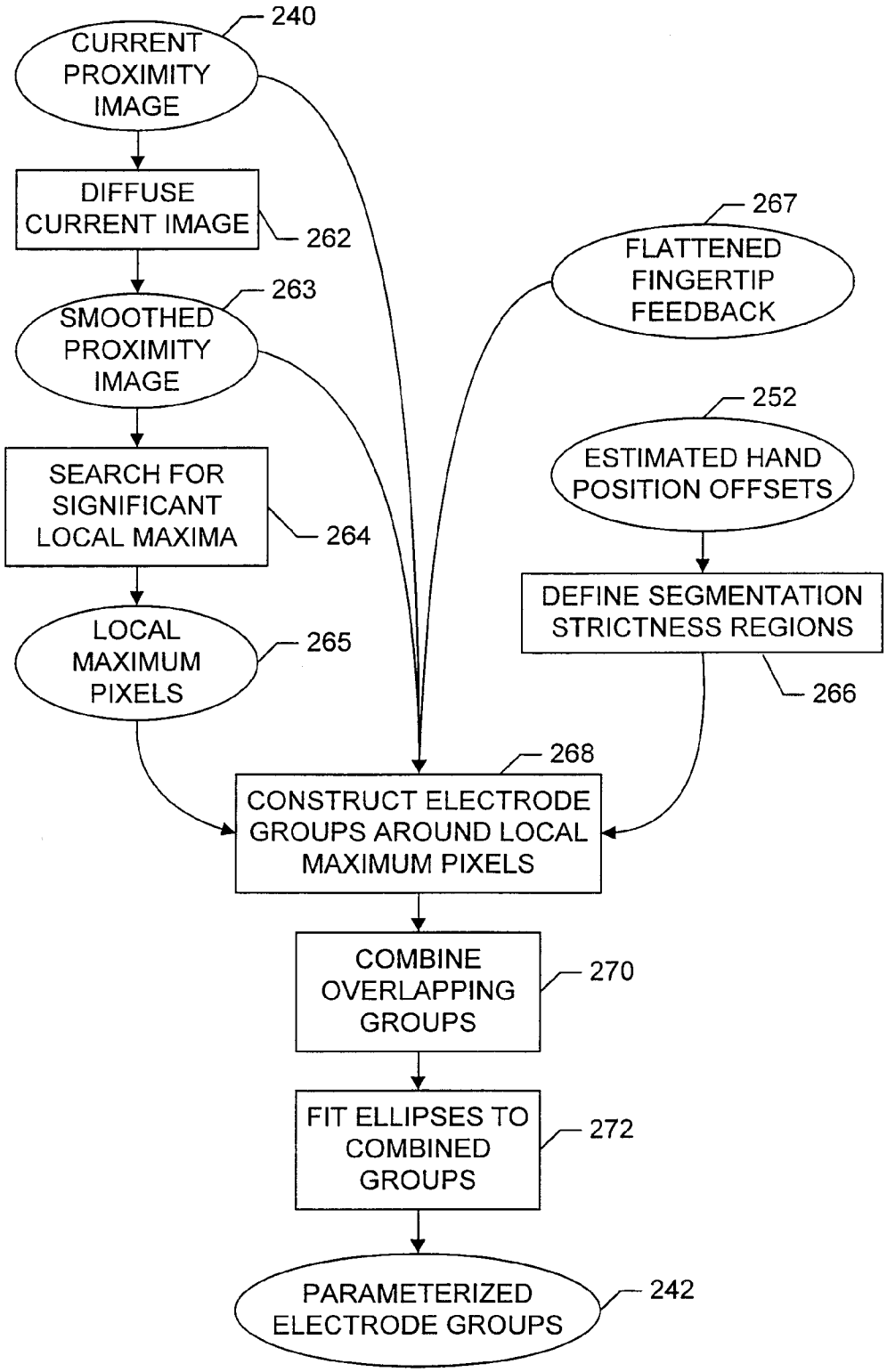
240
CURRENT PROXIMITY IMAGE
DIFFUSE CURRENT IMAGE
262
263
SMOOTHED PROXIMITY IMAGE
SEARCH FOR SIGNIFICANT LOCAL MAXIMA
264
LOCAL MAXIMUM PIXELS
265
267
FLATTENED FINGERTIP FEEDBACK
252
ESTIMATED HAND POSITION OFFSETS
DEFINE SEGMENTATION STRICTNESS REGIONS
266
268
CONSTRUCT ELECTRODE GROUPS AROUND LOCAL MAXIMUM PIXELS
COMBINE OVERLAPPING GROUPS
270
FIT ELLIPSES TO COMBINED GROUPS
272
PARAMETERIZED ELECTRODE GROUPS
242

*FIG. 18*

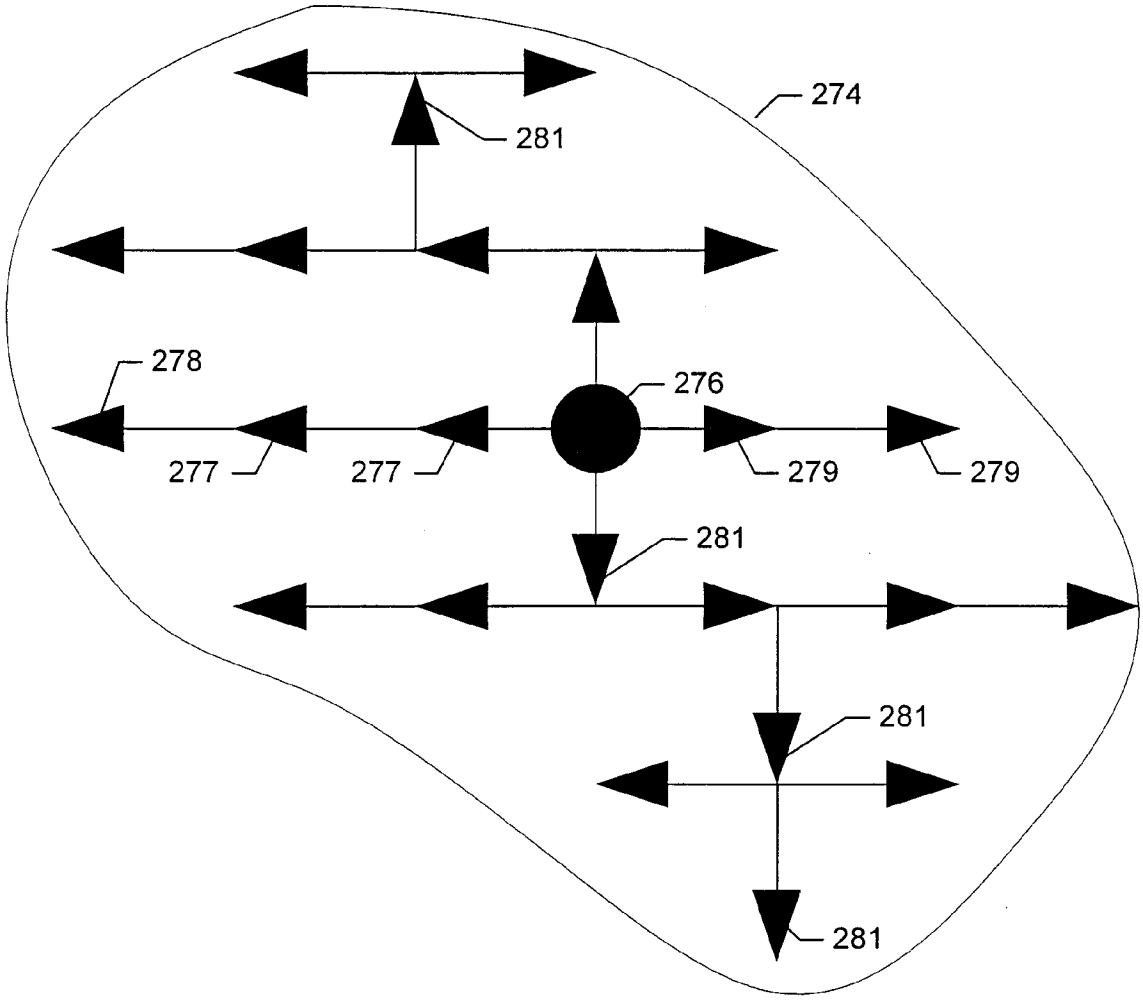


FIG. 19

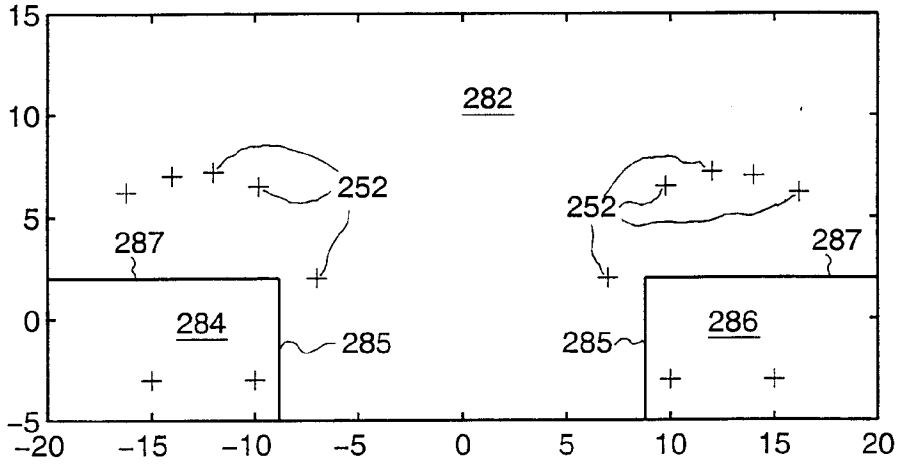


FIG. 20A

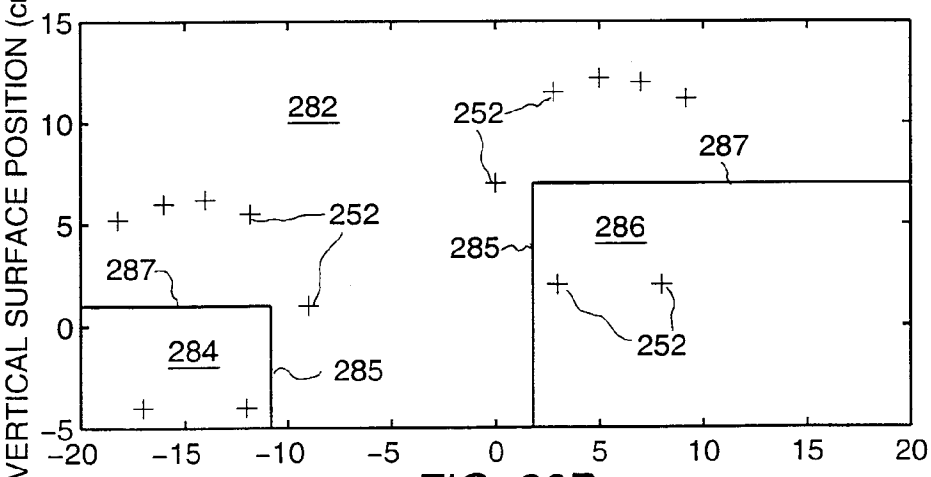


FIG. 20B

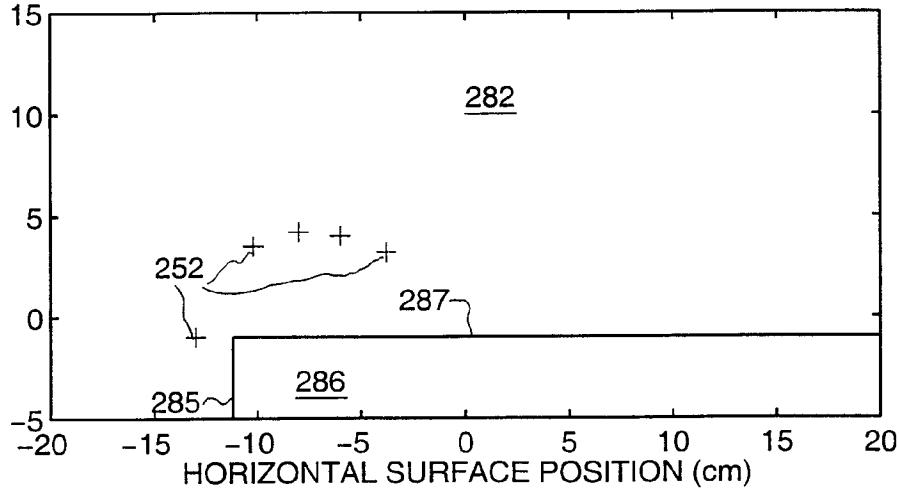


FIG. 20C

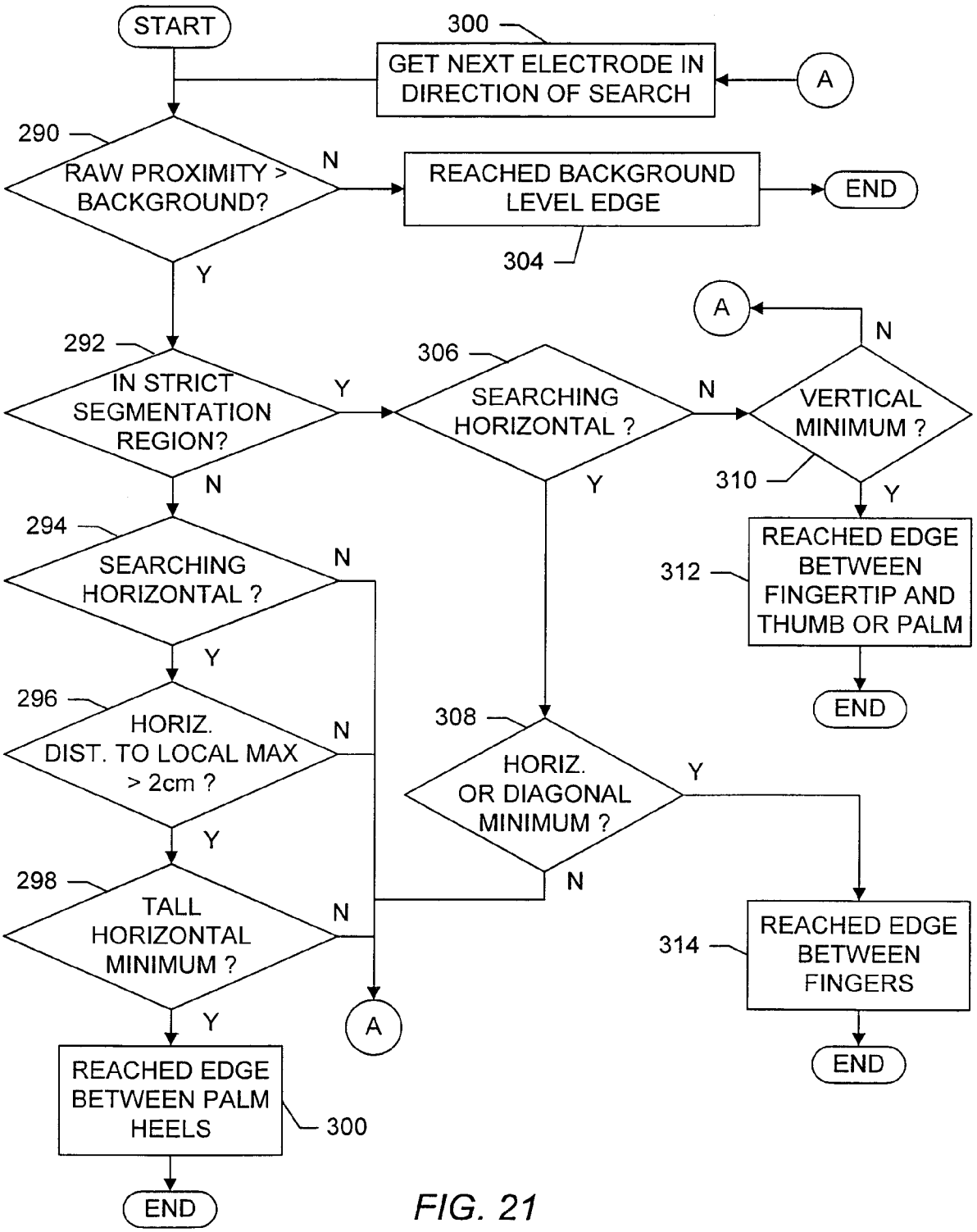
START
300
GET NEXT ELECTRODE IN DIRECTION OF SEARCH
A
290
RAW PROXIMITY > BACKGROUND?
N
REACHED BACKGROUND LEVEL EDGE
END
304
Y
A
N
292
IN STRICT SEGMENTATION REGION?
Y
306
SEARCHING HORIZONTAL ?
N
VERTICAL MINIMUM ?
N
Y
310
Y
294
SEARCHING HORIZONTAL ?
N
312
REACHED EDGE BETWEEN FINGERTIP AND THUMB OR PALM
Y
END
296
HORIZ. DIST. TO LOCAL MAX > 2cm ?
N
308
HORIZ. OR DIAGONAL MINIMUM ?
Y
Y
298
TALL HORIZONTAL MINIMUM ?
N
N
314
REACHED EDGE BETWEEN FINGERS
Y
A
END
REACHED EDGE BETWEEN PALM HEELS
300
END

FIG. 21

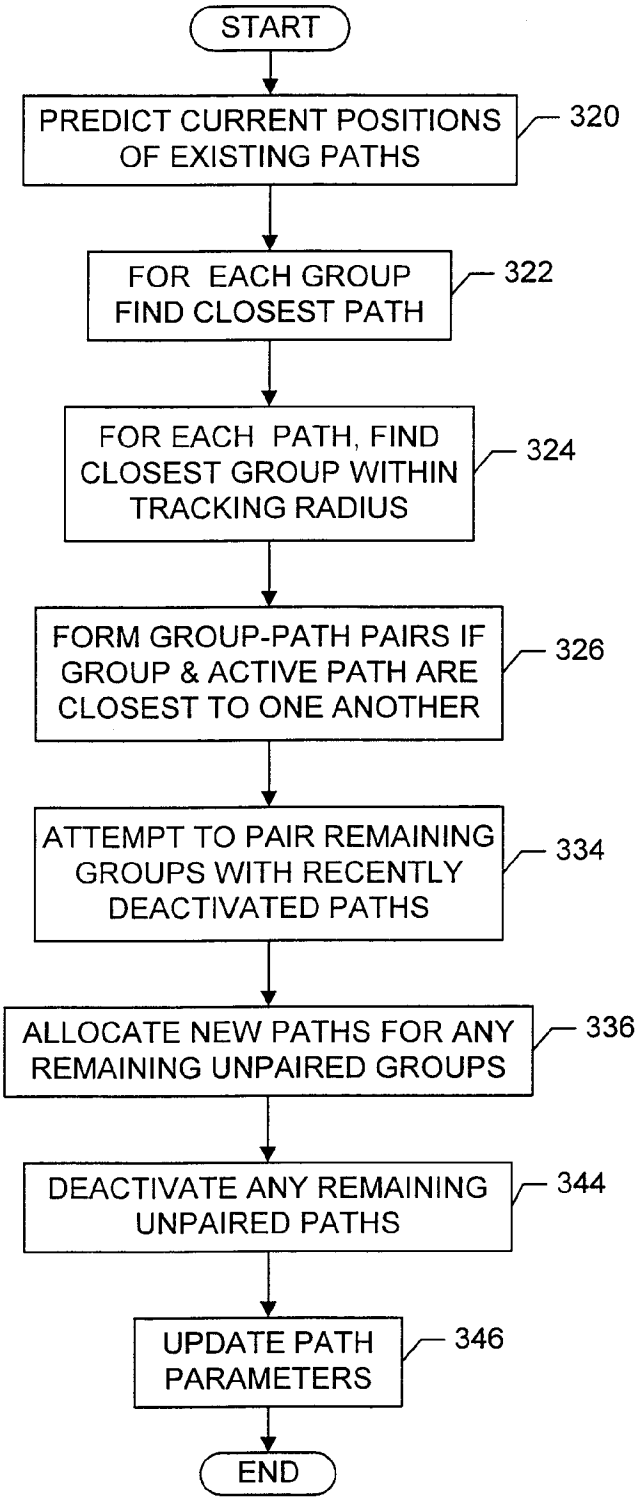


*FIG. 22*

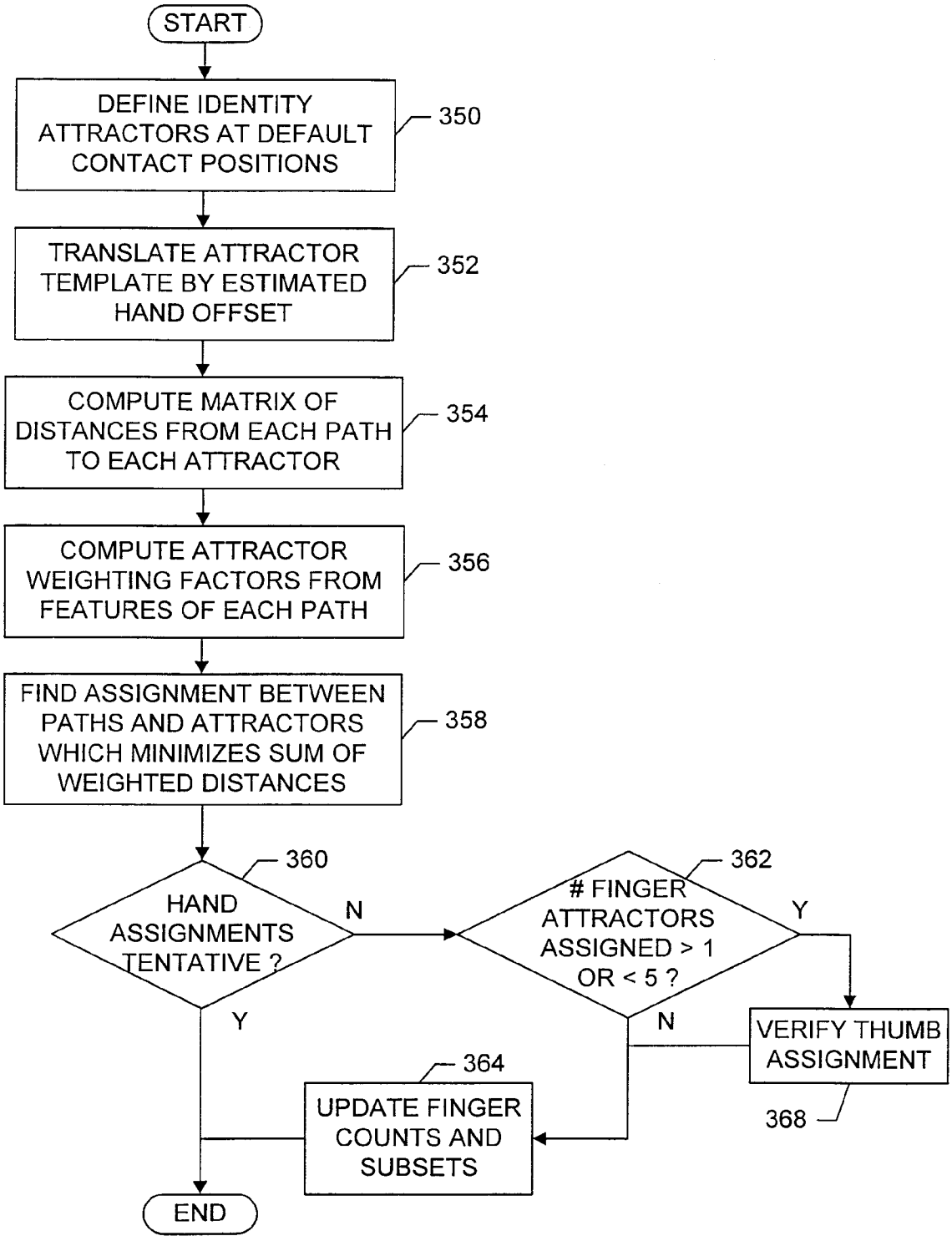


*FIG. 23*

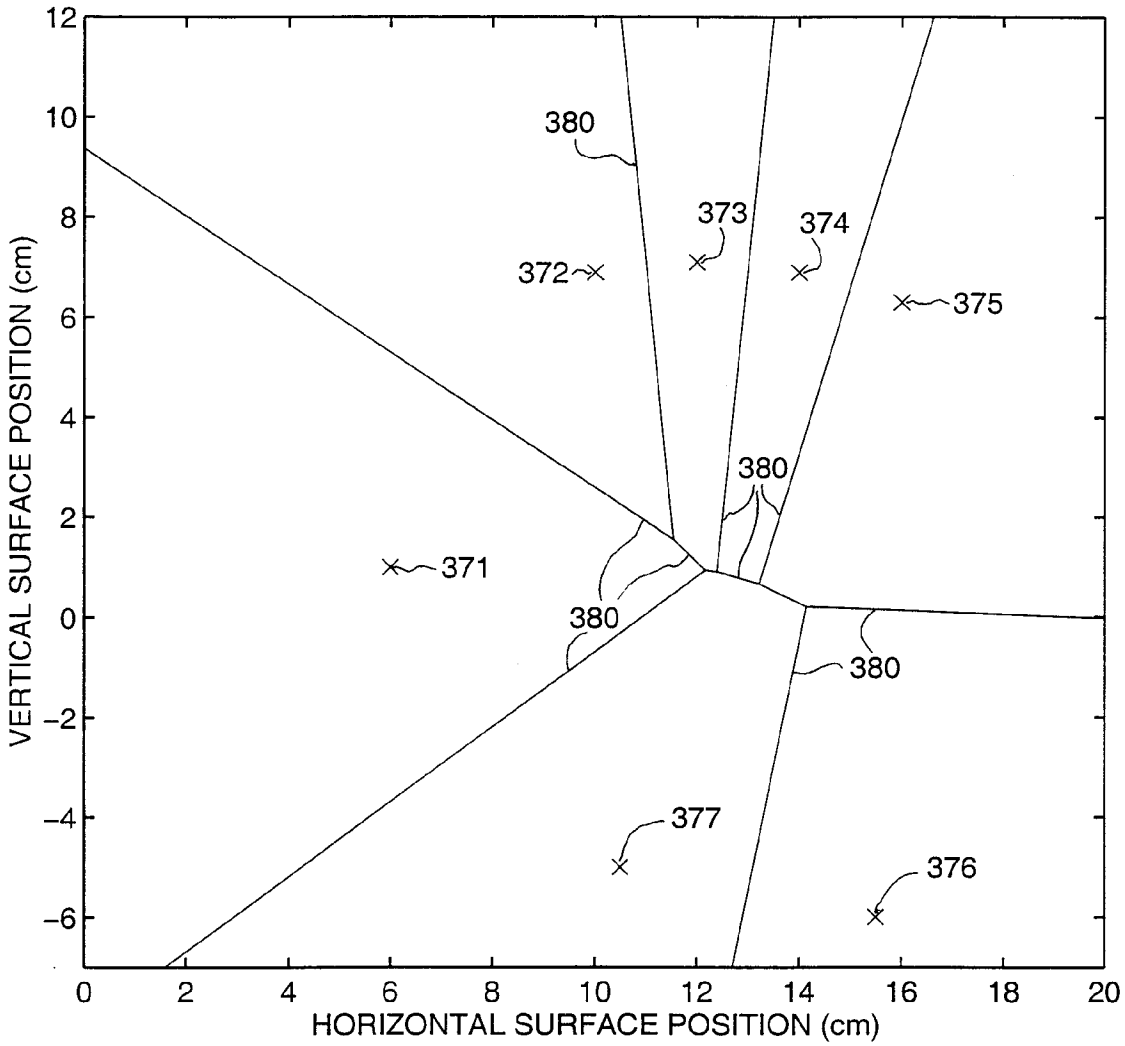


*FIG. 24*

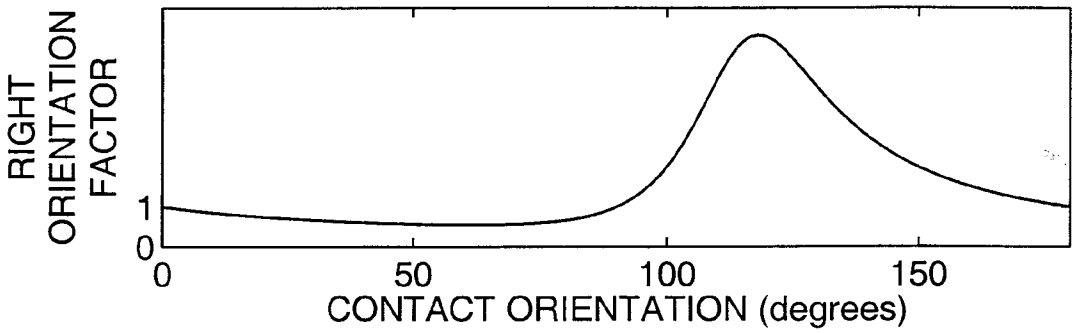


FIG. 25A

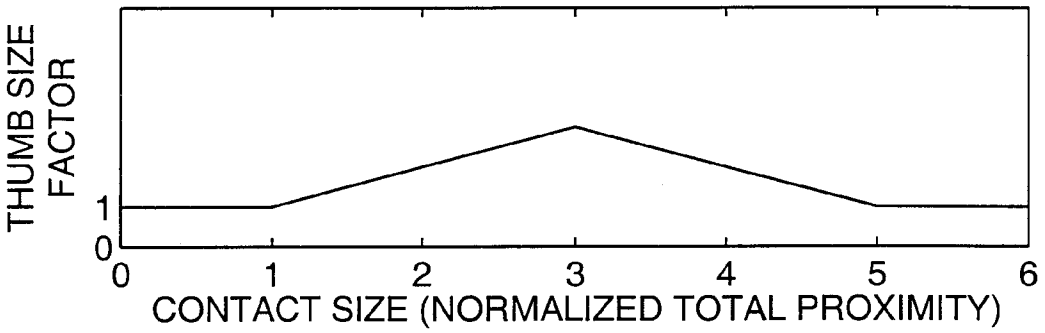


FIG. 25B

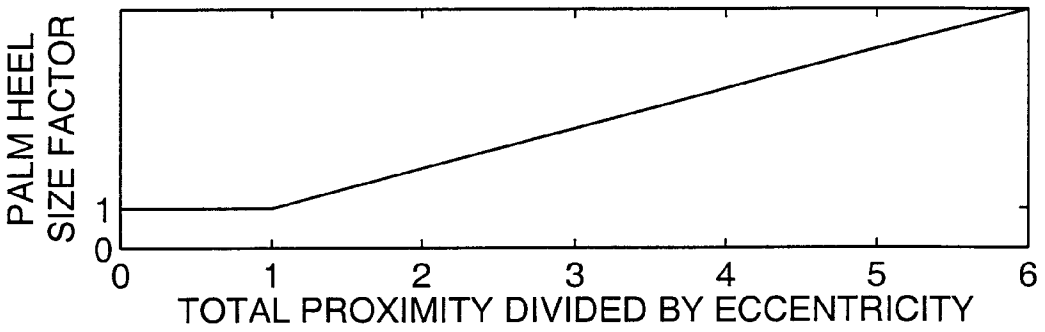


FIG. 25C

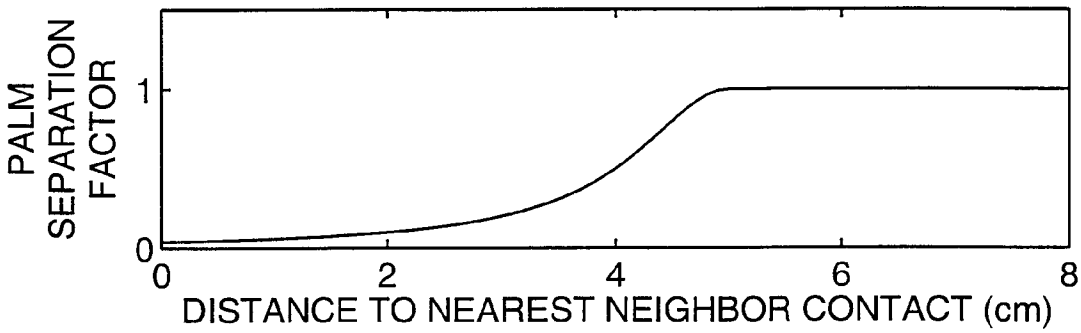


FIG. 25D

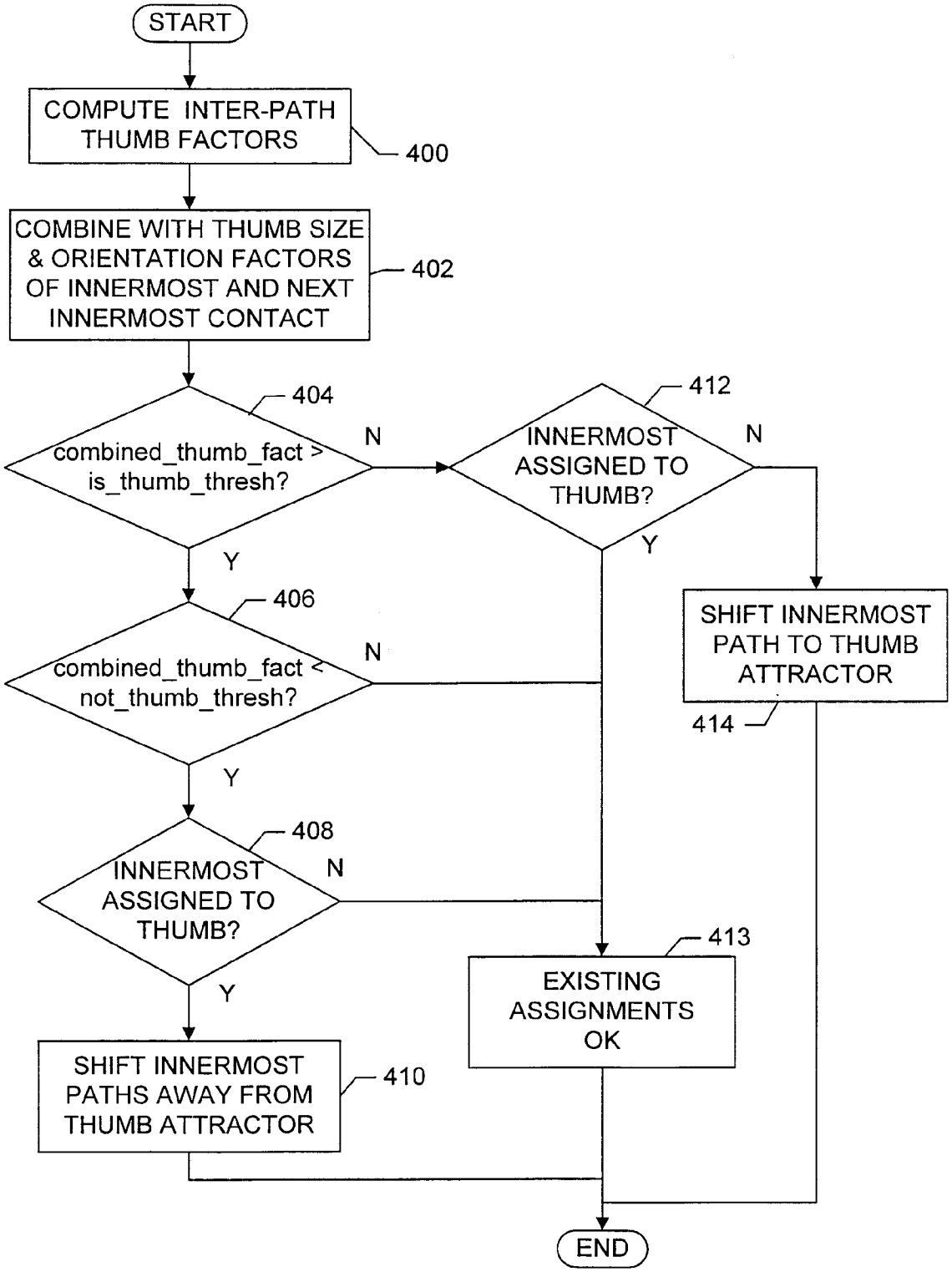
START
COMPUTE INTER-PATH THUMB FACTORS
400
COMBINE WITH THUMB SIZE & ORIENTATION FACTORS OF INNERMOST AND NEXT INNERMOST CONTACT
402
404
combined_thumb_fact > is_thumb_thresh?
N
Y
412
INNERMOST ASSIGNED TO THUMB?
N
Y
SHIFT INNERMOST PATH TO THUMB ATTRACTOR
414
406
combined_thumb_fact < not_thumb_thresh?
N
Y
408
INNERMOST ASSIGNED TO THUMB?
N
Y
413
EXISTING ASSIGNMENTS OK
SHIFT INNERMOST PATHS AWAY FROM THUMB ATTRACTOR
410
END

FIG. 26

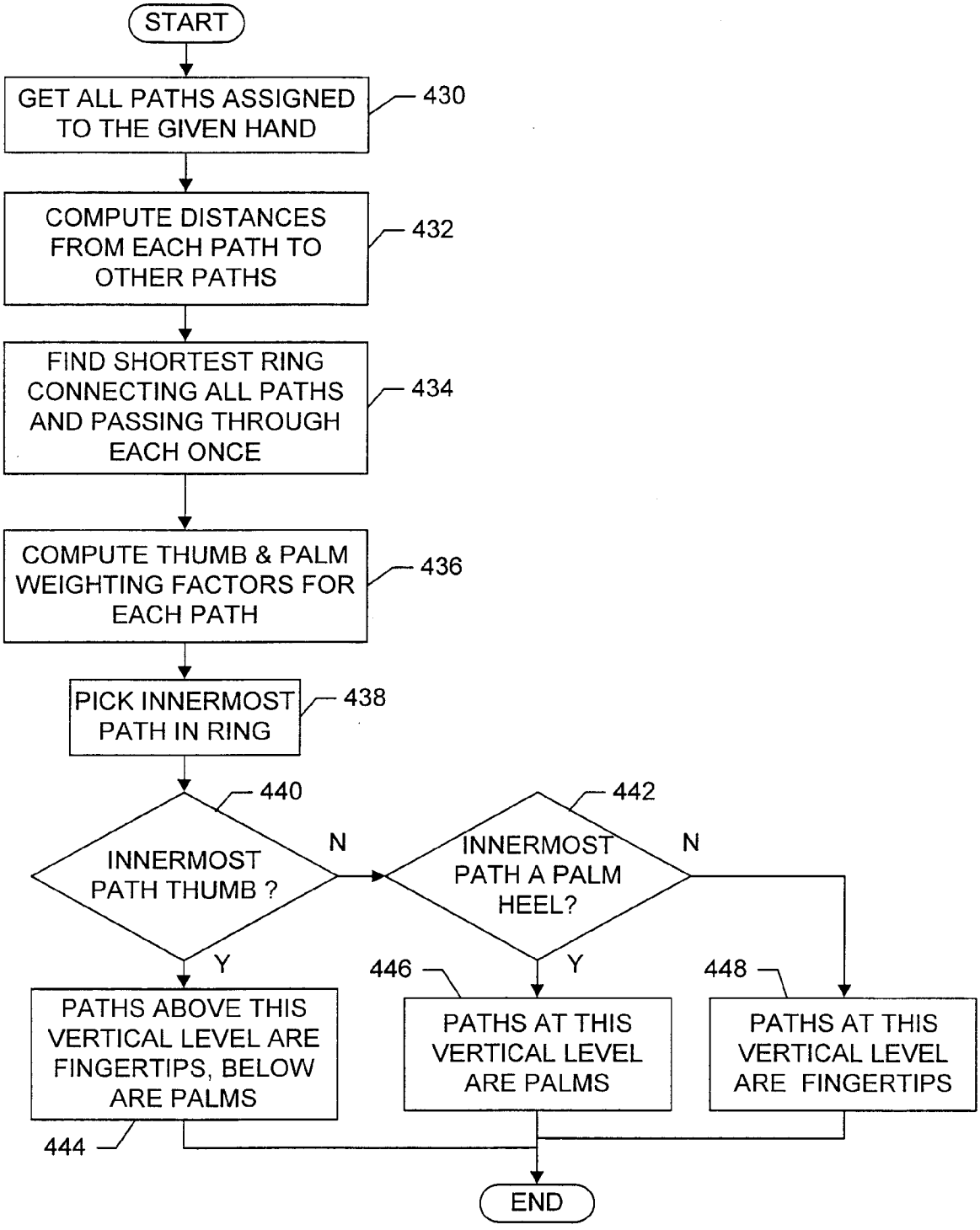


*FIG. 27*

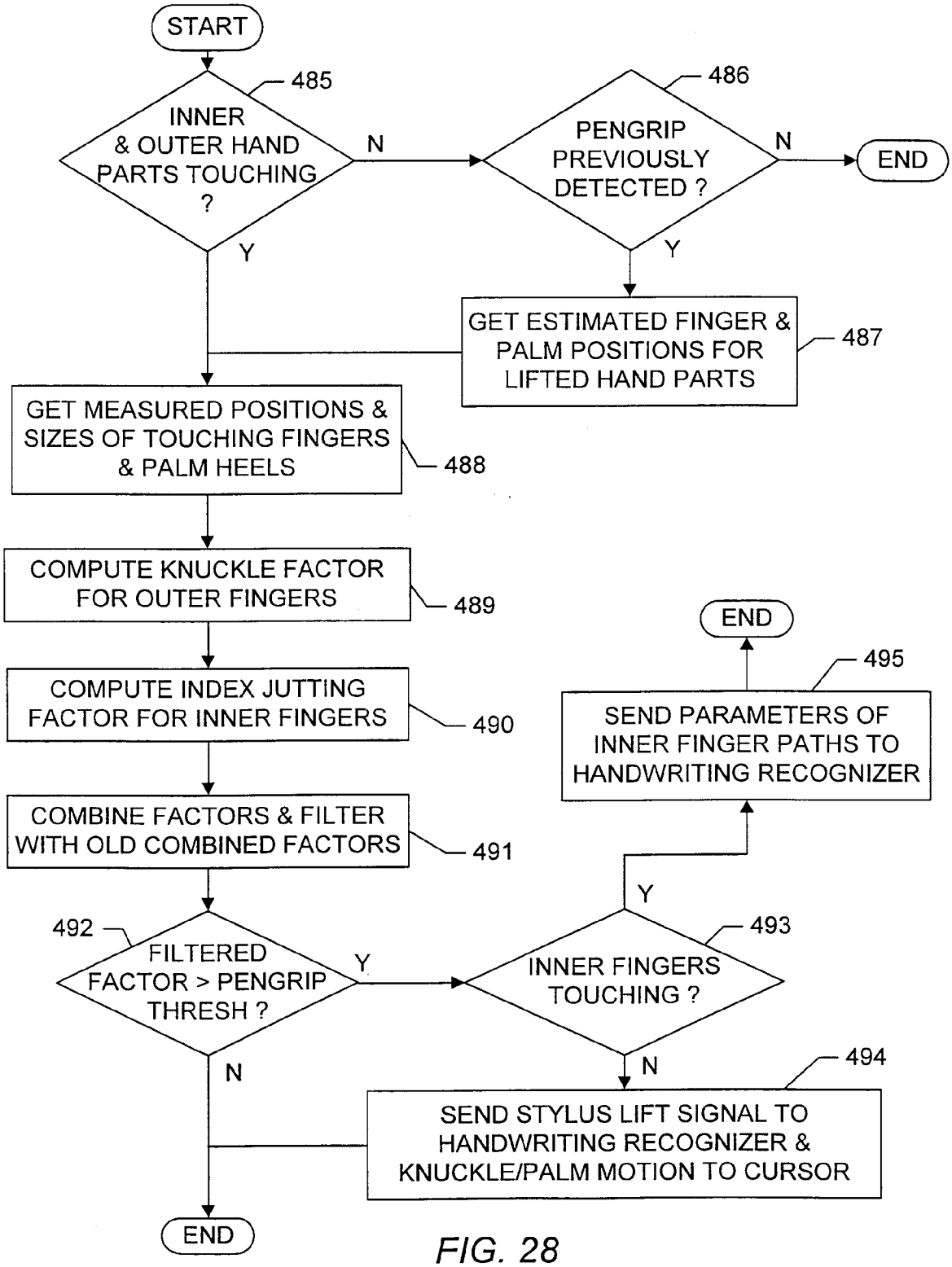


FIG. 28

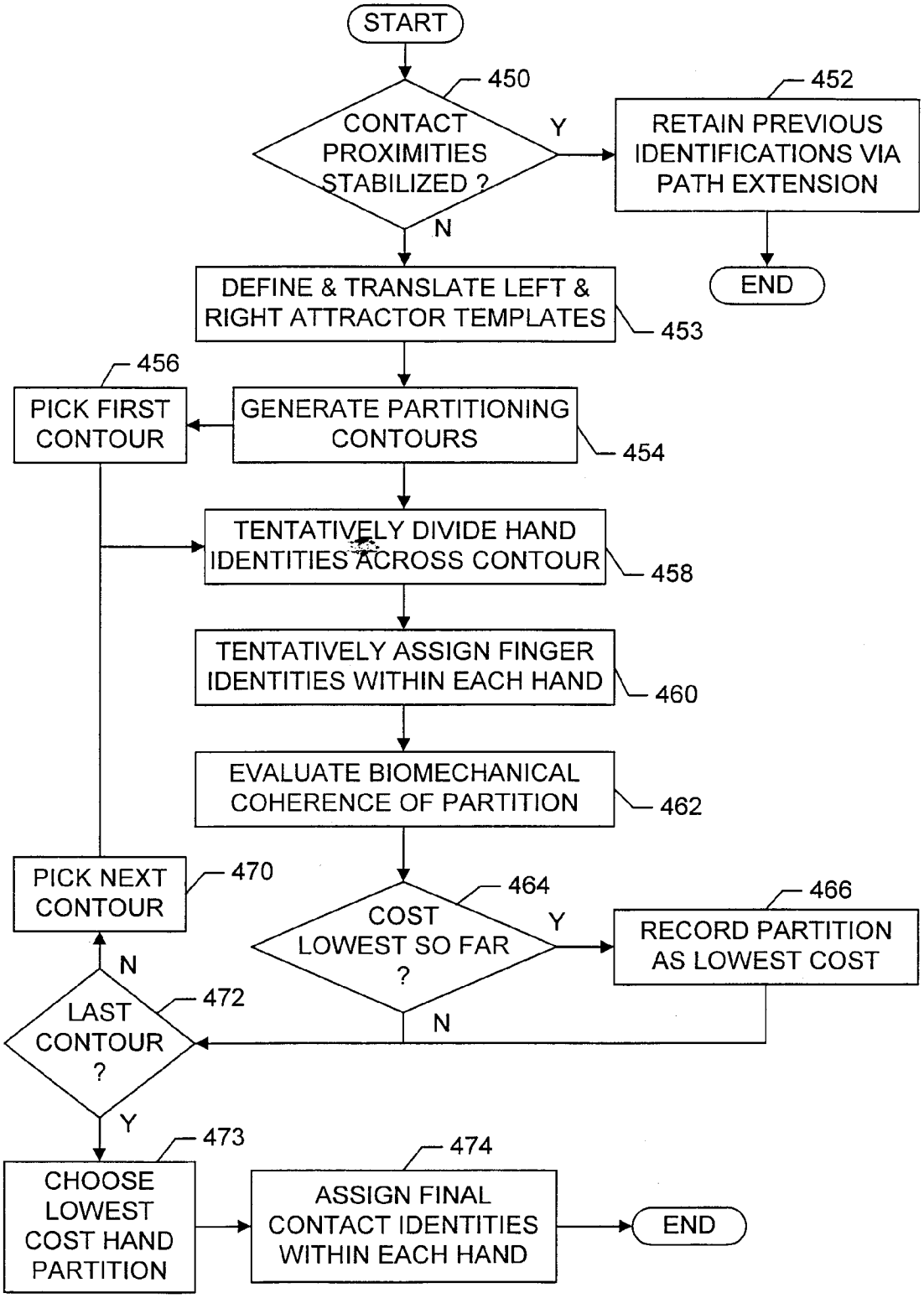


*FIG. 29*

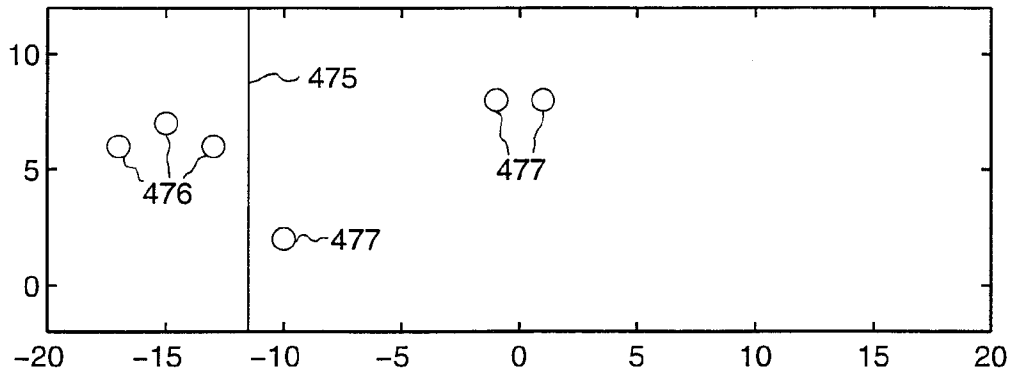


FIG. 30A

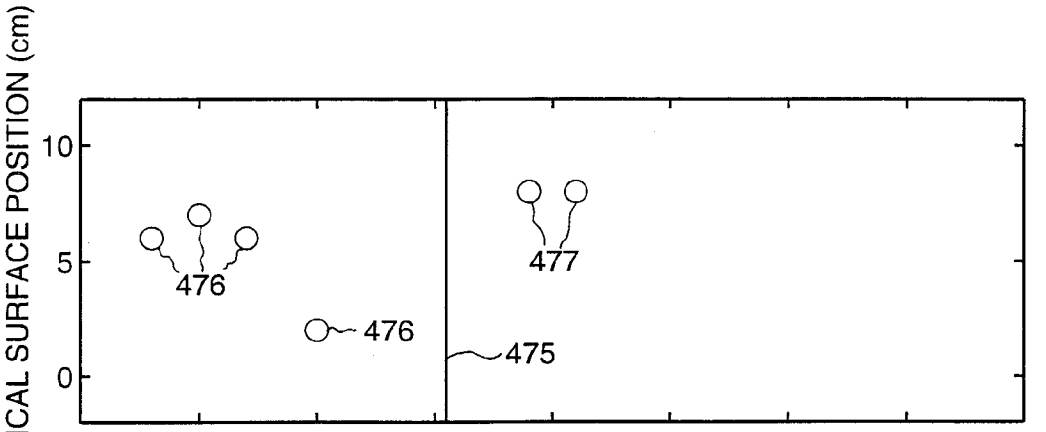


FIG. 30B

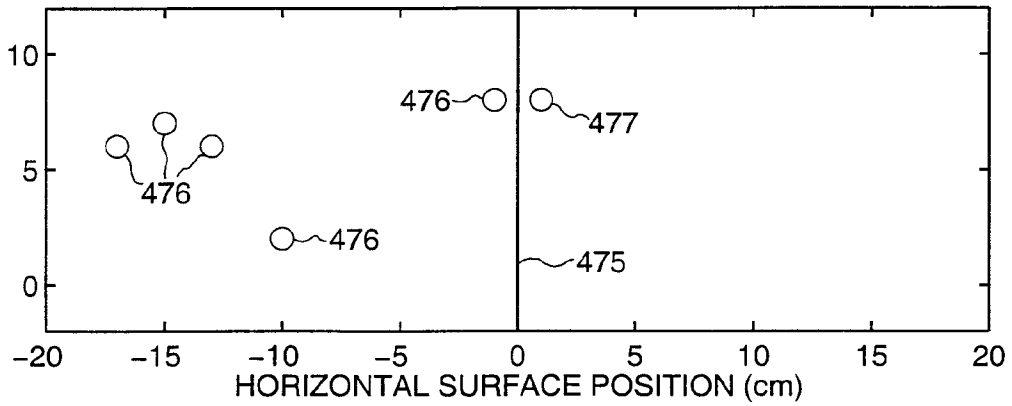


FIG. 30C

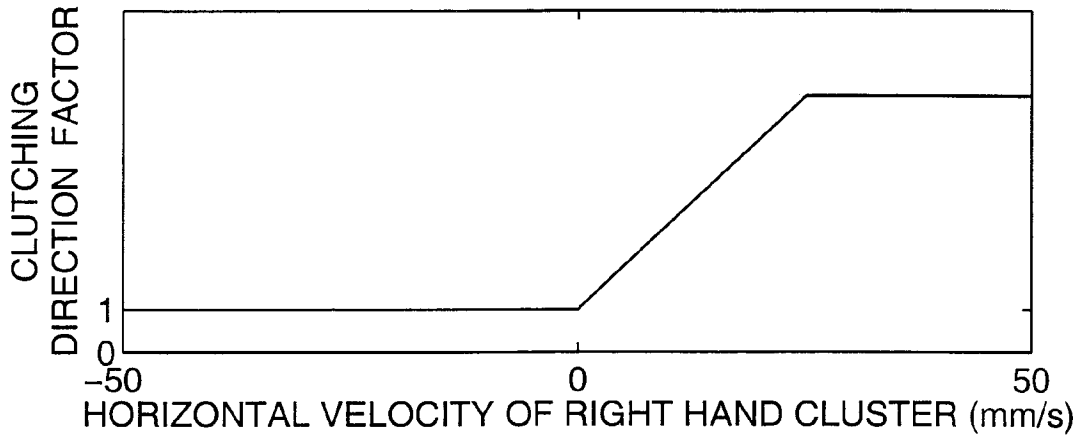


FIG. 31A

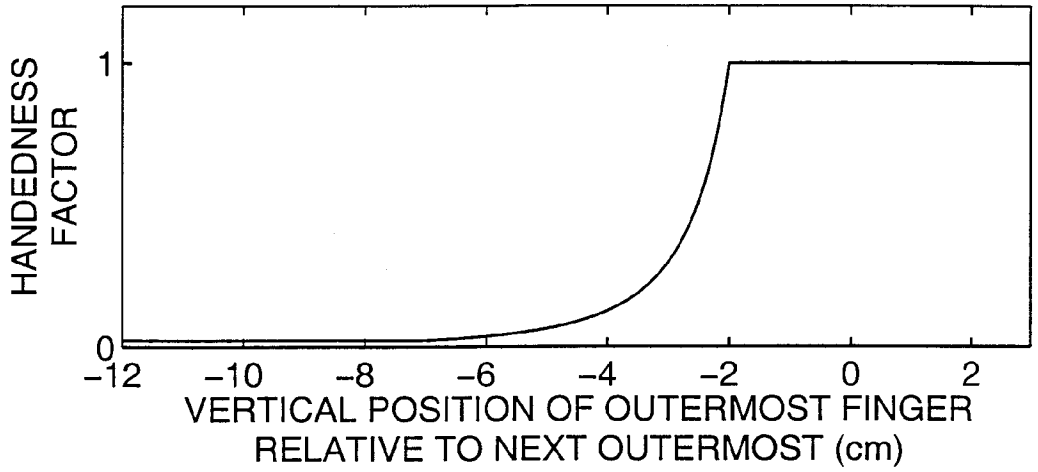


FIG. 31B

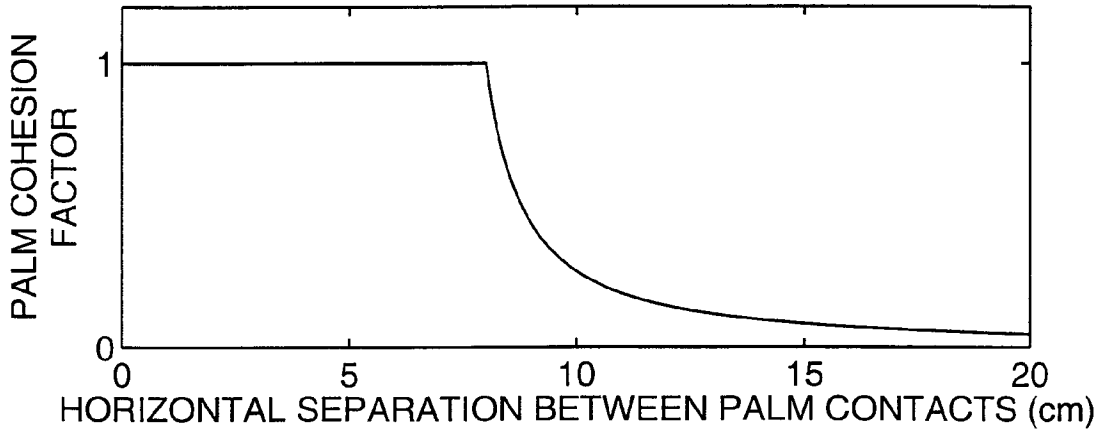


FIG. 31C

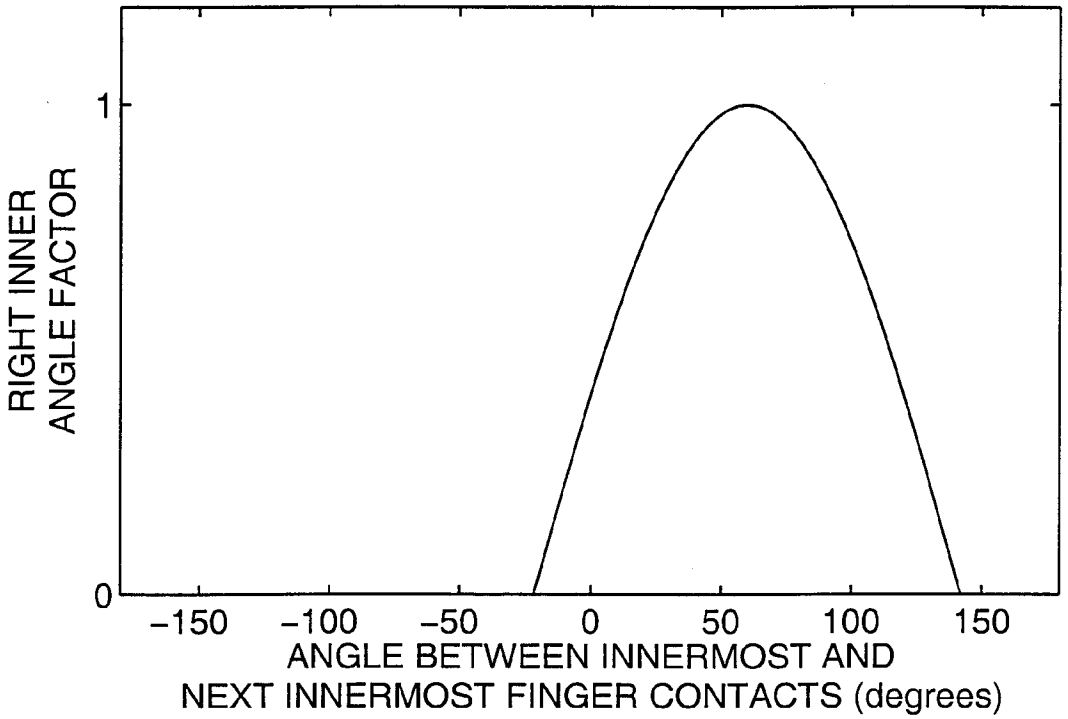


*FIG. 32*

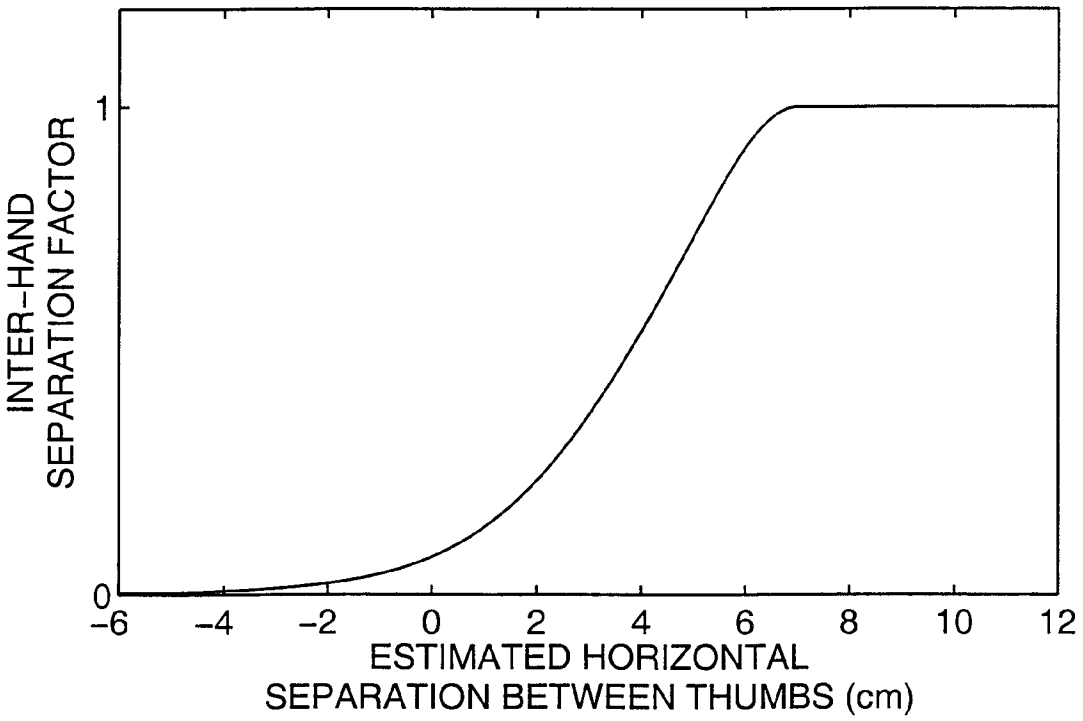


*FIG. 33*

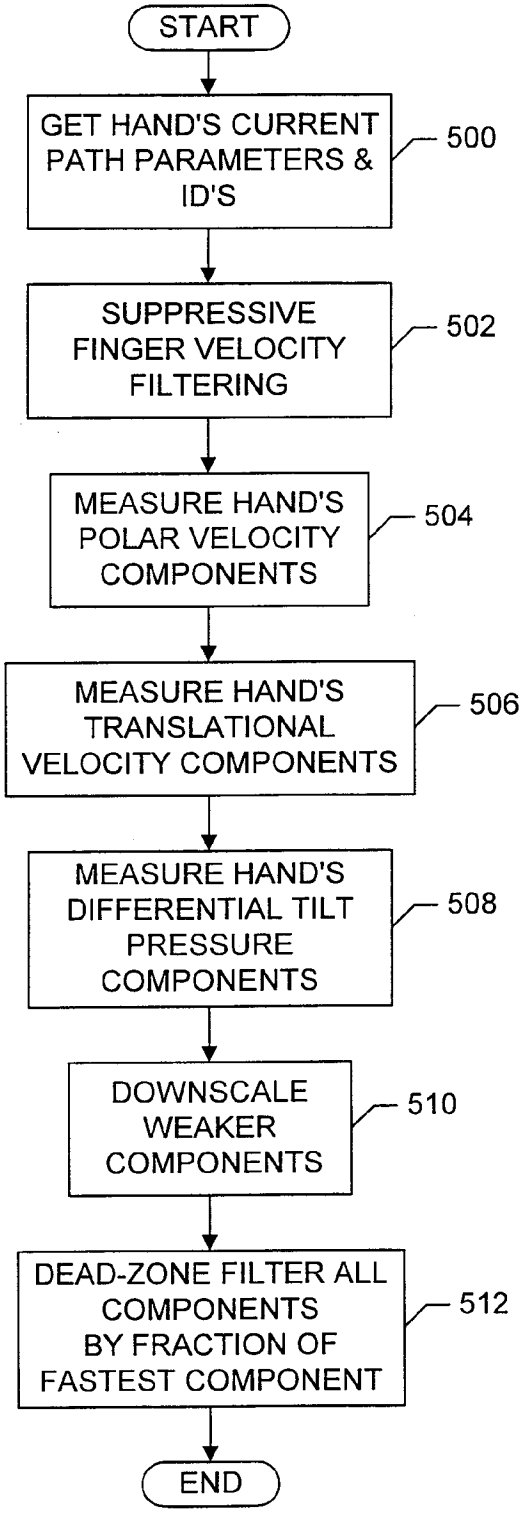


*FIG. 34*

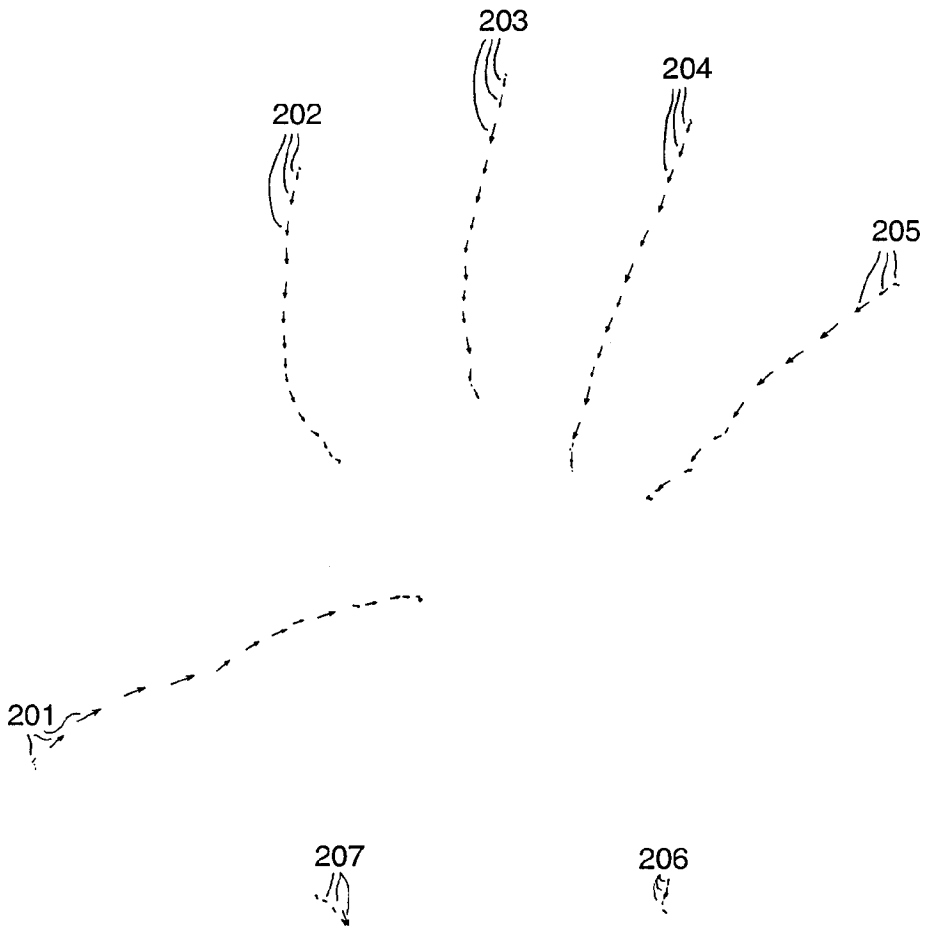


FIG. 35

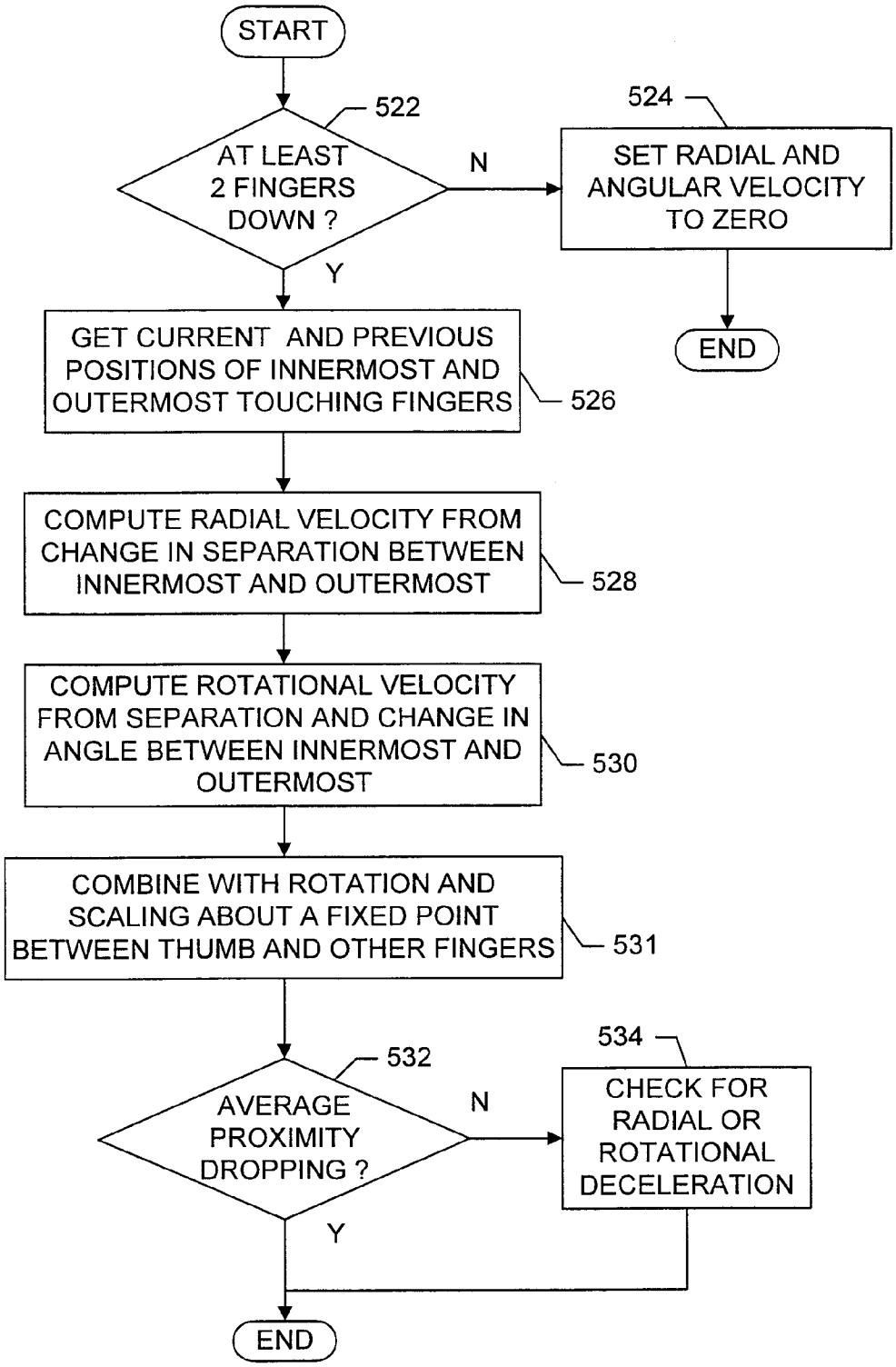
START
522
AT LEAST 2 FINGERS DOWN ?
N
524
SET RADIAL AND ANGULAR VELOCITY TO ZERO
END
Y
GET CURRENT AND PREVIOUS POSITIONS OF INNERMOST AND OUTERMOST TOUCHING FINGERS
526
COMPUTE RADIAL VELOCITY FROM CHANGE IN SEPARATION BETWEEN INNERMOST AND OUTERMOST
528
COMPUTE ROTATIONAL VELOCITY FROM SEPARATION AND CHANGE IN ANGLE BETWEEN INNERMOST AND OUTERMOST
530
COMBINE WITH ROTATION AND SCALING ABOUT A FIXED POINT BETWEEN THUMB AND OTHER FINGERS
531
532
AVERAGE PROXIMITY DROPPING ?
N
534
CHECK FOR RADIAL OR ROTATIONAL DECELERATION
Y
END

*FIG. 36*

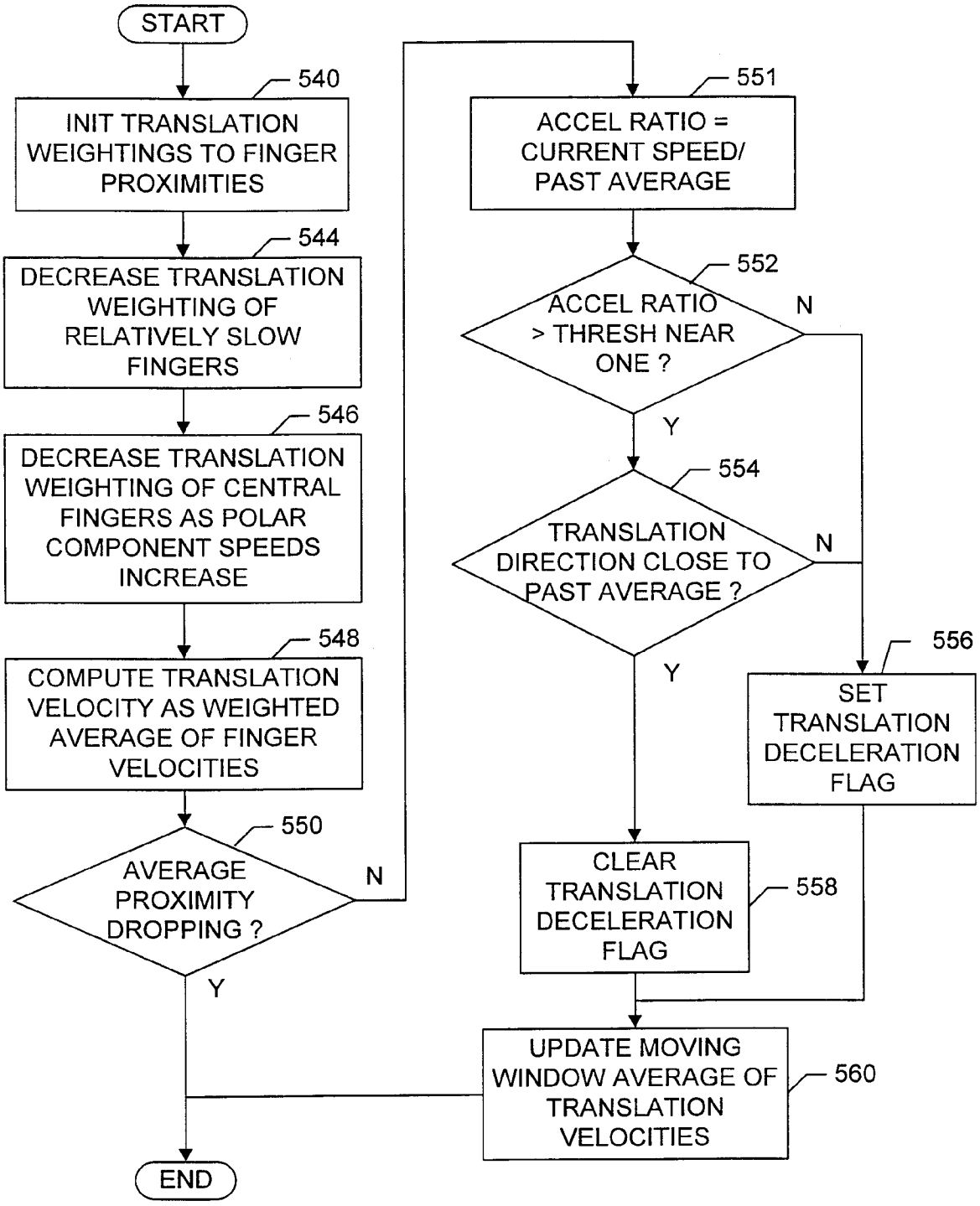
START
540
INIT TRANSLATION WEIGHTINGS TO FINGER PROXIMITIES
544
DECREASE TRANSLATION WEIGHTING OF RELATIVELY SLOW FINGERS
546
DECREASE TRANSLATION WEIGHTING OF CENTRAL FINGERS AS POLAR COMPONENT SPEEDS INCREASE
548
COMPUTE TRANSLATION VELOCITY AS WEIGHTED AVERAGE OF FINGER VELOCITIES
550
AVERAGE PROXIMITY DROPPING ?
N
Y
551
ACCEL RATIO = CURRENT SPEED/ PAST AVERAGE
552
ACCEL RATIO > THRESH NEAR ONE ?
N
Y
554
TRANSLATION DIRECTION CLOSE TO PAST AVERAGE ?
N
Y
556
SET TRANSLATION DECELERATION FLAG
CLEAR TRANSLATION DECELERATION FLAG
558
UPDATE MOVING WINDOW AVERAGE OF TRANSLATION VELOCITIES
560
END

FIG. 37

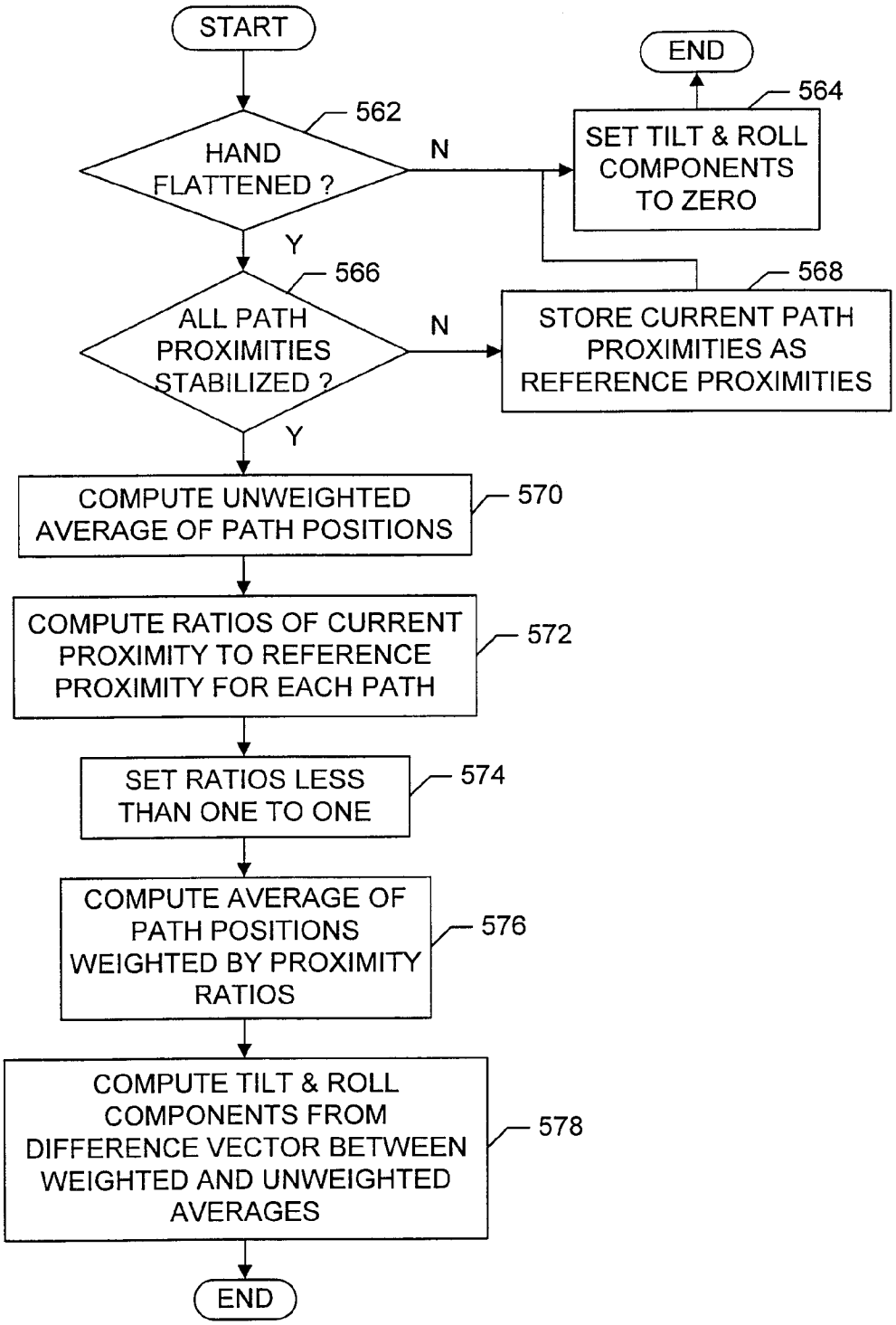


FIG. 38

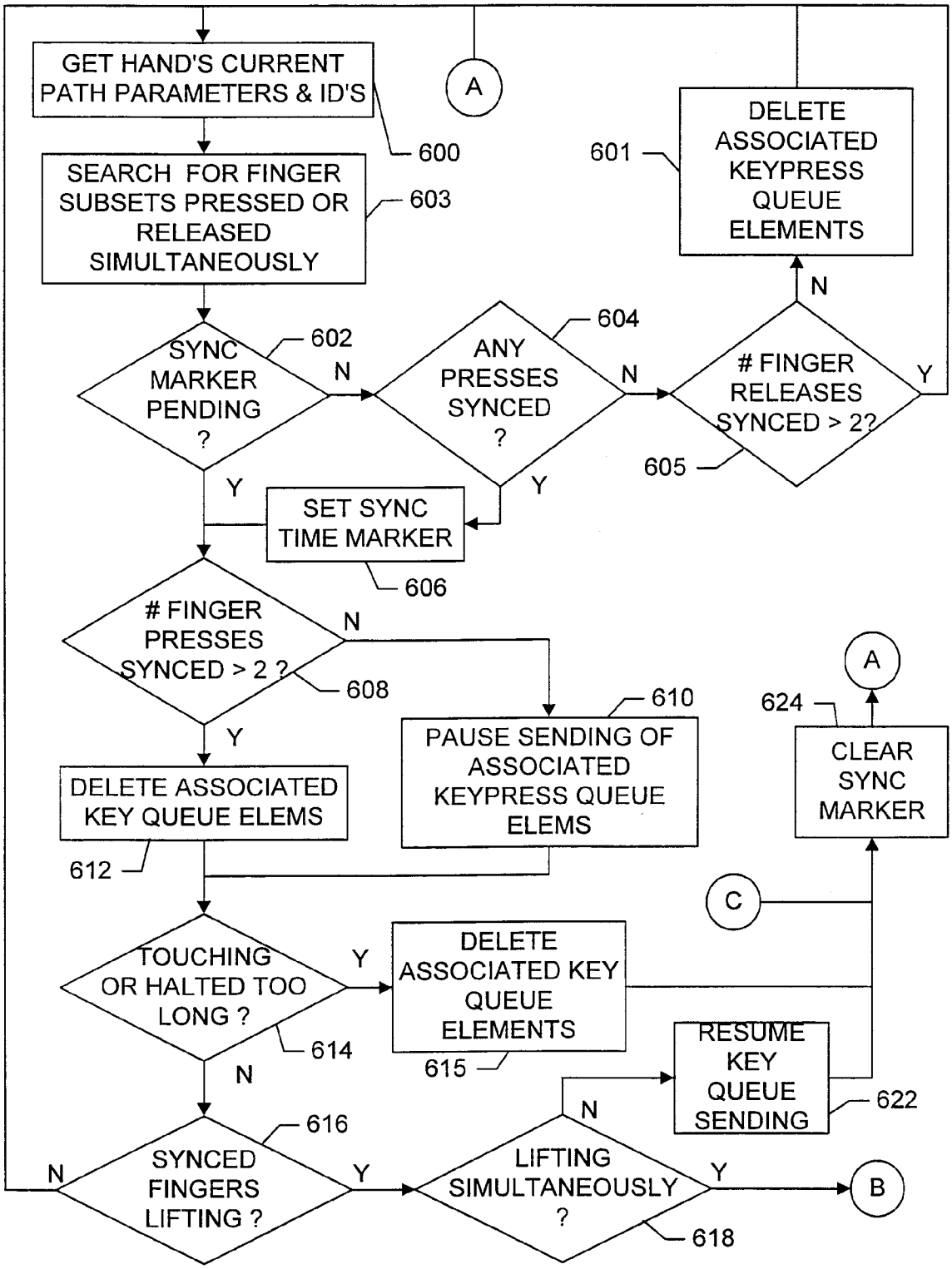


*FIG. 39A*

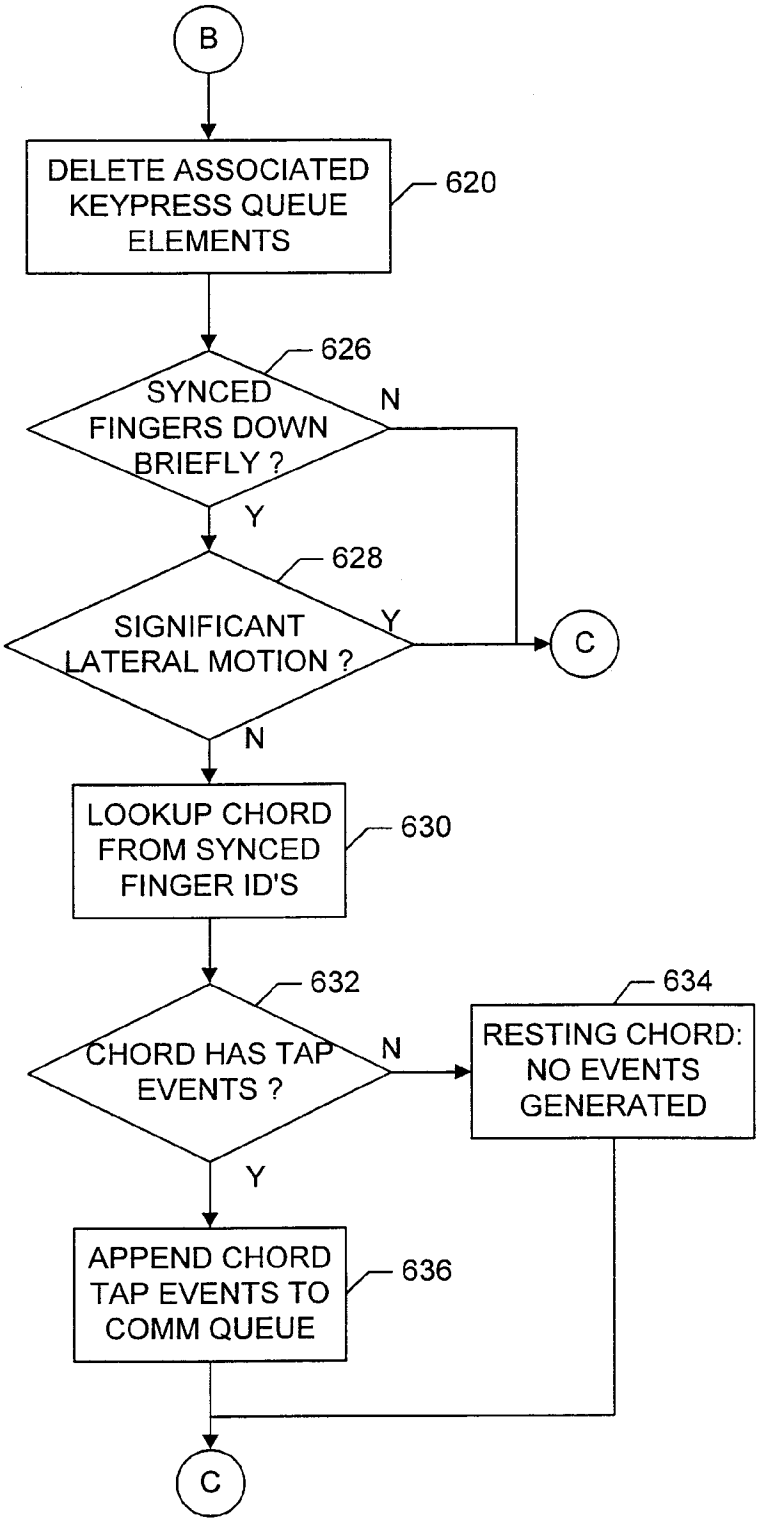


*FIG. 39B*

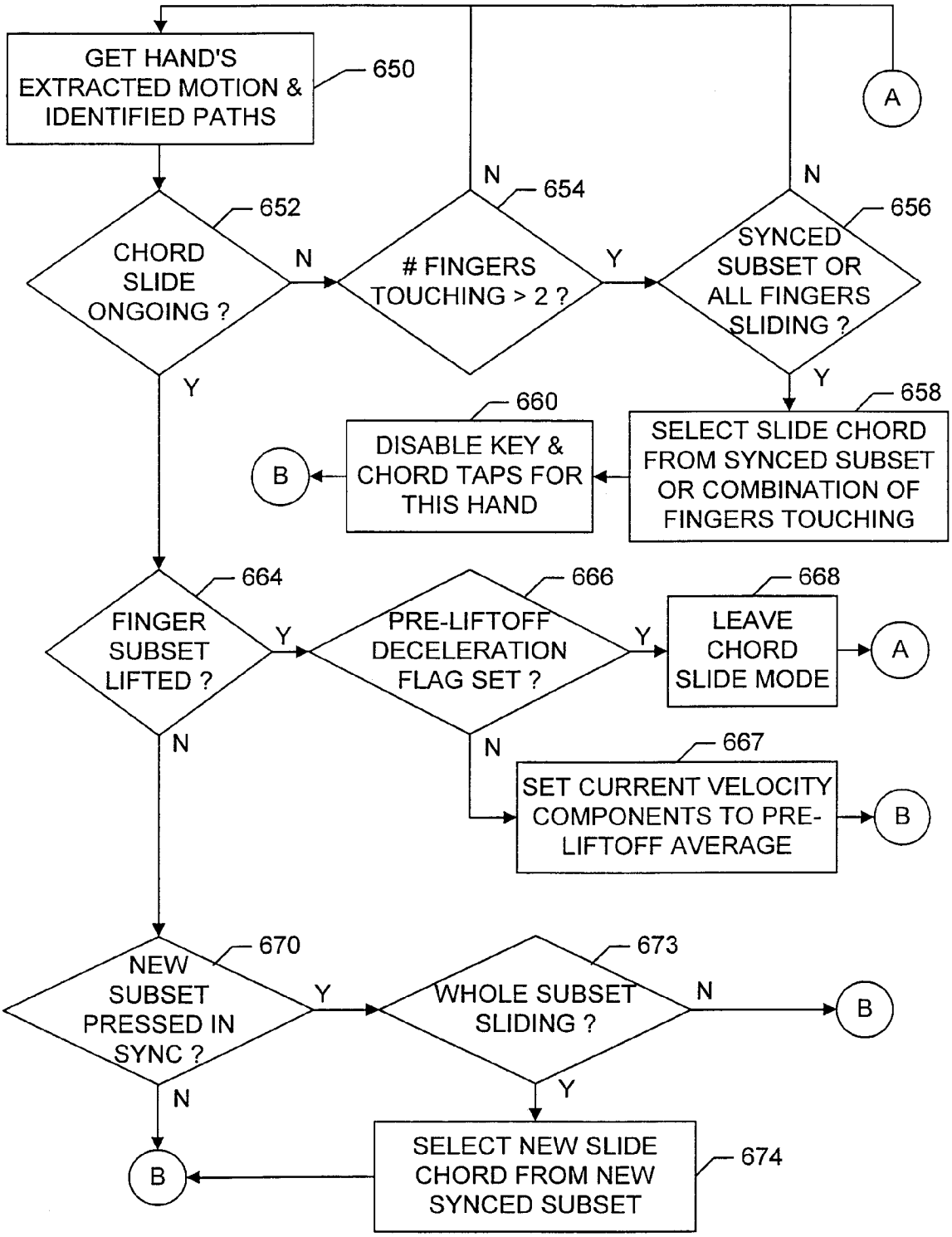


*FIG. 40A*

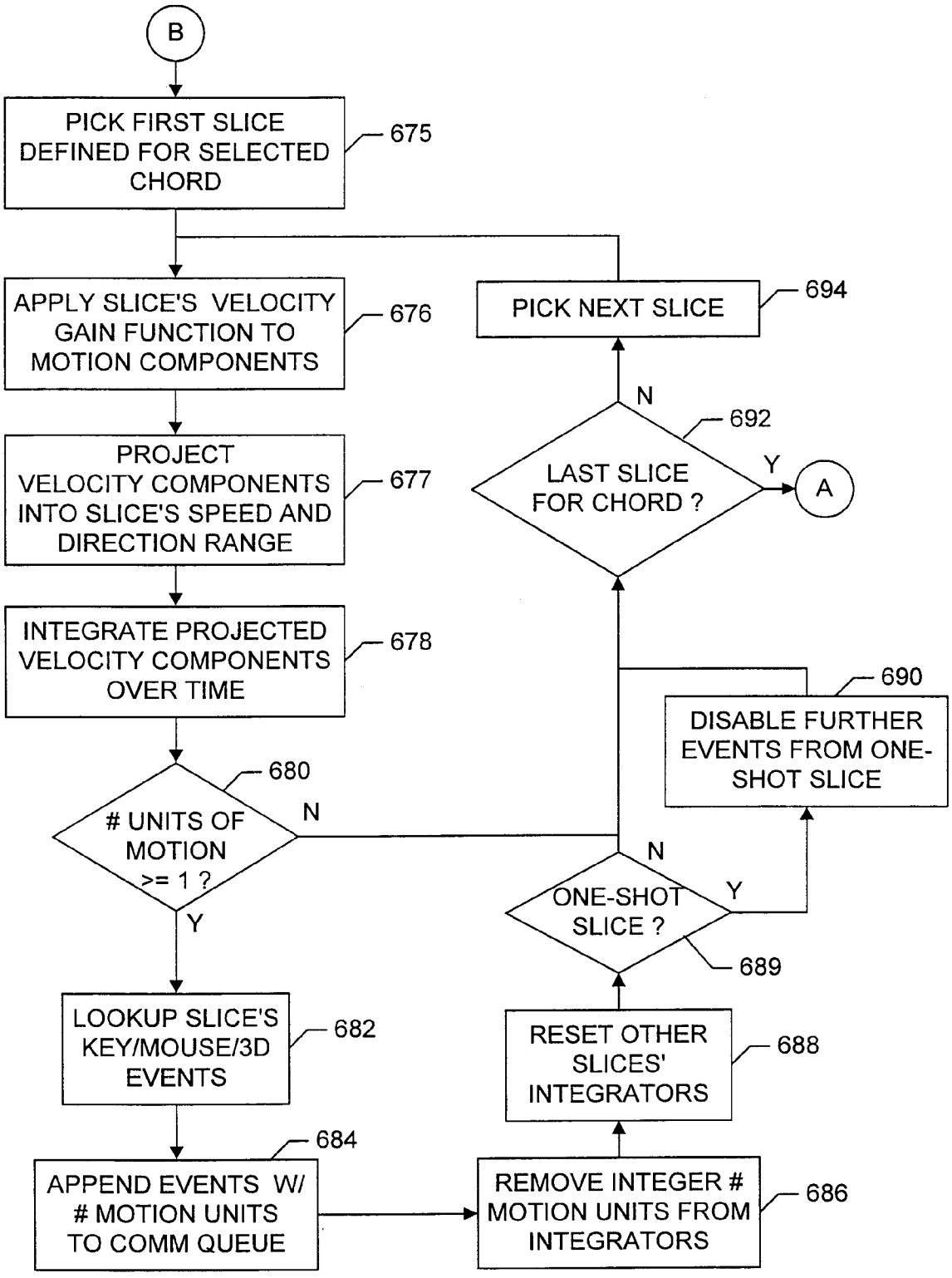


*FIG. 40B*

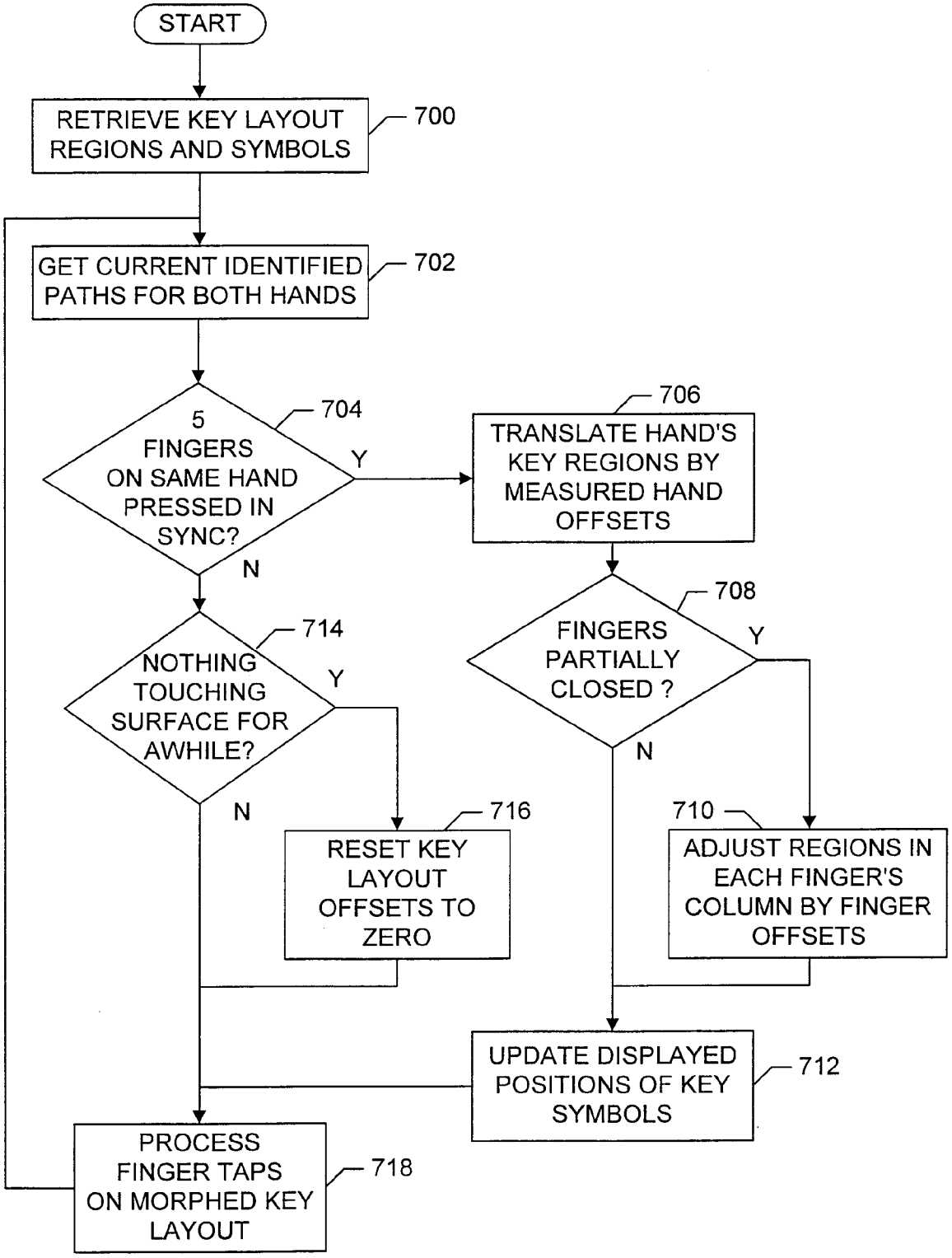


*FIG. 41*

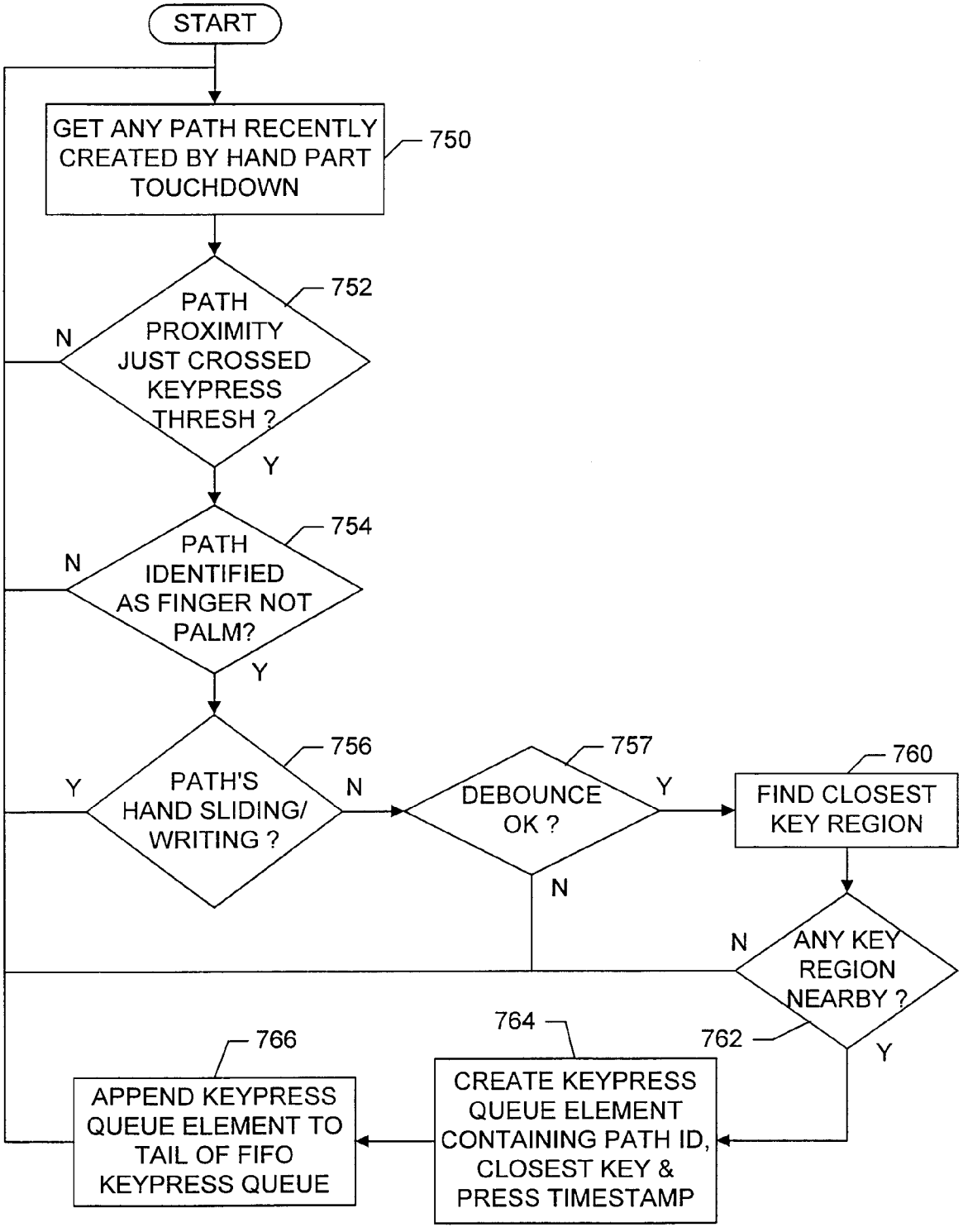


*FIG. 42*

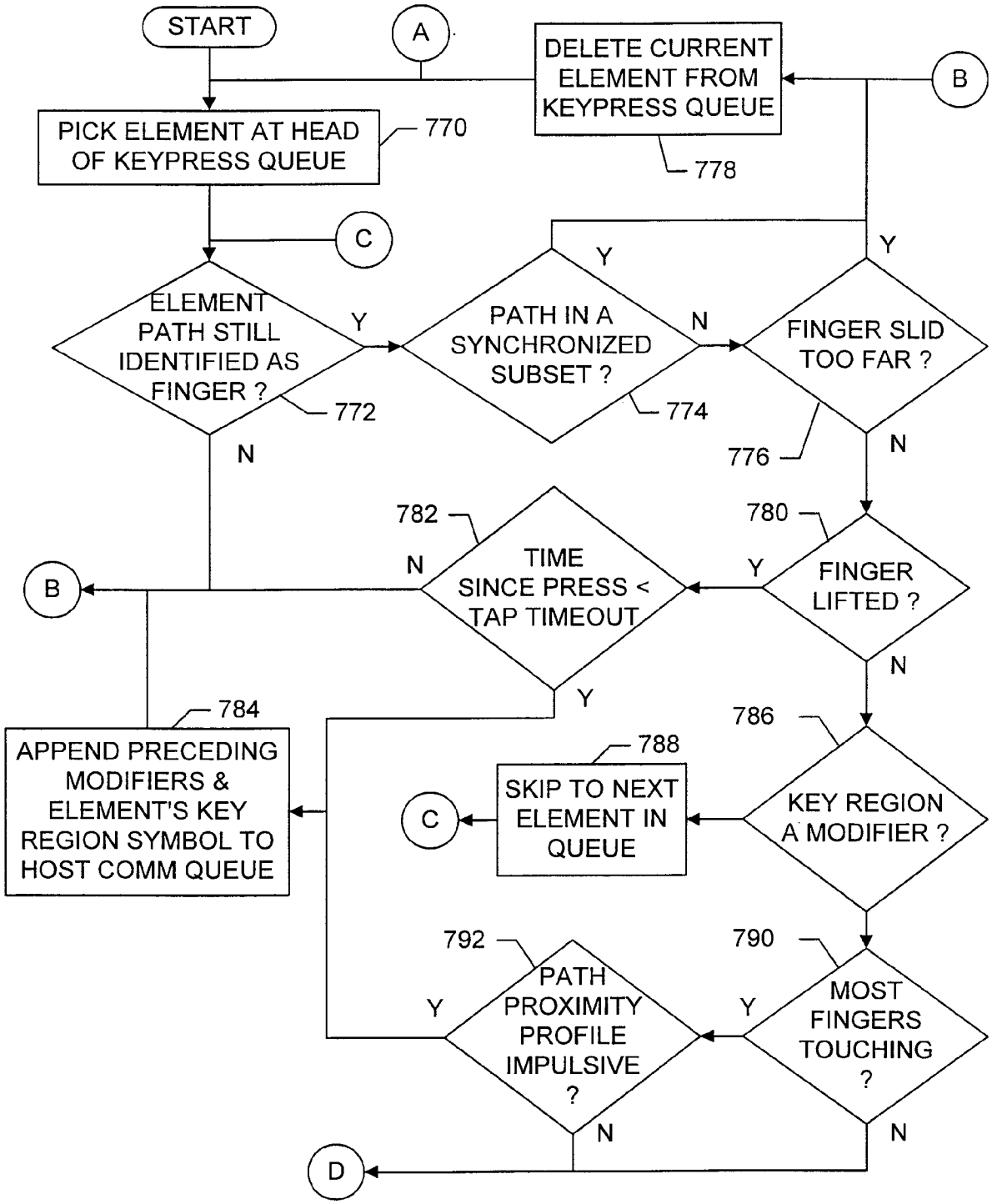


FIG. 43A

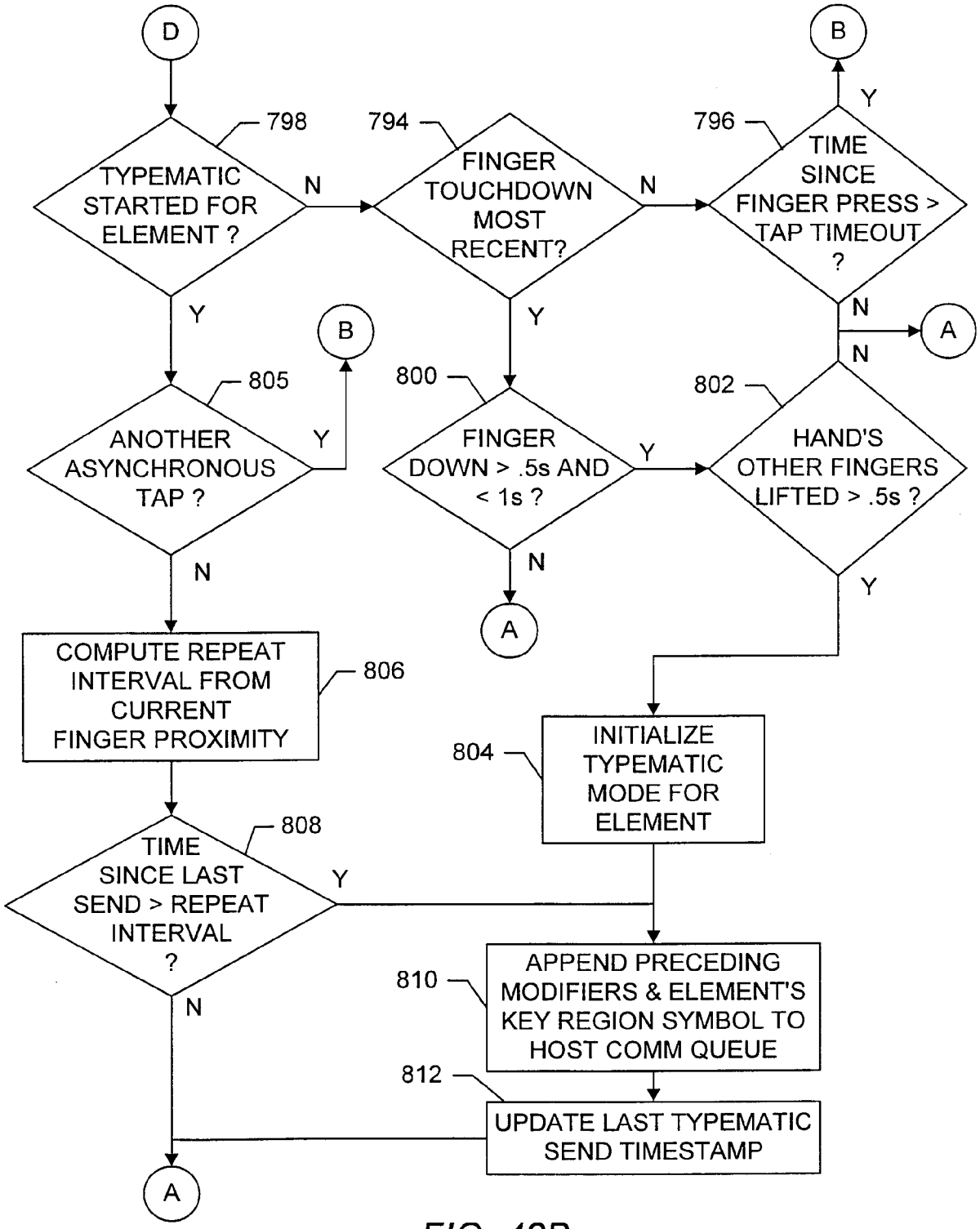


*FIG. 43B*

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/01454

**A. CLASSIFICATION OF SUBJECT MATTER**

IPC(6) :G09G 5/00
US CL :345/173

According to International Patent Classification (IPC) or to both national classification and IPC

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 345/173

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| Y | US 5,543,591 A (GILLESPIE ET AL) 06 August 1996, see abstract, column 5, lines 50-67, column 6, lines 1-5, 43-50, see figure 1. | 17, 22-24 |
| Y | US 5,305,017 A (GERPHEIDE et al) 19 April 1994, column 5, lines 12-50, see figure 1. | 17, 22-24 |
| A | US 5,563,632 A (ROBERTS) 08 October 1996, see abstract, column 3, lines 18-41, see figure 1. | 1-16, 18-21 |
| A,E | US 5,880,411 A (GILLESPIE et al) 09 March 1999, column 5, lines 29-67, column 6, lines 1-9, column 10, lines 18-37, see figure 1. | 25-121 |
| A | US 5,376,948 A (ROBERTS) 27 December 1994, see abstract and figure 1. | 1-16, 18-21 |

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

| | | | |
|---|---|---|---|
| * | Special categories of cited documents: | "T" | later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention |
| "A" | document defining the general state of the art which is not considered to be of particular relevance | "X" | document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone |
| "E" | earlier document published on or after the international filing date | "Y" | document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art |
| "L" | document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) | "&" | document member of the same patent family |
| "O" | document referring to an oral disclosure, use, exhibition or other means | | |
| "P" | document published prior to the international filing date but later than the priority date claimed | | |

| | |
|---|---|
| Date of the actual completion of the international search<br>29 MARCH 1999 | Date of mailing of the international search report<br>14 MAY 1999 |
| Name and mailing address of the ISA/US<br>Commissioner of Patents and Trademarks<br>Box PCT<br>Washington, D.C. 20231<br>Facsimile No. (703) 305-3230 | Authorized officer<br>RONALD LANEAU James R. Matthews<br>Telephone No. (703) 305-3973 |

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/01454

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A,P | US 5,821,930 A (HANSEN) 13 October 1998, see abstract. | 1-16, 18-21, 25-121 |

8221 Pattern Recognition Letters
14(1993)August, No.8, Amsterdam, NL


Pattern Recognition Letters 14 (1993) 625–630
North-Holland

August 1993

PATREC 1094

# A hashing-oriented nearest neighbor searching scheme

Chin-Chen Chang

*Institute of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 601, ROC*

Tzong-Chen Wu

*Department of Information Management, National Taiwan Institute of Technology, Taipei, Taiwan 107, ROC*

Received 25 February 1992

*Abstract*

Chang, C.-C. and T.-C. Wu, A hashing-oriented nearest neighbor searching scheme, Pattern Recognition Letters 14 (1993) 625–630.

This paper presents a hashing-oriented nearest neighbor searching scheme. Given $n$ points in the Euclidean two-dimensional plane, we first construct a Voronoi diagram and record the Voronoi vertices and the Voronoi edges. By passing each Voronoi vertex, we use two perpendicular lines, one is horizontal and the other is vertical, to partition the plane into some rectangular subdivisions. Here, each rectangular subdivision dominates at most two given points. Then we establish two hashing functions corresponding to horizontal slabs and vertical slabs, respectively. By applying the established hashing functions, we can quickly determine the proper rectangular subdivision containing the query point. After that, we compare the distances between the query point and the dominated points to determine the nearest neighbor. The searching time by our scheme is O(1). The preprocessing time and the amount of required storage are $O(n^2 + t)$, respectively, where $n$ is the number of given points and $t$ is the size of the hashing table needed by the established hashing functions.

*Keywords.* Nearest neighbor searching, computational geometry, Voronoi diagram, hashing.

## 1. Introduction

We are given $n$ points $P_i$, for $i=1,2,\ldots,n$, in the Euclidean two-dimensional plane. Let $Q$ be a query point. We want to find a point $P_k$ such that the distance between $Q$ and $P_k$ is the minimum. The problem of finding such $P_k$ is known as the nearest neighbor searching for $Q$. For convenience, we use 'the plane' or '$E^2$' instead of 'the Euclidean two-dimensional plane' throughout this paper.

The nearest neighbor searching problem is frequently encountered in many applications, such as pattern recognition and database similarity retrieval [5]. There are several known algorithms for solving the nearest enighbor searching problem [4, 6]. An intuitive method is to compute all distances of $(Q, P_i)$'s and then find the $P_k$ with the minimum distance. Clearly, this intuitive method requires $O(n)$ time to compute all distances of $(Q, P_i)$'s for each query.

*Correspondence to:* Tzong-Chen Wu, Department of Information Management, National Taiwan Institute of Technology, Taipei, Taiwan 107, ROC.

14(1993)August, No.8, Amsterdam, NL



Shamos and Hoey [7] pointed out that the construction of the Voronoi diagram can be used for solving the nearest neighbor searching problem. Let $D(p,q)$ be the Euclidean distance between points $p$ and $q$. The Voronoi polygon associated with $P_i$ is defined as

$$VP_i = \{x \in E^2 \mid D(x,P_i) \leqslant D(x,P_j), \text{ for } i \neq j\}.$$

The Voronoi diagram is the network of all Voronoi polygons associated with these given points. Figure 1 shows the Voronoi diagram on nine points in the plane. A Voronoi polygon is either bounded or unbounded. The vertices of the Voronoi diagram are referred to as Voronoi vertices, and its line segments are called Voronoi edges. From the definition, a Voronoi edge is the perpendicular bisector of two nearest neighbor points and a Voronoi vertex is the intersection of two Voronoi edges.

By the divide-and-conquer approach, a Voronoi diagram can be constructed in $O(n \log n)$ time, and this is optimal [6]. Moreover, the Voronoi diagram has the following two properties:

(1) If a point $Q$ is in $VP_i$, then $P_i$ is the nearest neighbor of $Q$.

(2) A Voronoi diagram on $n$ points has at most $2n-5$ Voronoi vertices and $3n-6$ Voronoi edges, for $n \geqslant 3$.

By the above properties, a constructed Voronoi diagram is well suitable for solving the nearest neighbor searching problem. Shamos and Hoey had also shown [7] that once the Voronoi diagram on the given $n$ points is constructed, the nearest neighbor searching problem can be performed in $O(\log n)$ time, using $O(n)$ storage and $O(n \log n)$ preprocessing time, which is optimal.

In this paper, we shall propose a hashing-oriented nearest neighbor searching scheme. On the given points in the plane, we first construct a Voronoi diagram and record the Voronoi vertices. Based on the slabbing method proposed by Shamos [6], we establish two hashing functions corresponding to $x$-coordinates and $y$-coordinates, respectively. Then we can perform the nearest neighbor searching in $O(1)$ time by the established hashing functions. Before describing our scheme, we first briefly review previous related works in the next section.

## 2. Previous related works

Given $n$ points in the plane. Shamos [6] proposed an elegant algorithm, known as the slabbing method, for solving the nearest neighbor searching problem. His algorithm has two stages. One is the preprocessing stage, the other is the query stage. In the proprocessing stage, we first construct a Voronoi diagram on the given points. Then we partition the plane into parallel slabs associated with every Voronoi vertex. Following the Voronoi diagram shown in Figure 1, Figure 2 shows the partitioned result by the slabbing method. In the query stage, we first find the slab to which the query point belongs by using a binary search. When the proper slab has been found, we perform a local search in

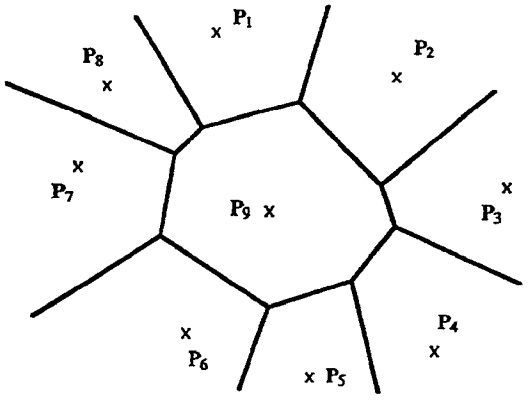


Figure 1. A Voronoi diagram on nine points.

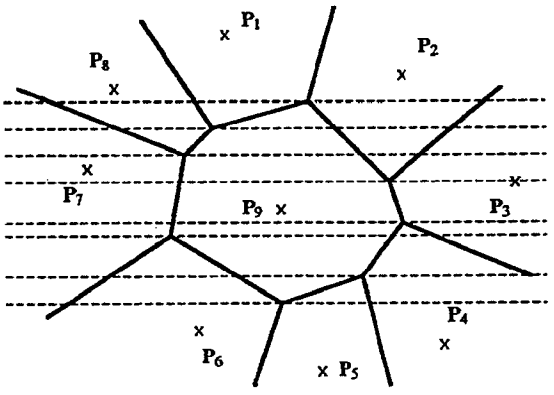


Figure 2. The partitioned result by the slabbing method.

8221 Pattern Recognition Letters
14(1993)August, No.8, Amsterdam, NL



this slab to determine which Voronoi polygon contains the query point. Once the proper Voronoi polygon is determined, the nearest neighbor of the query point is determined consequently. The slabbing method needs $O(n^2)$ preprocessing time, including $O(n \log n)$ time for constructing the Voronoi diagram and $O(n^2)$ time for partitioning the plane into slabs, and $O(n^2)$ storage. The query time requires $O(\log n)$, including $O(\log n)$ for the global search of slabs and $O(\log n)$ for the local search in the proper slab.

Later, Chang and Lin [2] proposed a method, called the double slabbing method, to improve the Shamos method. By passing each Voronoi vertex, we use two perpendicular lines, one is the horizontal line and the other is the vertical line, to partition the plane into some rectangular subdivisions. The partitioned result by using the double slabbing method is as shown in Figure 3. As pointed out in [2], each rectangular subdivision intersects at most one Voronoi edge. That is, each rectangular subdivision dominates two points at most. And one of the dominated points is the nearest neighbor to the query point contained in the rectangular subdivision. Consequently, once the proper rectangular subdivision containing the query point is determined, we can report the nearest neighbor of the query point in $O(1)$ time. However, searching the proper subdivision needs $O(\log n)$ time by conducting a binary search. Also, the double slabbing method needs $O(n^2)$ time and $O(n^2)$ storage in the preprocessing stage. The query time is $O(\log n)$.

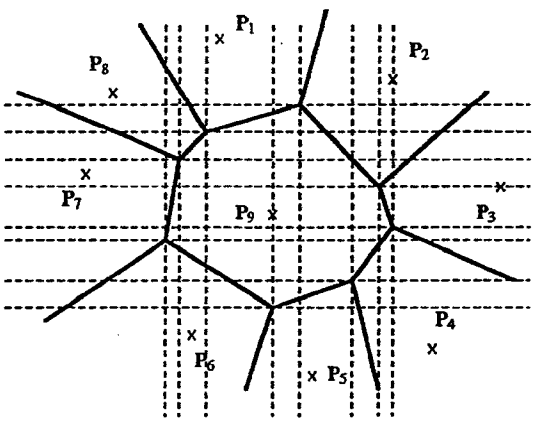


Figure 3. Partitioned result by the double slabbing method.

It is known that hashing is a simple and fast searching technique. The merit of using hashing functions in searching problems is that once the hashing functions are established, it takes little time to compute the hashing value for reporting the query [1]. Shen and Lee [8] first applied the hashing technique to solve the nearest neighbor searching problem. In their method, the plane is first divided into equal rectangular subdivisions such that each subdivision contains an almost constant number of points. For instance, Figure 4 illustrates the partitioned subdivisions, denoted as $D_i$, for $i = 1, 2, \ldots, 9$.

Since the plane is divided into rectangular subdivisions with equal sizes, we can easily determine which subdivision contains the query point by using simple hashing functions corresponding to the $x$-coordinates and $y$-coordinates, respectively. Suppose that the query point $Q$ is hashed into the subdivision $D_9$. Here, we only can say that the given points in $D_9$ are 'maybe' the nearest neighbor to $Q$. To confirm the answer, we need a backtracking searching and perform additional comparisons in the neighboring subdivisions, say $D_1, D_2, \ldots, D_8$, to find the 'exact' nearest neighbor to $Q$. That is, we need to search exchaustively in each possible subdivision. Figure 5 shows a backtracking path for searching the possible subdivisions. The backtracking searching is very inefficient and consumes much computation time.

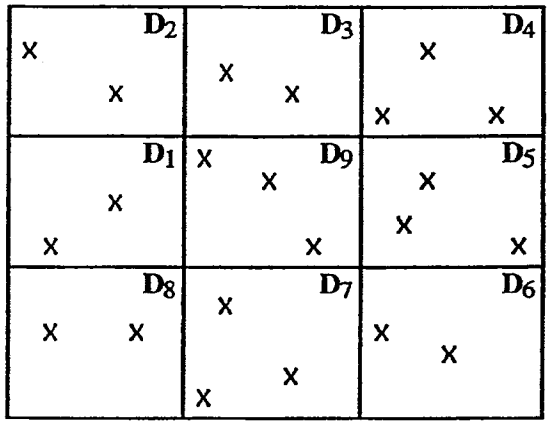


Figure 4. The partitioned result by Shen and Lee's method.

14(1993)August, No.8, Amsterdam, NL



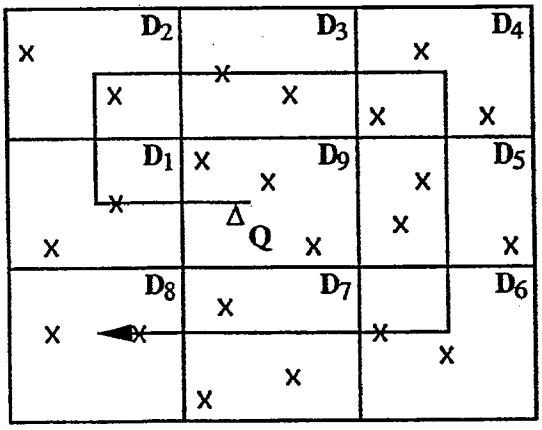


Figure 5. Backtracking path.

## 3. Our scheme

Suppose that there are $n+2$ contiguous intervals $[-\infty, x_0), [x_0, x_1), \ldots, [x_{n-1}, x_n), [x_n, \infty)$ in a straight line, as shown in Figure 6. Without loss of generality, we assume that $x_0 < x_1 < \cdots < x_n$. Let $[-\infty, x_0)$ be the 0th interval, $[x_n, \infty)$ be the $(n+1)$th interval, and $[x_{i-1}, x_i)$ be the $i$th interval for $i = 1, 2, \ldots, n-1$. Let $S(x_{i-1}, x_i)$ be the length (or distance) of the interval $[x_{i-1}, x_i)$, i.e., $S(x_{i-1}, x_i) = x_i - x_{i-1}$, and $s_{\min} = \min\{S(x_{i-1}, x_i)\}_{i=1,\ldots,n}$. Given a number $r$ in the straight line, we want to establish a hashing function that can quickly report the proper interval containing $r$.

For the construction of the hashing function, an auxiliary storage $AS$ is used. We first allocate $t$ contiguous memory space for the auxiliary storage $AS$, where $t = \lceil (x_n - x_0)/s_{\min} \rceil$, where $\lceil\ \rceil$ is a ceiling function. For simplicity, we call the $i$th location in $AS$ as $AS(i)$. Next, we determine the values contained in $AS$. If $x_0 + j \times s_{\min} < x_j$, then we store $i$ in $AS(j)$, for $j = 1, 2, \ldots, t-1$. For the location $AS(t)$, the value $(n+1)$ is stored in it. Here, $AS(j)$'s store the indices of the intervals. After that, a simple hashing function corresponding to the intervals is established as below:

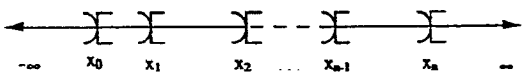


Figure 6. The intervals in a straight line.

(i) If $r < x_0$, then $r$ is in the 0th interval.

(ii) If $r \geq x_n$, then $r$ is in the $(n+1)$th interval.

(iii) If $r \geq x_{A(h(r))-1}$, then $r$ is in the $A(h(r))$th interval else $r$ is in the $(A(h(r))-1)$th interval, where $h(r) = \lceil (r - x_0)/s_{\min} \rceil$.

We call this scheme the interval-based hashing scheme. Figure 7 illustrates the hashing function for intervals.

In the following, we begin to describe our scheme for solving the nearest neighbor searching problem. Our scheme is divided into two stages. One is the preprocessing stage and the other is the query stage. The preprocessing stage is stated as follows.

**Preprocessing Stage**

*Input.* $n$ points.

*Output.* A hashing function for $x$-coordinates and a hashing function for $y$-coordinates.

*Step 1.* Construct the Voronoi diagram on the given $n$ input points by a certain available known algorithm. Then record its Voronoi vertices and Voronoi edges. Let the Voronoi vertices be numbered from 0 to $k$.

*Step 2.* Use the double slabbing method to partition the plane into $(k+2)^2$ rectangular subdivisions. Then do the following:

(2.1) Record the dominated points for each subdivision.

(2.2) Record $(k+2)$ intervals $[-\infty, x_0), [x_0, x_1), \ldots, [x_{k-1}, x_k), [x_k, \infty)$ for all the horizontal slabs.

(2.3) Record $(k+2)$ intervals $[-\infty, y_0), [y_0, y_1), \ldots, [y_{k-1}, y_k), [y_k, \infty)$ for all the vertical slabs.

*Step 3.* Construct two hashing functions corre-

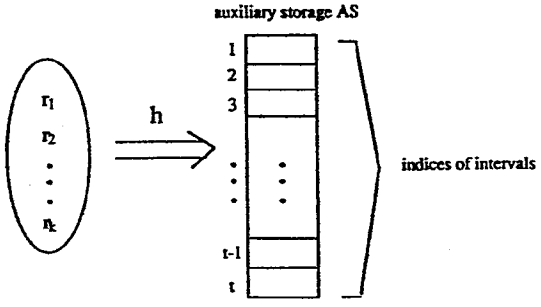


Figure 7. The hashing function for intervals.

8221 Pattern Recognition Letters
14(1993)August, No.8, Amsterdam, NL



sponding to horizontal slabs and vertical slabs by the interval-based hashing scheme, respectively.

*Step 4.* Output the rectangular subdivisions with their dominated points.

*Step 5.* Output the hashing function for horizontal slabs: $h_x(r)=\lceil (r-x_0)/s_{x\min}\rceil$ as well as the hashing table $AS_x$, where $s_{x\min}$ is the minimum length of the lengths of $[x_{i-1},x_i)$'s.

*Step 6.* Output the hashing function for vertical slabs: $h_y(r)=\lceil (r-y_0)/s_{y\min}\rceil$ as well as the hashing table $AS_y$, where $s_{y\min}$ is the minimum length of the lengths of $[y_{i-1},y_i)$'s.

Once the hashing functions corresponding to horizontal slabs and vertical slabs are established, we can quickly report the nearest neighbor to a query point $Q$. Let $q_x$ be the $x$-coordinate and $q_y$ be the $y$-coordinate of $Q$. The query stage is stated in the following.

**Query Stage**

*Input.* A query point $Q$.

*Output.* The nearest neighbor of $Q$.

*Step 1.* Apply the hashing functions $h_x$ and $h_y$ established in the Preprocessing Stage to compute $h_x(q_x)$ and $h_y(q_y)$. Then determine the proper subdivision containing $Q$ according to $h_x(q_x)$ and $h_y(q_y)$.

*Step 2.* Retrieve the dominated points $P_{i1},P_{i2},\ldots,P_{ir}$ of the proper subdivision containing $Q$.

*Step 3.* Compute the distances $D(Q,P_{ij})$, for $j=1,2,\ldots,r$ and report the point $P_{ik}$ satisfying that $D(Q,P_{ik})$ is the minimum among $D(Q,P_{ij})$'s.

## 4. Analysis of our scheme

We first analyze the computational complexity of our scheme. In the preprocessing stage, the computational time heavily depends on Step 1, Step 2 and Step 3. As pointed out in [7], the Voronoi diagram can be constructed in $O(n\log n)$ time. Since there are at most $2n-5$ Voronoi vertices and $3n-6$ Voronoi edges, we need $O(n^2)$ time for partitioning the plane and recording the dominated points of subdivisions in Step 2. The construction of the hashing functions needs $O(t)$ time, where

$$t=\max\{\lceil (x_n-x_0)/s_{x\min}\rceil,\ \lceil (y_n-y_0)/s_{y\min}\rceil\}.$$

Thus, we require $O(n^2+t)$ time in the preprocessing stage. In the query stage, Step 1 and Step 2 can be completely performed in constant time. Consequently, we require only $O(1)$ time to report the nearest neighbor of a query point.

The amount of the required storage by our scheme is $O(n^2+t)$, including $O(n^2)$ storage for the rectangular subdivisions and their dominated points and $O(t)$ storage for the hashing tables. However, we can only use $\lceil \log(k+1)\rceil$ bits to store the index of the slabbing intervals, where $k$ is the number of Voronoi vertices.

## 5. Conclusions

By combining the double slabbing method and the hashing technique, we have proposed a scheme for solving the nearest neighbor searching problem. Basically, three measures are used to evaluate the nearest neighbor searching problem [3]:

(1) the searching time, that is, the operations required to report the query,

(2) the preprocessing time, that is, the operations required to construct the data structure postulated by the search algorithm, and

(3) the amount of storage required by the preprocessed data structure.

Our scheme requires $O(1)$ searching time for reporting the nearest neighbor to a query point. However, we need $O(n^2+t)$ preprocessing time and $O(n^2+t)$ auxiliary storage, where $t$ is dependent on the minimum length of the slabbing intervals.

Since the CPU is getting more and more fast and the memory storage is getting more and more cheap, the overhead of the preprocessing time and the required auxiliary storage by the hashing functions is tolerable in case that the minimum length of the slabbing intervals is not too small to produce a large $t$.

## References


[1] Aho, A.V., J.E. Hopcroft and J.D. Ullman (1974). *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, MA.

[2] Chang, C.C. and C.H. Lin (1985). A fast nearest neighbor search algorithm. *Proc. 1985 National Computer Symposium*, Taiwan, 625–635 (in Chinese).

[3] Lee, D.T. (1982). On *k*-nearest neighbor Voronoi diagrams in the plane. *IEEE Trans. Comput.* 31 (6), 478–487.

[4] Lee, D.T. and F.P. Preparata (1984). Computational geometry—a survey. *IEEE Trans. Comput.* 33 (12), 1072–1101.

[5] Preparata, F.P. and M.I. Shamos (1985). *Computational Geometry—An Introduction.* Springer, New York.

[6] Shamos, M.I. (1978). Computational Geometry, Ph.D. dissertation, Yale University, New Haven, CT.

[7] Shamos, M.I. and D. Hoey (1975). Closest-point problems. *Proc. 16th IEEE Annual Symposium on Foundations of Computer Science*, Berkeley, CA, Oct. 1975, 151–162.

[8] Shen, C.W. and R.C.T. Lee (1978). A nearest neighbor search technique with short zero-in time. *Proc. 1978 Internat. Computer Software and Applications Conf.*, Nov. 1978, 470–475.

ELSEVIER

Robotics and Autonomous Systems 19 (1997) 347–358

Robotics and Autonomous Systems

# Vision for man–machine interaction

James L. Crowley [1]

*IMAG Project PRIMA, Institut National Polytechnique de Grenoble, 46 Ave Félix Viallet, 38031 Grenoble, France*

**Abstract**

Computer vision provides a powerful tool for the interaction between man and machine. The barrier between physical objects (paper, pencils, calculators) and their electronic counterparts limits both the integration of computing into human tasks, and the population willing to adapt to the required input devices. Computer vision, coupled with video projection using low cost devices, makes it possible for a human to use any convenient object, including fingers, as digital input devices. In such an "augmented reality", information is projected onto ordinary objects and acquired by watching the way objects are manipulated. In the first part of this paper we describe experiments with techniques for watching the hands and recognizing gestures.

Vision of the face is an important aspect of human-to-human communication. We have been experimenting with the use of computer vision "watch the face". In the second part of this paper we describe techniques for detecting, tracking and recognizing faces. When combined with real time image processing and active control of camera parameters, these techniques can greatly reduce the communications bandwidth required for videophone and videoconference communications.

*Keywords:* Computer vision; Multi-modal man–machine interaction; Eigenfaces; Tracking; Principal components analysis

## 1. Vision and man–machine interaction

One of the effects of the continued exponential growth in available computing power has been an exponential decrease in the cost of hardware for real time computer vision. This trend has been accelerated by the recent integration of image acquisition processing hardware for multi-media applications in personal computers. Lowered cost has meant more widespread experimentation in real time computer vision, creating a rapid evolution in robustness and reliability and the development of architectures for integrated vision systems [9].

Man–machine interaction provides a fertile applications domain for this technological evolution. The barrier between physical objects (paper, pencils, calculators) and their electronic counterparts limits both the integration of computing into human tasks, and the population willing to adapt to the required input devices. Computer vision, coupled with video projection using low cost devices, makes it possible for human to use any convenient object, including fingers, as digital input devices. Computer vision can permit a machine to track, identify and watch the face of a user. This offers the possibility of reducing bandwidth for videotelephone applications, for following the attention of a user by tracking his fixation point, and for exploiting facial expression as an additional information channel between man and machine.

Traditional computer vision techniques have been oriented toward using contrast contours (edges) to describe polyhedral objects. This approach has proved

[1] E-mail: jlc@imag.fr.


*PII* S0921-8890(96)00061-9

fragile even for man-made objects in a laboratory environment, and inappropriate for watching deformable non-polyhedric objects such as hands as faces. Thus man–machine interaction requires computer vision scientists to "go back to basics" to design techniques adapted to the problem. The following sections describe experiments with techniques for watching hands and faces.

## 2. Watching hands: Gestures as an input device

Human gesture serves three functional roles [6]: semiotic, ergotic, and epistemic.

- The *semiotic* function of gesture is to communicate meaningful information. The structure of a semiotic gesture is conventional and commonly results from shared cultural experience. The good-bye gesture, the American sign language, the operational gestures used to guide airplanes on the ground, and even the vulgar "finger", each illustrates the semotic function of gesture.
- The *ergotic* function of gesture is associated with the notion of work. It corresponds to the capacity of humans to manipulate the real world, to create artifacts, or to change the state of the environment by "direct manipulation". Shaping pottery from clay, wiping dust, etc. result from ergotic gestures.
- The *epistemic* function of gesture allows humans to learn from the environment through tactile experience. By moving your hand over an object, you appreciate its structure, you may discover the material it is made of, as well as other properties.

All three functions may be augmented using an *instrument*: Examples include a handkerchief for the semiotic good-bye gesture, a turn-table for the ergotic shape-up gesture of pottery, or a dedicated artifact to explore the world (for example, a retro-active system such as the pantograph [18] to sense the invisible).

In human–computer interaction, gesture has been primarily exploited for its ergotic function: typing on a keyboard, moving a mouse, and clicking buttons. The epistemic role of gesture has emerged effectively from pen computing and virtual reality: ergotic gestures applied to an electronic pen, to a data-glove or to a body-suit are transformed into meaningful expressions for the computer system. Special purpose interaction languages have been defined, typically 2-D pen gestures as in the Apple Newton, or 3-D hand gestures to navigate in virtual spaces or to control objects remotely [2].

With the exception of the electronic pen and the keyboard both of which have their non-computerized counterparts, mouses, data-gloves, and body-suits are "artificial add-on's" that wire user down to the computer. They are not real end-user instruments (as a hammer would be), but convenient tricks for computer scientists to sense human gesture.

We claim that computer vision can transform ordinary artifacts and even body parts into effective input devices. Krueger's seminal work on the video place [13], followed recently by Wellner's concept of digital desk [22] show that the camera can be used as a non-intrusive sensor for human gesture. However, to be effective the processing behind the camera must be fast and robust. The techniques used by Krueger and Wellner are simple concept demonstrations. They are fast but fragile and work only within highly constrained environments.

We are exploring advanced computer vision techniques to non-intrusively observe human gesture in a fast and robust manner. In Section 2.2, we present FingerPaint, an experiment in the use of cross-correlation as a means of tracking *natural pointing* devices for a digital desk. By "natural pointing device", we mean a bare finger or any real world artifact such as a pen or an eraser.

### 2.1. Projecting the workspace

In the digital desk a computer screen is projected onto a physical desk using a video projector, such as a liqiud-crystal "data-show" working with standard overhead projector. A video camera is set up to watch the workspace such that the surface of the projected image and the surface of the imaged area coincide. This coincidence cannot match "pixel to pixel" unless the camera and projector occupy the same physical space and use the same optics. Since this is impossible, it is necessary to master the transformation between the real workspace, and the imaged area. This transformation is a mapping between two planes.

The projection of a plane to another plane is an affine transformation. Thus the video projector can be used to project a reference frame onto the physical desk in the form of a set of points. The camera image

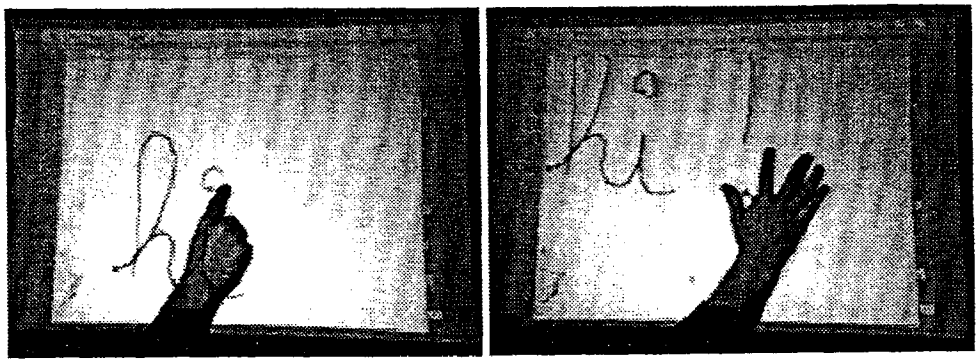

Fig. 1. Drawing and placing with "fingerPaint" (from [3]).

of these four points permits the calibration of six coefficients $(A, B, C, D, E, F)$ which transform image coordinates $(i, j)$ to workspace coordinates $(x, y)$.

$$x = Ai + Bj + C, \qquad y = Di + Ej + F.$$

The visual processes required for the digital desk are relatively simple. The basic operation is tracking of a pointing device such as finger, a pencil or an eraser. Such tracking should be supported by methods to determine what device to track and to detect when tracking has failed. A method is also required to detect the equivalent of "mouse-down" and "mouse-up" events.

The tracking problem can be expressed as: "Given an observation of an object at time $t$, determine the most likely position of the same object at time $t+\Delta T$". If different objects can be used as a pointing device, then the system must include some form of "trigger" which includes presentation of the pointing device to the system. The observation of the pointing device gives a small neighborhood, $w(n, m)$, of an image $p(i, j)$. This neighborhood will serve as a "reference template". The tracking problem can then be expressed: given the position of the pointing device in the $k$th image, determine the most likely position of the pointing device in the $(k+1)$th image. The size of the tracked neighborhood must be determined such that the neighborhood includes a sufficiently large portion of the object to be tracked with a minimum of the background.

We have experimented with a number of different approaches to track pointing devices: these include color, correlation tracking, principal components and active contours (snakes) [3,5]. The active contour model [12] presented problems which we believe can be resolved, but which will require additional experiments. Our current demonstration uses cross-correlation and principal components analysis.

### 2.2. FingerPaint

As a simple demonstration, we constructed a program called "FingerPaint".[2] Fingerpaint runs on an Apple Quadra AV/840 and uses a workspace projected with an overhead projector using a liquid-crystal display "data-show". A CCD camera with an 18 mm lens observes this workspace and provides visual input. "Finger-down" and "finger-up" events are simulated using the space bar of the keyboard but they could be sensed using a microphone attached to the surface of the desk. As illustrated in Fig. 1, any "natural pointing device" such as finger can be used to draw pictures and letters, or to move a drawing.

The image at time $(k+1)\Delta T$ to be searched will be noted as $p_{k+1}(i, j)$. The search process can generally be accelerated by restricting the search to a region of this image, denoted $s(i, j)$, and called a "Region of Interest". Our system uses a square search region of size $M \times M$ centered on the location where the reference template was detected in the previous image.

The robustness of the tracking system is reasonable but, as discussed below, its performance is inadequate with respect to Fitt's law [6]. Preliminary experiments with local users indicate, however, that the current

[2] The FingerPaint system has been implemented by Francois Berard.

performance is acceptable for investigation purposes. In addition, the widespread availability of image acquisition and processing hardware adequate for real time correlation should alleviate our current performance problem. Most importantly, this demonstration has permitted us to explore the problems involved in watching the gesture.

### 2.3. Correlation as a tracking technique for the digital desk

The basic operation for a digital desk application is tracking some pointing such as a finger, a pencil or an eraser. The tracking problem can be expressed as: "Given an observation of an object at time $t$, determine the most likely location of the same object at time $t + \Delta T$". The pointing device can be modeled as a reference template. The reference template is a small neighborhood, i.e., a window $w(m, n)$ of a picture $p(i, j)$ obtained at some prior time $t$. The reference template is compared to an image neighborhood $(i, j)$, by computing a sum of squared differences (SSD) between the $N \times N$ template and the neighborhood of the image whose upper left corner is at $(i, j)$.

$$\mathrm{SSD}(i, j) = \sum_{m=0}^{N} \sum_{n=0}^{N} (p_k(i + m, j + n) - w(m, n))^2.$$

An SSD value is computed for each pixel, $(i, j)$, within the $M \times M$ search region. A perfect match between the template and the neighborhood gives an SSD value of 0. Such a perfect match is rarely obtained because of differences and appearance due to lighting and other effects. These effects can be minimized by normalizing the energy in the template and the neighborhood.

Completing the squares of the SSD equation gives three terms, which can be written as

$$\mathrm{SSD}(i, j) = E_p^2(i, j) + E_W^2 - 2\langle p_k(i + mj + n), w(m, n)\rangle.$$

The term $\langle p_k(i+m, j+n), w(m, n)\rangle$ is the inner product of the template $w(m, n)$ with the neighborhood $p_k(i, j)$. The term $E_p^2(i, j)$ represents the energy in the image neighborhood and $E_W^2$ is the energy in the reference window. The neighborhood and reference window may be normalized by dividing by $E_P^2(i, j)$ and $E_W^2$, to give a normalized cross-correlation (NCC):

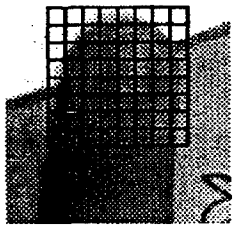

Fig. 2. Reference template for a finger.

$$\mathrm{NCC}(i, j) = \frac{\langle p_k(i + m, j + n)w(m, n)\rangle}{E_P^2(i, j)E_W^2}.$$

NCC produces a peak with a value of 1.0 at a perfect match between window and neighborhood, and is relatively robust in the presence of noise, changes in scale and gray level, and image deformations [16]. Hardware exists for computing an NCC at video rates. However, in software, it is more efficient to use SSD.

Implementing cross-correlation by SSD requires solving practical problems such as determining the sizes of the reference template and of the search region, triggering and breaking tracking, and updating the reference template.

### 2.4. The size of the reference template

The size of the reference template must be determined such that it includes a sufficiently large portion of the device to be tracked and a minimum of the background. If the template window is too large, correlation may be corrupted by the background. On the other extreme, if the template is composed only of the interior of the pointing device then the reference template will be relatively uniform, and a high correlation peak will be obtained with any uniform region of the image, including other parts of the pointing device. For a reasonable correlation peak, the reference template size should be just large enough to include the boundary of the pointing device, which contains the information used for detection and localization.

In FingerPaint, our workspace is of size 40 cm × 32 cm. This surface is mapped onto an image of 192 × 144 pixels, giving pixel sizes of 2 mm × 2.2 mm. At this resolution, a finger gives a correlation template of size 8 × 8 pixels or 16 mm × 18 mm, as shown in Fig. 2.

### 2.5. The size of the search region

The size $M$ of the search region depends on the speed of the pointing device. Considerations based on Fitt's law indicate a need for tracking speeds of up to 180 cm/s. To verify this, we performed an experiment in which a finger was filmed making typical pointing movements in our workspace. The maximum speed observed in this experiment was $V_m = 139$ cm/s. Expressed in pixels this gives $V_m = 695$ pixels per second.

Given an image processing cycle time of $\Delta T$ seconds and a maximum pointer speed of $V_m$ pixels per second, it is possible to specify that the pointing device will be found within a radius of $M = \Delta T V_m$ pixels of its position in the previous frame. For images of 192 × 144 pixels, our built-in digitizer permits us to register images at a maximum frame rate of 24 frames per second, giving a cycle time of $\Delta T_{max} = 41.7$ ms. This represents an upper limit on image acquisition speed which is attainable only if image tracking were to take no computation time.

The computational cost of cross-correlation is directly proportional to the number of pixels in the search region. Reducing the number of pixels will decrease the time needed for the inner loop of correlation by the same amount. This, in turn, increases the number of times that correlation can be operated within a unit time, further decreasing the region over which the search must be performed. Thus there is an inverse relation between the width of the search region, $M$, and the maximum tracking speed, $V_m$. The smaller the search region, the faster the finger movement that can be tracked, up to a limit set by digitizing hardware.

The fastest tracking movement can be expected at a relatively small search region. This is confirmed by experiments. To verify the inverse relation between $M$ and $V_m$, we systematically varied the size of the search region from $M = 10$ to $M = 46$ pixels and measured the cycle time that was obtained. The maximum speed of 126 pixels per second is obtained with $M = 26$. Although this is 5.5 times less than the maximum desirable speed (i.e., 695 pixesl per second), the system is quite usable to perform drawing and placements in a "natural" way.

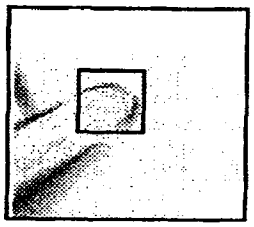

Fig. 3. Temporal difference of images in the reference square.

### 2.6. Triggering and breaking tracking

When tracking is not active, the system monitors an $N \times N$ pixel "tracking trigger", $T_k(i, j)$, located in the lower right corner of the workspace. As each image is acquired at time $k$, the contents of this tracking trigger are subtracted from the contents at the previous image $k - 1$. This creates a difference image as shown in Fig. 3. The energy of the difference image is computed as

$$E_k = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} (T_k(m, n) - T_{k-1}(m, n))^2.$$

When a pointing device enters the tracking trigger, the energy rises above a threshold. In order to assure that the tracking device is adequately positioned, the system waits until the difference energy drops back below the threshold before acquiring the reference template. At that point, the contents of the tracking trigger, $T_k(m, n)$, are saved as a reference image, and the tracking process is initiated.

Tracking continues as long as the minimum value of SSD remains below a relatively high threshold. However, it can happen that the tracker locks on to a pattern on the digital desk (for example a photo of the pointing device!). To cover this eventuality, if the tracked location of the pointer stops moving for more than a few seconds (say 10), the system begins again to observe the difference energy in the tracking trigger. If the trigger energy rises above threshold, the tracker will break the previous track and re-initialize the reference pattern with the new contents of the tracking trigger.

### 2.7. Updating the reference mask

As the user moves the pointing device around the workspace, there is a natural tendency for the device

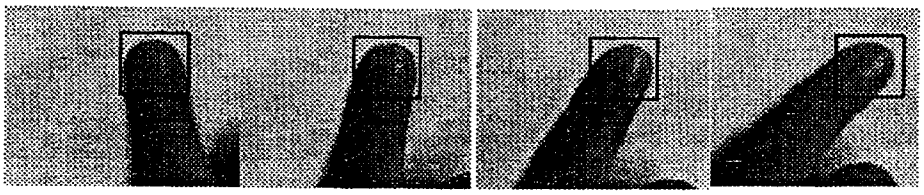

Fig. 4. Change in reference template as a function of finger orientation.

to rotate, as shown in Fig. 4. This, in turn, may cause loss of tracking. In order to avoid loss of tracking, the smallest SSD value from each search is compared to a threshold. If the smallest SSD rises above this threshold, then the reference template is updated using the contents of the image at time $k-1$ at the detected position.

Tracking fingertips is an example of a simple fast visual process which can be used to change the nature of the interaction between man and machine. Vision can also be used to make the machine aware of the user by detecting, tracking and watching his face.

## 3. Faces: Detecting, tracking and watching the user

Face to face communication plays an important role in human to human communication. Thus it is natural to assume that an important quantity of non-verbal information can be obtained for man–machine interaction by watching faces. However, even more than hands, face interpretation poses difficult problems for established machine vision techniques. In this section we briefly report on experiments with simple techniques for detecting, tracking and interpreting images of faces. The key to robustness in such tracking and interpretation is the integration of complementary techniques.

### *3.1. Why watch a face?*

Detection and interpretation of a face image can have a number of applications in machine vision. The most obvious use is to know whether a person is present in front of a computer screen. This makes a cute, but very expensive, screen saver. It is also possible to use face recognition as a substitute for a login password, presenting a person with his preferred workspace as soon as he appears in front of the computer system. Slightly more practical is the use of computer vision to watch the eyes and lips of a user. Eye-tracking can be used to determine whether the user is looking at the computer screen and to which part his fixation is posed. This could conceivably be used to activate the currently active window in an interface. Observing the mouth to detect lip movements can be used to detect speech acts in order to trigger a speech recognition system.

None of the above uses would appear to be compelling enough to justify the cost of a camera and digitizer. However, there is an application for which people are ready to pay the costs of such hardware: video communications. Recognizing and tracking faces can have several important uses for the applications of video telephones and video conferencing. We are currently experimenting with combining face interpretation with a rudimentary sound processing system to determine the origin of spoken words and associate speech with faces. Each of the applications which we envisage require active computer control of the direction (pan and tilt) zoom (focal length), focus and aperture of a camera. Fortunately, such cameras are appearing in the market.

In the video-telephone application, we use an active camera to regulate zoom, pan, tilt, focus and aperture so as to keep the face of the user centered in the image at the proper size and focus, and with an appropriate light level. Such active camera control is not simply for esthetics. Keeping the face at the same place, same scale and same intensity level can dramatically reduce the information to be transmitted. One such possible coding is to define (on-line) a face space using principal components analysis (defined below) of the sample images from the last few minutes. Once the face basis vectors are transmitted, subsequent images can be transmitted as a short vector of face space coefficients. Effective use of this technique is only possible with

active camera control. Other image codings can also be accelerated if the face image is normalized in position, size, gray level and held in focus.

In the videoconference scenario, a passive camera with a wide-angle lens provides a global view of the persons seated around a conference table. An active camera provides a close-up of whichever person is speaking. When no one is speaking, and during transitions of the close-up camera, the wide-angle camera view can be transmitted. Face detection, operating on the wide-angle images, can be used to estimate the location of conference participants around the table. When a participant speaks, the high-resolution camera can zoom onto the face of the speaker and hold the face in the center of the image.

What are the technologies required for the above applications? Both scenarios require an ability to detect and track faces, and an ability to servo control pan, tilt, zoom, focus and aperture so as to maintain a selected face in the desired position, scale and contrast level. From a hardware standpoint, such an application requires a camera for which these axes can be controlled. Such camera heads are increasingly appearing in the market. For example, we have purchased a small RS232 controllable camera from a Japanese manufacturer which produces excellent color images for title more than the price of a normal color camera.

A second hardware requirement is the ability to digitize and process images at something close to video rates. The classic bottleneck here is communication of the image between the frame-grabber and the processor. Fortunately, the rush to multi-media applications has pushed a number of vendors to produce workstations in which a frame-grabber is linked to the processor by such a high speed bus. Typical hardware available for a reasonable cost permits acquisition of up to 20 frames per second at full image size and full video rates for reduced resolution images. Adding simple image processing can reduce frame rates to 2–10 Hz (depending on image resolution). Such workstations are suitable for concept demonstrations and for experiments needed to define performance specifications. An additional factor of 2 (18 months) in bandwidth and processing power will bring us to full video rates.

The questions we ask in the laboratory are: What software algorithms can be used for face detection, tracking and recognition, and what are the system's concepts needed to tie these processes together. System's concepts have been the subject of our ESPRIT Basic Research Project "Vision as Process", described in the book [9] or the paper [8] for more details.

### 3.2. Detection: Finding a face with color

One of the simplest methods for detecting pixels which might be part of a face is to look for skin color. Human skin has a particular hue and saturation. The intensity, however, will vary as a function of the relative direction to the illumination. Of course, the perceived hue and saturation will be the product of the color of the ambient light and the skin color [20].

We have found that candidate pixels for faces and hands can be detected very rapidly using a normalized color histogram. Histogram color can be normalized for changes in intensity by dividing the color vector by the luminance. This permits us to convert a color vector $[R, G, B]$ having three dimensions into a vector $[r, g]$ of normalized color having two dimensions. The normalized color histogram $H(r, g)$ provides a fast means of skin detection. The histogram is initialized by observing a patch of the skin and counting the number of times each normalized color value occurs. The histogram contains the number of pixels which exhibit a particular color vector $[r, g]$. Dividing by the total number of pixels in the histogram, $N$, gives the probability of obtaining a particular vector given that the pixel observes skin:

$$P(\text{color/skin}) = \frac{1}{N} H(r, g).$$

Bayes rule can be used to determine the probability of skin given the color has values $[r, g]$:

$$P(\text{skin/color}) = \frac{P(\text{skin})}{P(\text{color})} P(\text{color/skin}).$$

$P(\text{skin})$ can be taken as constant. $P(\text{color})$ is the global statistic for the vector $[r, g]$. In practice this ratio is often approximated by a constant. The result at each pixel is the estimate of the probability of skin. An example (unfortunately printed here in black and white) is shown in Figs. 5(a)–(d).

Histogram matching can provide a very fast indicator that a face is present at a part of an image. However, reliability requires that this information be confirmed

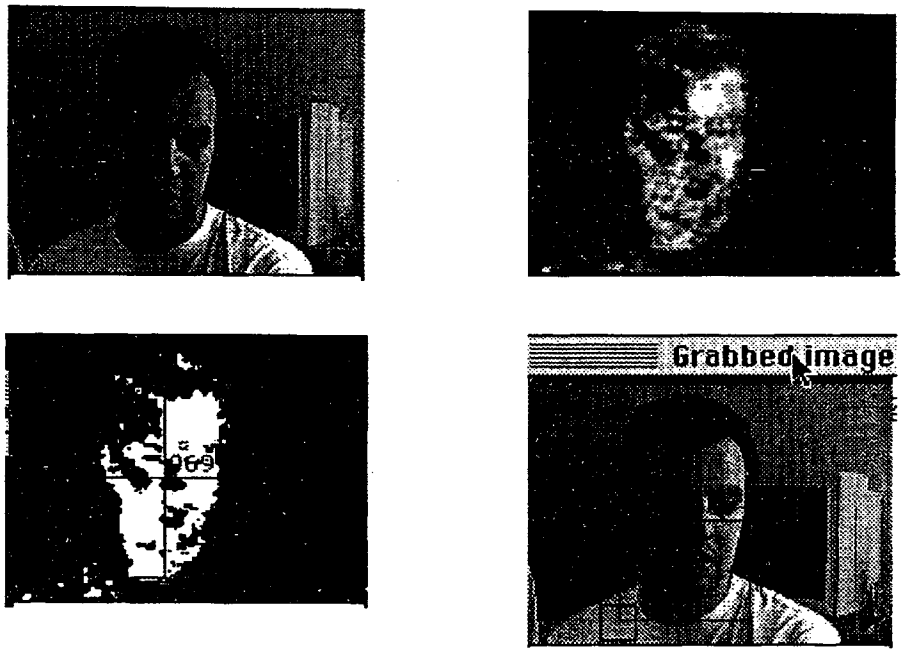


Fig. 5. (a) Black and white rendition of color image of Y.H. Berne. (b) Probability of skin in color image of Y.H. Berne. (c) Threshold skin probability with bounding box of connected components. (d) Bounding box for the face.

by another direction means. Such a means can be provided by a blink detector.

### 3.3. Finding a face by blink detection

A human must periodically blink to keep his eyes moist. Blinking is involuntary and fast. Most people do not notice when they blink. However, detecting a blinking pattern in an image sequence is an easy and reliable means to detect the presence of a face. Blinking provides a space–time signal which is easily detected and unique to faces. The fact that both eyes blink together provides a rendundance which permits blinking to be discriminated from other motions in the scene. The fact that the eyes are symmetrically positioned with a fixed separation provides a means to normalize the size and orientation of the head.

We have built a simple blink detector which works as follows: As each image is acquired, the previous image is subtracted. The resulting difference image generally contains a small boundary region around the outside of the head. If the eyes happen to be closed in one of the two images, there are also two small roundish regions over the eyes where the difference is significant.

The difference image is thresholded, and a connected components algorithm is run on the thresholded image. A bounding box is computed for each connected component. A candidate for an eye must have a bounding box within a particular horizontal and vertical size. Two such candidates must be detected with a horizontal separation of a certain range of sizes, and little vertical difference in the vertical separation. When this configuration of two small bounding boxes is detected, a pair of blinking eyes is hypothesized. The position in the image is determined from the center of the line between the bounding boxes. The distance to the face is measured from the separation. This permits to determine the size of a window which is used to extract the face from the image. This simple technique has proven quite reliable for determining the position and size of faces.

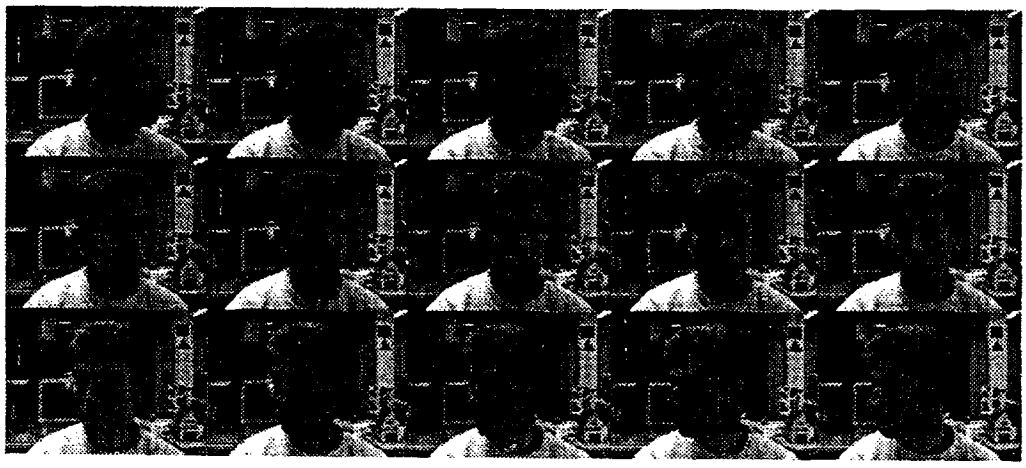

Fig. 6. Every fifth image from a sequence of 70 images of Jerome Martin turning.

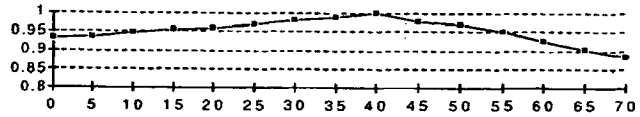


Fig. 7. Correlation values for correlation of face number 40 with the face images from Fig. 6.

### 3.4. Tracking the face by correlation (SSD)

Detection of a face by color is fast and reliable, but not always precise. Detection by blinking is precise, but requires capturing an image pair during a blink. Correlation can be used to complete these two techniques and to hold the face centered in the image as the head moves. The crucial question is how large a window to correlate ($N$) and over how large a search region to search.

We have found that a $7 \times 7$ search region extracted from the center of a face is sufficient to determine the precise location of the face in the next image. The size of the search region is determined by the speed with which a face can move between two frames. Calling on the finger tracking results above, we optimized the search region to maximize the frequency of image acquisition. With our current hardware this is provided by a search region to $26 \times 26$ pixels, but must be adjusted when several image processes are running in parallel.

Correlation can also be used in isolation to track a face. As a test of the robustness of correlation, we acquired a sequence of images of a head turning (shown in Fig. 6). We then correlated the center image from the sequence to produce the correlation graph shown in Fig. 7. The graph shows the results of zero-mean normalized correlation for the 40th face image with the other images in this set.

### 3.5. Identifying the face by eigenspace decomposition

The pixels which compose an image can be considered as very long vector. Thus, an image of a prototypical face can be considered as a basis for describing other images by inner product. A set of such images can form a space for describing a shape. For example, images number 0, 30 and 60 from Fig. 6 can be used to define a space of a face turning. A particular face image can be represented by the vector of three inner products obtained with these three images. It is necessary to compute this inner product at an appropriate

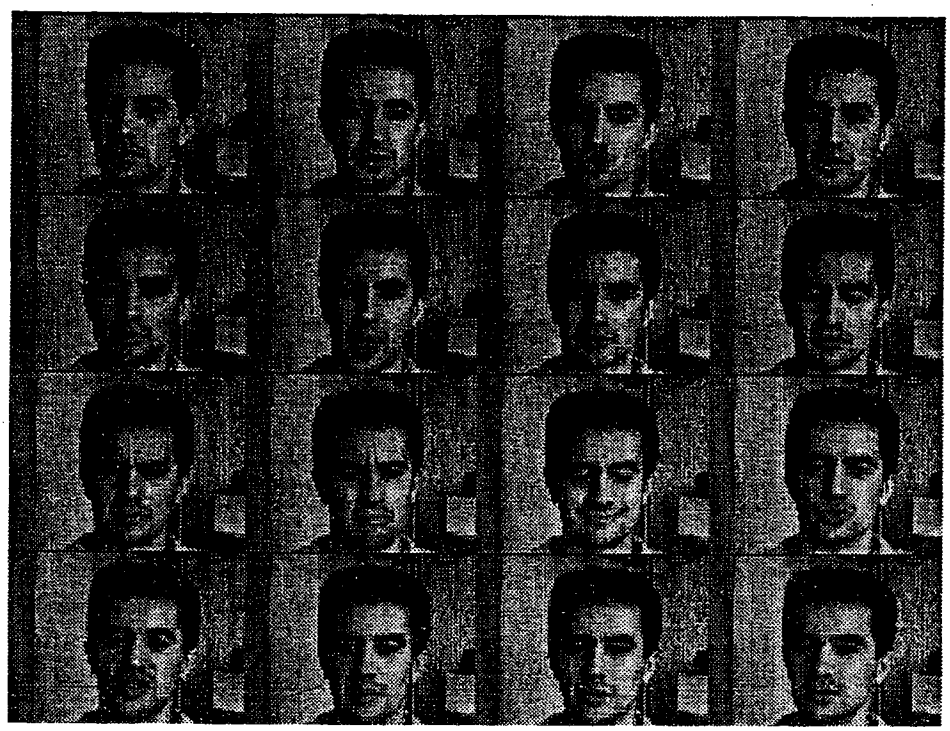

Fig. 8. A small face data base composed of 16 images.

location, but since correlation is a sequence of inner products, it is possible to find the peak correlation, and then describe the image by the vector of inner products obtained at that position.

The problem with this approach is that it can rapidly become expensive as the number of images increases. However, the image set can be reduced to minimal orthogonal basis set, and the correlation with this basis set used to describe the contents. This is the idea behind the eigenspace coding made popular by Turk and Pentland [21]. Correlation with a set of eigenimages is commonly thought of as a technique for recognition. However, it is also possible to use such a coding as a compact image transmission code provided that the position, scale and contrast are suitably normalized.

To construct an eigenspace, we begin with a database of images, as for example shown in Fig. 8. We then compute an average image as shown in Fig. 9. Finally, the technique of Turk [21] is used to compute the principal components of this space. The

Fig. 9. The average face from the face base in Fig. 8.

principal components of the covariance matrix form an orthogonal basis set, which are the axis of the eigenspace as shown in Fig. 10.

One of the simplest applications of the eigenfaces method is the recognition of a subject. We have

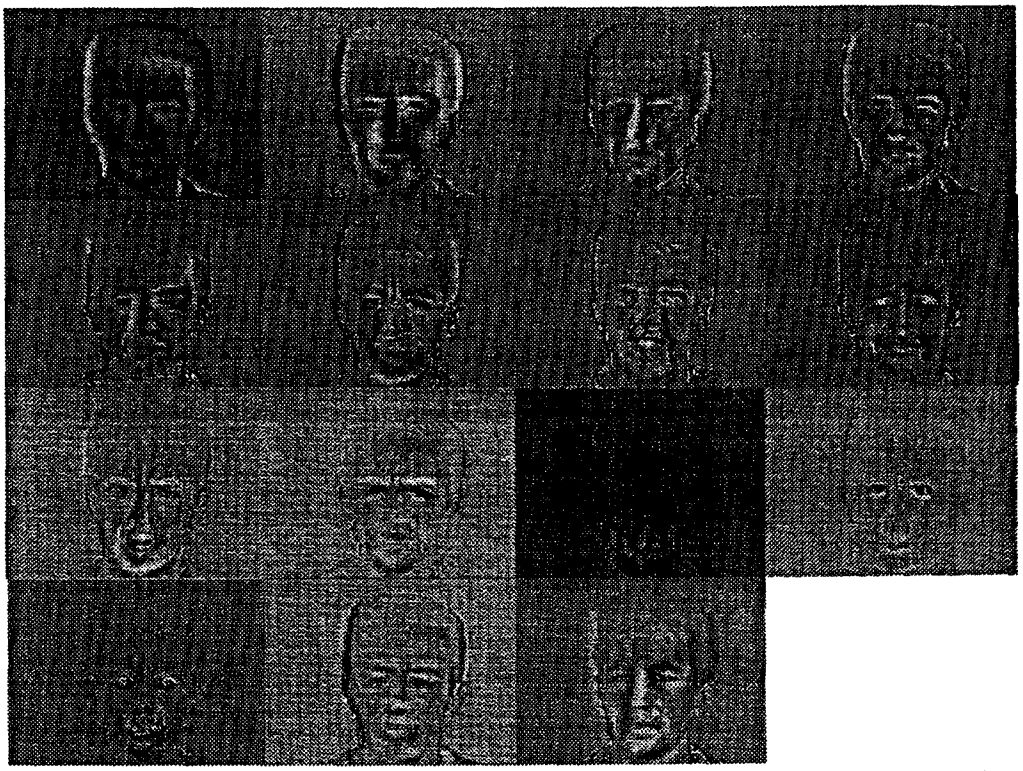

Fig. 10. The 15 principal component images from the face base in Fig. 8.

prepared a simple demo which works as follows. At the beginning of a session, the system classifies the subjects face in order to determine if the subject is known. Classification is performed by multiplying the normalized face by each of the principal component images in order to obtain a vector. The vector positions the image in the "face space" defined by the current eigenfaces. If the face is near a position of this space which corresponds to a known subject, then the subject's image from the face-space database is displayed. If the vector is not near a known subject, the subject is classified as unknown and no face is displayed. Using the eigenface technique, our Quadra 700 with no additional hardware can digitize and classify a face within a 108 × 120 image for a database of 12 images at about 1 frame per second.

It is possible to use the eigenface technique to measure parameters. One example of this is for eye-tracking. We train a set of images of the subject looking in different directions and use these images to form an eigenspace. During execution of task, a high-resolution window is placed over the subjects eyes, and the position in the eigenspace is compute. The nearest principal components are used to interpolate the current horizontal and vertical direction.

We are experimenting with this technique to determine the trade-off between resolution of the windows on the eyes, the number of eigen-images needed, and the precision which we can obtain in eye-tracking. The goal is to be able to drive a pointing device, such as a mouse with such eye-tracking.

Facial expression contains useful information about the user's state of mind. The eigenfaces principle can be easily extended to classifying the user's facial expression. A set of facial expressions are obtained as the subject performs his task. These facial expressions are then used to form an eigenspace. At each instant, the system determines the face expression class which

most closely corresponds to the user's current expression. In this way, we can experiment by anticipating the user's "mood" based on facial expression.

## 4. Conclusion

In this paper we have presented some preliminary results with detection and tracking of fingers and faces. We have also experimented with eye-tracking using snakes and other techniques based on signal processing, as well as face interpretation using eigenfaces. It is increasingly apparent that computer vision has the potential to provide a rich new source of interaction modes for computer–human interaction. Vision can make the machine "aware" of the user, his movements and environment, in ways that are yet to be invented.

## References


[1] P. Anandan, Measuring visual motion from image sequence, Ph. D. Dissertation and COINS Technical Report 87-21, University of Massachusetts, Amherst, 1987.

[2] T. Baudel and M. Beaudouin-Lafon, Charade: Remote control of objects using free-hand gestures, *Communications of the ACM* 36 (7) (1993) 28–35.

[3] F. Berard, Vision par ordinateur pour la réalité augmentée: Application au bureau numérique, Mémoire du D.E.A. en Informatique, Univeristé Joseph Fourier, 1994.

[4] A. Blake and M. Israd, 3D Position, attitude and shape input using video tracking of hands and lips, *ACM – SIGGRAPH Ann. Conf. on Computer Graphics* (1994).

[5] C. Cadoz, Les réalités virtuelles. Dominos, Flammarion, 1994.

[6] S.K. Card, T.P. Moran and N. Newell, *The Psychology of Human–Computer Interaction* (Lawrence Erlbaum, London, 1983).

[7] J. Coutaz, Interfaces hommes-ordinateur conception et réalisation, Dunod Informatique, 1994.

[8] J.L. Crowley and J.M. Bedrune, Integration and control of reactive visual processes, *Proc. European Conf. on Computer Vision (ECCV)-'94*, Stockholm (May 1994).

[9] J.L. Crowley and H. Christensen, *Vision as Process* (Springer, Heidelberg, 1994).

[10] C. Harris, Tracking with rigid models, in: *Active Vision* (MIT Press, Cambridge, MA, 1992).

[11] H. Inoue, T. Tashikawa and M.I. Inaba, Robot vision system with a correlation chip for real time tracking, optical flow, and depth map generation, *Proc. IEEE Conf. on Robotics and Automation*, Nice (1992).

[12] M. Kass, A. Witkin and D. Terzopoulos, Snakes: Active contour models, *Proc. 1st Int. Conf. on Computer Vision* (1987) 259–268.

[13] M. Krueger, *Artificial Reality II* (Addison-Wesley, Reading, MA, 1991).

[14] P. Maes, T. Darrel, B. Blumberg and A. Pentland, The ALIVE system: Full body interaction with animated autonomous agent, M.I.T Media Laboratory Perceptual Computing Technical Report No. 257, 194.

[15] J. Martin, Suivi et interprétation de Geste: Application de la vision par ordinateur 'a l'Interaction homme-machine, Rapport DEA Informatique, IMAG-INPG, 1995.

[16] J. Martin and J.L. Crowley, Experimental comparison of correlation techniques, *IAS-4, Int. Conf. on Intelligent Autonomous Systems*, Karlsruhe (1995).

[17] W. Newman and P. Wellner, A desk supporting computer-based interaction with paper documents, *Proc. CHI'92* (1992) 587–592.

[18] C. Ramstein, The pantograph: A large workspace haptic device for multimodal human computer interaction, *CHI'94 Interactive Experience, Adjunct Proc.* (1994) 57–58.

[19] J.M. Rehg and T. Kanade, DigitEyes: Vision-based human hand tracking, Carnegie Mellon University Technical Report CMU-CS-93-220, 1993.

[20] B. Schiele and A. Waibul, Gaze tracking based on face color, *Int. Workshop on Face and Gesture Recognition*, Zurich (1995).

[21] M. Turk and A. Pentland, Eigenfaces for recognition *Journal of Cognitive Neuroscience* 3 (1) (1991) 71–86.

[22] P. Wellner, Interacting with paper on the Digital/Desk, *communications of the ACM* 36 (7) (1993).

[23] P. Wellner, W. Mackay and R. Gold, Computer-augmented environments: Back to the real world, *Communications of the ACM* (special issue) 36 (7) (1993).

[24] J.M. Wozencraft and I.M. Jacobs, *Principals of Communication Engineering* (Wiley, New York, 1965).


**James L. Crowley** holds the post of Professor at the Institut National Polytechnique de Grenoble (INPG). He teaches courses in Artificial Intelligence, Machine Vision, and Robotics at l'Ecole National Supérieure d'Informatique et de Mathématiques Appliqués (ENSIMAG). Within the Laboratory GRAVIR of the Institut IMAG, Professor Crowley directs the project PRIMA. PRIMA has as its goal the development of techniques for integrating perception, action and reasoning. He is coordiantor of the European Computer Vision Network (ECVnet), a EC "Network of Excellence". HE is also coordinator of the DG XII Human Capital and Mobility network SMARTnet, whose subject is the development of techniques for a mobile autonomous surveillance robot. Professor Crowley has published two books, two special issues of journals, and over 90 articles on vision and mobile robotics.

To appear in ECCV-94, Stockholm, Sweden, May 2-6, 1994.

# Recognizing Hand Gestures *

James Davis and Mubarak Shah **

Computer Vision Laboratory, University of Central Florida, Orlando FL 32816, USA

**Abstract.** This paper presents a method for recognizing human-hand gestures using a model-based approach. A Finite State Machine is used to model four qualitatively distinct phases of a generic gesture. Fingertips are tracked in multiple frames to compute motion trajectories, which are then used for finding the start and stop position of the gesture. Gestures are represented as a list of vectors and are then matched to stored gesture vector models using table lookup based on vector displacements. Results are presented showing recognition of seven gestures using images sampled at 4 Hz on a SPARC-1 without any special hardware. The seven gestures are representatives for actions of Left, Right, Up, Down, Grab, Rotate, and Stop.

## 1 Introduction

It is essential for computer systems to possess the ability to recognize meaningful gestures if they are to interact naturally with people. Humans use gestures in daily life as a means of communication, e.g., pointing to an object to bring someone's attention to the object, waving "hello" to a friend, requesting $n$ of something by raising $n$ fingers, etc. The best example of communication through gestures is given by sign language. American Sign Language (ASL) incorporates the entire English alphabet along with many gestures representing words and phrases [3], which permits people to exchange information in a non-verbal manner.

Currently, the human-computer interface is through a keyboard and/or mouse. Physically challenged people may have difficulties with such input devices and may require a new means of entering commands or data into the computer. Gesture, speech, and touch inputs are few possible means of addressing such users' needs to solve this problem. Using Computer Vision, a computer can recognize and perform the user's gesture command, thus alleviating the need for a keyboard. Some applications for such a vision system are the remote control of a robotic arm, guiding a computer presentation system, and executing computer operational commands such as opening a window or program.

This paper presents a gesture recognition method using Computer Vision, which permits human users adorned with a specially marked glove to command

* The research reported here was supported by the National Science Foundation grants CDA-9200369 and IRI-9220768.

** For an extended version of this paper, please send e-mail to shah@sono.cs.ucf.edu.

a computer system to carry out predefined gesture action commands. Our system has been created to recognize a sequence of multiple gestures in which a subset of the gestures is comprised of select ASL letters (See Fig. 1). Each gesture begins with the hand in the "hello" position and ends in the recognizable gesture position. The current library of gestures contains seven gestures: *Left*, *Right*, *Up*, *Down*, *Rotate*, *Grab*, and *Stop*. The user must begin in the designated start position and is able to make gestures until the termination gesture (*Stop*) is recognized.

There are several advantages of this system over other methods. First, it uses inexpensive black-and-white video. Incorporating color markers on a glove as interest points [2] requires costly color imaging, whereas a binary marked glove, as used in this research, can be detected in low-cost black-and-white imaging. Second, a simple vision glove is employed, i.e., no mechanical glove with LEDs or bulky wires. Current gesture input devices require the user to be linked to the computer, reducing autonomy [1]. Vision input overcomes this problem. Third, a duration parameter for gestures is incorporated. For example, if this recognition system were connected to a robotic arm and the user makes a gesture for *Left*, the robotic arm would continue to move left until the user moves the hand from the gesture back to the start position. Therefore the user can control the execution duration of the robotic arm. Finally, due to Finite State Machine (FSM) implementation of a generic gesture, no warping of the image sequences is necessary. That is, the number of frames in each gesture can be variable.

## 2 Related Work

Baudel and Beaudouin-Lafon [1] implemented a system for the remote control of computer-aided presentations using hand gestures. In this system, the user wears a VPL DataGlove which is linked to the computer. The glove can measure the bending of fingers and the position and orientation of the hand in 3-D space. The user issues commands for the presentation by pointing at a predefined active zone and then performing the gesture for the desired command. Gesture models include information pertaining to the start position, arm motion (dynamic phase), and stop position of the gesture. The command set includes such commands as *next page*, *previous page*, *next chapter*, *previous chapter*, *table of contents*, *mark page*, and *highlight area*. Two main types of errors that can occur with this system are system errors and user errors. System errors relate to the difficulties identifying gestures that differ only in the dynamic phase, while user errors correspond to hesitations while issuing a command. With trained users, the recognition rate was 90% to 98%. This system does not use vision to recognize gestures, but instead uses a linked hardware system to track the hand and arm movements, which makes movement less natural for the user.

Cipolla, Okamoto, and Kuno [2] present a real-time structure-from-motion (SFM) method in which the 3-D visual interpretation of hand gestures is used in a man-machine interface. A glove with colored markers attached is used as input to the vision system. Movement of the hand results in motion between the images

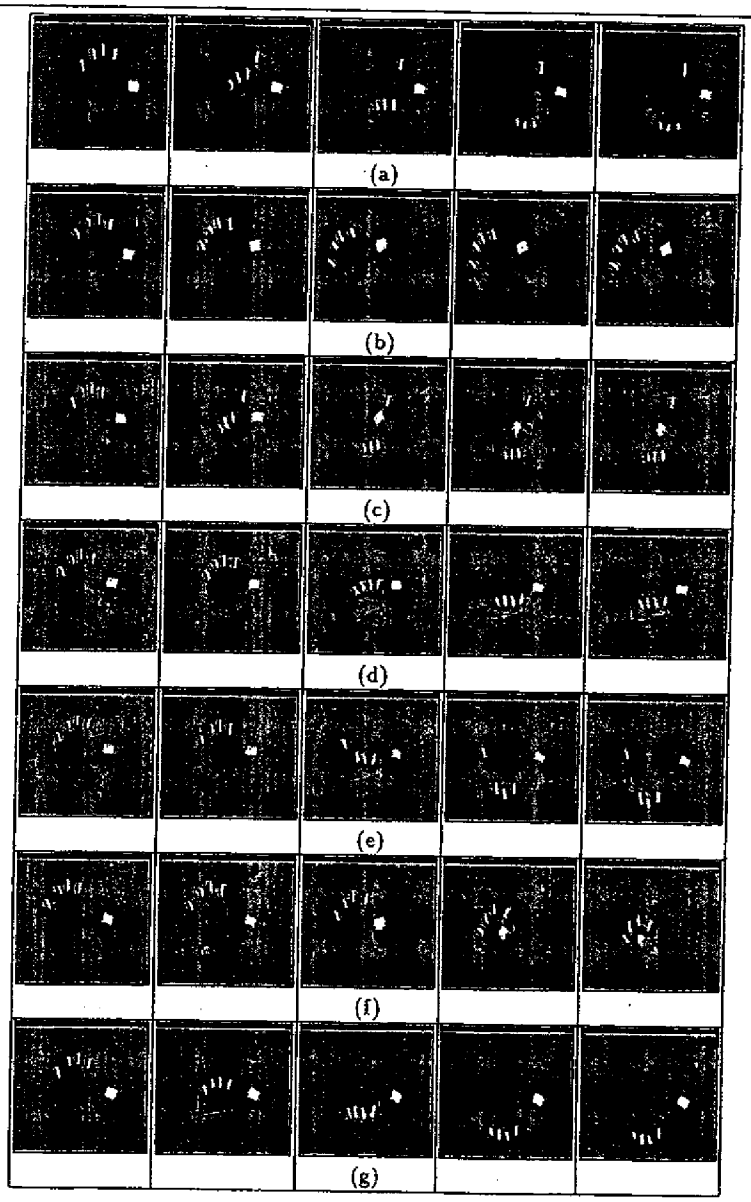


Fig. 1. Gestures. (a) Left. (b) Right. (c) Up. (d) Down. (e) Rotate. (f) Grab. (g) Stop.

of the colored markers. The authors use the parallax motion vector, divergence, curl, and deformation components of the affine transformation of an arbitrary triangle, with the colored points at each vertex, to determine the projection of the axis of rotation, change in scale, and cyclotorsion. This information is then used to alter an image of a model. The information extracted from the colored markers does not give the position of the *entire* hand (each finger), it only provides a triangular plane for the SFM algorithm. The SFM method used here assumes rigid objects, which is not true in the case of hand gestures.

Fukumoto, Mase, and Suenaga [5] present a system called *Finger-Pointer* which recognizes pointing actions and simple hand forms in real-time. The system uses stereo image sequences and does not require the operator to wear any special glove. It also requires no special image processing hardware. Using stereo images, their system uses the 3-D location of fingers rather than the 2-D location. The coordinates of the operator's fingertip and the direction it is pointing is determined from the stereo images and then a cursor is displayed in the target location on the opposing screen. The system is robust in that it is able to detect the pointing regardless of the operator's pointing style. Applications of this system can be similar to the gesture controlled computer-aided presentations of Baudel and Beaudouin-Lafon [1] and also can be used in a video browser with a VCR.

Darrell and Pentland [4] have also proposed a glove-free environment approach for gesture recognition. Objects are represented using sets of view models, and then are matched to stored gesture patterns using dynamic time warping. Each gesture is dynamically time-warped to make it of the same length as the longest model. Matching is based upon the normalized correlation between the image and the set of 2-D view models where the view models are comprised of one or more example images of a view of an object. This method requires the use of special-purpose hardware to achieve real-time performance, and uses gray level correlation which can be highly sensitive to noise. Also, their method was only tested in distinguishing between two gestures.

## 3 Generic Gesture

For a system to recognize a sequence of gestures, it must be able to determine what state the user's hand is in, i.e., whether or not the hand is dormant, moving, or in gesture position. Our approach relies on the qualitatively distinct events (phases) in gestures, rather than on frame by frame correlation. Each gesture the user performs begins with the hand in the start position (all fingers upright, as if one was about to wave "hello" to another person). Next, the user moves the fingers and/or entire hand to the gesture position. Once in position, the system will attempt to recognize and then execute the gesture command until the hand begins moving back to the start position. The system will then wait for the next gesture to occur. Thus, the user is constrained to the following four *phases* for making a gesture.

1. Keep hand still (fixed) in start position until motion to gesture begins.
2. Move fingers smoothly as hand moves to gesture position.
3. Keep hand in gesture position for desired duration of gesture command.
4. Move fingers smoothly as hand moves back to start position.

Since these four phases occur in a fixed order, an FSM can be used to guide the flow and recognition of gestures, based on the motion characteristics of the hand (See Fig. 2). A 1 or 0 in the state diagram represents motion or no motion respectively, between two successive images. A *three frame similarity* constraint, which states that, "at least three consecutive images must have the same motion properties to advance to the next phase," was found to inhibit premature phase advancement.

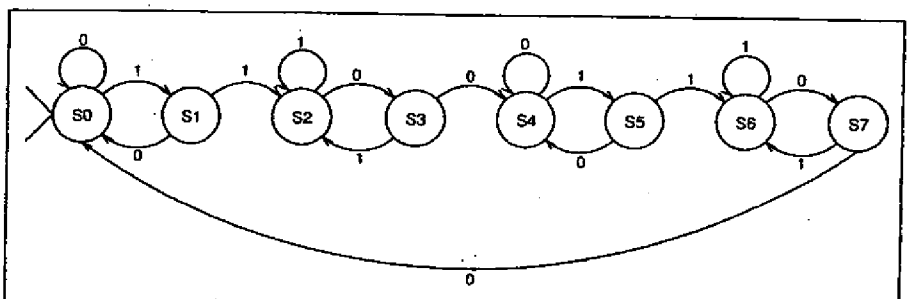


**Fig. 2.** State diagram. States S0 and S1 depict the initial phase (Phase 1), states S2 and S3 depict the motion to gesture phase (Phase 2), states S4 and S5 depict the gesture recognition phase (Phase 3) where an external device such as a robotic arm could execute the gesture until exiting this phase, and states S6 and S7 depict the motion to initial phase (Phase 4).

Due to the nature of this machine, no *warping* of image sequences is necessary, i.e., it is not required to have a fixed number of images for each gesture sequence. The FSM compensates for varying numbers of images by looping at the current phase as long as the *three frame similarity* constraint is satisfied. The *actual* number of frames which constitute the motion of a gesture yields no information for use with this system. The only useful information is the start and end position of the fingertips. The system does not care *how* the fingers or hand arrive in the gesture position; it wants to know the location of each fingertip before and when the gesture is made. Only the locations and total displacement of the fingertips play a crucial role in gesture recognition, as compared to other motion characteristics such as instantaneous velocity. Therefore, we need only to track each fingertip from the initial position to the final gesture position. The FSM permits the determination of which phase the user is currently executing, and it also tracks the fingertips of a variable-length frame sequence to the gesture position.

In our method, each image in the sequence is analyzed to find the location of the fingertips, and then motion correspondence is used to track these points to the resulting gesture position (Section 4). The trajectories computed by the motion correspondence algorithm are then converted to vector form to be matched with the stored gestures (Sections 5 and 6).

## 4 Fingertip Detection and Motion Correspondence

The goal of fingertip detection is to identify the 2-D location of the marked fingertips on the vision glove. The location of the fingertips determines the position of the fingers at any time. Since we are using a sequence of images in which the intensity of the fingertips is known a priori to be significantly different from the remaining regions, a multi-modal histogram of the image can be generated and used to segment the image into fingertip regions (See Fig. 3). We represent the five fingertip regions using centroids for ease of calculations, storage, display, etc. and also for motion correspondence (See Fig. 3.d).

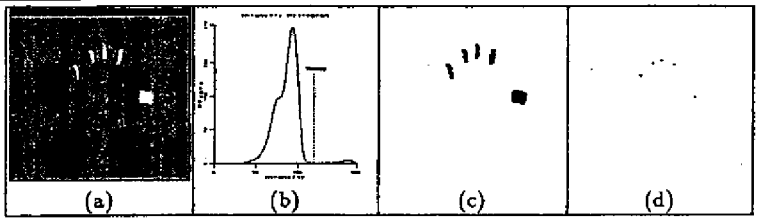


**Fig. 3.** Point detection process. (a) Initial image. (b) Smoothed histogram of entire image with threshold set at dashed line. In each image, the rightmost peak corresponds to the fingertip regions (brightest regions). (c) Binary image obtained from (a) by applying threshold determined from histogram in (b). (d) Centroids of five regions corresponding to five fingertips found in (c).

Motion correspondence maps points in one image to points in the next image such that no two points are mapped onto the same point. A path, known as a trajectory, is generated for each of the $m$ points, starting with the points in the first image and ending with the points in the $n$th image.

Rangarajan and Shah's [6] motion correspondence algorithm was chosen for its exploitation of a *proximal uniformity* constraint, which says objects follow smooth paths and cover a small (proximal) distance in a small time. It was stated previously, in the Phase 2 gesture constraint, that the fingers must move smoothly to the gesture position. Additionally, the *three frame similarity* constraint for motion, which requires at least three frames of motion, implies that the fingertips move a small (proximal) distance in each successive frame. Therefore,

the algorithm, using a proximal uniformity constraint, agrees with the previously stated gesture motion constraints.

The authors' algorithm establishes correspondence among points by minimizing a *proximal uniformity function* $\delta$, which prefers the proximal uniform path, such that

$$\delta(X_p^{k-1}, X_q^k, X_r^{k+1}) = \frac{\| \overline{X_p^{k-1} X_q^k} - \overline{X_q^k X_r^{k+1}} \|}{\sum_{x=1}^{m} \sum_{z=1}^{m} \| \overline{X_x^{k-1} X_{\Phi^{k-1}(x)}^k} - \overline{X_{\Phi^{k-1}(x)}^k X_z^{k+1}} \|} + \frac{\| \overline{X_q^k X_r^{k+1}} \|}{\sum_{x=1}^{m} \sum_{z=1}^{m} \| \overline{X_{\Phi^{k-1}(x)}^k X_z^{k+1}} \|} \quad (1)$$

where $\Phi^k$ is one to one onto correspondence between points of image $k$ and image $k+1$, $1 \leq p, q, r \leq m, 2 \leq k \leq m-1, q = \Phi^{k-1}(p)$, $\overline{X_q^k X_r^{k+1}}$ is the vector from point $q$ in image $k$ to point $r$ in image $k+1$, and $\| X \|$ denotes the magnitude of vector $X$ [6]. The first term in the equation represents the smoothness constraint and the second represents the proximity constraint.

## 5 Gesture Modeling

In general, human finger movements are linear, with extrema moving from an extended position down to palm/wrist area, e.g., from the hand in the "hello" position to the hand making a fist. Even though we have the ability of limited rotational movement in the fingers, we mostly move the fingers up and down to the palm, with the thumb moving left and right over the palm. Since the fingers move relatively linearly (some move curvilinearly at times), we can approximate each fingertip trajectory by a single vector (See Fig. 4). Each vector will originate at the location of the corresponding fingertip before motion to the gesture position, and will terminate at the location of the gesture position. We disregard the actual path each fingertip makes because, as stated previously, we are concerned with only the beginning and ending location of each fingertip. Therefore, if there is some curvature to the fingertip trajectory, it will be disregarded. The motion information leading to the gesture position is not needed. Motion correspondence is used only to map the starting points to the ending points by means of tracking the points in-between in the trajectories. See Fig. 4 for vector representations of the gesture set.

A library model is created from averaging $m$ test models of the same gesture and is represented in a data structure which contains

1. The gesture name.
2. The mean direction and mean magnitude, i.e., mean displacement, for each fingertip vector.
3. The gesture's motion code.

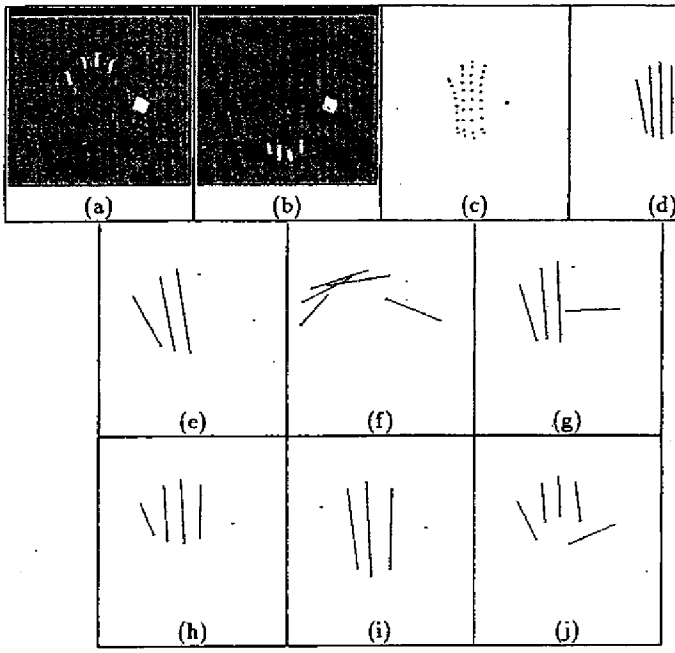


**Fig. 4.** Vector extraction and representation of gestures. (a)&(b) Image sequence – first and last images of gesture "Stop" shown. (c) Fingertip trajectories for entire gesture sequence. (d) Vector representation of trajectories. Other gesture vector representations: (e) Left, (f) Right, (g) Up, (h) Down, (i) Rotate, and (j) Grab.

The direction, $\Theta$, and magnitude, or displacement, of a fingertip vector is determined from the starting point $(x_0, y_0)$ and stopping point $(x_n, y_n)$ of its trajectory and are easily calculated respectively by

$$\Theta = \arctan \frac{y_n - y_0}{x_n - x_0} \ , \tag{2}$$

$$Disp = \sqrt{(x_n - x_0)^2 + (y_n - y_0)^2} \ . \tag{3}$$

We use a five-bit motion code to store the motion activity of the five fingertips for the gesture, which acts as a table key for the model. Each bit of this code corresponds to a specific fingertip vector, with the least significant bit storing finger 1's (thumb's) motion information and progressing to the most significant bit where finger 5's (little finger's) motion information is stored. A bit is set if its respective fingertip vector has motion, i.e., it's fingertip vector magnitude is

above some displacement threshold. Thus, the motion code for a gesture with significant motion in fingertip vectors 3, 4, and 5 only is represented as 11100. This binary number in decimal notation is 28, which is stored as the gesture's motion code.

## 6 Gesture Matching

Gesture matching consists of comparing the unknown gesture with the models to determine whether the unknown gesture matches with any model gesture in the system vocabulary.

Motion codes enable the matching scheme to consider only those models which have the same motion code as the unknown gesture and also provide information to which motion category the unknown gesture belongs. The library models, when loaded into memory, can be stored in an array of linear linked lists in which the array is indexed by the motion codes (0–31). During the matching stage, the unknown gesture is only compared with the library models that are indexed by the unknown gesture's motion code.

With only a subset of library models to compare to the unknown gesture, we have reduced the search complexity, which is now dependent on the different motion codes of the current library of gestures. A match is then determined by comparison between the stored models and the unknown gesture. A match is made if all vector fields (magnitude and direction for each fingertip) in the unknown gesture are within some threshold of the corresponding model entries.

## 7 Results

Ten sequences of over 200 frames each were digitized at 4 Hz, stored, and then used for the recognition program. Each run was performed in the same fashion, starting with the gesture for *Left* and progressing to the ending gesture *Stop*, as shown horizontally in Table 1. An image set in which the fixed order shown in Table 1 was altered resulted in perfect recognition, which implies order is not a concern.

The number of images for each sequence depended on the duration of each gesture performed. The overall results on the ten sequences of images yielded almost perfect scores with the exception of run 9, where an error in the sequence caused the remaining gestures to be unrecognizable. A shift of the hand above the threshold limit or occlusion of points due to lighting conditions may cause premature advancement of one phase to another, which in turn may result in the FSM continuing asynchronously with the image sequence.

Recognition of a nine $128 \times 128$ frame sequence sampled at 4 Hz took a CPU time of 890 ms on a SPARC-1 (99 ms processing time per frame) with no special hardware. Our experiments show that sampling at a rate of 30 Hz is not necessary for gesture recognition since the processing time needed for our method is small enough for implementation in real-time with images sampled up to 10 Hz.

Table 1. Table 1: Results. √ – Recognized, X – Not Recognized, * – Error in Sequence.

| Run | Frames | Left | Right | Up | Down | Rotate | Grab | Stop |
|---|---|---|---|---|---|---|---|---|
| 1 | 200 | √ | √ | √ | √ | √ | √ | √ |
| 2 | 250 | √ | √ | √ | √ | √ | √ | √ |
| 3 | 250 | √ | √ | √ | X | √ | √ | √ |
| 4 | 250 | √ | √ | √ | √ | √ | √ | √ |
| 5 | 300 | √ | √ | √ | √ | √ | √ | √ |
| 6 | 300 | √ | √ | √ | √ | √ | √ | √ |
| 7 | 300 | √ | √ | √ | √ | √ | √ | √ |
| 8 | 300 | √ | √ | √ | √ | √ | √ | √ |
| 9 | 300 | √ | √ | √ | √ | * | * | * |
| 10 | 300 | √ | √ | √ | √ | √ | √ | √ |

## 8 Conclusion

In this paper, we have developed a Computer Vision method for recognizing sequences of human-hand gestures within a gloved environment. A specialized FSM was constructed as an alternative to image sequence warping. We utilize vectors for representing the direction and displacement of the fingertips for the gesture. Modeling gestures as a set of vectors with a motion code allows the reduction of complexity in the model form and matching. We presented the performance of this method on real image sequences. Future work pursued includes extending the gesture vocabulary, removing the glove environment, and relaxing the start/stop requirement.

## References


1. Baudel T., Beaudouin-Lafon M.: Charade: Remote control of objects using free-hand gestures. CACM. July (1993) 28–35
2. Cipolla R., Okamoto Y., Kuno Y.: Robust structure from motion using motion parallax. ICCV. (1993) 374–382
3. Costello E.: Signing: How to speak with your hands. Bantam Books, New York (1993)
4. Darrell T., Pentland A.: Space-time gestures. CVPR. (1993) 335–340
5. Fukumoto, M., Mase, K., Suenaga, Y.: Real-time detection of pointing actions for a glove-free interface. IAPR Workshop on Machine Vision Applications. Dec. 7–9 (1992) 473–476
6. Rangarajan, K., Shah, M.: Establishing motion correspondence. CVGIP: Image Understanding. 54 July (1991) 56–73

Europäisches Patentamt
European Patent Office
Office européen des brevets

European Patent Office
Postbus 5818
2280 HV RIJSWIJK
NETHERLANDS
Tel. +31 (0)70 340-2040
Fax +31 (0)70 340-3016

**For any questions about this communication:**
Tel.:+31 (0)70 340 45 00

Wombwell, Francis
Potts, Kerr & Co.
15, Hamilton Square
Birkenhead
Merseyside CH41 6BR
GRANDE BRETAGNE

Date
12.12.08

| Reference | Application No./Patent No. |
|---|---|
| FB9380E/E14549E | 06016855.6 - 1245 / 1717681 |

Applicant/Proprietor
Apple Inc.

## Communication

The extended European search report is enclosed.

The extended European search report includes, pursuant to Rule 62 EPC, the European search report (R. 61 EPC) or the partial European search report/ declaration of no search (R. 63 EPC) and the European search opinion.

Copies of documents cited in the European search report are attached.

☑ 2 additional set(s) of copies of such documents is (are) enclosed as well.

The following have been approved:

☑ Abstract ☑ Title

☐ The Abstract was modified and the definitive text is attached to this communication.

The following figure(s) will be published together with the abstract: 1

### Refund of the search fee

If applicable under Article 9 Rules relating to fees, a separate communication from the Receiving Section on the refund of the search fee will be sent later.

Europäisches Patentamt • European Patent Office • Office européen des brevets

The examination is being carried out on the **following application documents**:

**Description, Pages**

1-81 as originally filed

**Claims, Numbers**

1-7 as originally filed

**Drawings, Sheets**

1/45-45/45 as originally filed

1. The subject-matter of claim 1 appears to involve an inventive step having regard to the prior art cited in the search report.

2. The following claims, however, do not meet the requirements of Article 84 EPC and are therefore not allowable, for the following reasons:

2.1. The following expressions are vague and unclear and leave the reader in doubt as to the meaning of the technical features to which they refer:
Claim 1: "the opposable thumb", "translation weighting", "scaling velocity component", "translation weighted average", "translational velocity components".
Claim 4: "polar component speeds"

2.2. The expression used in claim 1 "the proximity sensor" refers to a feature which has not been previously defined in the claim.

2.3. The following expressions merely attempt to define the subject-matter in terms of the result to be achieved:

Claim 1: "tracking...the trajectories of individual hand parts", "finding an innermost and an outermost finger contact from contacts identified as fingers on the given hand", "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".

Claim 2: "a measure of scaling velocity selective for symmetric scaling about a fixed point between the thumb and other fingers".

Claim 3: "...is supplemented with a measure of rotational velocity selective for symmetric rotational about a fixed point between thumb and other fingers"

Claim 4: "...so as to prevent vertical translation bias performing hand scaling and rotation but otherwise include all available fingers in the translation average"

Claim 5: "the translational weightings are related to the ratio of each finger's speed to the speed of the fastest finger so that if the user chooses to move fewer fingers than are on the surface the gain between individual finger motion and cursor motion does not decrease"

Claim 6: "downscaling each velocity component in proportion to a function of its average speed compared to the other average component speeds" and "dead-zone filtering each downscaled velocity component wherein the width of the dead-zone depends on the distribution of the current component speeds"

Claim 7: " the orientation of an ellipse fitted to the thumb contact after each successive sensor array scan is transmitted as an additional degree of freedom control signal"

Such a definition is only allowable under the conditions elaborated in the Guidelines C-III, 4.10. In this instance, however, such a formulation is not allowable because it

appears possible to define the subject-matter in more concrete terms, viz. in terms of how the effect is to be achieved.

2.4. The statement in the description "...the spirit of the invention..." on page 81 implies that the subject-matter for which protection is sought may be different to that defined by the claims, thereby resulting in lack of clarity of the claims (Article 84 EPC) when used to interpret them (see the Guidelines, C-III, 4.4). This statement should therefore be deleted.

3. When filing an amended set of claims, the applicant should also take into account the following remarks:

3.1. The features of the claims should be provided with reference signs placed in parentheses to increase the intelligibility of the claims (Rule 43(7) EPC). This applies to both the preamble and characterising portion (see Guidelines, C-III, 4.19).

3.2. The applicant should bring the description into conformity with the amended claims. Care should be taken during revision, especially of the introductory portion and any statements of problem or advantage, not to add subject-matter which extends beyond the content of the application as originally filed (Article 123(2) EPC).

3.3. In order to facilitate the examination of the conformity of the amended application with the requirements of Article 123(2) EPC, the applicant is requested to clearly identify the amendments carried out, irrespective of whether they concern amendments by addition, replacement or deletion, and to indicate the passages of the application as filed on which these amendments are based.

If the applicant regards it as appropriate these indications could be submitted in handwritten form on a copy of the relevant parts of the application as filed.

Europäisches Patentamt
European Patent Office
Office européen des brevets

# EUROPEAN SEARCH REPORT

Application Number
EP 06 01 6855

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (IPC) |
|---|---|---|---|
| A | LEE S K ET AL: "A Multi-Touch Three Dimensional Touch-Sensitive Tablet" PROCEEDINGS OF CHI: ACM CONFERENCE ON HUMAN FACTORS IN COMPUTINGSYSTEMS, XX, XX, 1 April 1985 (1985-04-01), pages 21-25, XP009074849 * the whole document * | 1-7 | INV. G06F3/033 |
| A | US 5 479 528 A (SPEETER THOMAS H [US]) 26 December 1995 (1995-12-26) * column 4, line 45 - column 8, line 49; figures 9,10 * | 1-7 | |
| A | DAVIS J ET AL: "Recognizing hand gestures" EUROPEAN CONFERENCE ON COMPUTER VISION, BERLIN, DE, vol. 1, 2 May 1994 (1994-05-02), pages 331-340, XP002360860 * the whole document * | 1-7 | |
| A | US 5 495 077 A (MILLER ROBERT J [US] ET AL) 27 February 1996 (1996-02-27) * the whole document * | 1-7 | TECHNICAL FIELDS SEARCHED (IPC) G06F G06K G06T |
| D,A | QUEK F K H: "UNENCUMBERED GESTURAL INTERACTION" IEEE MULTIMEDIA, IEEE SERVICE CENTER, NEW YORK, NY, US, vol. 3, no. 4, 1 December 1996 (1996-12-01), pages 36-47, XP000636689 ISSN: 1070-986X * the whole document * | 1-7 | |

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| The Hague | 4 December 2008 | Arranz, José |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document



EPO FORM 1503 03.82 (P04C01)

## ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 06 01 6855

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

04-12-2008

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| US 5479528 A | 26-12-1995 | NONE | |
| US 5495077 A | 27-02-1996 | US 5543588 A | 06-08-1996 |
| | | US 5914465 A | 22-06-1999 |
| | | US 6239389 B1 | 29-05-2001 |

Europäisches Patentamt
European Patent Office
Office européen des brevets

European Patent Office
Postbus 5818
2280 HV RIJSWIJK
NETHERLANDS
Tel. +31 (0)70 340-2040
Fax +31 (0)70 340-3016

**For any questions about this communication:**
Tel.:+31 (0)70 340 45 00

Wombwell, Francis
Potts, Kerr & Co.
15, Hamilton Square
Birkenhead
Merseyside CH41 6BR
GRANDE BRETAGNE

Date
23.12.08

| Reference | Application No./Patent No. |
|---|---|
| FB9380G/E14549E | 06016831.7 - 1245 / 1717678 |

Applicant/Proprietor
Apple Inc.

## Communication

The extended European search report is enclosed.

The extended European search report includes, pursuant to Rule 62 EPC, the European search report (R. 61 EPC) or the partial European search report/ declaration of no search (R. 63 EPC) and the European search opinion.

Copies of documents cited in the European search report are attached.

☑ 2 additional set(s) of copies of such documents is (are) enclosed as well.

The following have been approved:

☑ Abstract ☑ Title

☐ The Abstract was modified and the definitive text is attached to this communication.

The following figure(s) will be published together with the abstract: 1

### Refund of the search fee

If applicable under Article 9 Rules relating to fees, a separate communication from the Receiving Section on the refund of the search fee will be sent later.

The examination is being carried out on the **following application documents**:

**Description, Pages**

1-81 as originally filed

**Claims, Numbers**

1-24 as originally filed

**Drawings, Sheets**

1/45-45/45 as originally filed

1. Reference is made to the following documents; the numbering will be adhered to in the rest of the procedure:

   D1: EP-A-0 817 000 (IBM [US]) 7 January 1998 (1998-01-07)
   D2: US-A-5 479 528 (SPEETER THOMAS H [US]) 26 December 1995 (1995-12-26)
   D3: EP-A-0 622 722 (RANK XEROX LTD [GB] XEROX CORP [US]) 2 November 1994 (1994-11-02)
   D4: LEE S K ET AL: "A Multi-Touch Three Dimensional Touch-Sensitive Tablet" PROCEEDINGS OF CHI: ACM CONFERENCE ON HUMAN FACTORS IN COMPUTINGSYSTEMS, XX, XX, 1 April 1985 (1985-04-01), pages 21-25, XP009074849

2. The application does not meet the requirements of Article 84 EPC, for the following

reasons.

2.1. Claims 1,15 have been drafted as separate independent claims. Under Article 84 EPC in combination with Rule 43(2) EPC, an application may contain more than one independent claim in a particular category only if the subject-matter claimed falls within one or more of the exceptional situations set out in paragraph (a), (b) or (c) of Rule 43(2) EPC. This is not the case in the present application, however, for the following reasons: Although claims 1,15 have been drafted as separate independent claims, they appear to relate effectively to the same subject-matter and to differ from each other only with regard to the definition of the subject-matter for which protection is sought and in respect of the terminology used for the features of that subject-matter.

2.2. The expression used in claim 1, "...method for supporting divers hand input activities such...", is vague and unclear and leaves the reader in doubt as to the meaning of the technical features to which it refers. It is not clear which is the level of contribution or result achieved by the claimed method steps when "supporting" the hand input activities.

2.3. Claim 1 defines the step of "detecting when hand parts touch down or lift of simultaneously", which is the only step relating to detecting hand parts defined in the claim. The following step however defines the step of "producing discrete...when the user asynchronously taps a finger...on the key regions", which is therefore in contradiction which the previously defined "detection" step. Such an embodiment, i.e. producing a signal when the user asynchronously taps a finger based on detecting when hand parts touch down or lift of simultaneously is is not supported by the description as required by Article 84 EPC.

2.4. The statement in the description "...the spirit of the invention..." on page 81 implies that the subject-matter for which protection is sought may be different to that defined by the claims, thereby resulting in lack of clarity of the claims (Article 84 EPC) when used to interpret them (see the Guidelines, C-III, 4.4). This statement should therefore be deleted.

3. Furthermore, notwithstanding the above-mentioned lack of clarity, the subject-matter of claim 1 does not involve an inventive step within the meaning of Article 56 EPC, and the requirements of Article 52(1) EPC are not therefore met.

3.1. D1 discloses a method which enables a user to switch between different input activities by placing his hands in different configurations comprising distinguishable combinations of relative hand contact timing (Col.5, line 54), shape (Col.12, line 23-36 ), size (Col.9, line 41), position (Col.5, lines 20-36), motion (Col.5, lines 20-36) and or identity (Col.5, lines 48-58) across a succession of surface proximity images. The input method of D1 furthermore makes a distinction in the number of finger used for invoking different functions or commands (Col.6, lines 30-32).

The distinguishing features merely attempt to define different associations of commands and gestures to the associations defined in D1, i.e. tap command with two fingers tapping synchronously, gesture command when the user slides two or more fingers or discrete key symbols when the user taps asynchronously. The defined association of input gestures and functions or commands however, merely represent an obvious and straightforward alternative implementation of the teaching of D1 that the skilled person would make, in accordance with circumstances, without the exercise of inventive skill.

Hence, the subject-matter of claim 1 lacks an inventive step.

4. A similar objection applies mutatis mutandis, to claim 15.

5. Dependent claims 2-14,16-24 do not appear to contain any additional features which, in combination with the features of any claim to which they refer, meet the requirements of the EPC with respect to inventive step, the reasons being as follows: they are obvious in the light of the disclosures of documents D1-D4 and the common knowledge of a person skilled in the art.

6. It is not at present apparent which part of the application could serve as a basis for a new, allowable claim. Should the applicant nevertheless regard some particular matter as patentable, an independent claim should be filed taking account of Rule

43(1) EPC. The applicant should also indicate in the letter of reply the difference of the subject-matter of the new claim vis-à-vis the state of the art and the significance thereof.

When filing an amended set of claims, the applicant should also take into account the following remarks:

6.1. To meet the requirements of Rule 42(1)(b) EPC, the documents D1,D2 should be identified in the description and the relevant background art disclosed therein should be briefly discussed.

6.2. Any amended independent claim should be filed in the two-part from (cf. Rule 43(1) EPC).

6.3. The features of the claims should be provided with reference signs placed in parentheses to increase the intelligibility of the claims (Rule 43(7) EPC). This applies to both the preamble and characterising portion (see Guidelines, C-III, 4.19).

6.4. The applicant should bring the description into conformity with the amended claims. Care should be taken during revision, especially of the introductory portion and any statements of problem or advantage, not to add subject-matter which extends beyond the content of the application as originally filed (Article 123(2) EPC).

6.5. In order to facilitate the examination of the conformity of the amended application with the requirements of Article 123(2) EPC, the applicant is requested to clearly identify the amendments carried out, irrespective of whether they concern amendments by addition, replacement or deletion, and to indicate the passages of the application as filed on which these amendments are based.

If the applicant regards it as appropriate these indications could be submitted in handwritten form on a copy of the relevant parts of the application as filed.

Europäisches Patentamt
European Patent Office
Office européen des brevets

# EUROPEAN SEARCH REPORT

Application Number
EP 06 01 6831

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (IPC) |
|---|---|---|---|
| X | EP 0 817 000 A (IBM [US]) 7 January 1998 (1998-01-07) * column 4, line 4 - column 12, line 36; figures 1-9 * | 1-24 | INV. G06F3/033 G06F3/048 |
| A | US 5 479 528 A (SPEETER THOMAS H [US]) 26 December 1995 (1995-12-26) * column 1, line 60 - column 8, line 52; figures 1-10 * | 1-24 | |
| A | EP 0 622 722 A (RANK XEROX LTD [GB] XEROX CORP [US]) 2 November 1994 (1994-11-02) * column 5, line 10 - column 12, line 23; figures 1-6 * | 1-24 | |
| A | LEE S K ET AL: "A Multi-Touch Three Dimensional Touch-Sensitive Tablet" PROCEEDINGS OF CHI: ACM CONFERENCE ON HUMAN FACTORS IN COMPUTINGSYSTEMS, XX, XX, 1 April 1985 (1985-04-01), pages 21-25, XP009074849 * the whole document * | 1-24 | |

TECHNICAL FIELDS SEARCHED (IPC)
G06F

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| The Hague | 12 December 2008 | Arranz, José |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document



EPO FORM 1503 03.82 (P04C01)

## ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 06 01 6831

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

12-12-2008

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| EP 0817000 A | 07-01-1998 | DE 69709991 D1 | 14-03-2002 |
| | | DE 69709991 T2 | 26-09-2002 |
| | | JP 3504462 B2 | 08-03-2004 |
| | | JP 10063424 A | 06-03-1998 |
| | | US 5856824 A | 05-01-1999 |
| US 5479528 A | 26-12-1995 | NONE | |
| EP 0622722 A | 02-11-1994 | DE 69430967 D1 | 22-08-2002 |
| | | DE 69430967 T2 | 07-11-2002 |
| | | JP 3383065 B2 | 04-03-2003 |
| | | JP 7168949 A | 04-07-1995 |
| | | US 5511148 A | 23-04-1996 |

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

Europäisches Patentamt
European Patent Office
Office européen des brevets

European Patent Office
Postbus 5818
2280 HV RIJSWIJK
NETHERLANDS
Tel. +31 (0)70 340-2040
Fax +31 (0)70 340-3016



For any questions about this communication:
Tel.:+31 (0)70 340 45 00

Wombwell, Francis
Potts, Kerr & Co.
15, Hamilton Square
Birkenhead
Merseyside CH41 6BR
GRANDE BRETAGNE

Date: 09.01.09

| Reference | Application No./Patent No. |
|---|---|
| FB9380C/E14549E | 06016832.5 - 1245 / 1717679 |

Applicant/Proprietor: Apple Inc.

## Communication

The extended European search report is enclosed.

The extended European search report includes, pursuant to Rule 62 EPC, the European search report (R. 61 EPC) or the partial European search report/ declaration of no search (R. 63 EPC) and the European search opinion.

Copies of documents cited in the European search report are attached.

☑ 2 additional set(s) of copies of such documents is (are) enclosed as well.

The following have been approved:

☑ Abstract ☑ Title

☐ The Abstract was modified and the definitive text is attached to this communication.

The following figure(s) will be published together with the abstract: 1

### Refund of the search fee

If applicable under Article 9 Rules relating to fees, a separate communication from the Receiving Section on the refund of the search fee will be sent later.

Europäisches Patentamt • European Patent Office • Office européen des brevets

# EUROPEAN SEARCH REPORT

Application Number
EP 06 01 6832

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (IPC) |
|---|---|---|---|
| P,X | WESTERMAN W: "HAND TRACKING, FINGER IDENTIFICATION, AND CHORDIC MANIPULATION ON A MULTI-TOUCH SURFACE" DISSERTATION UNIVERSITY OF DELAWARE,, 1 January 1999 (1999-01-01), pages 1-333, XP002486836 * pages 138-187 * | 1-18 | INV. G06F3/033 |
| A | US 5 479 528 A (SPEETER THOMAS H [US]) 26 December 1995 (1995-12-26) * column 4, line 45 - column 8, line 52; figures 9,10 * | 1-18 | |
| A | CROWLEY J L: "Vision for man-machine interaction" ROBOTICS AND AUTONOMOUS SYSTEMS, ELSEVIER SCIENCE PUBLISHERS, AMSTERDAM, NL, vol. 19, no. 3-4, 1 March 1997 (1997-03-01), pages 347-358, XP004075327 ISSN: 0921-8890 * the whole document * | 1-18 | |
| A | CHIN-CHEN CHANG ET AL: "A HASHING-ORIENTED NEAREST NEIGHBOR SEARCHING SCHEME" PATTERN RECOGNITION LETTERS, ELSEVIER, AMSTERDAM, NL, vol. 14, no. 8, 1 August 1993 (1993-08-01), pages 625-630, XP000383902 ISSN: 0167-8655 * the whole document * | 1-18 | |

-/--

TECHNICAL FIELDS SEARCHED (IPC)
G06F
G06K

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| The Hague | 23 December 2008 | Arranz, José |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

# EUROPEAN SEARCH REPORT

Application Number
EP 06 01 6832

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (IPC) |
|---|---|---|---|
| A | NIREI K ET AL: "Human hand tracking from binocular image sequences" INDUSTRIAL ELECTRONICS, CONTROL, AND INSTRUMENTATION, 1996., PROCEEDIN GS OF THE 1996 IEEE IECON 22ND INTERNATIONAL CONFERENCE ON TAIPEI, TAIWAN 5-10 AUG. 1996, NEW YORK, NY, USA,IEEE, US, vol. 1, 5 August 1996 (1996-08-05), pages 297-302, XP010203420 ISBN: 978-0-7803-2775-7 * the whole document * | 1-18 | |
| A | HEAP T ET AL: "Towards 3D hand tracking using a deformable model" AUTOMATIC FACE AND GESTURE RECOGNITION, 1996., PROCEEDINGS OF THE SECO ND INTERNATIONAL CONFERENCE ON KILLINGTON, VT, USA 14-16 OCT. 1996, LOS ALAMITOS, CA, USA,IEEE COMPUT. SOC, US, 14 October 1996 (1996-10-14), pages 140-145, XP010200411 ISBN: 978-0-8186-7713-7 * the whole document * | 1-18 | TECHNICAL FIELDS SEARCHED (IPC) |

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| The Hague | 23 December 2008 | Arranz, José |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

## ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 06 01 6832

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

23-12-2008

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| US 5479528 A | 26-12-1995 | NONE | |

# Towards 3D Hand Tracking using a Deformable Model

Tony Heap and David Hogg
School of Computer Studies, University of Leeds, Leeds LS2 9JT, UK.
(ajh, dch)@scs.leeds.ac.uk

## Abstract

*In this paper we first describe how we have constructed a 3D deformable Point Distribution Model of the human hand, capturing training data semi-automatically from volume images via a physically-based model. We then show how we have attempted to use this model in tracking an unmarked hand moving with 6 degrees of freedom (plus deformation) in real time using a single video camera. In the course of this we show how to improve on a weighted least-squares pose parameter approximation at little computational cost. We note the successes and shortcomings of our system and discuss how it might be improved.*

## 1 Motivations

There has long been a need for a vision-based hand tracking system which is capable of tracking movement with 6 degrees of freedom (DOF), along with articulation information, whilst being as unintrusive as possible. The use of hand markings or coloured gloves and the need for highly constrained environments are undesirable. Such a system should also be as widely available as possible; this implies low-cost technology, so ideally a single camera input should be used and real-time performance should be possible on a standard workstation.

From the plethora of work on vision-based hand tracking, relatively few have tackled the task of extracting full 6 DOF hand position *and* articulation. Notable successes have been due to Dorner [6], whose goal was American Sign Language interpretation, and Rehg and Kanade [13] who developed a system called *DigitEyes*. Dorner relies on gloves with coloured markers to aid tracking (and incurs a speed penalty in the associated marker detection); Rehg however tracks only from edge information, and achieves a healthy 10Hz frame rate, but has to revert to stereo input in order to track full hand articulation. Both make use of a manually-constructed articulated hand model with *a priori* knowledge of inter-joint distances and valid pivot angles, and non-linear methods are employed in pose estimation.

With a view to addressing some of these limitations, and in order to attack the problem from a new angle, our tracker is based on a 3D version of the Point Distribution Model (PDM) [4]; this is a statistically-derived deformable model which affords several advantages:

- The model is constructed from real-life examples of hands in various positions, giving an accurate model which implicitly captures the ways in which a hand's shape can and can't vary. The specificity of the model proves to be invaluable when tracking from a single 2D image, from which data is both noisy and relatively sparse.
- The hand is modelled as a surface mesh, from which the positions of expected contours are easily derived. By sampling at every mesh vertex large amounts of good position information can be obtained, even in the case of partial occlusion or noise from background clutter.
- The technique uses linear mathematics in most calculations, which allows for fast tracking rates.
- The hand posture is characterised by only a few scalars, easing gesture analysis.

The required training information is extracted semi-automatically from 3D Magnetic Resonance Images using a deformable surface mesh.

The model is used to track a hand in real-time (currently 18 frames/second on a standard 134MHz Silicon Graphics Indy workstation) using a single video camera for input. The model is projected (orthographically) onto input images and edge detection is used to move and deform the model to fit image evidence.

The remainder of this paper is split into three sections. In the first the construction of the 3D PDM is described, in the second it is shown how this model is applied to hand tracking, and in the third the performance of the tracker is discussed, its shortcomings are highlighted and suggestions for improvement are made.

0-8186-7713-9/96 $5.00 © 1996 IEEE

## 2 The hand model

### 2.1 Point Distribution Models

A PDM is a deformable model built from the statistical analysis of examples of the object being modelled. Given a collection of 3D training images of an object, the Cartesian coordinates of $N$ strategically-chosen landmark points are recorded for each image. Training example $e$ is represented by a vector $\mathbf{x}_e = (x_{e1}, y_{e1}, z_{e1}, \ldots, x_{eN}, y_{eN}, z_{eN})$.

The examples undergo least-squares alignment and scaling to unit size; the pointwise mean shape $\overline{\mathbf{x}}$ is then calculated. Modes of variation are found using Principal Component Analysis (PCA) on the deviations of examples from the mean. These modes are represented by $3N$ orthonormal eigenvectors $\mathbf{v_j}$. A unit-sized object shape $\mathbf{x}^U$ is generated by adding linear combinations of the $t$ most significant variation vectors to the mean shape:

$$\mathbf{x}^U = \overline{\mathbf{x}} + \sum_{j=1}^{t} b_j \mathbf{v_j} \qquad (1)$$

where $b_j$ is the weighting for the $j^{th}$ variation vector.

By ensuring $t \ll N$, only the important deformations are extracted, discarding training data noise, and thus object shape and variation can be captured compactly.

### 2.2 Acquiring training data

A key requirement for building such a model is the collection of landmark coordinate data from training images. Doing this manually for a 3D model is impractical due to the considerable effort required for image-model registration. Automatic mesh growing/deforming is hampered by the need for point correspondences between examples.

These setbacks can be overcome by capturing training data semi-automatically using a physically-based model. A mesh (we used a Simplex Mesh [5]) is constructed on the surface of the hand in the first training image. This mesh is deformed to fit subsequent training examples under the action of various forces. A few manually-positioned *guiding* forces pull key features (such as the fingertips) roughly into position, and 3D edge detection is used to construct forces at *every* vertex to drive the model to an exact fit. Internal forces also act to constrain the model shape (for smoothness and evenness). Full details can be found in [8].

The mesh vertices can be used directly as landmark points for the PDM.

### 2.3 Model construction

Surface meshes were fitted to 8 different hand images, all from the same person. From these, a PDM with 7 modes of variation was constructed. Most of the significant deformation (over 93%) is captured by the first five modes. Figure 1 shows the two main variation modes. It should be noted that 8 training examples are too few to use as a basis for a PDM. The modes of variation produced mainly constitute linear interpolations between the different hand shapes in the training set. For this reason, the method is somewhat similar to *key frames*, as used by Blake [3].

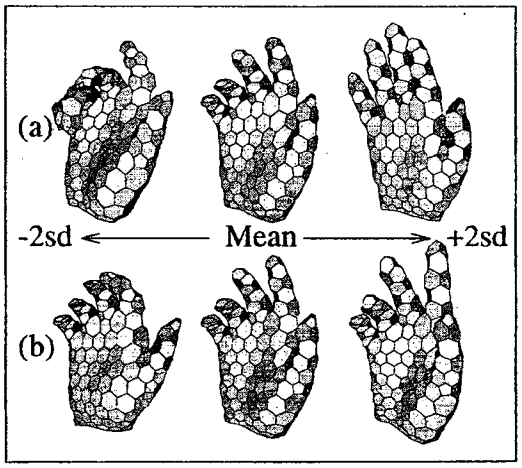


Figure 1. The first (a) and second (b) modes of variation of the 3D hand PDM.

## 3 Tracking

There has been much work on using PDMs for object location and tracking in both two and three dimensions. In most of this previous work, the dimensionality of the model has matched that of the input image (ie. 2D model for 2D images [12, 2, 7], 3D model for 3D images [11]). Work on matching a 3D model to a 2D image has so far assumed a ground plane constraint and only one degree of rotational freedom [15, 16]. We are attempting to match a 3D PDM to a 2D image under full 6 DOF.

The key to model-based object location is finding the set of model parameter values which cause the model to best fit the image data. In our case the parameters are a translation vector $\mathbf{u} = (u, v, w)$, a $3 \times 3$ orthonormal rotation matrix $\mathbf{R}$, a scale factor $s$ and the five significant deformation parameters $b_j$ (giving a total of 12 DOF). Iterative pose refinement is used – given a fair initial guess at an object's location, local image information (eg. edge data) is extracted and used to calculate a small change in the model parameters which will improve the fit.

To compare the model to the image, it is necessary to project the model onto the image. The model is first de-