EXHIBIT 4.11

in the same column or a column immediately adjacent to $G_{maxcol[i_{prev}]}$. For these columns j such that $G_{maxcol[i_{prev}]} - 1 <= j <= G_{maxcol[i_{prev}]} + 1$, a full partial minimum test of both next and previous pixels is applied instead of just checking for an increase in the next pixel in the direction of search, *i.e.*, a horizontal boundary is only indicated if $S_{i,j_{prev}}[n] > S_{i,j}[n] < S_{i,j_{next}}[n]$. This tolerates tall contacts which are oriented at a slant and therefore have proximity ridges which are not perfectly vertical. If $S_{i,j_{prev}}[n]$ was not checked and the search started from a column which did not have maximum proximity in the current row, diagonal ridges would be erroneously rejected.

For columns farther away from the previous row's maximum pixel, the horizontal test is slackened to include general increases in proximity in the horizontal, diagonal, and vertical directions of search. The horizontal boundary test simply becomes $S_{i,j}[n] < S_{i,j_{next}}[n]$, and a test $S_{i,j}[n] < S_{i_{next},j}[n]$ of the vertical neighbor in the next row to be searched indicates presence of a diagonal proximity valley caused by a contact diagonally adjacent to the contact being searched. Either test can stop the horizontal search in the current direction, establishing a horizontal boundary for the current row. By only testing that the proximity of the next pixel is greater than the current pixel without testing that the previous pixel is also greater than or equal to the current pixel, a search which starts in a sloppy region and meanders into a strict region will be forced to stop if proximities in the strict region are increasing in the direction of search, even if the actual saddle point occurred in the sloppy region and was ignored. This helps prevent palms from being joined to thumbs or flattened fingers if the sloppy segmentation regions get too close to the fingers.

When deciding whether to advance to the next row i_{next}, vertical partial minimum tests are applied at the column $j = G_{maxcol[i]}$ of the pixel in the current row found to have the maximum proximity. These tests also have a diagonal component, positively indicating a vertical boundary if $S_{i,j}[n] < S_{i_{next},j}[n]$, $S_{i,j}[n] < S_{i_{next},j-1}[n]$

or $S_{i,j}[n] < S_{i_{next},j+1}[n]$.

3.2.6.3 Flattened Finger Segmentation

To allow fingertip groups to merge reliably with the finger's proximal phalange contacts (Figure 2.7) without allowing thumbs, forepalms, or palm heels to merge with fingertips, the system ignores vertical partial minima in the strict segmentation region under a long series of conditions. First, such merging must be enabled by identification system feedback (Figure 3.2) indicating that most of the contacts identified as fingers are flattening onto the surface. This feedback takes the form of a flag which is set when in the previous image the geometric mean of the contact heights of all fingers but the innermost surpasses a threshold. This feedback distinguishes proximal phalanges of flattened fingers (Figure 2.7) from a thumb behind a curled fingertip (Figure 2.9) using the fact that it is biomechanically difficult to flatten one long finger without flattening the other long fingers. The geometric mean of multiple finger sizes and exclusion of the innermost finger ensures that phalange merging is not enabled by a misidentified, long thumb contact. If phalange merging were to be enabled without this multiple finger size requirement, any thumb contact traveling within a few centimeters behind a row of curled fingertips would merge with one of them.

Once the system has established from previous images that the fingers are at least beginning to flatten, several more measurements are made to ensure the fingertips are merged with only the proximal phalanges and not the forepalms or palm heels which could be flattening as well. First, the direction of search must be downward from the current group's local maximum. This helps to ensure that the search is coming from a fingertip, not the local maximum of the proximal phalange or forepalm. Additionally, the vertical location of the partial minimum being ignored must not be more than about 5 cm below the current group's local maximum, which corresponds to the maximum anatomical distance from the center of the distal

phalange (fingertip) to the center of the proximal phalange. The vertical location of the partial minimum must also be a couple centimeters above the horizontal dividing line of the hand's sloppy segmentation region, and only the first vertical partial minimum encountered will be ignored. These conditions all differentiate the proximal phalanges from the forepalms. Finally, the vertical partial minima should be in the same column or a column adjacent to the fingertip local maximum. If all of these conditions are met, search from a fingertip will continue past a vertical partial minimum to engulf a proximal phalange.

3.2.6.4 Contact Height Limitation Test

Despite all of the above precautions to prevent merging of fingers and palms, when the hand is flattened against the board very hard, images occasionally arise in which the only vertical minimum between a fingertip and palm heel is either in the sloppy segmentation region or between the fingertip and proximal phalange. In either case it will be ignored, merging a fingertip and palm heel into one group. The simplest way to prevent this is to observe that no hand part, finger, thumb or palm on the average adult hand is longer than about 8 cm. Therefore regardless of strict or sloppy segmentation region, search should stop and a vertical boundary be established whenever the next row to be searched is farther than about 8 cm from the row of the local maximum, $G_{localmax_{row}}$, where search started. This completely fills the chinks in the armor of the vertical contact boundary tests, preventing search from a palm heel local maximum from ever reaching an outstretched fingertip and vice versa.

3.2.6.5 Sloppy Segmentation Region Palm Heel Crease Test

The goal of sloppy region segmentation is to get as many of the electrodes influenced by the palms as possible into precisely two palm heels. This maximizes palm heel size and guarantees that the relatively fixed separation of about 5 cm

between palm heels can be measured. The relatively large palm heel sizes and the relatively large separations between palm heel centers help the identification system reliably distinguish fingers from palms. As long as the identification system has clear indication of two palm heels it cannot erroneously assign one of the palm heel identities to a thumb or pinky finger; *i.e.*, the combinatorial optimization constraints are stronger if the palm is consistently segmented into two parts. The palm heel separation is particularly critical in distinguishing a pair of adjacent fingertips starting a finger chord from a pair of palm heels when the fingertips are in the lower regions of the surface where palms are also expected. The identification system has means for identifying separate forepalms if necessary, but when possible they should be merged with one of the palm heels. Therefore, all partial minima in sloppy segmentation regions are ignored to maximize palm heel size except the large proximity valley or crease between the palm heels.

When the palm heels only touch the surface lightly, the proximity significance test will detect this crease and keep the two palm heel electrode groups separate. But as more pressure is applied, no electrode between them will remain at the background proximity level, and they will be separated only by a tall proximity valley with partial minima electrodes above the significance threshold. Fairly stringent tests are needed to detect this crease and not any other partial minima. First, anatomical constraints place the crease at least 2 cm from the center of either palm heel, so qualifying partial minima must be at a column j at least 2 cm from the column of the local maximum electrode where the search originated. Second, the proximity of the partial minimum electrode should be less than about half that of the local maximum, which requires that the proximity valley be fairly deep. Finally, the proximity valley must be fairly wide, so the partial minimum must extend past electrodes in next nearest neighbor columns, $S_{i,j-2}[n] > S_{i,j-1}[n] > S_{i,j}[n] < S_{i,j+1}[n] < S_{i,j+2}[n]$. These conditions allow horizontal boundaries between palm heels to be established in all rows which

the crease crosses. The only time the crease is not detected is when the entire center of the palm is pushed hard onto the surface, but when this is done the palm contacts are so large that the palm heel separation measurement is unimportant.

3.2.7 Combining Overlapping Groups

In sloppy segmentation regions it is possible for groups to overlap significantly because partial minima between local maxima do not act as boundaries. Typically when this happens the overlapping groups are part of a large fleshy contact such as a palm which, even after smoothing, has multiple local maxima. However, it is also necessary to get rid of the separate group around the local maxima of proximal phalanges when these are subsumed by fingertip groups. In the interest of presenting only one group per distinguishable fleshy contact to the rest of the system, the group combination stage of Figure 3.2 combines overlapping groups into single supergroups before parameter extraction.

Two groups are defined to be overlapping if the search originating local maximum electrode of one group is also an element of the other group. Two groups are connected if there is a sequence of overlapping groups between and including them: *i.e.*, overlap is transitive. A set of connected groups $\{G1..GN\}$ is combined into a supergroup GS by forming the semi-convex hull of the connected groups:

$$GS_{toprow} = \max_K GK_{toprow} \tag{3.5}$$

$$GS_{botrow} = \min_K GK_{botrow} \tag{3.6}$$

$$GS_{firstcol[i]} = \min_K GK_{firstcol[i]} \; \forall i : \; GS_{botrow} <= i <= GS_{toprow} \tag{3.7}$$

$$GS_{lastcol[i]} = \max_K GK_{lastcol[i]} \; \forall i : \; GS_{botrow} <= i <= GS_{toprow} \tag{3.8}$$

The semi-convex hull can include electrodes which are not part of any of the connected groups if a horizontal concavity lies between two of the connected groups, as occurs in Figure 3.19a). After consolidation into the supergroup is finished, the original connected set of groups $\{G1..GN\}$ is deleted.

3.2.8 Extracting Group Parameters

The last stage of the segmentation process extracts shape, size, and position parameters from each electrode group. Group centroid reflects hand contact position, and changes in contact position between successive images reflect finger velocity. The identification system utilizes geometric features of contacts such as total group proximity, eccentricity, and orientation to help distinguish finger, palm, and thumb contacts.

Though image smoothing does not affect the center of mass of a contact's proximity signal as measured over the whole image, it can diffuse some of the contact's signal outside of a group boundary into adjacent contacts or the image background. Diffusion can also increase the apparent radius or fitted ellipse axis lengths of a group. Therefore, to prevent biases during extraction, the parameterization process utilizes the undiffused image proximities $E_{ij}[n]$ instead of the smoothed proximities $S_{ij}[n]$ for all computations.

3.2.8.1 Centroid Computation

Given that G_E is the set of electrodes in group G, let $e_z = E_{e_i,e_j}[n]$ be the unsmoothed proximity of an electrode or pixel e, and let e_x and e_y be the coordinates on the surface of the electrode center in centimeters. To give a basic indicator of group position, the proximity-weighted center, or centroid, is computed from positions and proximities of the group's electrodes:

$$G_z = \sum_{e \in G_E} e_z \tag{3.9}$$

$$G_x = \sum_{e \in G_E} \frac{e_z e_x}{G_z} \tag{3.10}$$

$$G_y = \sum_{e \in G_E} \frac{e_z e_y}{G_z} \tag{3.11}$$

Note that since the total group proximity G_z integrates proximity over each pixel in the group, it depends upon both the size of a hand part, since large hand parts

tend to cause groups with more pixels, and upon the proximity to or pressure on the surface of a hand part.

Appendix B contains nonlinear interpolation methods used to ameliorate vertical interpolation biases and oscillations caused by the parallelogram electrodes.

3.2.8.2 Ellipse Fitting

While the user typically will not vary contact shape or orientation intentionally, such parameters will assist finger and hand identification in Chapter 4. Since most groups are convex, their shape is well approximated by ellipse parameters. The ellipse fitting procedure requires a unitary transformation of the group covariance matrix G_{cov} of second moments G_{xx}, G_{xy}, G_{yy}:

$$G_{cov} = \begin{bmatrix} G_{xx} & G_{xy} \\ G_{yx} & G_{yy} \end{bmatrix} \tag{3.12}$$

$$G_{xx} = \sum_{e \in G_E} e_z (G_x - e_x)^2 \tag{3.13}$$

$$G_{yx} = G_{xy} = \sum_{e \in G_E} e_z (G_x - e_x)(G_y - e_y) \tag{3.14}$$

$$G_{yy} = \sum_{e \in G_E} e_z (G_y - e_y)^2 \tag{3.15}$$

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

$$G_{major} = \sqrt{\lambda_0} \tag{3.16}$$

$$G_{minor} = \sqrt{\lambda_1} \tag{3.17}$$

$$G_\theta = \arctan\left(\frac{\lambda_0 - G_{xx}}{G_{xy}}\right) \tag{3.18}$$

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

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

$$G_\epsilon = \frac{G_{major}}{G_{minor}} \tag{3.19}$$

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

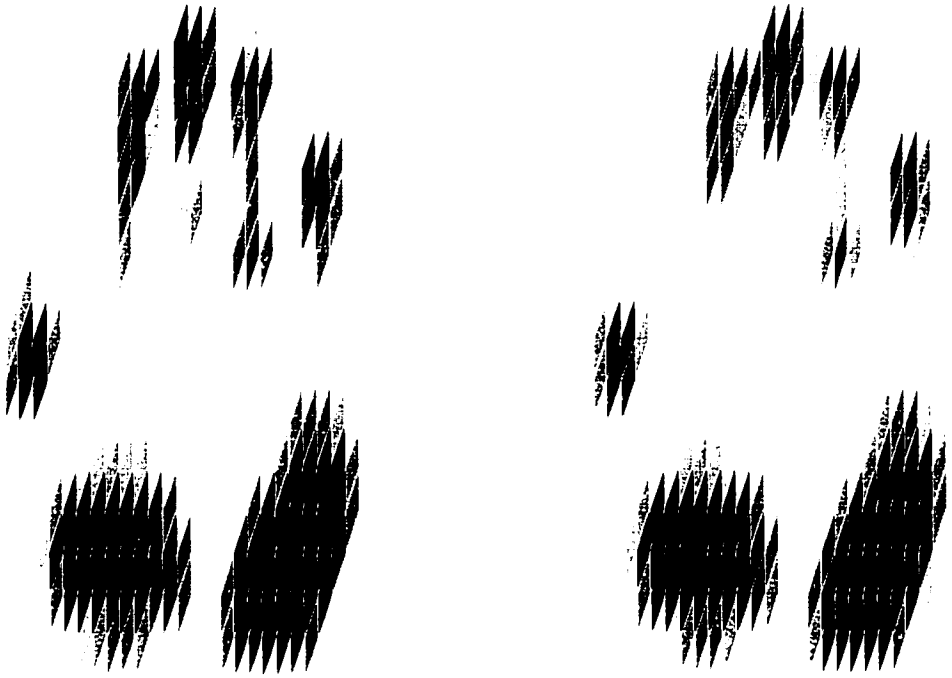
$$G_z := G_z / Z_{averageFingertip} \tag{3.20}$$

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

3.2.9 Performance of the Segmentation Methods

Daily typing and chordic manipulation on the MTS by the author has verified that segmentations are flawless for the most commonly utilized hand configurations, at least as performed by a skilled operator. Any quantitative evaluation of segmentation performance would depend highly on the relative frequency of easily segmented versus difficult to segment hand configurations which arise in the application being studied. Rather than attempt an application and operator-dependent quantitative evaluation, this section presents a sampling of hand configurations which illustrate the importance of the more complex segmentation rules and the rare cases in which segmentation fails.

A flattened hand with fingers squeezed together includes most of the contact juxtapositions which require proper alignment of the sloppy segmentation regions for correct segmentation. Figure 3.6 includes both the unsmoothed and diffused



a) b)

Figure 3.6: Unsmoothed a) and diffused b) proximity images of a flattened hand with the fingers squeezed against one another rather than spread out. Note that smoothed proximities of pixels near the center of each contact are more even but proximity has also bled outward around contact edges, as is especially noticeable for the thumb contact.

images of a flattened, squeezed hand. Note that the fingers are very close together, the proximal phalanges are touching the surface, and the outer palm heel causes two local maxima (darkest electrodes at lower left and upper right of contact) which are not merged by smoothing. Only the center of the palm and the forepalms remain suspended above the surface.

Note that usually each palm heel will only cause one local maximum, especially in the smoothed image. Multiple maxima which are distinct enough to remain separate during diffusion only appear sporadically due to odd distributions of hand pressure. Several images which only had one local maximum per palm heel were discarded before a hand posture was found which produced the dual maxima of Figure 3.6. Nevertheless, multiple maxima appear often enough that it is worthwhile to combine overlapping groups whenever possible. This avoids confusing the identification system with more than two palm heel groups per hand and avoids additional smoothing which could jeopardize segmentation of adjacent fingertips.

To demonstrate the types of segmentation failures which can occur when the hand position estimate is incorrect, the image of Figure 3.6 was segmented once with the strict segmentation region covering the whole image and again with the sloppy segmentation region covering the whole image. The former case will show what can happen when the palms unexpectedly touch down in the strict segmentation region, and the latter case will show what happens when fingertips unexpectedly touch down in the sloppy segmentation region, contradicting the last known hand position retained by the hand position estimator. Figure 3.7 displays segmentation maps in which electrodes numbered the same are members of the same segmentation group: the ordering of the numbers is arbitrary. Note that these segmentation maps and all that follow include only the final combined supergroups, not individual overlapping groups before they are combined.

Figure 3.7a shows the segmentation map obtained when the whole hand is processed with strict segmentation rules, *i.e.*, when the sloppy segmentation region is moved off the palms down to the lower right corner of the image. Since strict segmentation rules establish boundaries along the proximity valley between the dual outer palm heel local maxima, preventing each local maximum's group from overlapping the other, the outer palm heel contact remains erroneously broken into two

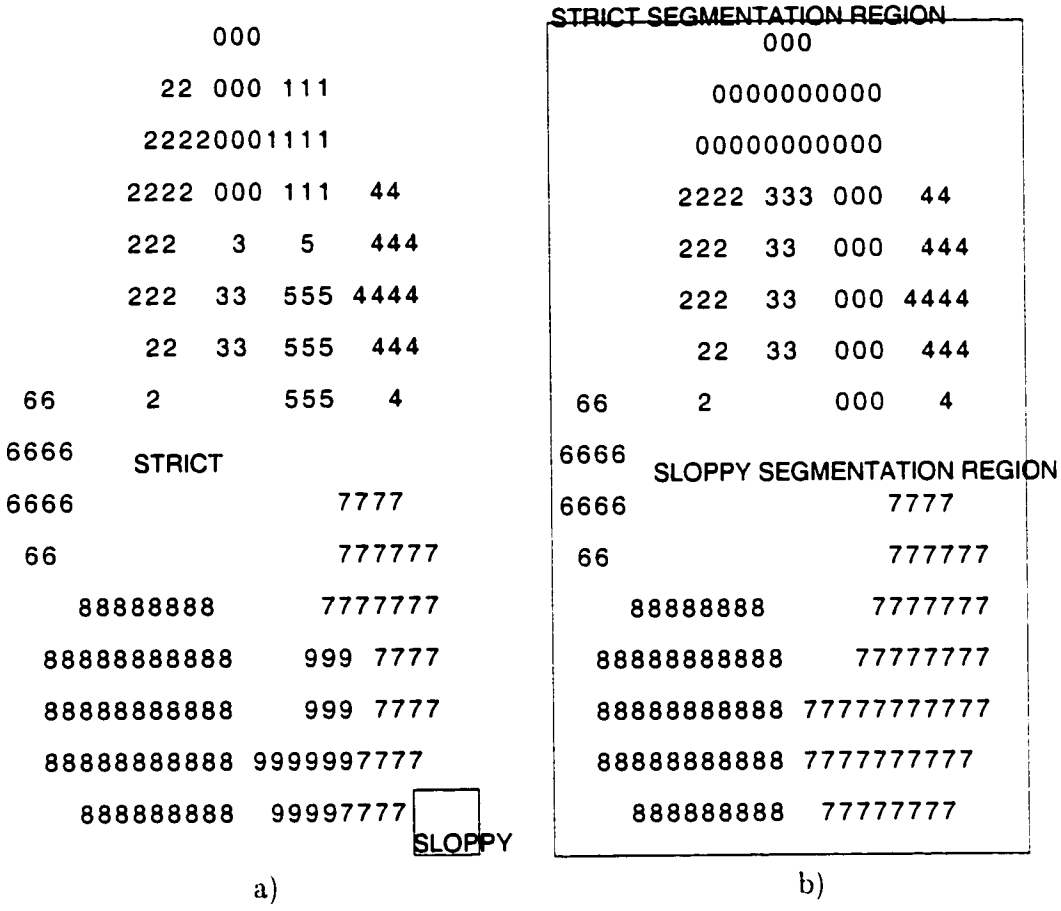


Figure 3.7: Segmentation results for the flattened hand using either a) strict segmentation rules for both fingers and palm heels or b) using sloppy segmentation rules for the whole hand. In a) the outer palm heel contact erroneously maintains two separate groups corresponding to the dual local maxima, and some of the electrodes at the edge of the palm heel contacts are excluded from all groups. Tolerance of flattened finger vertical minima is also disabled, causing splits in the segmentation between proximal and distal portions of the middle and ring fingers. In b) the lack of horizontal partial minimum detection causes the index, middle, and ring fingertips to erroneously merge into one group. Electrodes influenced by the thumb or pinky are grouped correctly in both a) and b).

groups. Comparing the size of the palm contacts of the original image (Figure 3.6a) to the segmentation map suggests that the strict segmentation rules have caused a few palm contact electrodes to be entirely left out of all the palm groups. Though not many electrodes have been excluded for this example image, sometimes this can cause the extracted group size and total proximity parameters to be erroneously low, making it harder for the identification system to distinguish fingertips from palms when the palms unexpectedly escape the sloppy segmentation region. The improper alignment of the sloppy segmentation region also disables tolerance of flattened finger vertical minima. This in turn causes proximal phalanges with distinct local maxima (those from the middle and ring fingers in this case) to fail to merge with their respective fingertip groups.

Figure 3.7b shows the opposite extreme in which the whole hand lies within the sloppy segmentation region and is segmented with sloppy segmentation rules. In this case the palm is properly segmented into two large heel groups which contain all palm contact electrodes. This occurs because sloppy segmentation rules allow the groups from each of the outer palm heel dual local maxima to overlap and thus be combined via their semi-convex hull. However, since sloppy segmentation rules do not include horizontal partial minima tests, the horizontally adjacent index, middle, and ring fingertips merge. Such incorrect fingertip merging can occur any time the fingertips touch down squeezed-together in a sloppy segmentation region. The proximal phalanges of the index and middle fingers remain separate from the merged fingertip group not because of the vertical minima between fingertips and proximal phalanges but because the semi-convex search pattern can only follow one vertical offshoot from each row, and in this case the search always traversed the proximal phalange of the ring finger.

Figure 3.8 shows the segmentation map when the sloppy segmentation region is properly aligned on the palm contacts. This segmentation region alignment or

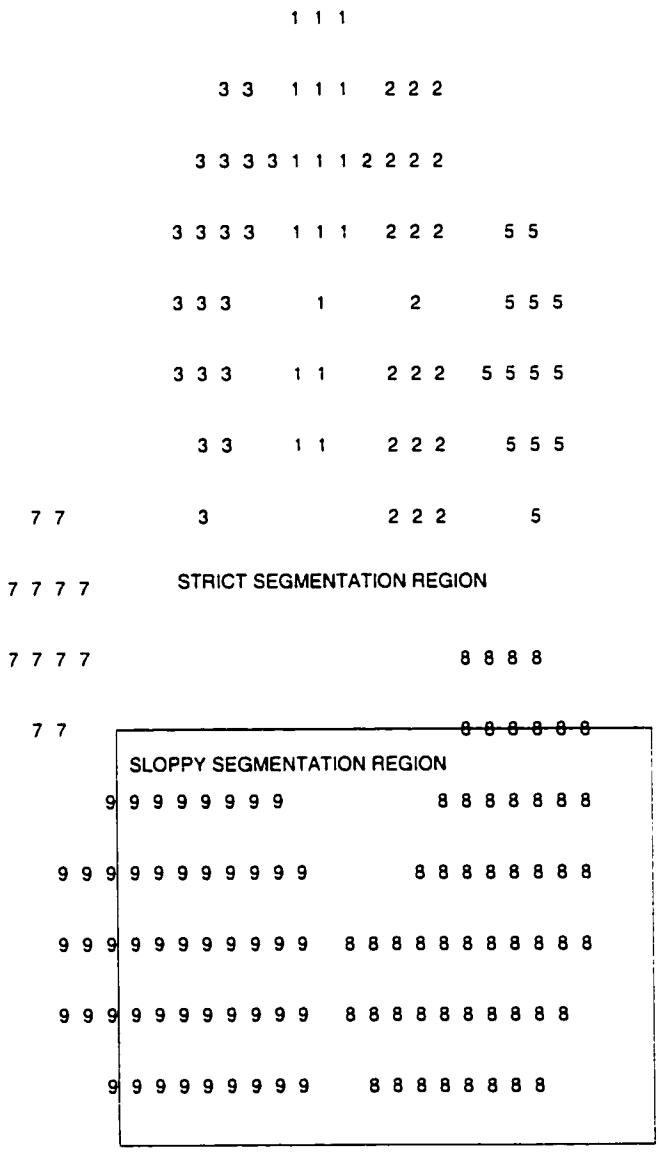


Figure 3.8: The correct segmentation for the flattened hand obtained by applying sloppy segmentation rules in the box around the palm heels and strict segmentation rules for the fingers. All electrodes proximal to the palm heels are included in their groups, yet the palm groups are split by the crease halfway between the palm heels. The whole of each finger, *i.e.*, both the distal tip and proximal phalange, combines into one group separate from the adjacent fingers.

any alignment within a couple centimeters of it achieves the correct segmentation for both fingers and palm heels. By the time the whole hand has flattened onto the surface, finger identifications and thus the hand position estimate will surely have stabilized to their correct values, invariably producing this correct segmentation under actual operating conditions.

In contrast to the squeezed, flattened hand which requires proper alignment of the sloppy region for correct segmentation, the default or neutral hand posture of Figure 2.8 is segmented correctly regardless of where strict or sloppy rules are applied. The segmentation map for the whole hand in the sloppy region is shown in Figure 3.9, but the same map is obtained with the sloppy region over palms only or with strict segmentation everywhere, with the exception that with strict segmentation of the palm heels a few of the electrodes on the periphery of the palm contacts can be excluded from the palm heel groups. The image can be segmented correctly regardless of segmentation region alignment because each distinguishable contact has only one local maximum and the contacts are all separated from one another by electrodes at background proximity levels. Thus the proximity significance test common to both strict and sloppy segmentation regions is sufficient to establish group boundaries. Note also that this easily segmented hand posture and variations upon it comprise the most commonly performed hand configurations.

A rotated variation (Figure 3.10) of this neutral hand configuration demonstrates the need for diagonal partial minima detection. Sloppy segmentation (Figure 3.11a) of the diagonally adjacent fingertips causes them to be merged into two groups. Strict segmentation of them (Figure 3.11b) detects the diagonal partial minima between them and keeps them in four separate groups. As will be shown in Figure 3.12, any further rotation causes fingertips to merge despite strict segmentation.

In Figure 3.12 the hand is rotated a full 90° from the neutral or default pos-

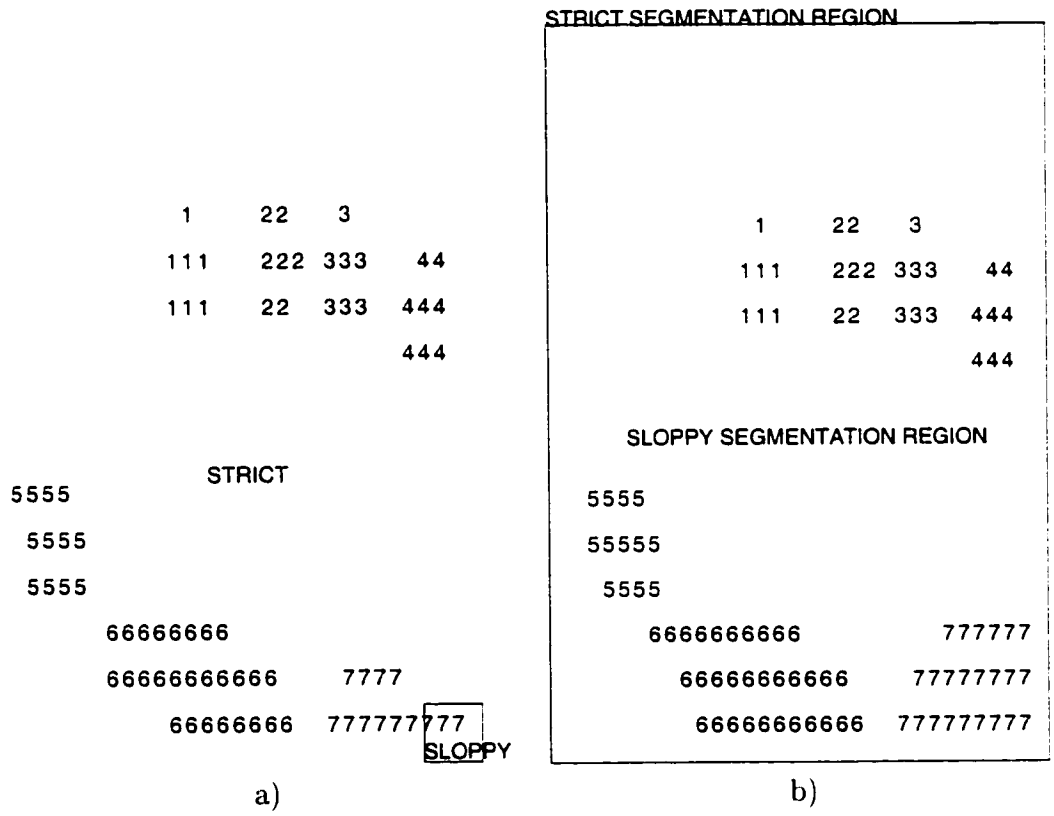


Figure 3.9: This correct segmentation of the neutral hand posture (Figure 2.8) is obtained regardless of where strict a) or sloppy b) segmentation rules are applied since contacts are relatively small and well-separated.

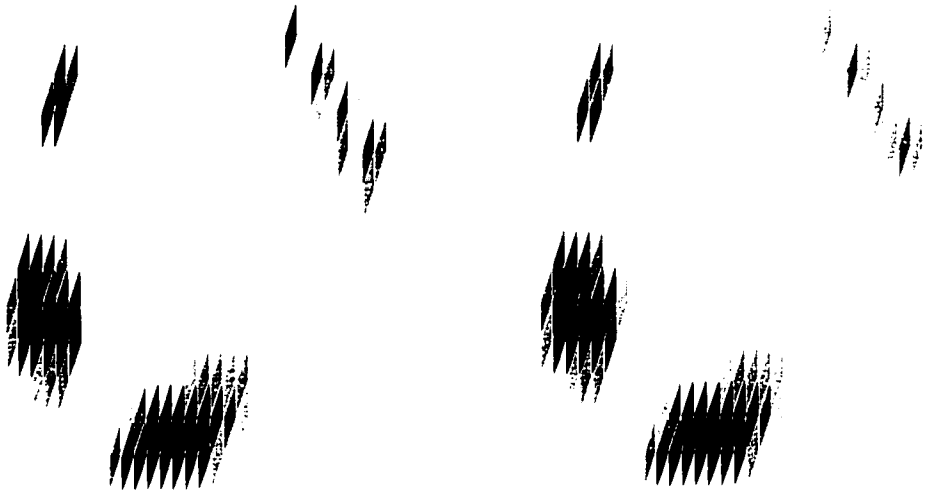


Figure 3.10: Unsmoothed a) and diffused b) proximity images of a partially closed right hand rotated clockwise 45°.

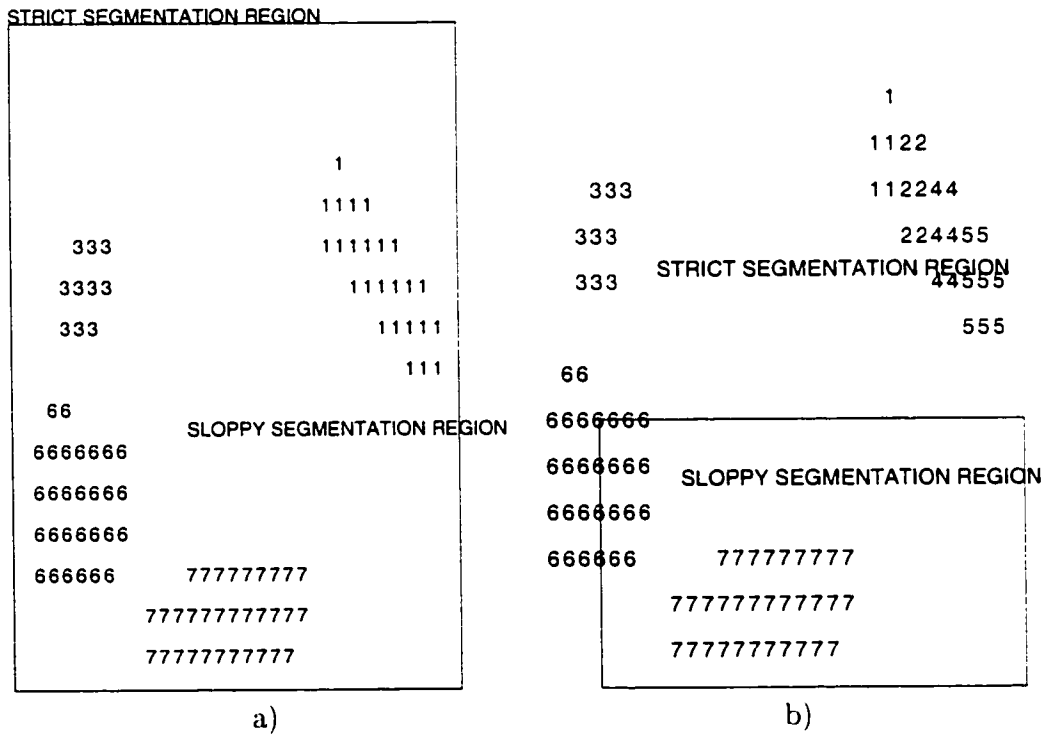
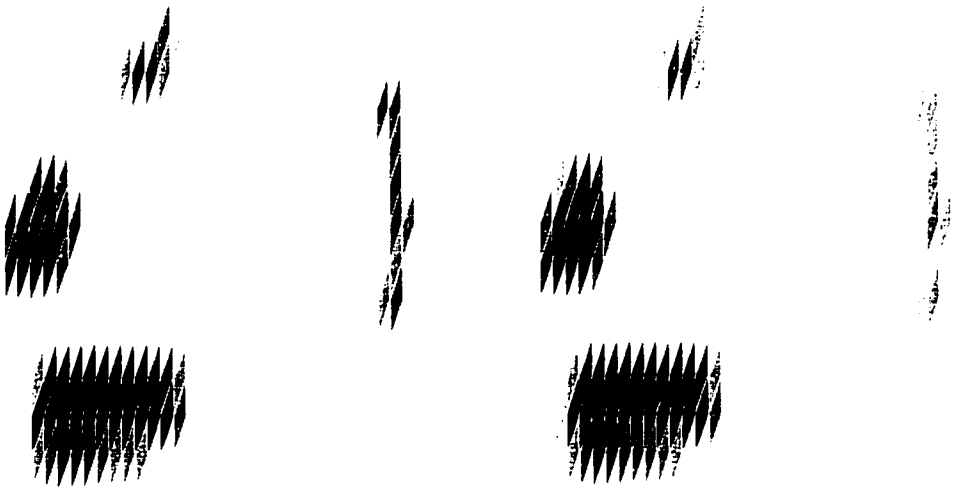


Figure 3.11: Sloppy segmentation a) of fingertips in a slanted row causes some of them to be merged. However, the diagonal minima tests of strict segmentation b) keep fingertip groups properly separated even when the row of fingertips is slanted as much as 45°.



a) b)

Figure 3.12: Unsmoothed a) and diffused b) proximity images of a partially closed right hand rotated clockwise 90°, fully sideways.

ture. As the segmentation map with properly aligned sloppy regions in Figure 3.13 indicates, vertical smearing by the parallelogram electrodes hides the proximity valleys between the fingertips, causing unavoidable merging even with strict segmentation rules. Note that performing this sideways hand configuration from the normal

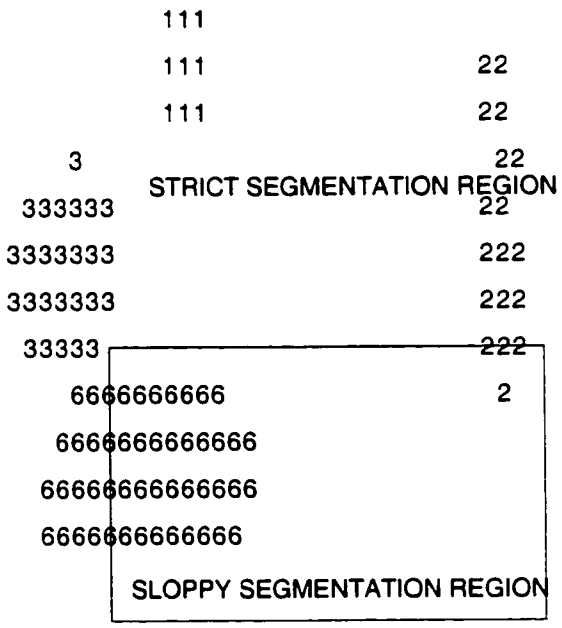


Figure 3.13: All segmentation rules fail to segment the column of fingertips because the vertical smearing by vertically interleaved parallelogram electrodes obscures the local proximity maxima normally caused by each fingertip.

sitting posture requires awkward contortions of the body, so the configuration would only be encountered regularly if the MTS was mounted on a pedestal so it could be approached from all sides.

The parallelogram electrode smearing also causes segmentation errors when the thumb passes just behind a fingertip. Figure 3.14a shows the thumb as close as

it can get to the index fingertip and still be segmented correctly with strict rules (Figure 3.14b). Figure 3.15a is an image of a closed hand with the thumb tip fully pushed up against the back of the fingertip. In this case the thumb tip and fingertip are so close that parallelogram electrode smearing obscures the vertical partial minima between them, causing them to be merged into one group (Figure 3.15b). Since the pen grip hand configuration (Section 2.3.4) contains this same juxtaposition of the thumb and index fingertip, it cannot be segmented properly with the current parallelogram electrode row spacing either. Note that if proximal phalange merging for flattened finger segmentation (Section 3.2.6.3) was erroneously enabled during segmentation of Figure 3.14a, the vertical partial minimum would have been ignored, causing thumb and fingertip to be merged as in Figure 3.15b.

Figure 3.17 is the correct segmentation map for the flattened hand with outstretched fingers of Figure 3.16. This map illustrates that a forepalm contact gets its own group if it has a separate local maximum and cannot be combined with the palm heels through mutual overlap. This is not considered an error because the identification system can reliable identify forepalms when the rest of the hand is flattened onto the surface. Note also that even though the proximity images show that the index finger is connected to the inner palm heel and the pinky finger is connected to the outer palm heel, the segmentation groups for fingers and palms remain separate.

Figure 3.18 exposes the shortcomings of flattened finger segmentation when the fingers are not oriented near vertical. If the fingers lie at a slant of more than about 25° off vertical, each finger begins breaking into multiple groups due to detection of diagonal partial minima instead of vertical partial minima. Figure 3.19b shows the resulting segmentation map with fingertip groups separated from proximal phalange groups even when the sloppy segmentation region is aligned properly. This failure is a consequence of the assumption that proximity ridges will always be

a)

```
 1    2
 11   22  333  44
55    22  33   44
55          3    4
555   STRICT SEGMENTATION REGION
555
```

```
SLOPPY SEGMENTATION REGION

 6666          777777
66666         7777777
6666666     77777777
 66666      77777777
```

b)

Figure 3.14: Unsmoothed proximity image and properly aligned segmentation map of a thumb passing about a centimeter behind the index fingertip.

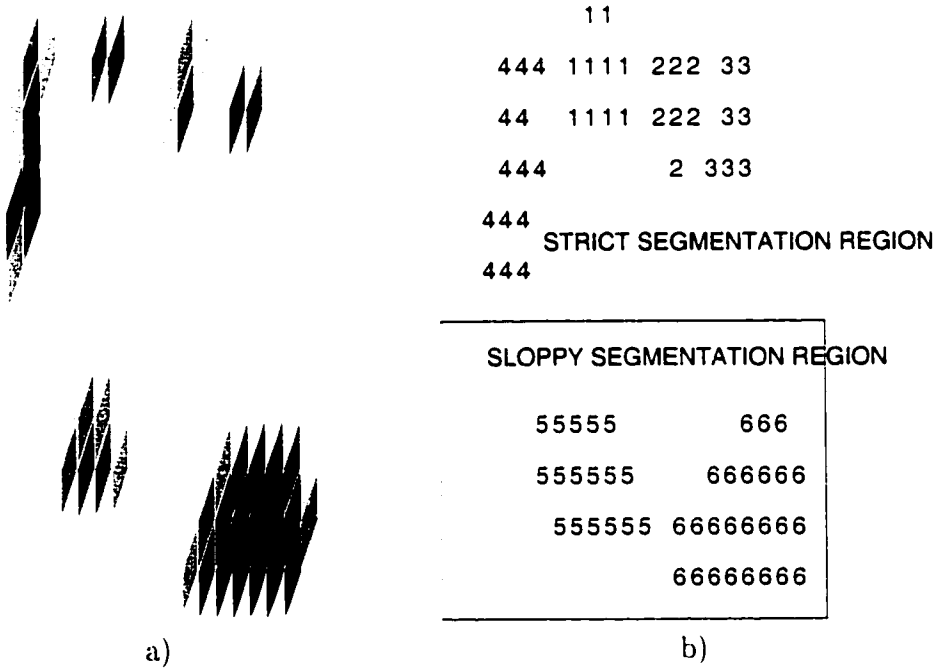


Figure 3.15: Unsmoothed proximity image and properly aligned segmentation map of a thumb touching the back of the index fingertip.

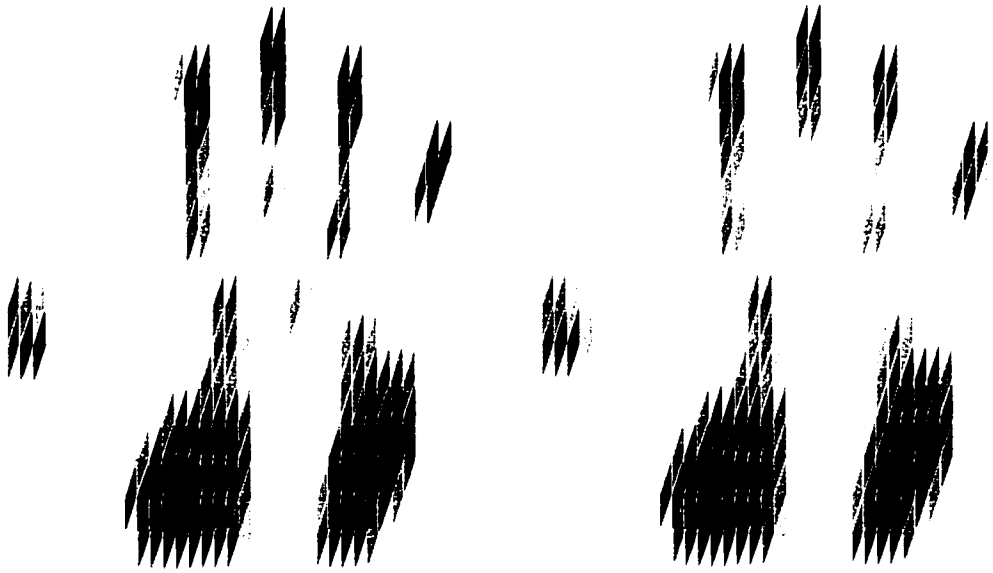


Figure 3.16: Unsmoothed a) and diffused b) proximity images of a right hand flattened onto the surface so hard that the forepalms are touching.

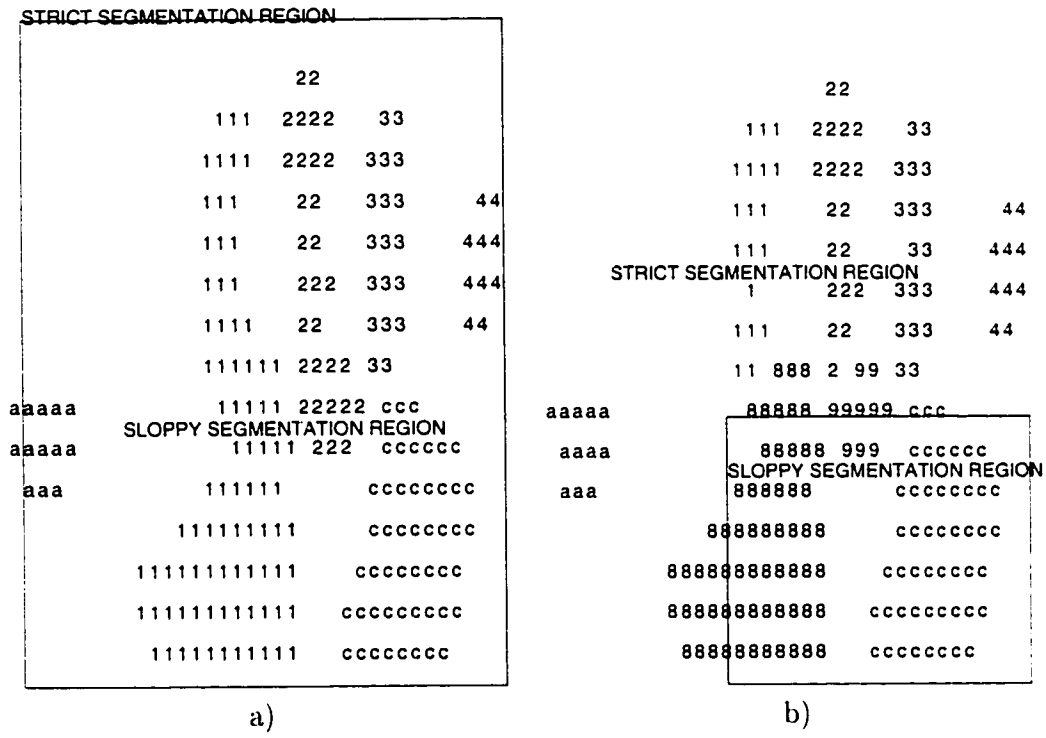


Figure 3.17: All sloppy segmentation a) of the flattened right hand of Figure 3.16. and the correct segmentation using properly aligned sloppy regions b).

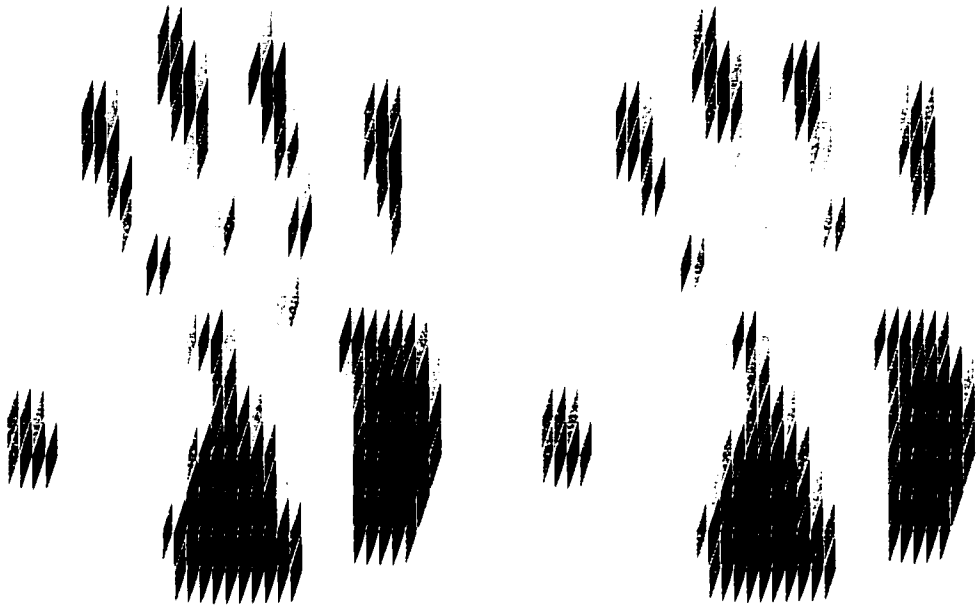


Figure 3.18: Unsmoothed a) and diffused b) proximity images of a flattened right hand rotated counter-clockwise about 30°.

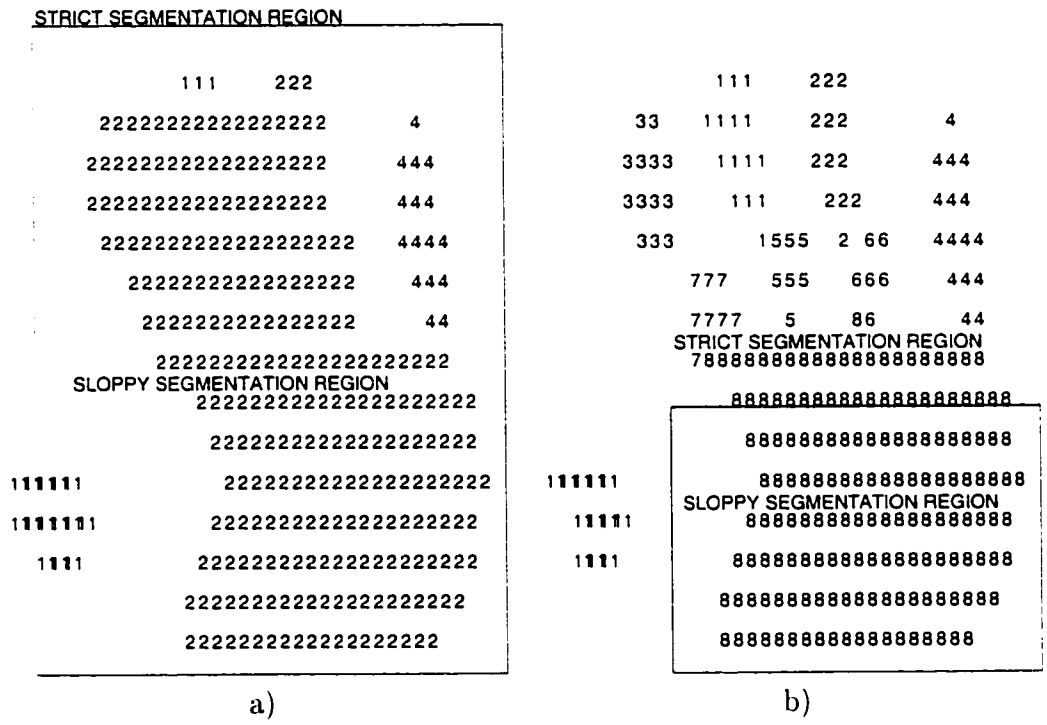


Figure 3.19: Segmentation maps for the rotated, flattened hand. Sloppy segmentation everywhere a) causes all groups to overlap. A single supergroup forms from the semi-convex hull around these groups and encloses the whole hand. With sloppy segmentation only over the palms b), the diagonal partial minima detection of strict segmentation keeps the fingertip and proximal phalange groups erroneously split.

connected through nearest diagonal neighbors. Sufficiently oblique ridges such as these which cause maxima in every other column are broken up by diagonal minima tests. Occasionally this type of failure splits a diagonally-oriented thumb pressed hard onto the surface into two groups. This failure would be less likely with a smaller row spacing which increases the angle between nearest diagonal neighbors to 45°. Combination via semi-convex hull of the entire flattened palm into a single group (Figure 3.19b) can be handled by the identification system and is not considered an error as long as the merging remains stable across successive proximity images.

Another measure of segmentation performance is the consistency of group membership across successive proximity images of a stationary hand. Segmentation consistency has a major impact on the path tracking to be discussed in section 3.3. If segmentations of a hand contact are not consistent across images, the measured hand contact centroid can erroneously shift between images and cause jitter in the contact velocity computed by the path tracking module. If large contacts suddenly merge or break apart in a new image the centroids may move so much that the path tracker concludes a hand part has lifted off or newly touched down. As long as these unstable hand parts are identified as forepalms or palm heels such spurious touchdowns will have no deleterious effects, but if the unstable hand parts are identified as fingers whose centroids are near keys of the key layout, the system can falsely interpret them as keypresses.

The segmentation of most unflattened hand postures is perfectly consistent, but when palms are fully flattened, forepalm-influenced electrodes can unstably shift between finger, palm heel, and independent forepalm groups, or the palm heels can unstably merge together for one image and break up in the next due to subtle changes in palm contact topology. Though such instabilities have been observed to perturb finger centroids enough to cause spurious key activations during hand flattening tests, full hand flattening does not occur during normal MTS operation

or hand resting. Therefore, like the other rare cases of segmentation failure, this problem will not be investigated further until it is known whether lower noise, higher resolution sensor arrays will make it disappear entirely.

3.3 Persistent Path Tracking

3.3.1 Introduction to the Path Tracking Problem

Electrode groups are transitory in the sense that the segmentation algorithm reconstructs them from scratch for each proximity image. It is then the responsibility of the path tracking process to chain together those groups from successive proximity images which correspond to the same physical hand contact. To determine where each hand part has moved since the last proximity image, the tracking algorithm must decide which current groups should be matched with which existing contact paths. As a general rule, a group and path arising from the same contact will be closer to one another than to other groups and paths. Also, biomechanical constraints on lateral finger velocity and acceleration limit how far a finger can travel between images. Therefore a group and path should not be matched unless they are within a distance known as the tracking radius of one another. Tracking performance can be improved by incorporating velocities measured along existing paths in previous images into the prediction of current surface contact location.

The path tracking process must also determine when a physical hand contact newly touches down or lifts off the surface. Since the typical lateral separation between fingers is greater than the tracking radius for reasonable image scan rates, finger touchdown and liftoff are easily detected by the fact that touchdown usually causes a new group to appear outside the tracking radii of existing paths, and liftoff will leave an active path without a group within its tracking radius. To prevent improper breaking of paths at high finger speeds, each path's tracking radius P_{rtrack} is made dependent on its existing speed and contact size.

Rubine [130] addressed the path tracking problem for up to three fingers detected by a "Sensor Frame" [107, 108]. He did not find it necessary to predict current contact locations from past path velocity or acceleration. He simply used the last known path position as the predicted position. Since Rubine only dealt with 3 fingers, *i.e.*, six possible one-to-one assignments of groups to paths, he simply evaluated all possible pairings and picked the one which minimized the sum of the distances between each path and its assigned group. However, counting all fingers, thumbs, and palm heel and forepalm contacts from both hands, up to 20 groups can be present at any one time on the MTS. Clearly the MTS cannot do a brute force enumeration of all 20! possible assignments, but the same assignment optimization algorithms invoked for finger identification in Section 4.4 could be utilized to efficiently minimize the assignment sum.

Any group-path pairing method must perform a global optimization in the sense that it must handle both the case that several groups are clustered around a single path, wherein only the closest group in the cluster should be assigned to the path, and the case that several paths are clustered around a single group, wherein only the closest path in the cluster should be assigned to the group. Thus the decision to pair a group and path cannot be made based solely upon the distance between them without considering the pairing distances of other groups or paths nearby. The pairing method presented here utilizes a rule related to the shared near neighbors clustering technique introduced by Jarvis and Patrick [72] instead of explicitly minimizing the sum of pairing distances. This rule only accepts assignments between groups and paths that are closest to one another. The nearest neighbor clustering rule and assignment sum minimization may produce slightly different pairings for very tight clusters of groups and paths, but given reasonable frame rates, adult finger spacings, and velocity-based path prediction, groups remain so much closer to their actual paths than other paths that the behavior of the

pairing method for marginal cases does not matter.

Consult Figure 3.20 for a summary of the steps in the path tracking process described below.

3.3.2 Prediction of Contact Location

Whether the prediction of path contact locations needs to include past path velocity depends on the minimum possible finger separation, maximum expected finger speed and acceleration, and the array scan or frame rate. If segmented properly, group centroids are usually separated by at least 1.5 cm. The MTS frame rate is currently 50 fps, but can easily be raised to 100 fps. Because hand and finger motions tend to be smooth, acceleration is quite low except at the beginning and end of slides. At typical mouse manipulation speeds a finger will travel less than 1 mm between frames. However, consideration of Fitts' Law [25] suggests hand pointing speeds can reach 180 cm/s. The fastest hand slides observed across the MTS have only reached 100 cm/s, but these still cause each finger to travel up to 2 cm between frames, a value comparable to the nominal fingertip separation. If a row of fingers quickly slides horizontally, one fingertip can thus appear right over the last known location of an adjacent fingertip unless this last known location is updated with past finger velocities.

Including previously measured velocity in the location prediction improves the prediction except when a finger suddenly starts or stops or changes direction. Since such high acceleration events occur less often than zero acceleration events, the benefits of velocity-based prediction outweigh the potentially bad predictions during finger acceleration. Let $P_x[n-1], P_y[n-1]$ be the position of path P from time step $n-1$ and $P_{vx}[n-1], P_{vy}[n-1]$ the last known velocity. The velocity-predicted path continuation is then:

$$P_{predx}[n] = P_x[n-1] + \triangle t P_{vx}[n-1] \tag{3.21}$$

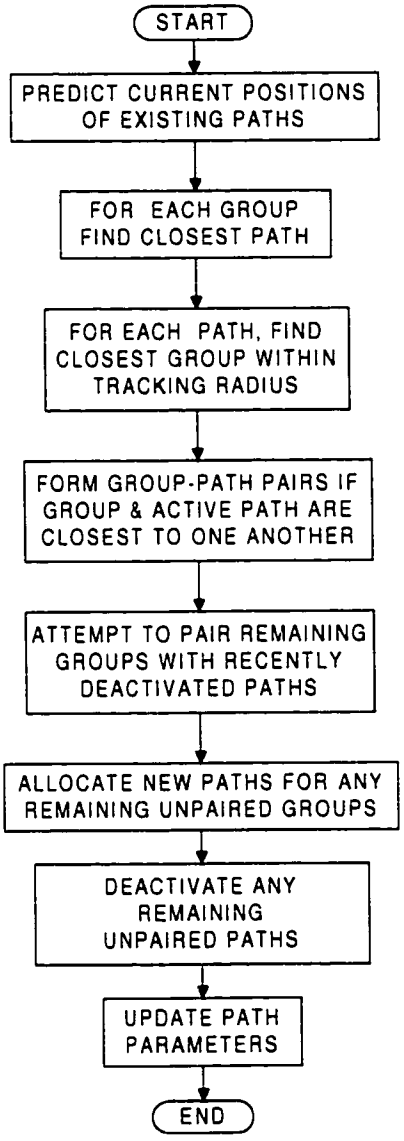


Figure 3.20: Flow chart summarizing the contact path tracking algorithm.

$$P_{predy}[n] = P_y[n-1] + \Delta t P_{vy}[n-1] \tag{3.22}$$

Possibly the reason Rubine found velocity-based predictions unnecessary was that the Sensor Frame had a higher frame rate or that he was only tracking finger motion with the hand in a relatively fixed position over a small area. The MTS is large enough for lateral slides of the whole hand and forearm. These easily reach speeds several times higher than individual finger motions from a stationary hand.

3.3.3 Mutually Closest Pairing Rule

Let the set of paths active in the previous image be $\mathcal{PA}$, and let the set of electrode groups constructed in the current image be $\mathcal{G}$. For each active group Gk, find the closest active path and record the distance to it:

$$Gk_{closestP} = \underset{Pl \in \mathcal{PA}}{\operatorname{argmin}}\, d^2(Gk, Pl) \qquad \forall\, Gk \in \mathcal{G} \tag{3.23}$$

$$Gk_{closestPdist^2} = \min_{Pl \in \mathcal{PA}} d^2(Gk, Pl) \qquad \forall\, Gk \in \mathcal{G} \tag{3.24}$$

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

$$d^2(Gk, Pl) = (Gk_x - Pl_{predx})^2 + (Gk_y - Pl_{predy})^2 \tag{3.25}$$

Then for each active path Pl, find the closest active group and record the distance to it:

$$Pl_{closestG} = \underset{Gk \in \mathcal{G}}{\operatorname{argmin}}\, d^2(Gk, Pl) \qquad \forall\, Pl \in \mathcal{PA} \tag{3.26}$$

$$Pl_{closestGdist^2} = \min_{Gk \in \mathcal{G}} d^2(Gk, Pl) \qquad \forall\, Pl \in \mathcal{PA} \tag{3.27}$$

A group Gk and path Pl are only paired with one another if they are closest to one another, *i.e.*, $Gk_{closestP}$ and $Pl_{closestG}$ refer to one another, and the distance between them is less than the tracking radius. The nominal tracking radius is about 1 cm. All of the following conditions must hold:

$$Gk_{closestP} \equiv Pl \tag{3.28}$$

$$Pl_{closestG} \equiv Gk \tag{3.29}$$

$$Pl_{closestGdist^2} < Pl^2_{rtrack} \tag{3.30}$$

Any active group which cannot be paired with an active path under these constraints is allocated a new path, representing touchdown of a new finger onto the surface. Any active path which cannot be so paired with an active group is deactivated, representing hand part liftoff from the surface.

To aid finger tap debouncing and detection of repetitive finger taps, it is useful to preserve continuity of path assignment between taps over the same location. When a new path needs to be allocated for an isolated group, *i.e.*, the group cannot be assigned to any active path, priority is given to any recently deactivated paths within the tracking radius of the group. The closest path which has been deactivated within the last second or so is reactivated, assigned to the group, and specially marked as reactivated. It's release time is stored in $P_{tpreviousrelease}$ to aid in detection of double-click timing. If no recently deactivated paths exist nearby, a totally new path is started.

3.3.4 Path Parameters

The final step of path tracking is to incorporate the extracted parameters of each group into its assigned path via standard filtering techniques. The MTS applies the simple autoregressive filter equations shown below to update the path position ($P_x[n]$, $P_y[n]$, $P_z[n]$), velocity ($P_v x[n]$, $P_v y[n]$), and shape $P_\theta[n]$, $P_\epsilon[n]$ parameters from corresponding group parameters. If a path P has just been started by group G at time step n, *i.e.*, a hand part has just touched down, its parameters are initialized as follows:

$$P_{press_t} = t \tag{3.31}$$

$$P_{press_x} = G_x \tag{3.32}$$

$$P_{press_y} = G_y \tag{3.33}$$

$$P_x[n] = G_x \tag{3.34}$$

$$P_y[n] = G_y \tag{3.35}$$

$$P_z[n] = G_z \tag{3.36}$$

$$P_\theta[n] = G_\theta \tag{3.37}$$

$$P_\epsilon[n] = G_\epsilon \tag{3.38}$$

$$P_{vx}[n] = 0 \tag{3.39}$$

$$P_{vy}[n] = 0 \tag{3.40}$$

$$P_{vz}[n] = G_z/\triangle t \tag{3.41}$$

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

$$\begin{aligned}
P_x[n] &= G_\alpha G_x + (1-G_\alpha)(P_{predict_x}[n-1]) && (3.42)\\
&= G_\alpha G_x + (1-G_\alpha)(P_x[n-1] + \triangle t P_{vx}[n-1]) && (3.43)\\
&= G_\alpha G_x + (1-G_\alpha)(2P_x[n-1] - P_x[n-2]) && (3.44)\\
P_y[n] &= G_\alpha G_y + (1-G_\alpha)(P_{predict_y}[n-1]) && (3.45)\\
&= G_\alpha G_y + (1-G_\alpha)(P_y[n-1] + \triangle t P_{vy}[n-1]) && (3.46)\\
&= G_\alpha G_y + (1-G_\alpha)(2P_y[n-1] - P_y[n-2]) && (3.47)\\
P_x[n] &= G_\alpha G_x + (1-G_\alpha)(P_{predx}[n-1]) && (3.48)\\
P_y[n] &= G_\alpha G_y + (1-G_\alpha)(P_{predy}[n-1]) && (3.49)\\
P_z[n] &= G_\alpha G_z + (1-G_\alpha)P_z[n-1] && (3.50)\\
P_\theta[n] &= G_\alpha G_\theta + (1-G_\alpha)P_\theta[n-1] && (3.51)\\
P_\epsilon[n] &= G_\alpha G_\epsilon + (1-G_\alpha)P_\epsilon[n-1] && (3.52)\\
P_{vx}[n] &= (P_x[n] - P_x[n-1])/\triangle t && (3.53)\\
P_{vy}[n] &= (P_y[n] - P_y[n-1])/\triangle t && (3.54)\\
P_{vz}[n] &= (P_z[n] - P_z[n-1])/\triangle t && (3.55)
\end{aligned}$$

It is also useful to compute the magnitude $P_{speed}[n]$ and angle $P_{dir}[n]$ from the velocity vector $(P_{vx}[n], P_{vx}[n])$. The filters are first or second order autoregressive where the amount of filter memory increases with $(1-G_\alpha)$. The second-order position

filter has zero tracking delay during constant velocity motion, but position filter output can lag behind actual finger position during accelerations or slightly overshoot during decelerations. Since the reliability of position measurements increases considerably with total proximity P_z, the low-pass filter pole G_α is decreased for groups with total proximities lower than normal:

$$G_\alpha \approx \begin{cases} .8 & \text{if } P_z > 1 \\ .4 + .4P_z & \text{else if } P_z <= 1 \end{cases} \tag{3.56}$$

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

Since the MTS does not yet recognize complex path gestures or handwriting, it does not need to store all past points of each finger trajectory. However, to aid detection of finger chords and taps, the MTS does retain temporal markers with finger position and size parameters from important stages of the path life cycle such as finger touch down ($P_{press_t}, P_{press_x}, P_{press_y}$), stabilizations in finger proximity ($P_{peak_t}, P_{peak_x}, P_{peak_y}, P_{peak_z}$), and finger liftoff ($P_{release_t}, P_{release_x}, P_{release_y}$). As a finger taps the surface, proximity quickly rises and then plateaus until the finger lifts off. To ensure the peak proximity marker is available fairly soon after touchdown, its parameters are actually captured when the rate of change of proximity $P_{vz}[n]$ falls below a very small positive threshold, indicating the beginning but not necessarily the peak of a proximity plateau. These important finger activity markers would become inconsistent if a faulty path tracking algorithm inadvertently reshuffled paths.

3.3.5 Path Tracking Results

The path tracking process described above performs flawlessly for normal operating hand speeds and a frame rate of 50 fps. It will chain groups caused by the same hand contacts indefinitely as they slide across the surface, starting new

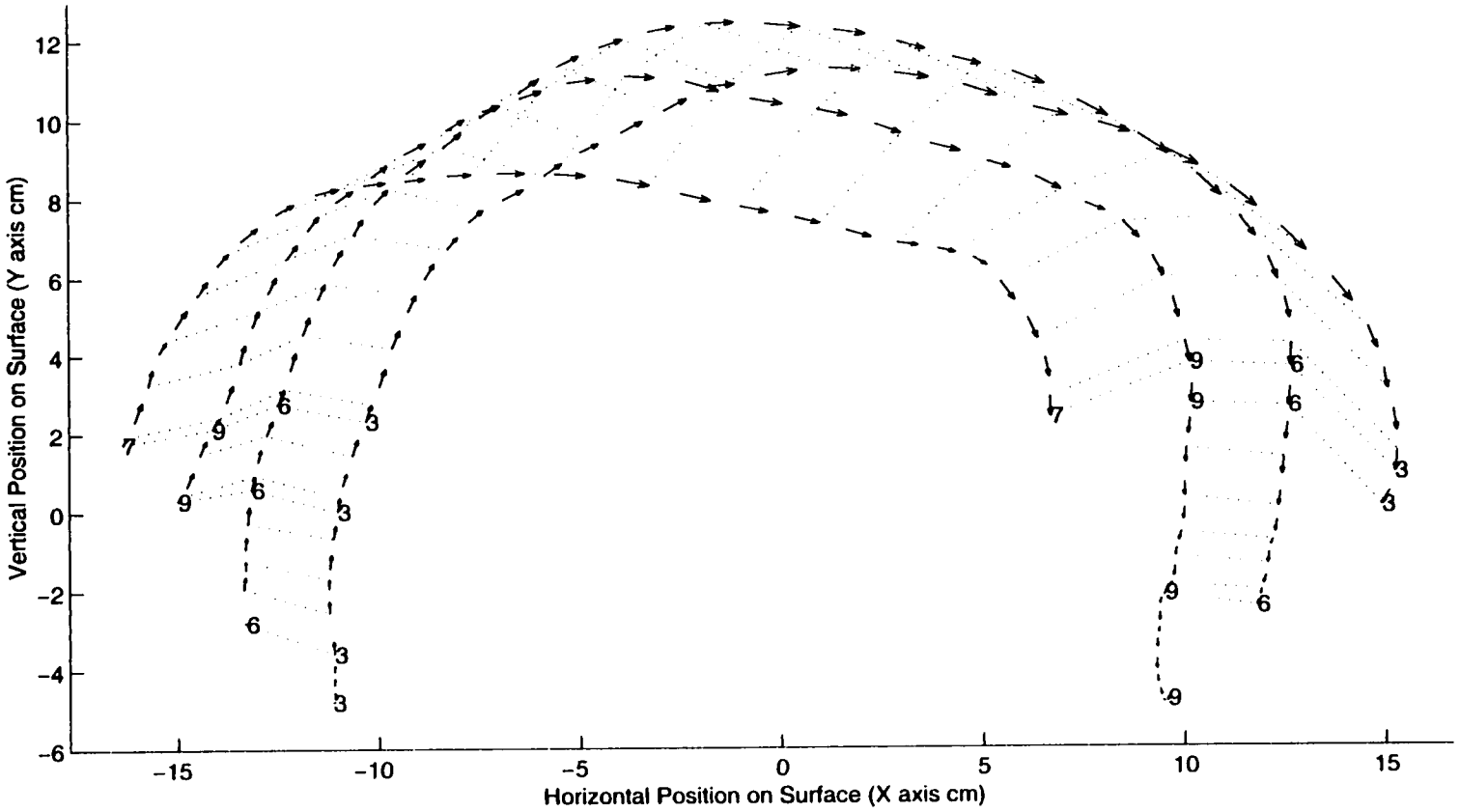


Figure 3.21: Trajectories of four left hand fingertips touching down asynchronously on the lower left of the surface and sliding in an arc to lift off at the lower right. Arrow tips and tails represent filtered contact positions $(Pi_x[n], Pi_y[n])$ in successive proximity images n. The four path indices i are printed at touchdown and liftoff to prove continuity of the tracking algorithm. Index order is arbitrary. To indicate relative timing of the trajectories, dotted lines connect all current contacts at certain timesteps.

paths only due to extreme deceleration or finger touch down. Figure 3.21 gives a representative example of tracking for four sliding fingers. The only case in which paths are known to get reshuffled or erroneously broken is when forepalm contact segmentations for fully flattened hands become unstable.

3.4 Summary

The centers of hand contacts invariably cause local maxima in smoothed proximity images, so proximity image segmentation algorithms can be very efficient by starting segmentation group growth at significant local maxima. Since most contacts are convex, semi-convex group search patterns and data structures can efficiently encapsulate each contact's electrodes without including electrodes from neighboring contacts. Though in the most commonly used hand configurations contacts are well-separated and easy to segment, other important configurations contain contacts separated only by ambiguous partial minima. Contextual feedback of past hand position and flattened finger status is necessary to decide which of these partial minima should be treated as contact edges. Careful tuning of the edge detection rules and segmentation regions so that electrode groups reliably correspond to functionally distinct hand parts such as fingertips, thumbs, or palm heels allows the contact identification system of the following chapter to be relatively simple yet robust. Vertical smearing by interleaved parallelogram electrodes causes most of the rare segmentation failures, so these failures can be best addressed by improvements in vertical sensor density which phase out parallelogram electrodes, rather than by improvements in the segmentation algorithms.

Path tracking is not terribly difficult given typical finger spacings and frame rates above 50 fps, especially if velocity-based prediction is utilized. Nevertheless, its flawlessness also simplifies the jobs of the identification system in the following chapter and the synchronization system in Chapter 5. Once the finger and hand identification system has correctly assigned an identity to a contact, continuation

of that contact's path by the path tracking system will be sufficient to retain the identity until the contact lifts off, thus avoiding computationally expensive and potentially destabilizing reidentification in every successive proximity image. Accurate contact tracking also ensures the validity of the path life cycle markers used to debounce key presses and detect synchronous touchdowns or liftoffs of multiple fingers.

Chapter 4

FINGER IDENTIFICATION AND HAND POSITION ESTIMATION

This chapter will tackle the main shortcoming of capacitive proximity sensing, establishing fingertip and palm heel identities for surface contacts when most of the intervening hand structure is missing from the proximity images. Finger identification has not been necessary for interactions with commercial touchpads or touchscreens because most cannot track more than one finger at a time, and those that can detect two or three fingers only utilize the contact count. If the MTS was only to be used for typing, trying to identify each surface contact might not be worthwhile because key taps should be distinguished by their spatial location, not which finger strikes the key. But recognition of the rich, bimanual chordic manipulations demonstrated in Chapter 5 demands reliable clustering of surface contacts with their originating hand as well as reliable finger ordering and thumb identification within each hand. Also, palm contacts must be properly identified so that operators can safely rest the entire hand anywhere on the surface without palm motion being misinterpreted as typing or chordic manipulation.

This chapter begins with a discussion of how the identification problem for the MTS relates to previous research on recognition of hand gestures captured by optical sensing systems. Then the chapter presents the algorithms and biomechanical constraints which the MTS utilizes for hand position estimation, finger identification, and hand identification. Finger identification is posed as an assignment

problem which must optimally match finger contacts to a ring of finger attractor points. The identification of single, isolated contacts is solely determined from the weighted Voronoi diagram formed by the attractor points. It is shown that with a contact-attractor distance-squared cost metric, the assignment algorithm effectively sorts multiple hand contacts around the ring with respect to the attractor orderings and inter-attractor angles, regardless of whether the attractor ring is at all centered on the contact cluster. Results will illustrate how intricate feedback between tracking system modules over successive scanning cycles causes quick convergence upon correct identifications for the comfortable range of hand motions.

Remember during the following discussions that the term "fingertip" can refer to any finger except the thumb. Thus the thumb never counts as a "fingertip," though the thumb is considered one of the five fingers. Also, palm heels never count as fingertips or fingers.

4.1 Hand Gesture Recognition

Techniques as diverse as rule-based inference [121], elastic graph matching [146], and Kohonen Feature Maps [16] have been utilized to recognize the free-space hand postures of various sign languages. Because the objective of these systems has been to recognize an alphabet of communicative gestures, most of the systems have only been designed to recognize representative static hand postures, though Wexelblat [163] and Boehm *et al.* [16] have concentrated on recognition of dynamic, continuous gestures sensed by DataGloves [148].

4.1.1 Communicative Gestures versus Manipulative Gestures

In contrast to these communicative gesture recognizers, the primary objective of the MTS is to recognize a variety of simple control gestures for manipulating graphical objects in two or three dimensions. The recognition task can then be separated into three parts: recognizing which graphical manipulation channel the

operator is selecting, tracking the manipulative hand and finger motions, and continuously extracting multiple degree-of-freedom (DOF) object control signals from the tracked motions. Compared to a mouse, recognizing the channel selection corresponds to checking which buttons are being held down, tracking hand motions corresponds to measuring the rotation of the mouse ball, and extracting independent motion components has no analogue since measurements from orthogonal ball rotation sensors are already independent. This chapter will tackle the channel selection and tracking problems in the context of proximity imaging devices by reliably attaching finger and hand identities to the contact paths constructed in the previous chapter. Multi-DOF extraction will not be addressed until Chapter 5.

The graphical manipulation objective places very different requirements on the recognition algorithm than the symbolic communication objective. A central tenet of this dissertation is that operator efficiency in rich graphical environments can increase tremendously by having several manipulation channels which the operator can switch between instantaneously with simple changes in hand configuration. However, not nearly as many channels or distinct hand configurations are necessary for improved graphical manipulation as are necessary to support a symbolic gesture language containing dozens of distinct signs. Operators may only be willing to memorize the hand configurations which select a few different graphical manipulation channels, and it may be hard to memorize mappings for or even imagine uses for more than a dozen channels. Fundamental manipulations such as mouse cursor pointing, mouse cursor dragging, text cursor pointing, text cursor selection, and window scrolling only demand five channels. A few more hand configurations might be utilized to select particular paintbrush, stylus, or eraser tools in drawing programs, avoiding frequent excursions to the drawing tool palette to select different tools. Any remaining hand configurations not needed for graphical manipulation

channels can still be reserved for symbol or command gestures. The MTS can recognize up to eight distinct symbol or command gestures per channel by mapping opposing motions in the rotational, scaling, and two translational DOFs to different symbols or commands.

4.1.2 Locating Fingers within Remote Optical Images

Another important difference between recognition of manipulation versus symbolic gestures is that symbol recognizers only need to discern the pattern of each hand configuration as a whole, while manipulation recognizers must also be able to precisely extract hand motion in several dimensions. To do this, manipulation recognizers must locate and track specific points on the hand such as the fingertips. In the context of passive optical sensing, several researchers have adapted their systems to recognize index finger pointing gestures and track the center of a fingertip. Crowley and Coutaz [30] track a finger on a digital desk by finding the peak in cross-correlation between each video image and a reference fingertip template image.

Ahmad [3] first locates the palm by assuming the palm center is the center of mass of the video image. Then he fits a circle around the palm, finds the center of the wrist from overall hand orientation and the palm-enclosing circle, and applies a Hough transform [9] to the angles with respect to wrist center of pixel groups outside the palm circle. The peaks of the Hough transform histogram then represent the finger angles, and the pixel groups farthest from the palm along these angles are assumed to represent the fingertip locations. Clearly this approach only works for an outstretched hand which approximately faces the camera.

Nolker and Ritter [115] successfully train a local linear mapping network to locate all five fingertips in a wide variety of hand configurations, including when the fingers are curled in touching the palms so that the background of the fingertips

consists of low contrast palm flesh. Nolker and Ritter also successfully extract the pointing direction of an index finger.

4.1.3 The Feasibility of Identification from Proximity Images

The problem of locating and identifying fingertips from proximity image information is substantially different than fingertip location from remote optical images. As described in the previous chapter, accurately measuring contact centroids and tracking contacts across proximity images are relatively easy once the contacts are properly segmented. However, determining which contact comes from which fingertip is greatly complicated by the invisibility of the intermediate finger structure which connects fingertips to the center of the hand.

4.1.3.1 Rubine's Encounter with Finger Identification

The only researcher known to have encountered the finger identification problem for proximity sensing systems is Rubine [130], who was trying to recognize complex multi-path gestures on the Sensor Frame. As an active optical system which sensed obstruction of light beams, the Sensor Frame (see further description on Page 38) [107, 108, 129] suffered the same limitations in detecting intermediate finger structure as finger capacitance sensing systems. Rubine's objective was to recognize complex gestures involving two or three fingers by feeding each finger's path into his path classifier and identifying the gestures by the resulting combination of path classifications. He considered general finger identification infeasible:

> For multi-path input devices which are actually attached to the hand or body, such as the DataGlove, there is no problem determining which path corresponds to which finger. Thus, it would be possible to build one classifier for thumb paths, another for forefinger paths, etc. The characteristics of the device are such that the question of path sorting does not arise.
>
> However, the Sensor Frame (and multifinger tablets) cannot tell which of the fingers is the thumb, which is the forefinger, and so on. Thus there is no *a priori* solution to the path sorting. The solution adopted here was

> to impose an ordering relation between paths. The consistency property is required of this ordering relation: the ordering of corresponding paths in similar gestures must be the same. *Rubine* [130], Page 81.

Rubine resorted to simple sorting of paths according to the temporal and spatial coordinates of their starting points. He claimed this was sufficient for most of the multi-path gestures he was trying to recognize, though it could be confused by similar gestures whose path starting points did not always have the same ordering.

Rubine went on to develop a path clustering technique for complex multi-path gesture recognition which avoided path sorting [130]. This clustering technique computed global features from the sums and differences of all combinations of path pairs. Instead of training a different path classifier for each sorted path, a single classifier was applied to all the global features. During training of the classifier, hierarchical cluster analysis grouped similar feature vectors, irrespective of path sorting.

4.1.3.2 Summary of Constraints on Contact Identity

Upon closer examination, there actually turn out to be quite a few anatomical and biomechanical constraints on the relative features and positions of hand contacts. Many have already been discussed in Section 2.3 on proximity image topology. The challenge will be that sometimes very few constraints are available in the current proximity image or system tracking state. Since some constraints are weak and ambiguous, they can only make up for the invisibility of hand structure when several are combined.

For example, the sizes and orientations of thumb and palm contacts are usually but not always unique and distinguishable from the fingertips. Sometimes the thumb and palm heels are not touching the surface at all and therefore do not appear in the proximity image. Though thumb and palm heels are usually larger than fingertips normal to the surface, they do not become larger instantaneously.

Unless they impact the surface impulsively, the thumb and palm heels can be just as small as normal fingertips for the first few images after touchdown, until their flesh compresses and flattens out. Likewise, if the operator intentionally touches the thumb or palm heel on the surface only lightly, actively suspending their weight, the contacts will never reach a large size. Thumb and palm contact orientations, in turn, cannot be measured reliably until a contact is as big or bigger than a normal fingertip, and these orientations will not always differ from fingertip orientation.

The rest of the constraints apply to inter-contact relationships and cannot simply be measured from geometric features of individual contacts. The interaction of finger joint kinematics and a surface greatly constrains the locations of fingers relative to one another. As is apparent in most of the sample images of Section 2.3, the fingertips tend to settle into a horizontal arc whose radius varies as fingers are flexed and extended (Figures 2.7–2.10). Although it is possible to make some of the fingers flex while the others remain extended, concurrent, uniform flexion as when gripping a foam ball is more natural. Most people can only cross their fingers when the fingers are fully extended, so finger crossover should not occur during typing and chordic manipulation because these activities are normally performed with the fingers partially flexed. The only possibility of partial crossover occurs under extreme rotation of the whole hand in the surface plane. Thus the ordering of fingertip identities within their arc should coincide with the ordering of their horizontal coordinates.

However, the thumb and palm heels can be interspersed nearly anywhere in this horizontal ordering. When considered all together, the thumb, fingertips, and palm heels from one hand tend to be arranged in a circular cluster with a limited radius. As more hand parts touch the surface, more inter-contact relationships become available to evaluate. By the time the whole hand, *i.e.*, five fingers and two palm heels, touches the surface, the inter-contact constraints alone are enough to

reliably identify all of a hand's contacts.

As discussed in Section 2.3.5, the position and rotation of each hand as a whole are also fairly well constrained. Although hand crossover occurs during advanced piano performance, it is hard to imagine how it would be of use in MTS gestures. Therefore hands are not expected to cross over one another, though they may slide to the other side of the surface. A split key layout and a slight arch in the surface about the vertical axis encourage operators to keep the hands well separated. Assuming the operator's torso faces the MTS at a fixed angle, the maximum expected range of hand rotation is about 90°.

A final important observation is that when not actively engaged in graphical manipulation, both hands tend to return to neutral postures with wrists straight and fingers slightly flexed. Thus the variation in posture or deviation from neutral tends to be less at the start of a manipulation than towards the end. A system which can extend throughout a gesture the correct identifications of the initial, neutral posture may get by without needing to identify the extreme, confusing finger arrangements which occur at the end of the gesture.

In summary, the following constraints are *sometimes* available for identification:

- diagonal orientation of thumb and inner palm heel.
- moderate size of flattened thumb contact and large size of flattened palm heels relative to fingertips.
- expected angles and separations between multiple contacts depend on identity of involved hand parts.
- each identifiable hand part can only cause one surface contact.
- crossover or overlap of fingers or hands unlikely.

- range of comfortable hand rotations fairly limited.
- gestures usually start from neutral postures rather than extreme postures.

4.1.3.3 Underconstrained Cases

As this chapter will show, the identification problem is not insurmountable, at least not with the help of the constraints noted above. However, there will still be instances in which the system will not be able to tell fingertips apart, usually because only some parts of a hand are touching the surface, limiting the efficacy of inter-contact constraints. For example, unless the hand is assumed to be hovering over home row, *i.e.*, the default hand position, there is no sure way to tell which fingertip caused an isolated tap on the surface when no other hand parts are touching the surface. However, as long as isolated finger taps are only interpreted as keystrokes on a conventionally distributed key layout, there is no real need to know the finger identity. The intended key symbol should be indicated by the position of the finger tap relative to the key regions in the layout, not by finger identity. Indeed, it would be unnecessarily restrictive to activate keys only if the operator struck them with a particular finger. Only chord typing schemes make rigid associations between single finger identities and key symbols, but in such schemes the hand remains in a relatively static position over the home row keys. Given the reasonable assumption that overall hand positions are fairly fixed during chord typing, the finger identification methods presented in this chapter are accurate enough to support chord typing on the MTS.

Since the actual purpose of finger identification on the MTS is to support chordic manipulation, not chord typing, the contrasting demands each places on an identification system should be further emphasized. Since chordic manipulations will involve slides over the entire surface, chordic manipulations cannot be assumed to start from any one hand position, though they will often start from the neutral

or default hand position. While contact size, orientation, and relative velocity features will be sufficient to distinguish the thumb from fingertips even in the face of such initial hand displacements, these features tend not to differ between fingertips. Therefore, unless all fingertips on a hand are touching the surface, the MTS will have no sure way to tell exactly which fingertips are touching if the fingertips deviate more than a centimeter from the horizontal locations predicted by the hand position estimate. Such deviations occur often enough that the MTS will not be able to reliably distinguish those chordic manipulations consisting of the same number of fingertips but different combinations of them, such as the index and middle fingertips versus ring and pinky fingertips.

4.1.4 Pooling of Fingertip Combinations

Careful design of the chord gesture set in Chapter 5 will pool potentially ambiguous finger combinations so that minor identification failures do not affect the MTS operator. As already discussed, far fewer chords are necessary to cover all conceivable graphical manipulation channels than to cover an entire symbolic alphabet. While the identification system will do its best to identify fingers correctly in all cases, the chordic manipulation recognizer will only depend upon correct detection of thumb presence and proper ordering and counting of the other fingertips. This will support selection of seven different chordic manipulation channels (see Table 5.1 on Page 246) on each hand by multiple-finger chords, as opposed to 26 which would be available on each hand if all combinations of two or more fingers were distinguished.

This pooling of performable fingertip combinations should not be considered an unhappy compromise; consideration of human factors also argues for it. Memorization and control of particular combinations of fingertips rather than particular numbers of fingertips may be just as cognitively and biomechanically demanding for the operator as identification is for the MTS. Given the freedom to choose and vary

which fingertips are used in chords composed of only one, two or three fingertips, operators naturally prefer to pick combinations in which all touching fingertips are adjacent rather than combinations in which a finger such as the ring finger is lifted but the surrounding fingers such as the middle and pinky must touch. In a chord typing study, Fukumoto and Tonomura [41] found that users can tap these finger chords in which all touching fingertips are adjacent twice as fast as other chords. Trained pianists could perform all chords about twice as fast as normal subjects, but awkward chords still took twice as long as chords composed of a contiguous group of fingertips.

Allowing the MTS operator to interchange fingertip combinations when accessing common manipulation channels such as pointing also helps avoid overuse of the finger muscles which press or suspend a particular set of fingers. The uniquely opposable motion of the thumb and relatively large region of sensory-motor cortex devoted to the thumb also suggest that the thumb may be cognitively distinct from the fingertips. The gesture set will abide by this distinction: chords including the thumb will be reserved for selection of command gesture channels, and chords initially composed solely of fingertips will select graphical manipulation channels, though the thumb can be added to these chords after the initial channel selection to capture its unique opposable motions.

4.2 Overview of the Hand Tracking and Identification System

For the reader's convenience, the tracking system flow diagram of Figure 3.1 is repeated in Figure 4.1. The image segmentation module described in Chapter 3 segments the current proximity image into groups of electrodes corresponding to the distinguishable hand parts. The path tracking module, also described in the previous chapter, links the groups from successive proximity images corresponding to the same hand part into persistent contact paths. It also computes parameters

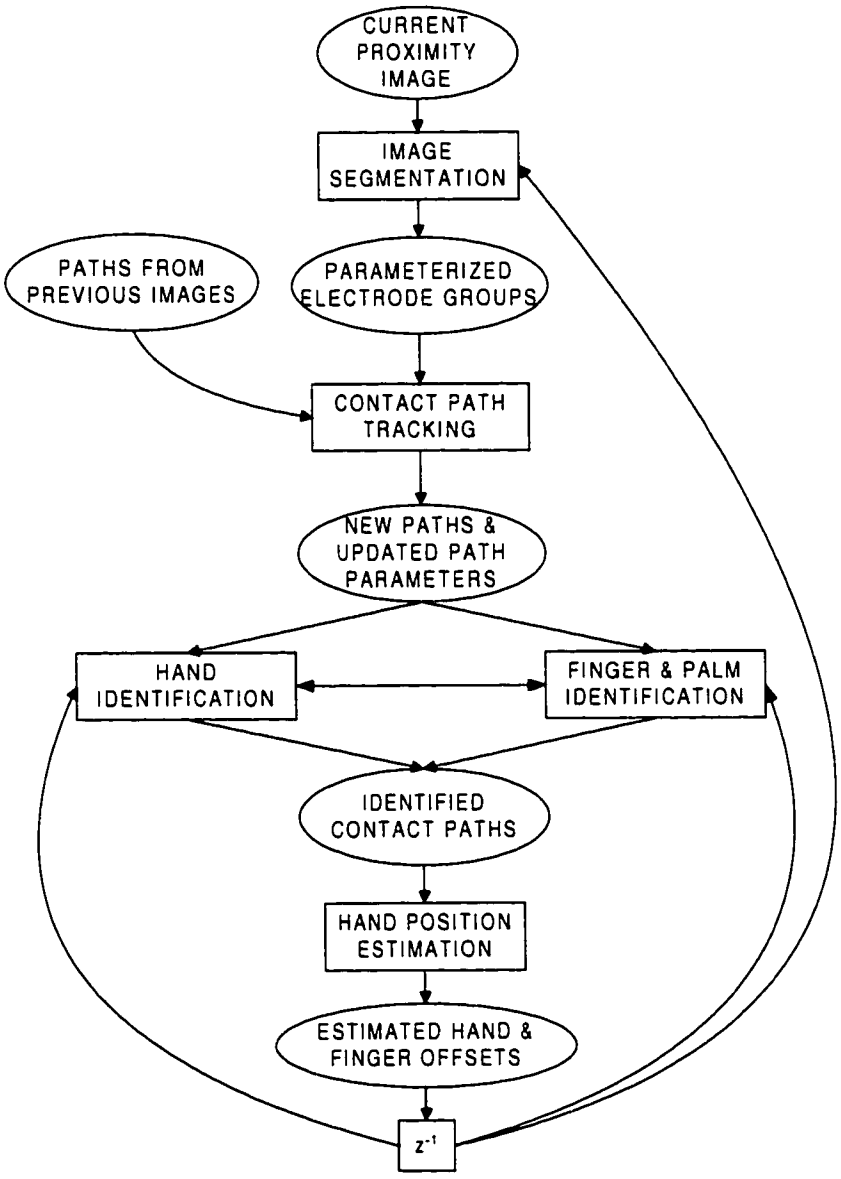


Figure 4.1: System-level diagram for hand and finger tracking and identification modules.

Table 4.1: Finger identity notation for identified path data structures.

Notation	*Common Hand Part Name*
$F0$	The Null/Dummy Hand Part
$F1$	Thumb Finger (not a fingertip)
$F2$	Index Finger
$F3$	Middle Finger
$F4$	Ring Finger
$F5$	Pinky Finger
$F6$	Outer (lateral) Palm Heel
$F7$	Inner (medial) Palm Heel
$F8 \ldots F11$	Forepalm Calluses/Overflow Contacts

related to the contact trajectories such as the instantaneous lateral velocity along each path.

The finger identification, hand identification, and hand position estimation modules are the focus of this chapter. The hand identification module will determine which hand causes each contact, and the finger identification module will establish a unique finger or palm heel identity for each contact within a hand. The finger and hand identification modules are hierarchically related, and both will employ combinatorial optimization methods to find the most biomechanically and anatomically consistent set of contact identifications. The output of the identification modules will be non-zero hand and finger indices attached to all contact paths. The identified contact paths will be referred to with the notation of Table 4.1. To denote a particular hand identity this notation can be prefixed with an L for left hand or R for right hand; for example, RF2 denotes the right index finger path. When referring to a particular hand as a whole, LH denotes the left hand and RH denotes the right hand.

The hand position estimator will use contact positions as well as the assigned contact identities to maintain a conservative estimate of overall hand position even

when a hand is not touching the surface. Feedback of the estimated hand positions to the finger identification process will help identify contacts when so few contacts appear in the image that the hand's contact cluster has no apparent structure. To aid hand identification, the estimated hand position will temporarily retain the last measured hand and finger positions after a hand completely lifts off the surface. Then, if the fingers quickly touch back down in the same region, they will more likely regain their previous identifications. Because this estimated hand position feedback is essential to the accurate functioning of the hand and finger identification algorithms, the hand position estimation algorithm will be described first.

Again, this overall system architecture grossly resembles that of most computer vision systems, with the hand and finger identification modules corresponding to the object recognition stage. In this case, the object recognition stage is more constrained than in most computer vision applications because upper limits are known on the number and type of objects which can appear in the image, *i.e.*, thumb, fingertip, or palm contacts from a left or right hand. Rather than picking the best match from a library of object templates, the object recognition problem reduces to an assignment problem of finding the optimal one-to-one mapping between surface contacts and finger identities. When fewer contacts are present on the surface than possible identities, dummy contacts will be created to keep the mapping one-to-one. Fingers whose identities end up mapped to a dummy contact are assumed to be lifted off the surface.

Unlike the path tracker, which is expected to create and maintain perfect continuity of each finger path from the first image frame in which the finger appears, the identifications are not required to be correct in the first frame. As a hand touches down, its individual parts may gradually appear over several image frames, gradually increasing the constraints available for identification. Given that

the segmentation, identification, and hand position estimation modules may be underconstrained without the feedback between them, they can only be expected to converge on a coherent solution over a few iterations of feedback between one another. Reshuffling of identifications early in a hand gesture is allowed so the system does not commit to an assignment before hand configuration clues have accumulated.

However, reshuffling back and forth during ambiguous cases is still undesirable, so the system must also have some assignment hysteresis. For images which contain no new information pertinent to identification, the identities of existing contacts are simply extended via path continuation. The identification algorithm need not even execute for such images, avoiding the risk that identities will get reshuffled. As soon as a proximity image appears with new constraints such as an additional contact or clearer features, the identification algorithm executes again. As will become apparent, the hand position estimates provide a weaker, analog form of hysteresis while fingers are temporarily lifted off the surface.

4.3 Hand Position Estimation

As indicated in Figure 4.1, the hand position estimator provides important biasing feedback to the identification and segmentation processes. The hand position estimates are intended to be a conservative guess of lateral hand position under all conditions, including when a hand is floating above the surface with no flesh visible in the proximity images. In case a hand is not touching the surface, its position estimate represents a best guess of where it will touch down again. When a hand partially touches the surface, the estimate combines current hand position measurements based upon the centroids and identities of its contacts with previous hand position estimates based upon earlier identifications which may have been more or less reliable than the current measurements.

4.3.1 Measuring Current Hand Position

Figure 4.2 shows the individual steps of the hand position estimation process, which must be repeated for each hand separately. First, an overall hand position must be measured from the contact positions in the current proximity image. The simplest method of obtaining a hand position measurement would be to average the positions of all the hand's contacts regardless of identity. If all hand parts were always touching the surface, as in the flattened hand of Figure 2.7, the resulting centroid would be a decent estimate, lying somewhere under the center of the palm since the fingers and palm heels typically form a ring around the center of the palm. However, consider when only one hand contact is available for the average. The measurement would wrongly assume the hand center is at the position of this lone contact. A lone hand contact is very unlikely to be from the hand center because the hand center usually does not even touch the surface until the rest of the hand is flattened onto the surface. If the lone contact is actually from the right thumb, the true hand center would be 4-8 cm to the right, or if the contact is actually from a palm heel, the true hand center would be 4-6 cm higher, or if the lone contact is from the middle finger, the true hand center would be 4-6 cm lower.

Instead of assuming each contact comes from the center of the hand when computing the average, the MTS's hand position measurement utilizes the within-hand identifications to compute for each contact an offset between its measured position, $(Fi_x[n], Fi_y[n])$, and the default position (Fi_{defx}, Fi_{defy}) of the particular finger or palm heel with its identity i. These default positions correspond to finger and palm positions when the hand is in a neutral posture with fingers partially closed (see Figure 2.8), as when resting on home row of the key layout. *With this identity-dependent offset computation, the position of a single, properly identified surface contact will be sufficient to sustain a fairly accurate hand position estimate.*

The next step averages the individual contact offsets to obtain the measured

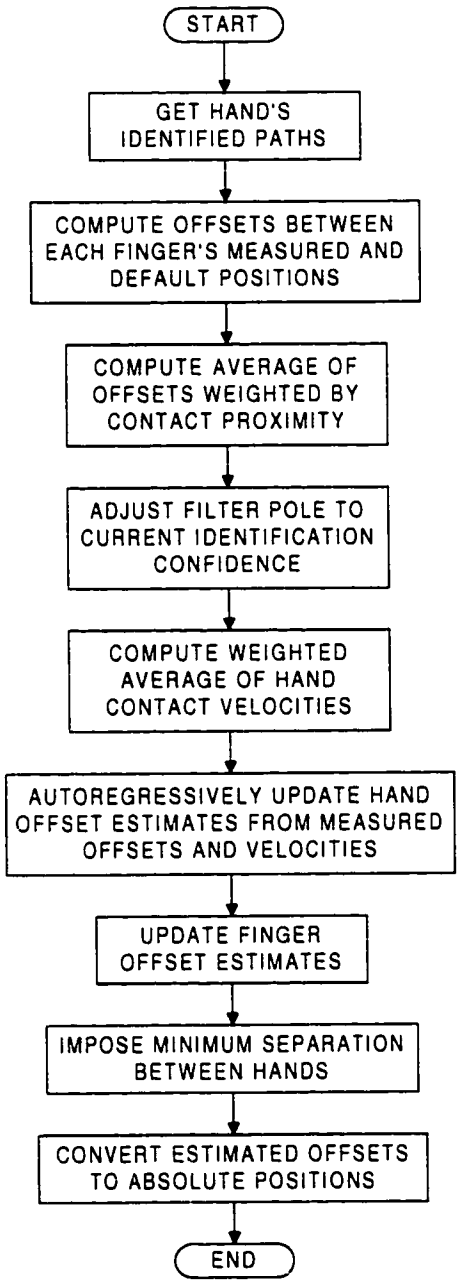


Figure 4.2: Flow chart of hand position estimation process.

hand offset ($H_{mox}[n], H_{moy}[n]$):

$$H_{mox}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n](Fi_x[n] - Fi_{defx})}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (4.1)$$

$$H_{moy}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n](Fi_y[n] - Fi_{defy})}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (4.2)$$

Preferably the weighting $Fi_{mow}[n]$ of each finger and palm heel is approximately its measured total proximity, *i.e.*, $Fi_{mow}[n] \approx Fi_z[n]$. This ensures that lifted fingers, whose proximity is zero, have no influence on the average, and that contacts with lower than normal proximity, whose measured positions and identities are less accurate, have low influence. Furthermore, if palm heels are touching, their large total proximities will dominate the average. This is beneficial because the palm heels, being immobile relative to the hand center compared to the highly flexible fingers, supply a more reliable indication of overall hand position.

When a hand is not touching the surface, *i.e.*, when all proximities are zero, the measured offsets are set to zero. This will cause the filtered hand position estimate below to decay toward the default hand position.

4.3.2 Identification Confidence and Filter Delay

As long as the contact identifications are correct, this identification-dependent method for hand position measurement eliminates the large errors caused by assuming lone contacts originate from the center of the hand. Flexing of fingers from their default positions will not perturb the measurement more than a couple centimeters. However, this method is susceptible to contact misidentification. It assumes identifications are always correct, which is not always the case, especially if the identification system only has a couple of contacts to optimize over. For example, if only one hand part is touching the surface and it is assigned the wrong finger or palm identity, the hand position measurement can be off by as much as 8 cm. If the lone contact is attributed to the wrong hand, it can easily cause $H_{mox}[n]$ to be

in error by 20cm. Therefore the current measured offsets are not used directly, but are averaged with previous offset estimates ($H_{eox}[n-1], H_{eoy}[n-1]$) using a simple first-order autoregressive filter, forming current offset estimates ($H_{eox}[n], H_{eoy}[n]$).

The filter pole $H_{oa}[n]$ should be adjusted according to confidence in the current contact identifications. Since finger identifications accumulate reliability as more parts of the hand contact the surface, one simple measure of identification confidence is the number of fingers which have touched down from the hand since the hand last left the surface. Contacts with large total proximities also improve identification reliability because they have strong disambiguating features such as size and orientation. Therefore $H_{oa}[n]$ is set roughly proportional to the sum of contact proximities for the hand:

$$H_{oa}[n] = \min(1, \beta + \frac{1}{7} \times \sum_{i=1}^{i=7} Fi_z[n]) \tag{4.3}$$

$H_{oa}[n]$ must of course be normalized to be between zero and one or the filter will be unstable. Thus when confidence in contact identifications is high, *i.e.*, when many parts of the hand firmly touch the surface, the autoregressive filter favors the current offset measurements. However, when only one or two contacts have reappeared since hand liftoff, the filter emphasizes previous offset estimates in the hope that they were based upon more reliable identifications.

To encourage correct segmentation and identification upon touchdown for a hand which has temporarily lifted off the surface, the filtered offsets must hold a conservative estimate of hand position while the hand is floating above the surface. If a hand lifts off the surface in the middle of a complex sequence of operations and must quickly touch down again, it will probably touch down close to where it lifted off. However, if the operation sequence has ended, the hand is likely to eventually return to the neutral posture, or default position, to rest. Therefore, the β term in Equation 4.3 is made small enough that while a hand is not touching the surface, the estimated offsets gradually decay to zero at about the same rate as a hand lazily

returns to default position. Alternatively, the estimated hand offsets can be made to decay toward zero at a constant speed rather than exponentially.

4.3.3 The Filter Equations

When $H_{o\alpha}[n]$ is small due to low identification confidence, the filter tracking delay becomes large enough to lag behind a pair of quickly moving fingers by several centimeters. The purpose of the filter is to react slowly to questionable changes in contact identity, not to smooth contact motion. Measurement of the current contact velocities $(Fi_{vx}[n], Fi_{vy}[n])$ occurs in the path tracking process (3.3.4) independent of finger identity. Therefore this motion tracking delay can be safely eliminated by adding the contact motion measured between images to the old offset estimate. Again the hand motion $(H_{mvx}[n], H_{mvy}[n])$ is averaged over the individual contact velocities:

$$H_{mvx}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n] Fi_{vx}[n]}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (4.4)$$

$$H_{mvy}[n] = \frac{\sum_{i=1}^{i=7} Fi_{mow}[n] Fi_{vy}[n]}{\sum_{i=1}^{i=7} Fi_{mow}[n]} \quad (4.5)$$

The estimated hand offsets $(H_{eox}[n], H_{eoy}[n])$ can now be computed using the complete filter equations:

$$H_{eox}[n] = H_{o\alpha}[n] H_{mox}[n] + (1 - H_{o\alpha}[n])(H_{eox}[n-1] + H_{mvx}[n]\Delta t) \quad (4.6)$$

$$H_{eoy}[n] = H_{o\alpha}[n] H_{moy}[n] + (1 - H_{o\alpha}[n])(H_{eoy}[n-1] + H_{mvy}[n]\Delta t) \quad (4.7)$$

The overall filter structure captured by these equations is also illustrated in Figure 4.3.

4.3.4 Enforcing Hand Separation

While the offset computations for each hand have been independent as described so far, it is advantageous to impose a minimum horizontal separation between

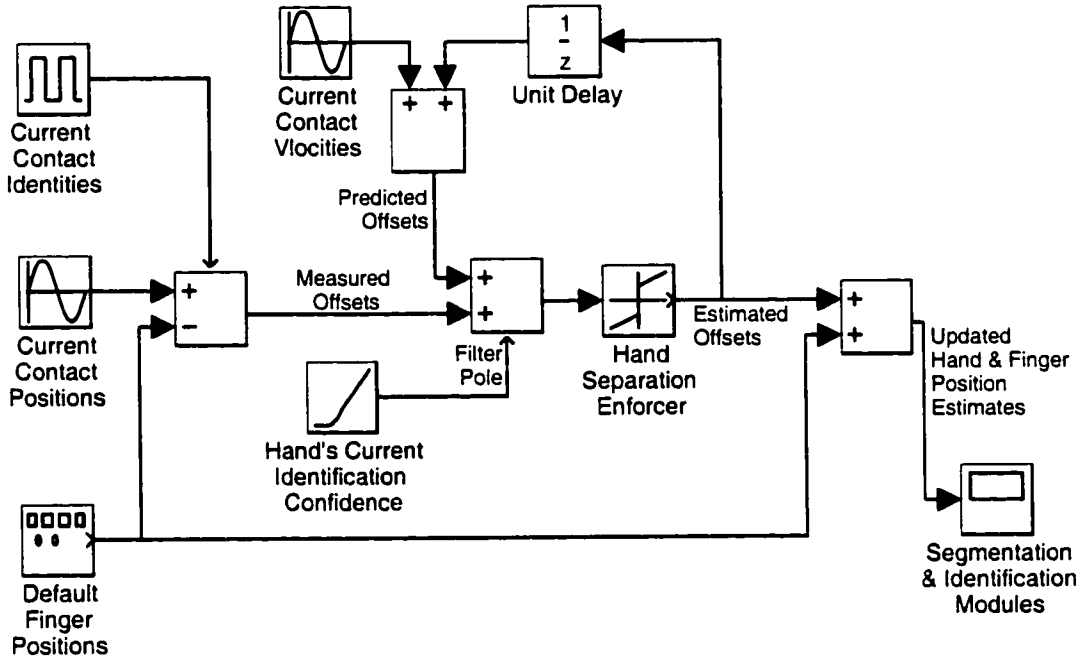


Figure 4.3: Filter diagram for hand position estimator. Note that the default finger positions are subtracted off in the first stage and added back in at the last stage so that the signals at intermediate stages are in the form of relative offsets from default.

the estimated left hand position and estimated right hand position such that when a hand such as the right hand slides to the opposite side of the board while the other hand is lifted, the estimated position of the other hand is displaced. In this case the estimated position of the lifted left hand would be forced from default to the far left of the surface, possibly off the surface completely. If the right hand is lifted and the left is not, an equation like the following can be applied to force the estimated right hand position out of the way:

$$RH_{eox}[n] := \min(RH_{eox}[n], (LF1_{defx} - RF1_{defx}) + LH_{eox}[n] + min\_hand\_sep) \quad (4.8)$$

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

4.3.5 Interactions with Segmentation and Identification Modules

The updated hand position estimates $(H_{eox}[n], H_{eoy}[n])$ are fed back to the segmentation and identification processes during analysis of the next proximity image. If the other processes need the estimate in absolute coordinates, they can simply add the supplied offsets to the default finger positions, but in many cases the relative offset representation is actually more convenient.

The updated hand position estimates tend to move so as to reinforce the current contact identifications. Assuming the current identifications are correct, this tends to stabilize them, to move the attractor ring so that the contacts line up better with their assigned attractors. But it can also reinforce incorrect identifications, which is why it is so important to limit the rate of change in estimated hand position with the filter pole $H_{o\alpha}[n]$ when the confidence in identifications is low. On the other hand, when confidence in identifications is high but the position estimate contradicts them, the position estimate should be allowed to change quickly to become consistent with the identifications.

4.4 Finger Identification

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

4.4.1 The Basic Attractor Ring

For the contacts assigned to each hand, the finger identification process (Figure 4.4) attempts to match contacts to a template of hand part attractor points, each attractor point having an identity which corresponds to a particular finger or palm heel. This matching between contact paths and attractors should be one-to-one, but in the case that some hand parts are not touching the surface, some attractors will be left unfilled, *i.e.*, assigned to dummy paths.

The relative locations of the attractor points are set to the approximate positions of their corresponding fingers and palms when the hand is in the default posture with fingers partially curled (Figure 2.8). These should be the same default finger and palm locations (Fi_{defx}, Fi_{defy}) employed in hand position estimation. The default fingertip positions should also match the centers of the home row keys in the key layout. Setting the distances and angles between attractor points from the half-closed hand posture causes the attractors to lie in a ring with a radius

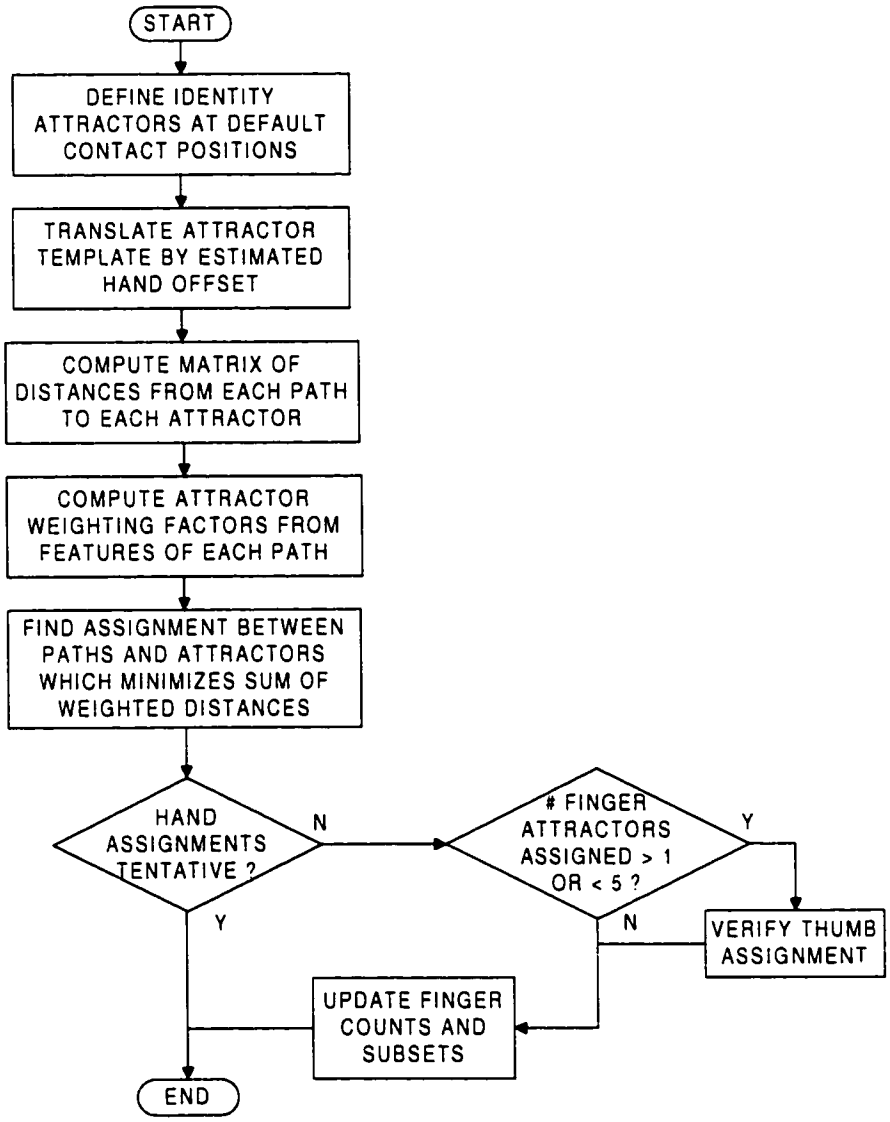


Figure 4.4: Flow chart of the finger and palm (within-hand) identification algorithm.

about halfway between that of an outstretched, flattened hand and a fist. This will allow the matching algorithm to perform well for a wide variety of finger flexions and extensions.

For optimal accuracy of contact-attractor matching, the ring should be kept roughly centered on the hand cluster. Therefore, the attractor ring for a given hand is translated as a whole by the hand's estimated position offset. The final attractor positions ($Aj_x[n], Aj_y[n]$) are then:

$$Aj_x[n] = H_{eox}[n] + Fj_{defx} \tag{4.9}$$

$$Aj_y[n] = H_{eoy}[n] + Fj_{defy} \tag{4.10}$$

4.4.2 Voronoi Diagram for Single Contact Identification

Figure 4.5 displays both the ring-like structure formed by the attractor points for the right hand and the Voronoi polygon or cell around each attractor. If the given hand is a left hand, the attractor ring must be mirrored about the vertical axis from that shown. Every geometric point within an attractor's Voronoi cell is closer to that attractor than any other attractor in the ring [116]. When there is only one contact in the hand cluster and its features are not distinguishing, *i.e.*, when only a single small part of a hand is touching the surface, the identification algorithm can simply determine which Voronoi cell the contact lies within and assign the contact to that cell's attractor. Thus the size and shape of each Voronoi cell indicates the range over which a particular hand part can touch down with respect to estimated hand center and still be identified correctly. Given that the Voronoi cells for fingertips are rather tall and narrow, one can conclude that the Voronoi cells will tolerate a high degree of finger flexion and extension but not so much hand rotation or unexpected horizontal displacement.

In the unweighted Voronoi diagram of Figure 4.5, the palm attractors are actually a couple centimeters lower than the measured default palm heel positions.

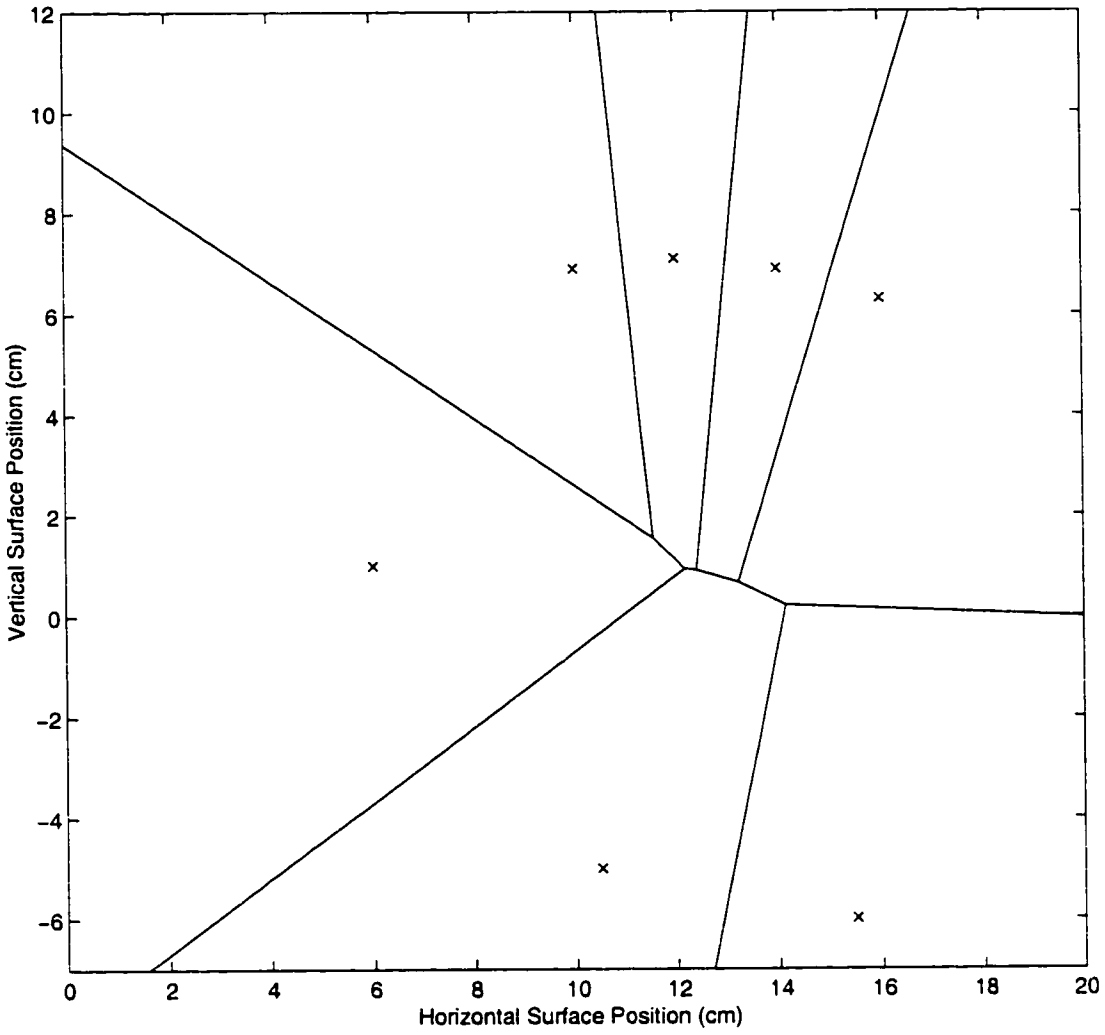


Figure 4.5: Voronoi cell diagram constructed around ring of hand part attractor points (x's labeled with finger identity indices). In this figure, the estimated right hand position offsets are zero, so the ring is not translated from the default finger positions.

Moving these attractors forward to their proper vertical positions of -4 cm would have enlarged the palm heel Voronoi cells too much at the cost of thumb and pinky Voronoi cell size. The palm heel Voronoi cells are so large that an abducted thumb or flexed pinky is occasionally misidentified as a palm heel. This can actually cause typing errors when hitting pinky keys in the lowest row such as $< Ctrl >$ or $< Alt >$ (see Figure 1.1 on Page 6) because the tapping finger can be misidentified as a palm and therefore ignored. Section 4.4.6.2 will introduce palm heel weightings which shrink the palm heel Voronoi cells, allowing the true default palm heel positions to be used.

4.4.3 Multiple Contacts Compete for Voronoi Cells

When multiple parts of a hard touch the surface, more than one may lie within the same Voronoi cell. *Since only one of the contacts can be assigned to any Voronoi cell's attractor at one time, the contacts lying in the same cell must compete for the cell's attractor and neighboring attractors.* A global optimization is necessary to determine which of the contacts goes to the attractor of the occupied Voronoi cell and which goes to neighboring attractors.

This global optimization finds the one-to-one assignment between attractors and contacts which minimizes the sum of weighted, squared distances between each attractor and its assigned contact. When there are fewer surface contacts than attractors, *i.e.*, when any hand parts are floating above the surface, the null path $P0$, which has zero distance to each attractor, acts as a dummy contact in place of any missing contacts. This ensures a one-to-one mapping can be found by making the total number of surface plus dummy contacts the same as the number of attractors. Let the squared distances in the surface plane between each contact path Pi and each translated attractor point Aj form a square matrix $[d_{ij}^2]$:

$$d_{ij}^2 = (Aj_x[n] - Pi_x[n])^2 + (Aj_y[n] - Pi_y[n])^2 \tag{4.11}$$

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

$$\sum_{i=1}^{7} d_{i\pi_i} \tag{4.12}$$

where contact i and attractor j are considered assigned to one another when $\pi_i \equiv j$. This combinatorial optimization problem, known more specifically in mathematics as an assignment problem, can be efficiently solved by a variety of well-known mathematical techniques. The following sections will review the solution techniques for the assignment problem, the reason the squared Euclidean distance is preferred over other distance metrics for finger identification, and how feature-dependent weightings of particular contact-attractor distances sustains correct identification over wider ranges of finger motion.

4.4.4 The Assignment Problem

The assignment problem is a special case from the classes of linear programming problems and integer programming problems. Its more general formulation as a linear programming problem [35] is to minimize over the parameters x_{ij} the sum:

$$\sum_{i=1}^{M} \sum_{j=1}^{M} c_{ij} x_{ij} \tag{4.13}$$

subject to the constraints:

$$\sum_{j=1}^{M} x_{ij} = 1 \qquad \forall i = 1, \ldots M \tag{4.14}$$

$$\sum_{i=1}^{M} x_{ij} = 1 \qquad \forall j = 1, \ldots M \tag{4.15}$$

$$x_{ij} = 0 \quad or \quad 1 \qquad \forall i, j = 1, \ldots M \tag{4.16}$$

where c_{ij} is an arbitrary cost of assigning contact i to attractor j, and contact i is considered assigned to attractor j only when $x_{ij} = 1$. Since there are $M!$ possible solution matrices x_{ij}, brute force enumeration is inappropriate unless M is very

small. Since this is a linear programming problem, it also has a dual formulation as a maximization problem, and specialized versions of the primal and dual simplex method can find the optimal solution [8, 10, 109]. However, the specially constrained structure of the solution matrix x_{ij} supports other efficient solution techniques.

Kuhn [83] was the first to develop the Hungarian Method for solving the assignment problem. This method was based on special matrix properties discovered by Egerváry [34] and König [80] which guide manipulations of matrix rows and columns similar to Gaussian elimination. Its worst case performance is $O(M^3)$, though performance varies widely for large M depending on implementation tricks [27, 102]. The Hungarian Method and improvements upon it have been the preferred solution method in the field of operations research, where the assignment problem often arises when trying to match workers' various skills to a variety of tasks or machines requiring different skills. In most of these applications the cost matrix is not derived from the distances between points in a plane. However, some operations research problems do construct a cost matrix from a set of matching distances. An example would be trying to assign M taxis to M passengers given the location where each passenger is to be picked up, the starting location of each taxi, and the assumption that the overhead for each taxi increases in proportion to the distance from last drop off (starting location) to next pick up. In these real-world operations research problems the unsquared rather than squared distance is usually the more relevant cost parameter.

As an integer programming problem, the assignment problem can be solved with the branch and bound heuristic [58], relaxation [11, 12] or network flow minimization techniques based upon the Shortest Augmenting Path Method [7, 31, 33, 75]. As a combinatorial optimization problem, localized [1] combinatorial search heuristics can be applied, though these do not guarantee the global minimum will be found. For the relatively small size of the finger assignment problem, any of these

solution techniques should be efficient enough for real-time finger identification.

Localized combinatorial search was the first technique implemented in the MTS software, and once methods to avoid convergence failures for it were understood (see Appendix C), there seemed to be no need to try an alternate implementation with any of the other well-known techniques. Moreover, localized combinatorial search has a couple advantages within the context of real-time finger identification. First, it can verify very quickly that a previous set of identifications is still at least locally optimum. Though its worst case performance will turn out to be $O(M^3)$ as well, it finds the global minimum fairly fast given an initialization near the global minimum. Second, it offers important insights into the design and analysis of the attractor ring when assignments costs are proportional to contact-attractor distances squared.

4.4.4.1 Localized Combinatorial Search

Combinatorial problems such as the assignment problem can be incrementally optimized using k-exchange neighborhoods [1]. In the case of the finger identification problem, a k-exchange neighborhood is a subset of attractors from the attractor ring. If k is 2, the subset will usually be a pair of adjacent attractors. Minimizing the sum of assignment distances within this subset by conditionally swapping the two contact-attractor assignments will always improve the total assignment sum. This is a consequence of the fact that the total cost, *i.e.*, the assignment sum, is a monotonic increasing function of the individual costs, *i.e.*, contact-attractor distances. Picking a sequence of k-exchange neighborhoods, *i.e.*, successive adjacent attractor pairs, and optimizing them with conditional swapping causes the total assignment sum to decrease toward a local minimum. Whether the local minimum found is actually the global minimum depends on the initial assignments and the ordering of the exchange neighborhood sequence.

4.4.4.2 Choosing Initial Assignments

After each sensor array scan, the paths of hand parts which have newly touched down are assigned to the closest available attractors in the ring. This operation is equivalent to picking a new path, constructing a Voronoi diagram from only the unfilled attractors, and assigning the path to the unfilled attractor whose Voronoi cell the contact lies within. This does not ensure that the initialization puts the new contact with the best attractor since the best attractor might be one that is already filled with a pre-existing path, but this usually puts the new path close to its correct attractor and therefore close to the global minimum.

4.4.4.3 The Swapping Condition

Let Aa and Ab be adjacent attractors in the ring, *i.e.*, $a = b \pm 1$, and let Pg and Ph, correspondingly, be their currently assigned contact paths. The one-to-one nature of the matching assignments is enforced with the double-links $Aa_{assignedpath} \Longleftrightarrow Pg_{assignedfinger}$ and $Ab_{assignedpath} \Longleftrightarrow Ph_{assignedfinger}$. These assignments are swapped, making $Aa_{assignedpath} \Longleftrightarrow Ph_{assignedfinger}$ and $Ab_{associated} \Longleftrightarrow Pg_{assignedfinger}$, if swapping reduces the sum of the contact-attractor distances:

$$d_{ha}^2 + d_{gb}^2 \overset{?}{<} d_{hb}^2 + d_{ga}^2 \tag{4.17}$$

In the method of simulated annealing, a noise term is added to the right side of inequality 4.17 so that some swaps are taken even when they do not decrease the assignment sum. In large problems this allows the optimization search to jump back out of local minima. The noise variance or annealing temperature is decreased over time to lock in the global minimum. As Appendix C shows, there are simpler ways than simulated annealing to avoid non-global minima in finger identification.

4.4.4.4 The k-exchange Sequence

The exchange neighborhood sequences utilized by localized combinatorial search heuristics are often application specific [1]. For finger identification, the roughly circular structure of the attractor ring supports a simple nearest neighbor traversal around the ring, as would a linear structure. The sequence simply proceeds around the attractor ring in either clockwise or counter-clockwise order, pausing for a conditional swap at each attractor encountered. Each swap test is applied between the current attractor and the next adjacent attractor around the ring. Travel around the ring repeats until one complete traversal is made without accepting any swaps.

This basically amounts to a bubble sort on the ring, utilizing the swapping condition for the current attractor pair as the bubble sort ordering relation. Notice that though a bubble sort has $O(M^2)$ performance [79], *i.e.*, M traversals of the ring are required in the worst case, a single traversal of the ring can verify that the sorting of contacts based on their positions in previous proximity images is still the correct sorting for their current positions. Also, the initialization of new contacts assignments as described in Section 4.4.4.2 tends to put contacts within a couple attractors of their proper attractor, so usually not more than two or three traversals of the ring are necessary. Worst case performance can be improved by utilizing a bidirectional bubble sort [79], *i.e.*, reversing the direction of the exchange sequence after each complete traversal of the ring.

Because transitivity of the ordering relations described in the next section will not always extend between opposite sides of the ring, contacts can get stuck in attractors on the opposite side of the ring. Such convergence failures are explained further in Appendix C. They can be avoided and the global minimum reached by expanding the exchange neighborhoods to include contact pair swaps between attractors on opposite sides of the ring instead of only between adjacent attractors on the ring. This expansion of the exchange neighborhood reduces worst case

performance to $O(M^3)$.

4.4.5 Geometric Interpretations of the Swapping Condition

This section explores geometric interpretations of the pair swapping condition of Equation 4.17 which aid in design of the attractor ring and understanding of its assignments.

4.4.5.1 Geometric Interpretation of Single Contact Swapping

The simplest case is when one of the contacts, say Ph, is a dummy or null contact. By definition of dummy contact, $d_{ha} = 0$ and $d_{hb} = 0$. Then the swapping inequality reduces to:

$$d_{gb}^2 \stackrel{?}{<} d_{ga}^2 \tag{4.18}$$

which is equivalent to:

$$d_{gb} \stackrel{?}{<} d_{ga} \tag{4.19}$$

since all distances are non-negative. As shown in Figure 4.6, the geometric interpretation of this swapping condition is that after the conditional swap, real contact Pg will be assigned to the closest attractor, *i.e.*, the attractor which is on the same side of the perpendicular bisector between the attractors as Pg is. The walls of Voronoi cells are actually composed of such perpendicular bisectors between various attractors, Figure 4.6 being the special case of a Voronoi diagram of only two attractors. The fact that Equation 4.18 is equivalent to Equation 4.19 simply means that squaring the distances has no impact on the swapping condition a dummy contact is involved; Voronoi diagrams constructed for a Euclidean distance *squared* metric are the same as Euclidean distance metric diagrams [116].

Another interpretation will be useful for comparison to the case when the attractors compete for two contacts. By projecting the contact g onto the line $\overline{ab}$ between the attractors, one can see that the swapping condition depends on the

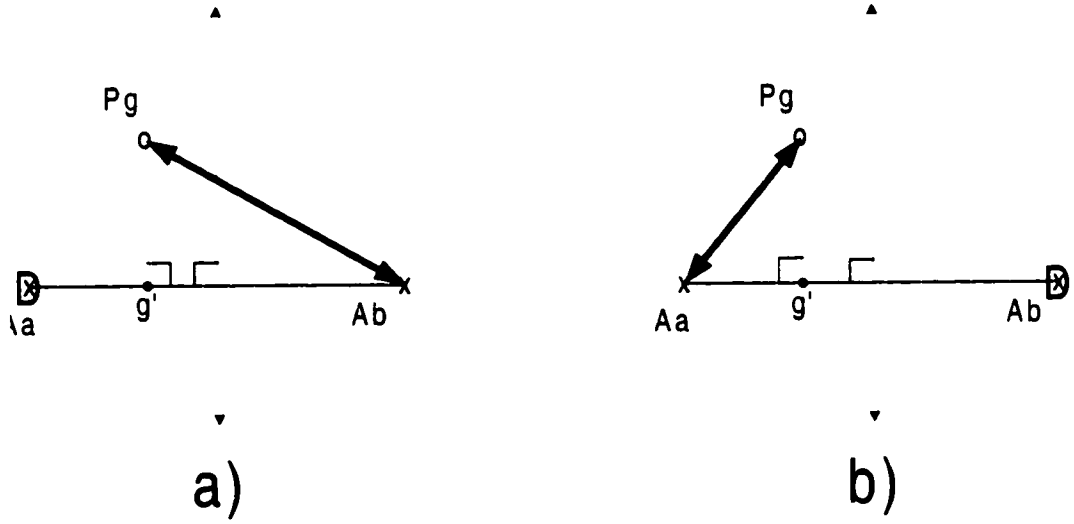


Figure 4.6: Geometric construction showing possible assignments (heavy arrows) when two attractors (crosses) compete for one real surface contact (circle). The losing attractor is always assigned a dummy contact (D) which has zero distance to every attractor. The swapping condition prefers case b) in which the assignment link does not cross the perpendicular bisector (dotted arrows) of the attractors.

horizontal position of the projection g' relative to the center point between the attractors:

$$g'_x \stackrel{?}{<} \frac{(a_x + b_x)}{2} \tag{4.20}$$

The comparison of the contact projection with absolute attractor pair position in Equation 4.20 should be contrasted to the ordering relation which will arise for contact pairs in Equation 4.35. Note Equation 4.35 does not depend upon absolute horizontal alignment of the attractor pair.

4.4.5.2 Geometric Interpretation of Contact Pair Swapping

Though Figure 4.6 and its generalization to Voronoi diagrams explain swapping behavior when two or more attractors are only competing for one real surface contact, the case when two attractors are competing for two contacts, as when the attractor ring is full of contacts, is not as straightforward unless a Euclidean distance squared metric is used. With the distance-squared metric, the swapping decision depends only on the ordering of the contacts as projected onto the line connecting the attractor pair.

This interesting property of the distance-squared metric can be proved with the help of the geometric constructions in Figure 4.7. Without loss of generality, assume attractors a and b lie along a horizontally oriented line. Now, form perpendiculars from each contact g and h to the line $\overline{ab}$ and mark the projected intersections g' and h', respectively. The swapping Inequality 4.17 can be restated in terms of the lengths of the vectors connecting pairs of these points:

$$||\vec{ah}||^2 + ||\vec{bg}||^2 \stackrel{?}{<} ||\vec{ag}||^2 + ||\vec{bh}||^2 \tag{4.21}$$

where the left side of the inequality represents the sum of the squared assignment lengths (heavy arrows) of Figure 4.7a, and the right side of the inequality represents the sum of the squared lengths for the opposite assignments in Figure 4.7b. By the

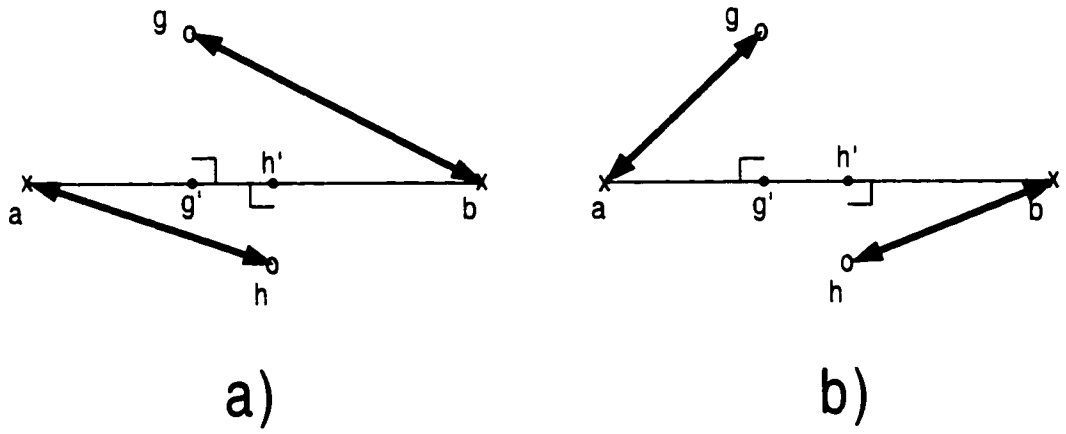


Figure 4.7: Geometric construction for comparing the costs of the two possible assignments (heavy arrows) between a pair of contacts (circles) and a pair of attractors (crosses). If squared-distance is the assignment cost metric, the swapping decision reduces to comparing the ordering of the contacts' projections (dotted perpendiculars) onto the (solid) line between the attractors. For the relative contact positions shown, the assignments of b) have lower cost than those of a) under the distance-squared metric.

Pythagorean theorem, these potentially diagonal vector lengths can be restated in terms of purely horizontal or vertical vector components:

$$||\vec{ag}||^2 = g'a_x^2 + g'g_y^2 \tag{4.22}$$

$$||\vec{bh}||^2 = h'b_x^2 + h'h_y^2 \tag{4.23}$$

$$||\vec{ah}||^2 = h'a_x^2 + h'h_y^2 \tag{4.24}$$

$$||\vec{bg}||^2 = g'b_x^2 + g'g_y^2 \tag{4.25}$$

and substituted into the inequality to obtain:

$$h'a_x^2 + h'h_y^2 + g'b_x^2 + g'g_y^2 \stackrel{?}{<} g'a_x^2 + g'g_y^2 + h'b_x^2 + h'h_y^2 \tag{4.26}$$

The vertical distances on each side cancel, leaving an expression that depends only on the horizontal separations of the contacts relative to the attractors:

$$h'a_x^2 + g'b_x^2 \stackrel{?}{<} g'a_x^2 + h'b_x^2 \tag{4.27}$$

Since $g'a_x^2$ and $h'a_x^2$ can be rewritten in terms of $h'b_x$, $g'b_x$, and ab_x:

$$g'a_x^2 = (ab_x - g'b_x)^2 \tag{4.28}$$

$$= ab_x^2 - 2ab_x g'b_x + g'b_x^2 \tag{4.29}$$

$$h'a_x^2 = (ab_x - h'b_x)^2 \tag{4.30}$$

$$= ab_x^2 - 2ab_x h'b_x + h'b_x^2 \tag{4.31}$$

substituting these expansions into Equation 4.27 and simplifying produces:

$$-2ab_x h'b_x \stackrel{?}{<} -2ab_x g'b_x \tag{4.32}$$

Assuming attractor a is to the left of attractor b so that ab_x is negative, and substituting $h'b_x = h'_x - b_x$ and $g'b_x = g'_x - b_x$, the swap condition reduces to an ordering relation of the contact coordinates as projected onto the line between the attractors:

$$g'_x \stackrel{?}{<} h'_x \tag{4.33}$$

This simple condition ensures that regardless of the contacts' orthogonal displacement from the line connecting attractors a and b, if contact g as projected onto line $\overline{ab}$ is to the left of the projection h' of contact h, contact g gets assigned to the left attractor a, and contact h gets assigned to the right attractor b. For the relative positions of h and g shown in Figure 4.7, Equation 4.33 is not true, and the swap to the assignments of Figure 4.7a is not taken because the existing assignments of Figure 4.7b already minimize the total distance.

Note that though the simplification to Equation 4.33 depends on the assumptions that the attractors lie on a horizontal line and attractor a is to the left of attractor b, the ordering relation is independent of the horizontal alignment of the attractor pair or the separation between the attractors. Rotation of the coordinate space trivially extends the result to attractor pairs lying on a non-horizontal line. This ordering relation can therefore provide translation and scale invariance to identification of multiple finger contacts.

For comparison to the case when the attractor pair competed for only one real contact, Figure 4.8 contains examples of two contact arrangements with the relevant bisectors shown. In Figure 4.8a and b, both contacts are between the attractors, but in Figure 4.8c and d one contact is to the left of the leftmost attractor. In addition to the perpendicular bisector $B_{ab\perp ab}$ of $\overline{ab}$, a bisector $B_{gh\perp ab}$ of the segment $\overline{gh}$ between the contacts is constructed *perpendicular to* $\overline{ab}$ in each case. Again, the distance-squared swapping criterion (Equation 4.17) ensures that if the contact-attractor links cross the contact bisector as in Figure 4.8a or c, the assignments will be swapped so that the contact to the right of the contact bisector $B_{gh\perp ab}$ always ends up assigned (Figure 4.8b or d) to the attractor to the right of the attractor bisector $B_{ab\perp ab}$, and the contact to the left of the contact bisector $B_{gh\perp ab}$ always ends up assigned to the attractor which is left of the attractor bisector $B_{ab\perp ab}$.

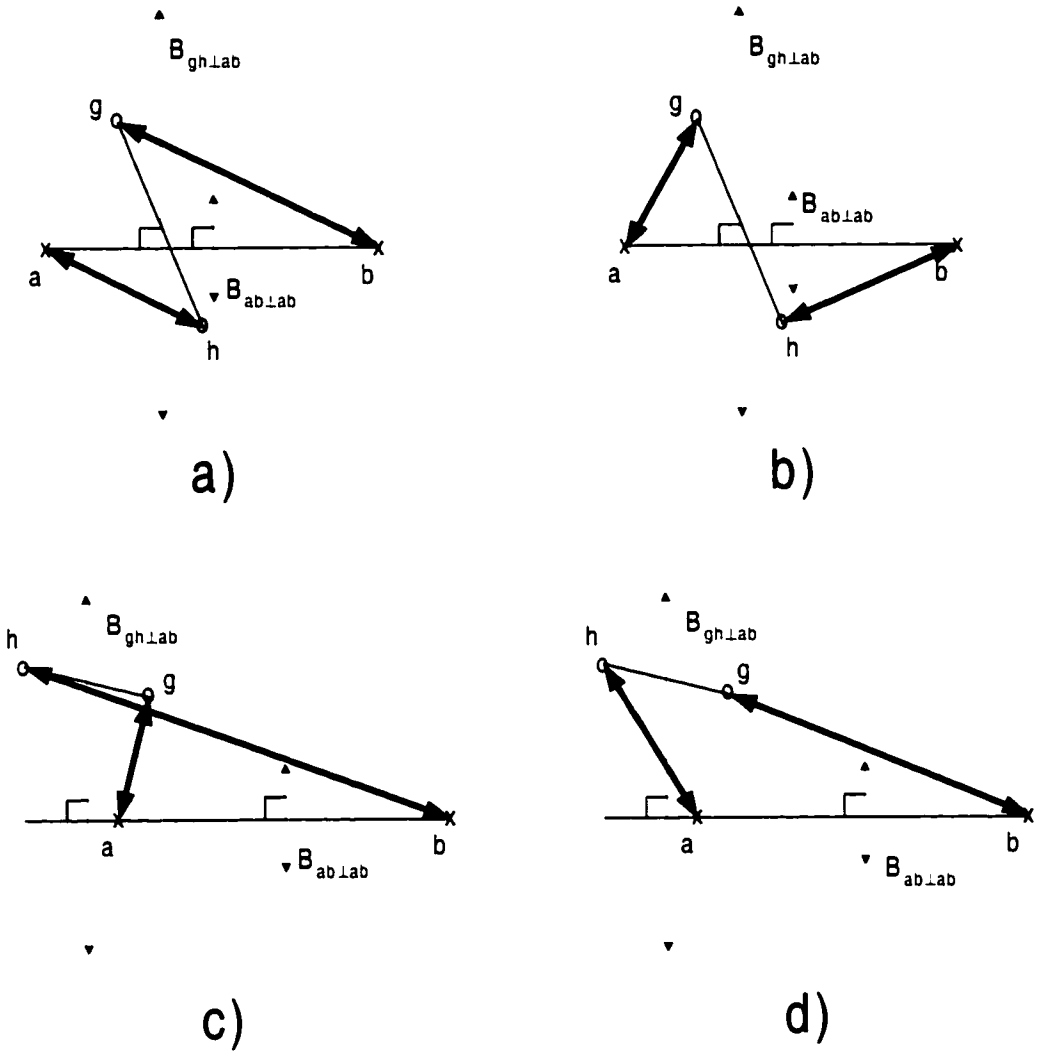


Figure 4.8: Visual comparison of distance-squared assignments via contact pair and attractor pair bisectors (dotted arrows) which are both perpendicular to the segment between the attractor (solid line). The assignments (heavy arrows) in b) and d) have lower distance-squared cost because they preserve the projected ordering. In other words, the contact to the left of the contact bisector, $B_{gh\perp ab}$, goes with the attractor to the left of the attractor bisector, $B_{ab\perp ab}$, and vice versa, regardless of the contact pair's displacement relative to the attractor pair.

4.4.5.3 Summary of Swapping Behavior using Distance-Squared Metrics

In summary, swapping behavior under distance-squared cost is quite different depending on whether the two attractors compete for one or two surface contacts.

- In the one surface contact case, the real contact simply goes to the closest attractor according to the absolute position of the bisector between attractors:

$$g_x \overset{?}{<} \frac{(a_x + b_x)}{2} \tag{4.34}$$

  A dummy contact is assigned to the farther attractor.

- With two competing surface contacts, the contacts are ordered with respect to the inter-attractor angle, but the absolute position of the attractor pair does not matter:

$$g_x \overset{?}{<} h_x \tag{4.35}$$

4.4.5.4 Contact Pair Swapping Behavior with Other Metrics

For metric spaces in which the Euclidean distance between each contact and attractor is taken to some power other than 2, the swapping condition does not simplify so nicely. For example, if the distance power is one, the inequality corresponding to Equation 4.27 includes length product terms:

$$ag_x^2 + bh_x^2 + 2||\vec{ag}||\,||\vec{bh}|| \overset{?}{<} ah_x^2 + bg_x^2 + 2||\vec{ah}||\,||\vec{bg}|| \tag{4.36}$$

which persist through the simplification:

$$||\vec{ag}||\,||\vec{bh}|| - ||\vec{ah}||\,||\vec{bg}|| \overset{?}{<} ab_x(g'b_x - h'b_x) \tag{4.37}$$

This suggests that the swap decision depends on the difference in the products of the link distances as well as the contact ordering as projected onto $\overline{ab}$.

Nevertheless, a couple special cases can illustrate the substantially different swapping behavior which arises with distance powers other than two. Consider

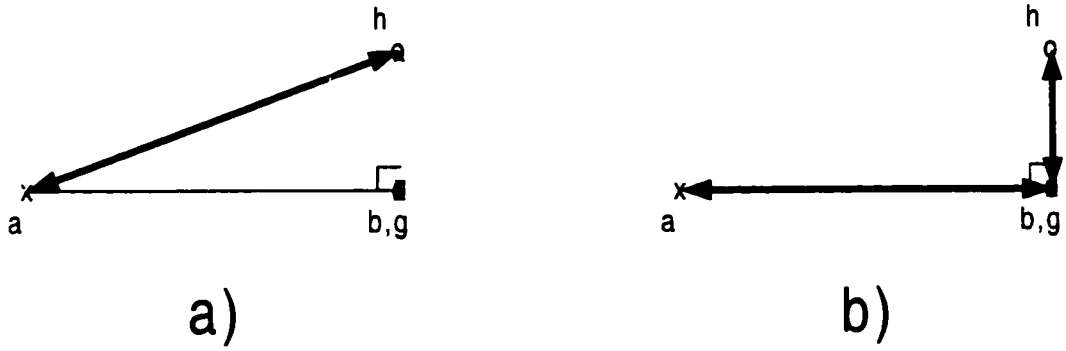


Figure 4.9: Special case when one contact (g) is perfectly aligned on an attractor (b) so that the attractor pair (crosses) and contact pair (circles) form a right triangle. Note the $\vec{bg}$ link in a) is invisible and has zero cost. Under the unsquared-Euclidean metric, a) has a lower total assignment cost than b). With the distance-squared metric, the assignment costs of a) and b) are the same, and with the distance-quadrupled metric, b) has the lower total cost.

Figure 4.9, in which one contact g is right on top of one attractor b, and the other h lies on the perpendicular passing through the same attractor b. Thus the contacts and attractors form a right triangle with both contact g and attractor b at the perpendicular corner, making $||\vec{bg}|| = 0$. The triangle inequality ensures that for the unsquared-Euclidean distance metric, the hypotenuse $||\vec{ah}||$ is shorter than the sum of the lengths of the sides $||\vec{ag}|| + ||\vec{bh}||$, so the assignments will be $g \Longleftrightarrow b$ and $a \Longleftrightarrow h$ as in Figure 4.9a.

For the distance-squared case, the Pythagorean theorem ensures that $||\vec{ah}||^2 = ||\vec{ag}||^2 + ||\vec{bh}||^2$. Therefore neither Figure 4.9a nor Figure 4.9b is preferred since they both produce the same total assignment costs. In other words, both contacts project onto $\overline{ab}$ at the same point, so their order must be chosen arbitrarily.

For the Euclidean-distance-quadrupled metric, the Pythagorean theorem can be invoked again to prove that Figure 4.9b has the lower assignment cost:

$$||\vec{ah}||^4 = (||\vec{ag}||^2 + ||\vec{bh}||^2)^2 \tag{4.38}$$

$$= ||\vec{ag}||^4 + 2||\vec{ag}||^2||\vec{bh}||^2 + ||\vec{bh}||^4 \quad (4.39)$$

$$< ||\vec{ag}||^4 + ||\vec{bh}||^4 \quad (4.40)$$

Note that in this case contact g gets assigned to attractor a even though its distance to attractor b is zero!

Another important case occurs when all attractors and contacts are collinear yet the contact pair is shifted away from the attractor pair as in Figure 4.10. Under

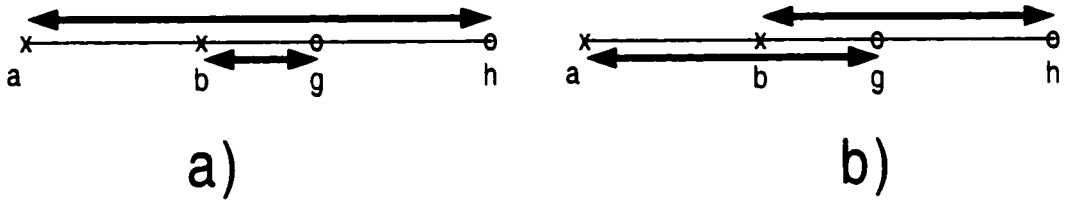


Figure 4.10: Special case when the attractor pair (crosses) and contact pair (circles) are collinear illustrates that unlike the $L2$ metric, the $L1$ metric does not preserve the contact ordering under lateral translation of the contact pair. Under the unsquared-Euclidean metric, the sum of the lengths of the assignment arrows is the same in a) and b) even though the assignments (heavy arrows) of a) reverse the horizontal ordering of the contacts with respect to their assigned attractors. The distance-squared metric produces a lower total cost for more uniform arrow lengths as in b). This causes the horizontal ordering of the contacts to be consistent with that of the attractors under arbitrary horizontal translation of the contact pair.

the unsquared-Euclidean metric, the total of the assignment arrow lengths is the same in Figure 4.10a as in Figure 4.10b even though the lengths within Figure 4.10a are disparate and reverse the horizontal ordering of the contacts with respect to the attractors. The sum of squared lengths favors the more uniform arrow lengths of Figure 4.10b because the long assignment in Figure 4.10a between contact a and attractor h produces a disproportionately large cost.

This preference for more uniform combinations of lengths or costs is one reason squared error is chosen as an error metric for a wide variety of optimization

problems. This section has shown that in the context of an assignment problem, the preference for uniform lengths has the additional consequence of picking the permutation which preserves ordering even when the cluster of contacts is translated with respect to the attractors. This is particularly beneficial for finger identification because the hand position estimates will sometimes be very wrong, causing total misalignment of the attractor ring with the cluster of hand contacts.

4.4.5.5 Distance-Squared Assignment as Sorting

In analogy to Figure 4.10, consider the case when a row of four fingertip contacts is horizontally misaligned with a row of four collinear fingertip attractors. Since the distance-squared swapping condition for any contact pair and attractor pair reduces to an ordering relation (Equation 4.35) on the contact horizontal coordinates, an appropriate exchange neighborhood sequence will sort the contacts with respect to the attractor ordering, producing the same result as a conventional sorting algorithm [79] applied only to the horizontal coordinates of the fingertips. Sorting under distance-squared assignment also tolerates scaling of fingertip spacings as easily as straight horizontal coordinate sorting.

However, assignment to the attractor ring using the distance-squared metric is more general than either horizontal coordinate sorting or assignment using the unsquared Euclidean metric. Recall that with the distance-squared contact pair swapping condition, the contacts are ordered according to their projection onto a line whose angle matches the angle between two given attractors. Thus when three or more attractors are not collinear, the ordering relation changes to fit the angle of the local attractor pair, providing sorting along arbitrary arcs or around a ring. Technically, the pairwise ordering relations will cease to be transitive when the attractors are not collinear, *i.e.* $f < g$ and $g < h$ no longer implies $f < h$. Though this creates the possibility of sorting failures as discussed in Appendix C, the sorting effect still works as long as the change in adjacent attractor angles along the arc is

fairly gradual. Though the unsquared-Euclidean metric can also sort along arcs or around the ring when the ring is perfectly aligned with the contacts, the example of Figure 4.10 shows that the unsquared-Euclidean metric has no preference for the proper fingertip ordering once the attractor ring becomes misaligned by more than one attractor spacing.

Unfortunately, these translation-invariant sorting properties of distance-squared assignment are only effective when the attractor ring is full or nearly full of contacts, *i.e.*, when most of the hand parts are touching the surface. On the edges of a cluster of only two to four finger contacts, empty attractors will compete for real contacts, potentially giving up dummy contacts. As was shown in Section 4.4.5.1, when two attractors compete for one real contact, the swapping condition is equivalent for both squared- and unsquared- distance metrics and offers no special protection against attractor ring misalignment. The absolute position comparison of Equation 4.34 causes dummy contacts to propagate toward those attractors which are farthest from the real contacts, leaving erroneous shifts in fingertip assignment, but the contact pair ordering relation still holds within groups of adjacent attractors retaining real surface contacts, ensuring that the fingertips are at least ordered correctly.

4.4.5.6 Analyzing Swaps on the Attractor Ring

When applied to an attractor ring which is full of fingertips, the geometric interpretations of the contact pair swapping conditions indicate exactly how much local deviation in finger arrangement will be tolerated before finger identities are erroneously swapped. For example, consider the attractor ring and five fingers in Figure 4.11. The fingers are arranged with the same relative angles as the attractors except the whole hand is translated about three centimeters toward the upper right with respect to the attractor ring. Since the contact pair swapping condition for the distance-squared metric is invariant to translation, and since the contact pairs

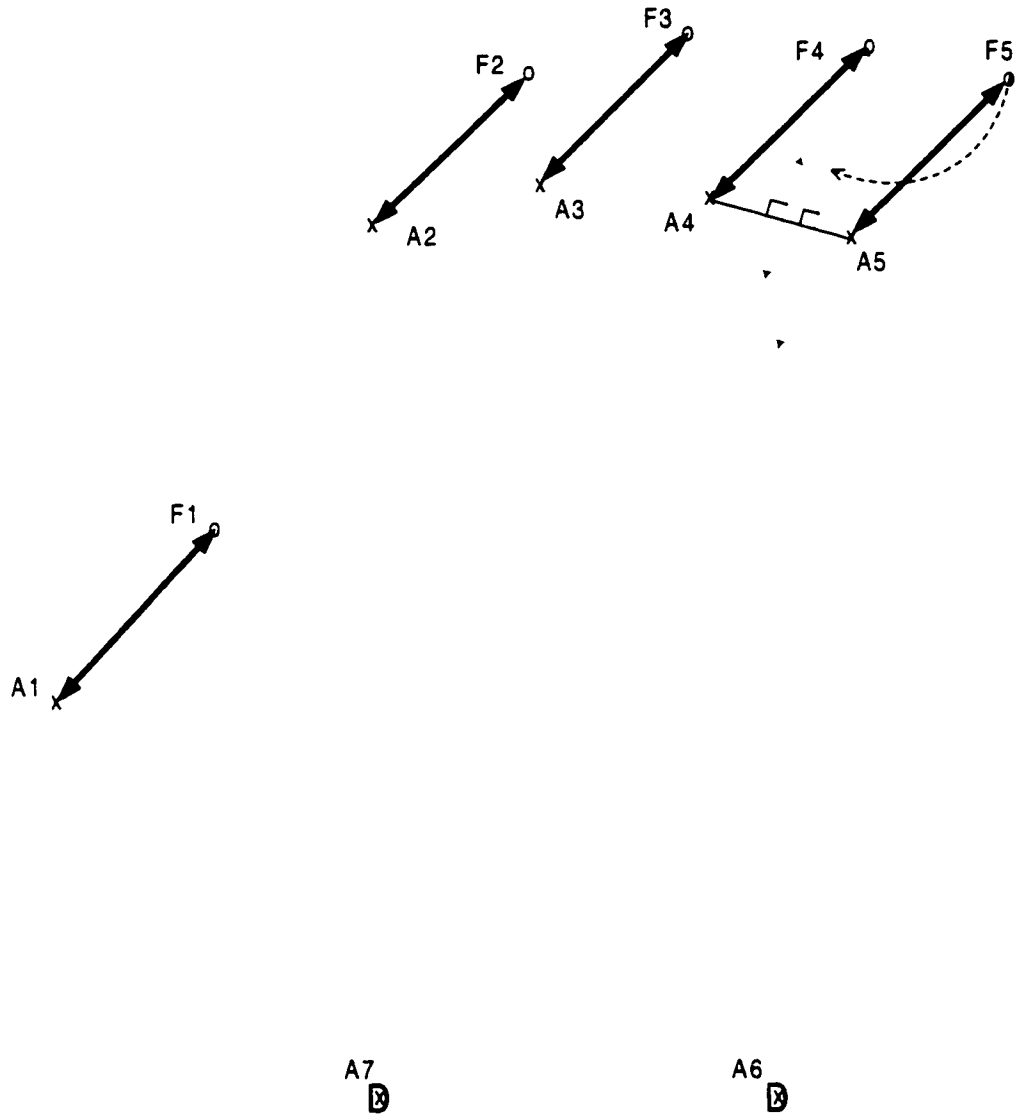


Figure 4.11: Tolerance of hand translation and finger pair deviation in assignment of five fingers to an attractor ring. Under the distance-squared metric, the correct assignments (heavy arrows) of five fingers will be produced for any translation. Furthermore, up to 90° of rotation of the pinky (F5) about the ring finger (F4) from the default angle is guaranteed to be tolerated. Erroneous identity swapping commences when the angle of the perpendicular bisector (long dotted arrow) between A4 and A5 is reached. Even though the palm heels are not touching, the open (or dummy-filled) palm attractors are too far down to receive any finger contacts.

constrain one another within the arc of finger attractors, the exchange sequence will find the correctly ordered finger assignments as shown.

Now suppose the pinky finger (F5) contact begins to rotate clockwise around the ring finger (F4). The pinky's identity will be maintained correctly for a rotation of up to 90° from the default ring-pinky angle. After a rotation of 90° both the ring (F4) and pinky (F5) contacts will be aligned on their bisector, perpendicular to the segment joining the ring (A4) and pinky (A5) attractors and parallel to the Voronoi cell wall (perpendicular bisector of (A4-A5) segment) between the ring and pinky attractors. In other words, the ring (F4) and pinky (F5) contact projections onto the ring-pinky attractor segment will be the same point, and their identities may swap unstably. Any rotation past 90° would definitely cause the ring fingertip (F4) to be assigned to the pinky attractor (A5) and the pinky fingertip (F5) to be assigned to the ring attractor (A4).

Thus *even in the face of attractor ring misalignment, the tolerance for finger pair crossover can be determined precisely.* Basically, the identifications are most stable when the inter-contact angles match the inter-attractor angles. Identities of two fingers will begin to be swapped erroneously as the angle between them approaches the angle of the perpendicular bisector between their attractors, which is usually available from the Voronoi diagram. With the unsquared-Euclidean metric, this rule would only hold when the contacts were roughly centered between their attractor pair on the attractor bisector. Given the amount of translation in Figure 4.11, the Euclidean metric probably would not produce the proper finger ordering even without finger pair rotation.

To understand the identifications which will result when only two to four fingers are touching the surface, leaving some finger attractors open, one must combine the geometric interpretation of assignment of one contact to an attractor pair (Section 4.4.5.1, Figure 4.6, Equation 4.20) with that of contact pair assignment

(Section 4.4.5.2, Figure 4.8, Equation 4.35). Since the single contact swapping condition is not as tolerant of translation as the contact pair swapping condition, the identifications will become more and more vulnerable to corrupting translations as more attractors are left unfilled. In general, finger identifications can be erroneously shifted to the right or left by as many attractors as the number of open finger attractors (5 - number of touching fingers) under sufficient horizontal translation. Finger contacts are rarely shifted into open palm attractors (Figure 4.11) because the palm attractors are so far below, meaning erroneous identification would require a vertical misalignment of more than five centimeters. Also, palm heels have fairly unique features which will be used to discriminately weight assignment to them in Section 4.4.6.

Suppose the pinky finger contact (F5) is removed from Figure 4.11 and a dummy contact (D) is assigned to the pinky attractor (A5) in its place. This situation is shown in Figure 4.12. As soon as the swapping test is applied to the ring (A4) and pinky (A5) attractor pair, the ring finger (F4) will be assigned to the pinky attractor (A5) since it is closer to the pinky attractor, *i.e.*, on the right side of the perpendicular bisector of the attractor pair. The ring attractor (A4) will receive the dummy contact (D). When swap testing proceeds to the A3-A4 attractor pair, A3 will receive the dummy contact (D) and A4 will receive F3 for similar reasons. After swap testing of the A2-A3 pair, the dummy contact (D) will arrive at A2, the index finger attractor, and stay there as shown in Figure 4.13. It will stay because for the given amount of hand translation, the thumb (F1) is still closer to the thumb attractor (A1) than the index finger attractor (A2). In other words, no real contacts lie in the index finger Voronoi cell, so the dummy contact remains there.

Notice that though the assignments of the three remaining fingertips have been erroneously shifted to the right, the distance-squared contact pair swapping condition will still maintain their proper ordering. The ordering is still governed by

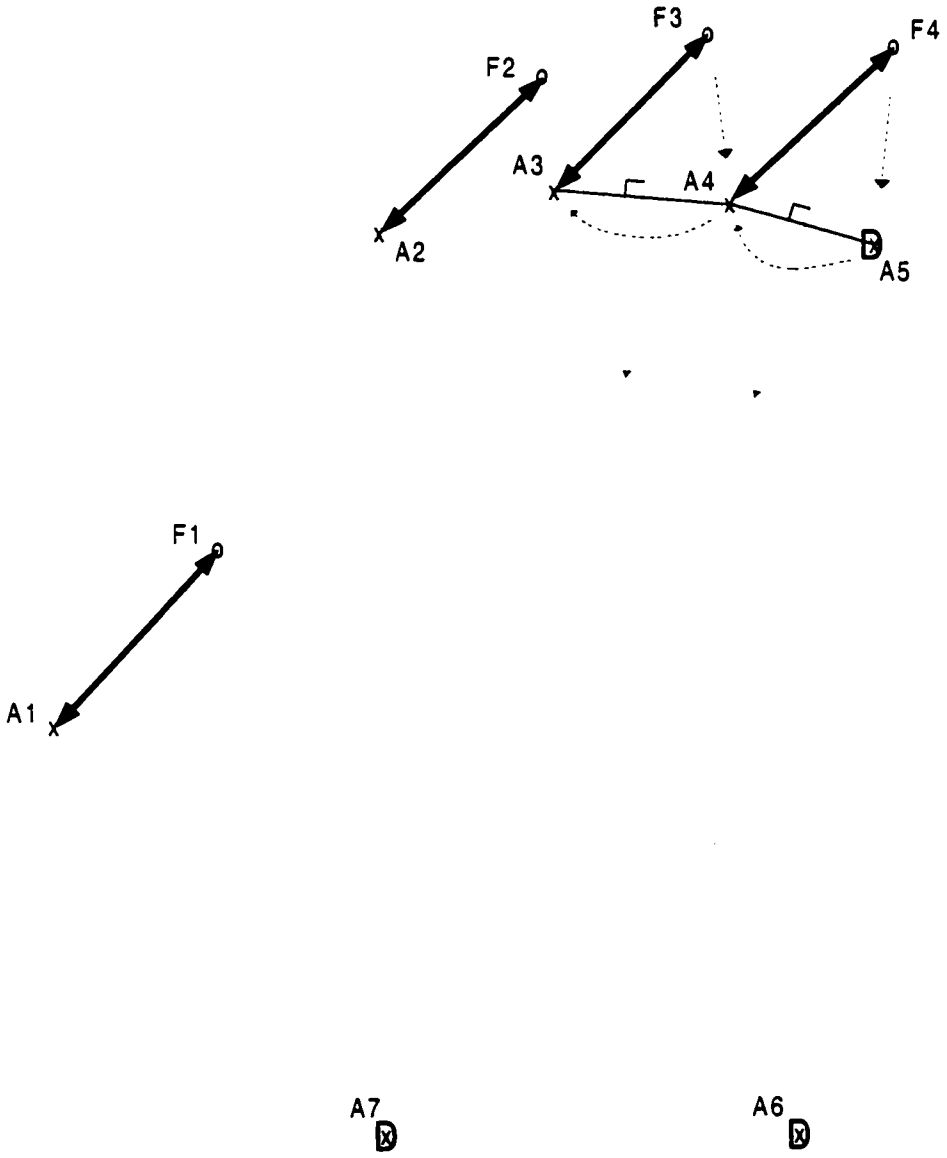


Figure 4.12: Identity swaps which occurs after the pinky finger contact is removed and replaced with a dummy contact (D). The assignments of the remaining fingertip contacts will shift right due to the hand translation as the dummy contact propagates toward the index finger attractor.

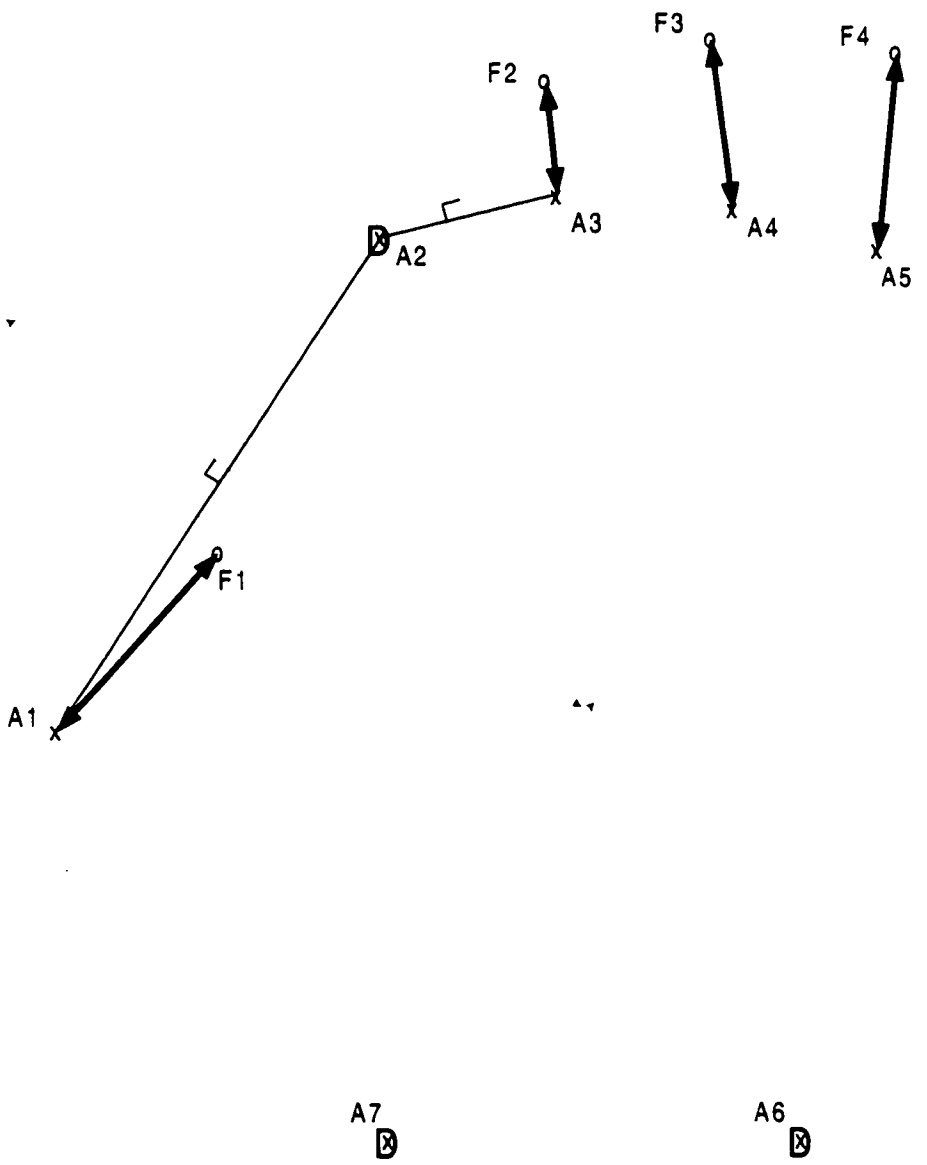


Figure 4.13: The dummy contact (D) propagates to the index finger attractor when the pinky finger is removed. It remains there because no real surface contacts lie in the index finger Voronoi cell.

inter-contact angles in the same way as in Figure 4.11, except that in this case the middle (F3) and ring (F4) finger identities will be swapped if the ring finger rotates more than 90° past the angle between the ring (A4) and pinky (A5) attractors.

4.4.6 Tuning the Attractor Ring with Weighted Voronoi Diagrams

As presented so far the attractors have been unweighted, which causes the Voronoi cells formed from perpendicular bisectors to be shaped like polygons. Multiplicatively weighting the distances to particular attractors causes the equidistant bisectors between contacts with different weightings to be circles instead of straight lines [116]. The center of these *Apollonius* circles lies along the line between the two attractors but to the outside of the attractor with the smaller weight [116]. Weighted Voronoi cells can therefore be composed of arcs as well as straight lines, and the *Apollonius* circles of attractors with large weights can contain "holes" caused by attractors with tiny weights [116]. By convention, the distances are divided by the weightings w_{ij}, *e.g.* $c_{ij} = d_{ij}^2/w_{ij}$, so that as the weighting of an attractor gets smaller, the attractor's Voronoi cell does too, and vice versa.

Static weightings of particular attractors can warp and resize the Voronoi cells to better match the range of motion of each hand part. For instance, since the thumb and pinky have a much wider range of motion than the palm heels, it will be advantageous to shrink the palm heel Voronoi cells and expand the thumb and pinky Voronoi cells into space vacated by the palm cells.

Dynamic weightings of particular attractors in proportion to the strengths of distinguishing contact features will also enlarge or shrink certain Voronoi cells in relation to those of neighboring attractors. Attractor points whose Voronoi cells appear dynamically enlarged to contacts with appropriate features are more likely to be assigned those contacts even in the face of large attractor ring alignment errors. While individual attractors can be weighted independently and the attractor ring as

a whole can be translated from the default hand position, note that the positions of individual attractors with respect to the rest of the ring never need to be changed.

4.4.6.1 Constant Additive Weighting to the Distance Matrix

Before applying any multiplicative weights to the contact-attractor distances, a constant additive weight of about 2 cm^2 is added to each distance matrix entry, d_{ij}^2. Since this offset is applied uniformly to every distance, it has no effect when assignments are unweighted except adding $2M$ to all assignment sums. Thus it does not affect the structure of the unweighted Voronoi diagram. Its purpose will be to ensure dynamic contact feature weightings are effective even when a contact is precisely on top of an attractor, that is when $d_{ij}^2 \approx 0$. This effectiveness depends on the fact that in such "compound-weighted" Voronoi diagrams, the combination of the constant offset and a tiny weighting can cause a Voronoi cell to disappear entirely [116]. Likewise, the combination of the constant offset and a huge weighting can cause the Voronoi cell of one attractor to take over the entire surface.

To illustrate this, suppose a contact's features are inconsistent with an attractor, causing its feature weighting w_{ij} for that attractor to be small, yet the contact lies right on top of the attractor. Without the constant distance offset, the weighted distance will still approach zero, *i.e.*, $d_{ij}^2/w_{ij} \approx 0$, not reflecting the feature mismatch. With the constant offset, the weighted distance $(d_{ij}^2 + 2)/w_{ij} \approx 2/w_{ij}$ will always reflect the weighting somewhat, even as the Euclidean distance goes to zero. The choice of 2 cm^2 reflects both empirical testing and the argument that since attractor ring alignment errors average a centimeter or two and fingers have ranges of motion of several centimeters from their default positions, precise alignment of a contact over an attractor is more happenstance than a strong indicator of contact identity.

What does it mean for a Voronoi cell to shrink and vanish? If the Voronoi cell is from the diagram of the entire attractor ring, it simply means that a sole

hand contact will not be assigned to the vanished cell's attractor, regardless of the contact's position on the surface. However, when multiple contacts are competing for attractors, an attractor whose cell has vanished from the Voronoi diagram can still receive a contact simply because all other nearby attractors may already be assigned to other contacts.

4.4.6.2 Static Palm Heel Weightings

In the unweighted attractor ring of Figure 4.5 on Page 141, the palm attractors had to be placed a couple centimeters lower than the measured default palm heel positions. Moving these attractors forward to their proper vertical positions of -4 cm would have enlarged the palm heel Voronoi cells too much at the cost of thumb and pinky Voronoi cell sizes. Since the thumb and pinky have much wider ranges of motion with respect to hand center than the palm heels, it would make much more sense for the palm heel Voronoi cells to be more compact than the thumb and pinky cells.

The MTS achieves compact palm heel Voronoi cells by including a small weighting on all squared distances between contacts and palm heel attractors. This weighting has been empirically chosen to be .25, *i.e.*, $w_{i6} = w_{i7} = .25$. Since the weights modify the squared distance, this effectively doubles the unsquared Euclidean distance to a palm heel attractor compared to distances to other attractors. As shown in Figure 4.14, this causes the bisectors between the inner palm heel and thumb and between the outer palm heel and pinky to contract into circles around the palm heels. Since the relative weightings between the palm heels are the same, the bisector between inner and outer palm heels continues to be a perpendicular line. Because the weightings limit the lateral and upward extent of the palm heel Voronoi cells, with the weightings it is safe to move the palm heel attractors forward to their actual measured default positions, as is done in Figure 4.14.

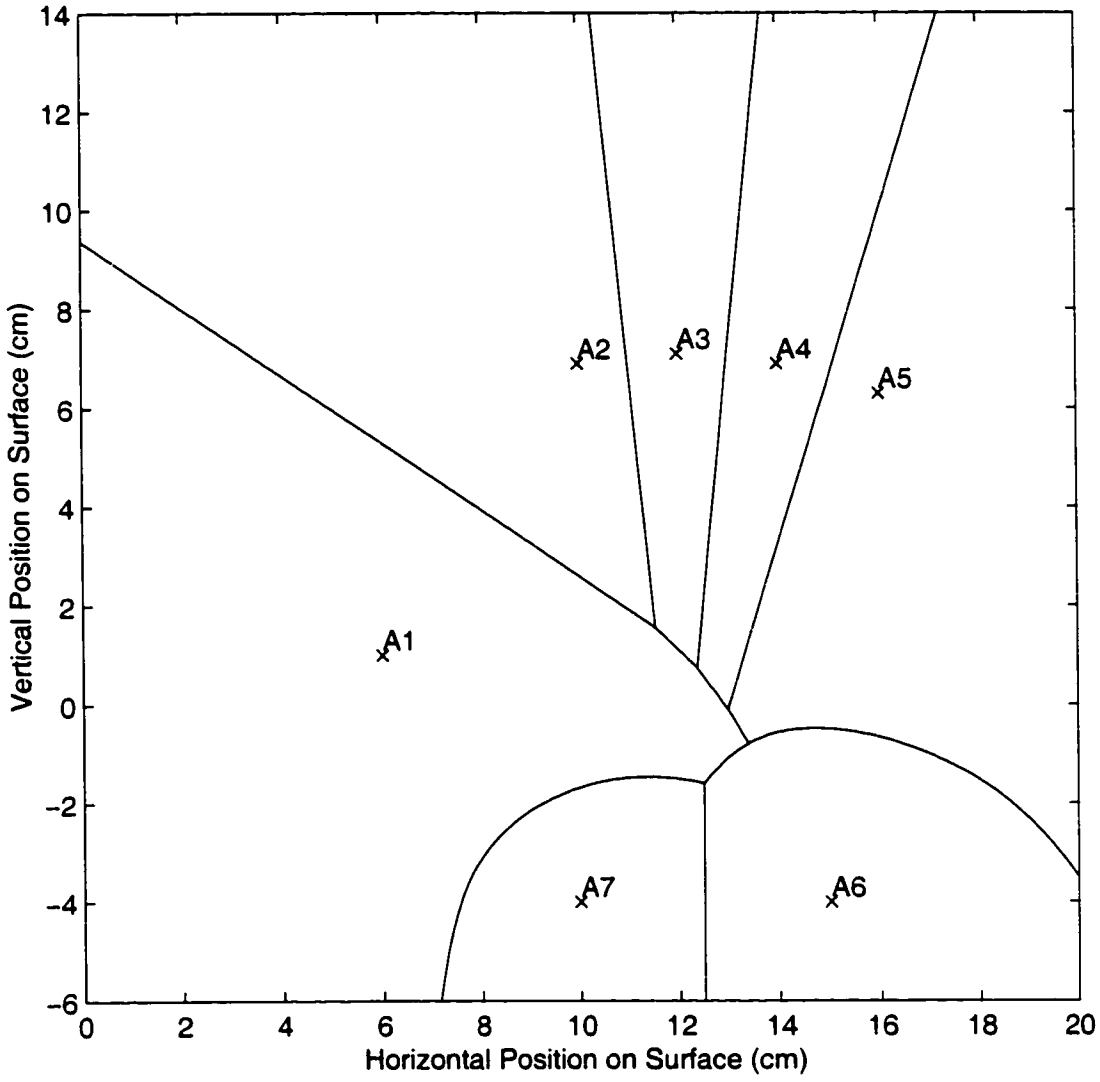


Figure 4.14: Voronoi diagram with distances from contacts anywhere in the plane to palm heels weighted to be twice as far as normal.

With these weighted palm heel attractors, the weighted Voronoi cells match the range of motion of each finger quite well. This is shown by the finger motion trajectories superimposed on the weighted Voronoi diagram in Figures 4.15 and 4.16. Figure 4.15 contains the entire range of finger flexion and extension, from outstretched hand to fist, for the author's hand. At the end of the trajectories the fingertips are actually curled under so the knuckles begin to touch the surface. The palms remain fixed on the surface throughout the flexing.

Note that each finger remains within its Voronoi cell over the entire motion range. This demonstrates that the isolated touchdown of any finger anywhere along its flexion range will result in correct identification as long as the horizontal alignment of the attractor ring is correct. The horizontal alignment as determined by the hand offset estimates will generally be perfect as shown when the hand is centered on its default position or if both palm heels are resting anywhere on the surface. If some other fingers are or were recently touching the surface and were properly identified, the horizontal offset estimate will still be within a couple centimeters of the correct alignment but probably will not be quite as good as shown. When all four fingertips are touching the surface, the global assignment optimization will find their correct identities by exclusion from the thumb and palm attractors, regardless of attractor ring alignment.

Figure 4.16 demonstrates the superb fit of the weighted Voronoi diagram when one finger at a time sweeps out its circular range of motion while all others remain resting in their default positions. Note how the circular sweeps of the thumb, index, and pinky fingers are closely circumscribed by their Voronoi cells. Though the hand started in default position, the author did not look at the Voronoi diagram while sweeping out the circles, so it is surprising that the finger sweeps so closely match a Voronoi diagram constructed only from default finger positions and empirically determined palm heel weightings.

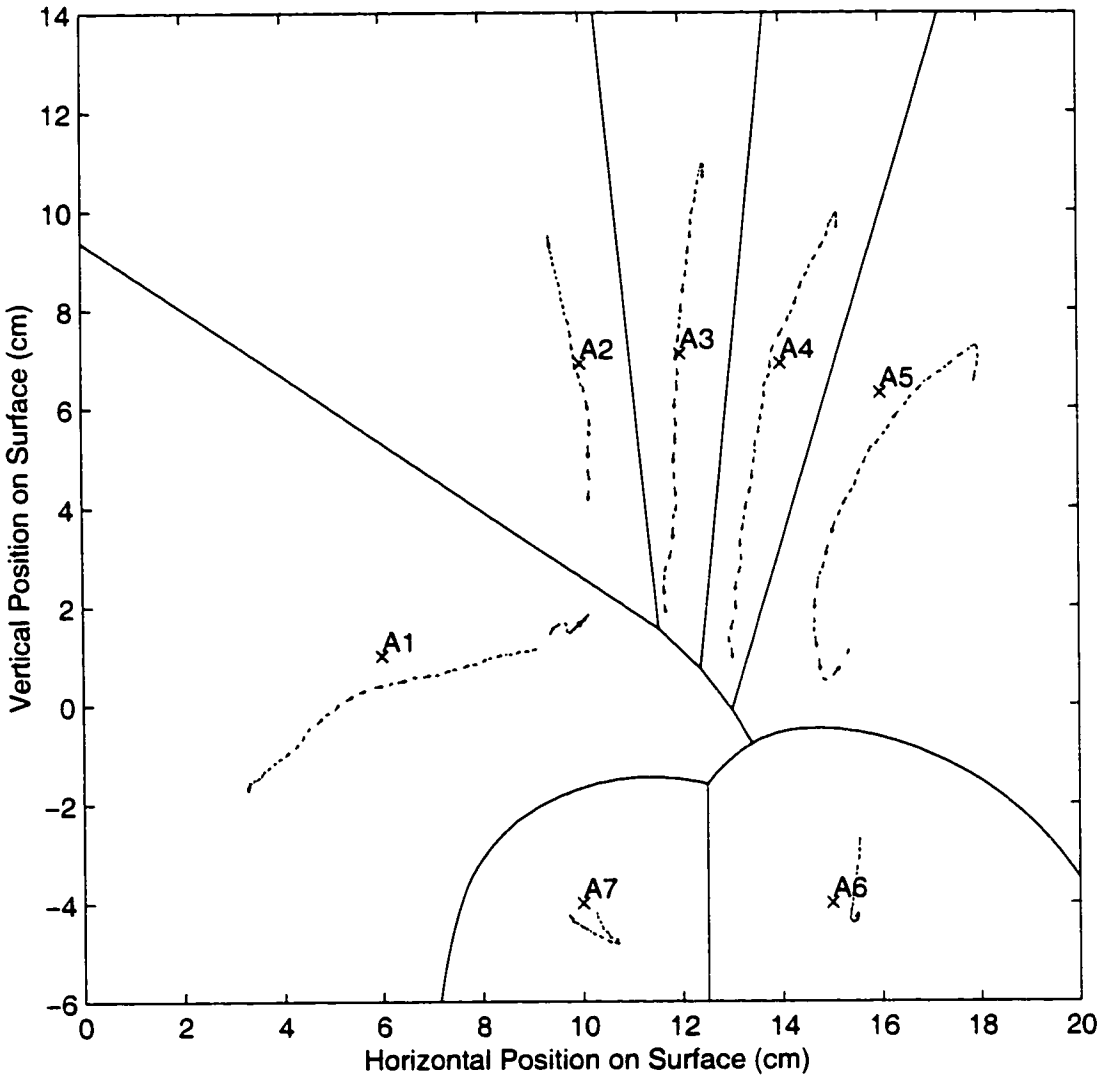


Figure 4.15: Weighted Voronoi diagram with flexing finger trajectories (tiny arrows) superimposed. The fingers of the author's hand started fully extended and outstretched and then flexed simultaneously into a fist. Note that the palm heels remain stationary over their attractors and each finger remains within its Voronoi cell over the entire motion range.

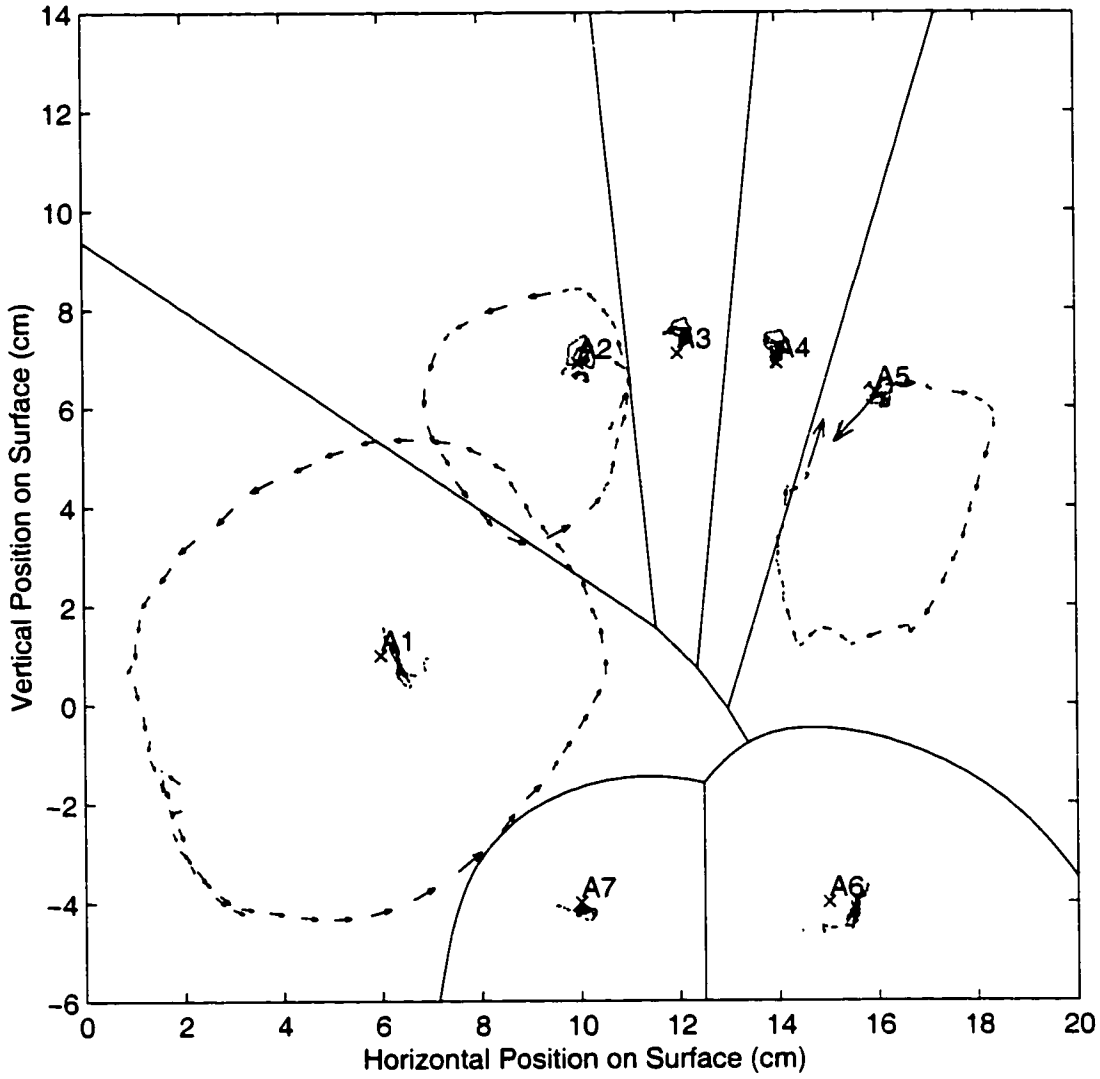


Figure 4.16: Weighted Voronoi diagram with rotating finger trajectories (tiny arrows) superimposed. All fingers of the author's hand started in their default positions right over the attractors. One at a time, the thumb, index and pinky fingers were picked up and made to sweep out their range of motion while the other fingers remained resting in their default positions. Note that the palms were allowed to lift off the surface and shift laterally to capture the full thumb range, but the palms remained on the surface at all other times. Because the middle and ring fingers have little room to move laterally while their adjacent fingers are resting, they were left out of the sweeping experiment.

4.4.6.3 Dynamic Feature Weightings

The dynamic contact-attractor distance weightings depend on whether the geometric features of the given contact match those expected from the hand part that the attractor represents. Since the thumb and palm heels exhibit the most distinguishing geometric features, weighting functions are computed for the thumb and palm heel attractors, and distances to fingertip attractors are unchanged. Each weighting function is the product of several factor versus feature relationships. Each weighting factor is designed to take on a default value of 1 when its feature measurement provides no distinguishing information, take on larger values if the measured contact feature uniquely resembles the given thumb or palm heel, and take on smaller values if the measured feature is inconsistent with the given attractor's hand part. Thus the larger a particular feature factor for a particular contact-attractor pair, the larger the attractor's Voronoi cell will appear to the contact, and the more likely the contact will fall within that cell and be assigned to its attractor.

Since each contact can have a different feature weighting for matching to the same attractor, each contact can encounter a differently warped Voronoi diagram. For example, a sufficiently large thumb weighting for a contact could make the thumb attractor so powerful that the Voronoi diagram encountered by the contact could contain only one Voronoi cell, a thumb Voronoi cell covering the whole surface. In actuality, thumb contact features are never unambiguous enough to warrant weightings this large.

The weighting functions were arrived at by trial and error. Since each weighting function can move the boundaries of one Voronoi cell, the experimenter typically decides from finger motion ranges where each boundary should be. The current boundary positions can be determined by repeatedly lifting and touching a finger over different spots until a surface position is found where finger identity becomes unstable, alternating between two Voronoi cells each time the finger touches. The

experimenter then adjusts the amplitude of the weighting function until the boundary moves to the desired location. Care must be taken to balance the thumb and palm weightings or the boundary between thumb and inner palm heel Voronoi cells may shift unintentionally.

4.4.6.4 Thumb and Inner Palm Orientation Factor

Figure 4.17 shows the right thumb and right inner palm heel orientation

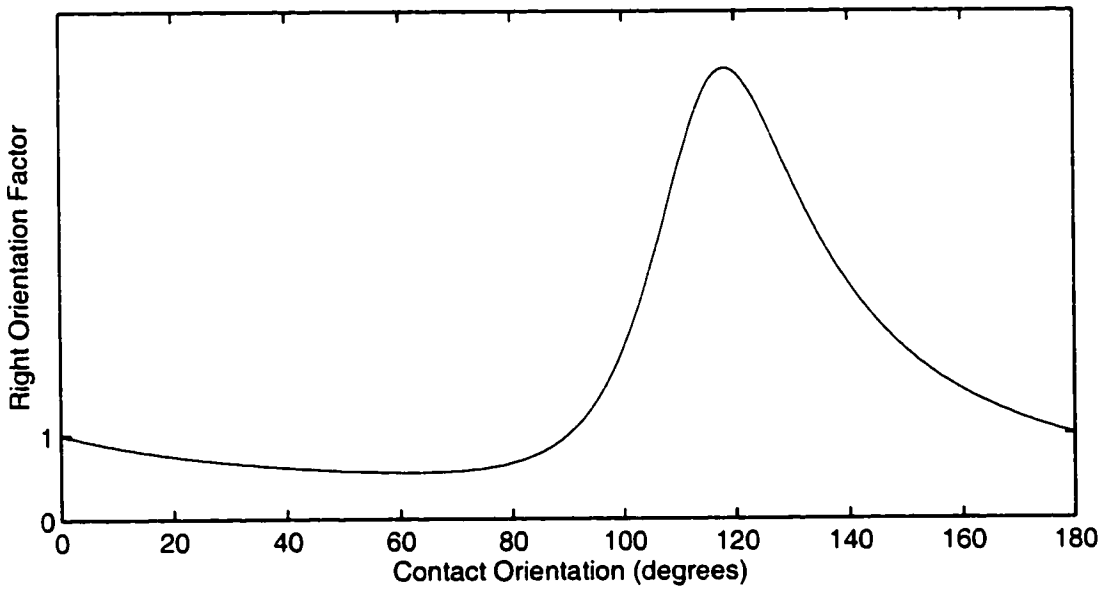


Figure 4.17: Right thumb and inner palm heel orientation factor, $Pi_{worient}[n]$, versus orientation of the contact's fitted ellipse, $Pi_{\theta}[n]$.

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

the opposite thumb or palm heel. The corresponding relationship for the left thumb and inner palm heel orientation factor is flipped about 90°.

4.4.6.5 Thumb Size Factor

Figure 4.18 approximately plots the thumb size factor. Since thumb size as

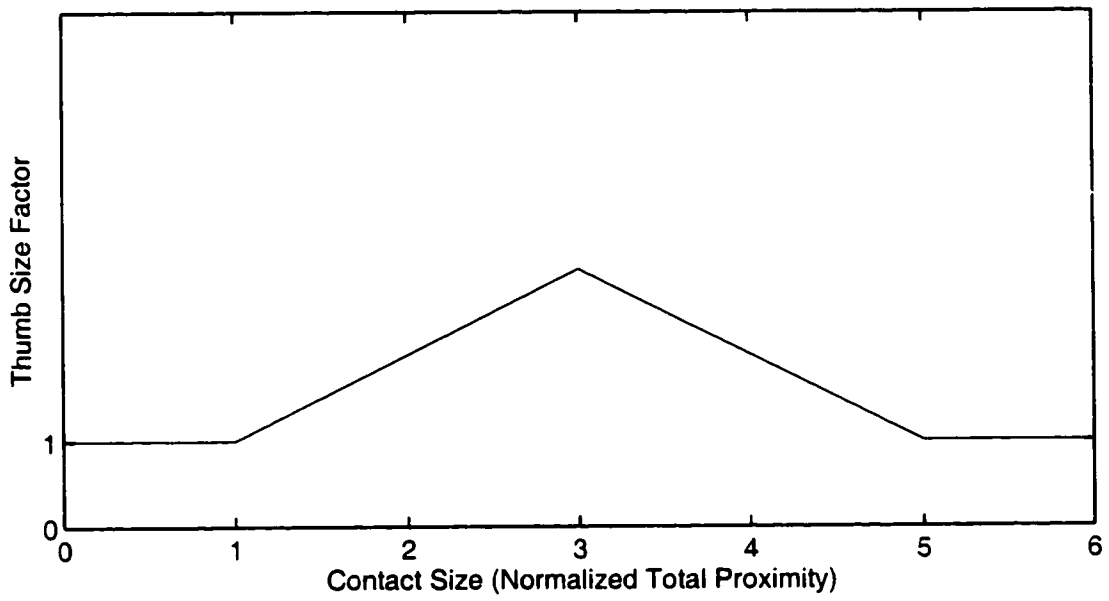


Figure 4.18: Thumb size factor, $Pi_{wthumb\_size}[n]$ versus a contact's total proximity, $Pi_z[n]$.

indicated by total proximity tends to peak at two or three times the size of the typical curled fingertip, the thumb size factor peaks at these sizes. Unlike palm heels, thumb contacts cannot be much larger than two or three times the default fingertip size, so the thumb factor drops back down for larger sizes. Since any hand part can appear small when touching the surface very lightly or just starting to touchdown, small size is not distinguishing, so the size factor defaults to 1 for very small contacts.

4.4.6.6 Palm Heel Size Factor

Figure 4.19 approximately plots the palm heel size factor. As more pressure

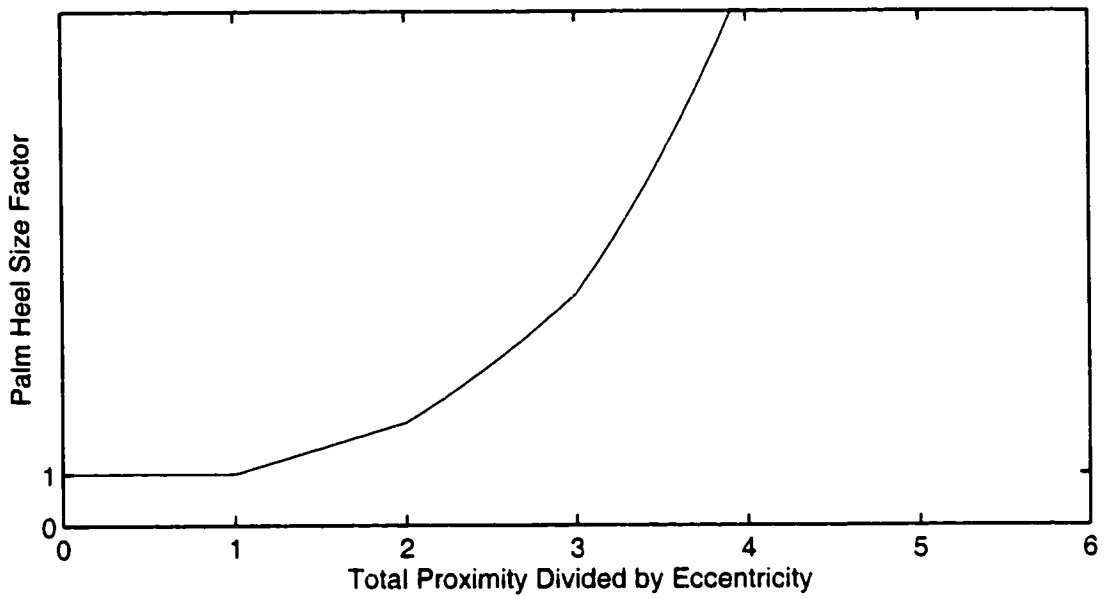


Figure 4.19: Palm heel size factor, Pi_{wpalm_size} versus the ratio of a contact's total proximity to its eccentricity, $P_z[n]/P_\epsilon[n]$.

is applied to the palms, the palm heel contacts can grow quite large, remaining fairly round as they do so. Thus the palm heel size factor is much like the thumb size factor except the palm factor is free to increase indefinitely. For palm heels larger than the maximum expected thumb size, the palm heel size factor becomes so large that the palm heel Voronoi cells engulf the entire surface. Thus if the full weight of the hands rests on the palm heels and fully flattens them, they will be correctly identified anywhere on the surface, regardless of attractor ring alignment.

However, fingertip contacts can grow by becoming taller as the fingers are flattened. But since finger width is constant, the eccentricity of an ellipse fitted to a growing fingertip contact increases in proportion to the height. To prevent flattened fingers from having a large palm factor, the size measure is modified to be the ratio of total contact proximity to contact eccentricity. This has little effect for palms,

whose eccentricity remains near 1, but cancels the high proximities of flattened fingertips. Though directly using the width from the contact's fitted ellipse would be less accurate for low resolution electrode arrays, the proximity to eccentricity ratio basically indicates contact width.

4.4.6.7 Palm Heel Separation Factor

Another important distinguishing feature of the palm heels is that wrist anatomy keeps the centroids of their contacts separated from one other and from the fingers by several centimeters. This is not true of the thumb and fingertips, which can be moved within a centimeter of one another via flexible joints. Minimum contact separation can be measured without knowing contact identities by searching all contacts for the nearest neighbor contact of a given contact and measuring the distance to that neighbor. As plotted approximately in Figure 4.20, the

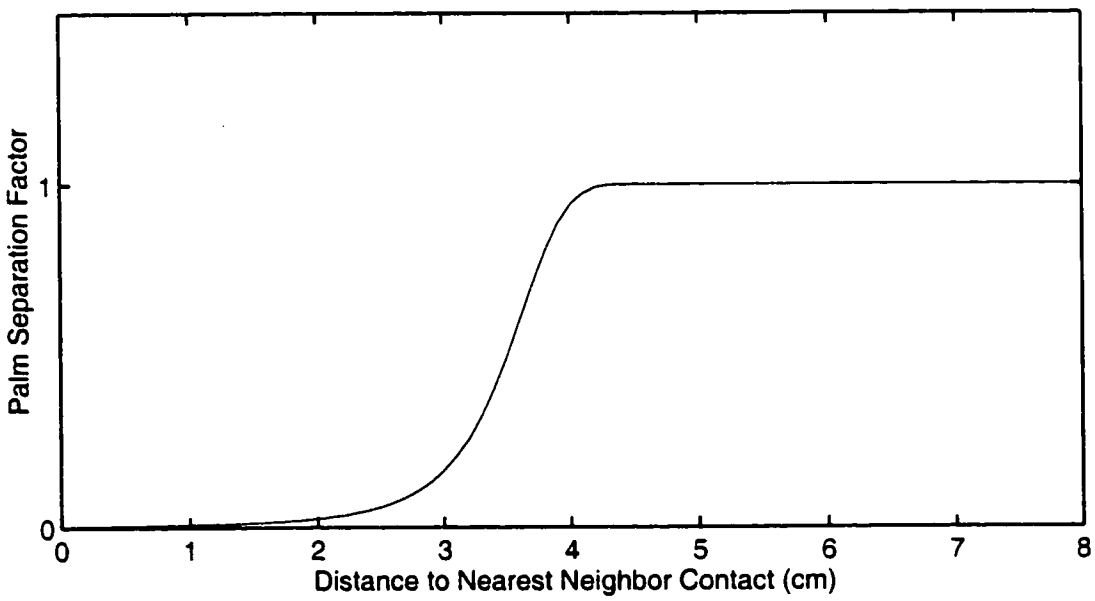


Figure 4.20: Palm heel separation factor, $Pi_{wpalm\_sep}[n]$ versus the Euclidean distance between contact Pi and its nearest neighbor contact.

palm separation factor quickly decreases as the separation between the contact and its nearest neighbor falls below a few centimeters, indicating that the given contact (and its nearest neighbor) are *not* palm heels.

Unlike the size and orientation factors, which only become reliable as the weight of the hands fully compresses the palms, the palm separation factor is especially helpful in distinguishing pairs of adjacent fingertips from palm heels because it applies equally well to light, small contacts. For small separations, this weighting factor becomes so small that the palm heel Voronoi cells vanish. Therefore, a pair of contacts which are within about 3 cm of one another will not be identified as palm heels, regardless of their position on the surface or the alignment of the attractor ring. However, the palm separation factor should only be made this influential if the segmentation system always either merges the two palm heels into one huge palm contact or divides the palm across its central vertical crease into exactly two heel contacts. If the segmentation system erroneously splits one of the palm heels into two contacts which are less than 3 cm apart, the small separation factor which results can cause both contacts to be identified erroneously as fingers, again regardless of the alignment of the attractor ring.

4.4.6.8 Forepalm Attractors and Weightings

The MTS includes four additional attractors near the center of the attractor ring to handle forepalms contacts and other extra groups from segmentation of flattened palms. This increases the size of the contact attractor distance matrix to 11×11. Since the forepalms typically do not touch the surface unless the rest of the hand is flattened onto the surface as well, the forepalm attractors must be weighted such that contacts near hand center are assigned to them only when the hand is flattened. When the hand is not flattened, contacts near hand center might be fingers accessing keys on the bottom row of the key layout and should not be identified as forepalms.

The easiest way to determine if the hand or palm is flattening is to measure the total proximity of all contacts assigned to the hand, $H_{totalz}[n]$. A forepalm weighting function is devised to be so small when the hand is not flattened that the forepalm Voronoi cells totally vanish:

$$w_{i,8-11} = \frac{1}{\max(2, (8 - H_{totalz}[n]))} \tag{4.41}$$

Only when the total hand proximity $H_{totalz}[n]$ becomes large due to finger or palm flattening will the forepalm weightings be large enough that their Voronoi cells appear near the center of the hand. To discourage fingers or palms away from hand center from being assigned to forepalm attractors, forepalm entries in the contact-attractor distance matrix are squared again, making forepalm assignment costs vary with the fourth power of Euclidean distance. The final forepalm assignment costs $[c_{i,8-11}]$ can then be written:

$$c_{i,8-11} = \max(2, (8 - H_{totalz}[n])) \times (d^2_{i,8-11} + 2)^2 \tag{4.42}$$

4.4.6.9 The Fully Weighted Assignment Cost Matrix

All of the static and dynamic weightings are combined to form a fully weighted assignment cost matrix $[c_{ij}]$, where:

$$c_{ij} = \begin{cases} (d^2_{ij} + 2)/(Pi_{wthumb\_size}[n] Pi_{worient}[n]) & \text{if } j == 1 \\ (d^2_{ij} + 2) & \text{if } 2 <= j <= 5 \\ 4(d^2_{ij} + 2)/(Pi_{wpalm\_size}[n] Pi_{wpalm\_sep}[n]) & \text{if } j == 6 \\ 4(d^2_{ij} + 2)/(Pi_{wpalm\_size}[n] Pi_{worient}[n] Pi_{wpalm\_sep}[n]) & \text{if } j == 7 \\ \max(2, (8 - H_{totalz}[n])) \times (d^2_{ij} + 2)^2 & \text{if } 8 <= j <= 11 \end{cases} \tag{4.43}$$

The basic assignment optimization of Equation 4.12 is then restated as finding the permutation $\{\pi_1, \ldots, \pi_{11}\}$ of integer hand part identities $\{1, \ldots, 11\}$ which minimizes:

$$\sum_{i=1}^{11} c_{i\pi_i} \tag{4.44}$$

where c_{ij} is the weighted distance from contact i to attractor j, and contact i and attractor j are considered assigned to one another when $\pi_i \equiv j$. Though the various weightings can warp the Voronoi cells in complex ways which were not anticipated in the design of the combinatorial search exchange sequence (Section 4.4.4.4), no convergence failures have been noticed as long as palm attractors are allowed to swap with any other attractor on the ring. Making the exchange neighborhoods of palm attractors include all other attractors ensures that when a palm Voronoi cell vanishes it can give up its contact to any finger attractor, or when a palm Voronoi cell engulfs the surface it can accept a contact from any finger attractor.

4.4.6.10 Tolerance of Different Hand Sizes

Though the default finger positions and thus the attractor positions were determined from the author's medium-sized male hand, anthropomorphic data suggests that the given attractor ring will perform well for most adult hands. Wagner's study [150] of anthopometry and biomechanics in the pianist's hand found less than a 10% standard deviation in all hand shape parameters except the prominences (relative lengths with hand flattened) of the thumb and pinky. While male hands are on average nearly 10% larger than female hands, the range of adult hand sizes for a given hand posture is much narrower than the range of finger flexion and extension tolerated in Figure 4.15. Thus the only people for which the attractor ring might need to be resized or reshaped would be very small children.

The feature weighting functions are more sensitive to hand size variation than the attractor ring itself. For example, the palm heel separation factor cuts off sharply at 4 cm and might become erroneously small for smaller hands whose palm heels are as close as 3 cm to one another. Luckily, palm heel separation would actually be much easier to measure and adapt to for individual operators than finger lengths since the palm heels are immobile relative to one another. The system could power up assuming a small palm heel separation to accomodate operators with small hands.

Then, the actual palm heel separation could be measured upon the first confirmed touchdown of both palm heels, and this measured palm heel separation could then be used to calibrate the inflection points of the palm heel separation function of Figure 4.20.

Fatter fingers cause larger total proximities for all contacts and thus cause the proximity inflection points of the thumb and palm heel size factors to become misaligned. Modifications to the thickness of the surface dielectric can also scale all proximities and thus disrupt the calibration of the proximity inflection points. Such variations in proximity scaling can potentially be dealt with by adapting the average fingertip proximity, $Z_{averageFingertip}$ in Equation 3.20, to peak fingertip proximities when the fingertips are normal to the surface.

4.4.7 Thumb Verification

The identifications produced by this attractor assignment method are highly reliable when all five fingers are touching the surface or when thumb and palm features are unambiguous. Checking that the horizontal coordinates for identified fingertip contacts are in increasing order easily verifies that fingertip identities are not erroneously swapped. However, when only two to four fingers are touching, yet no finger strongly exhibits thumb size or orientation features, the assignment of the innermost finger contact may wrongly indicate whether this contact is the thumb because distance-squared assignment is sometimes too lenient about thumb-fingertip angles and separations. In this case, the MTS employs a thumb verification process to take further measurements between the innermost finger contact and the other fingers. If these further measurements strongly suggest the innermost finger contact identity is wrong, the thumb verification process changes the assignment of the innermost finger contact. Once the finger assignments are verified, statistics about the assignments within each hand such as the number of touching fingertips

are compiled. These statistics provide convenient summaries of identification results for other modules.

Figure 4.21 shows the steps within the thumb verification process. The first is to compute several velocity, separation, and angle factors for the innermost contact identified as a finger relative to the other contacts identified as fingers. Since these inter-path measurements presuppose a contact identity ordering, they could not have easily been included as attractor distance weightings because contact identities are not known until the attractor distance minimization is complete. For the descriptions below, let FI be the innermost contact tentatively identified as a finger, FN be the next innermost finger contact, and FO be the outermost finger contact.

4.4.7.1 Inner Finger Separation Factor

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

$$inner\_separation\_fact = \min\left(1, \frac{\sqrt{(FI_x - FN_x)^2 + (FI_y - FN_y)^2}}{avg\_separation}\right) \qquad (4.45)$$

The factor is clipped to be greater than one since an innermost separation less than the average can occur regardless of whether thumb or index finger is the innermost finger touching the surface. In case there are only two finger contacts, a default average separation of 2-3 cm is used. This factor tends to become larger than one if the innermost contact is actually the thumb but remains near one if the innermost contact is a fingertip.

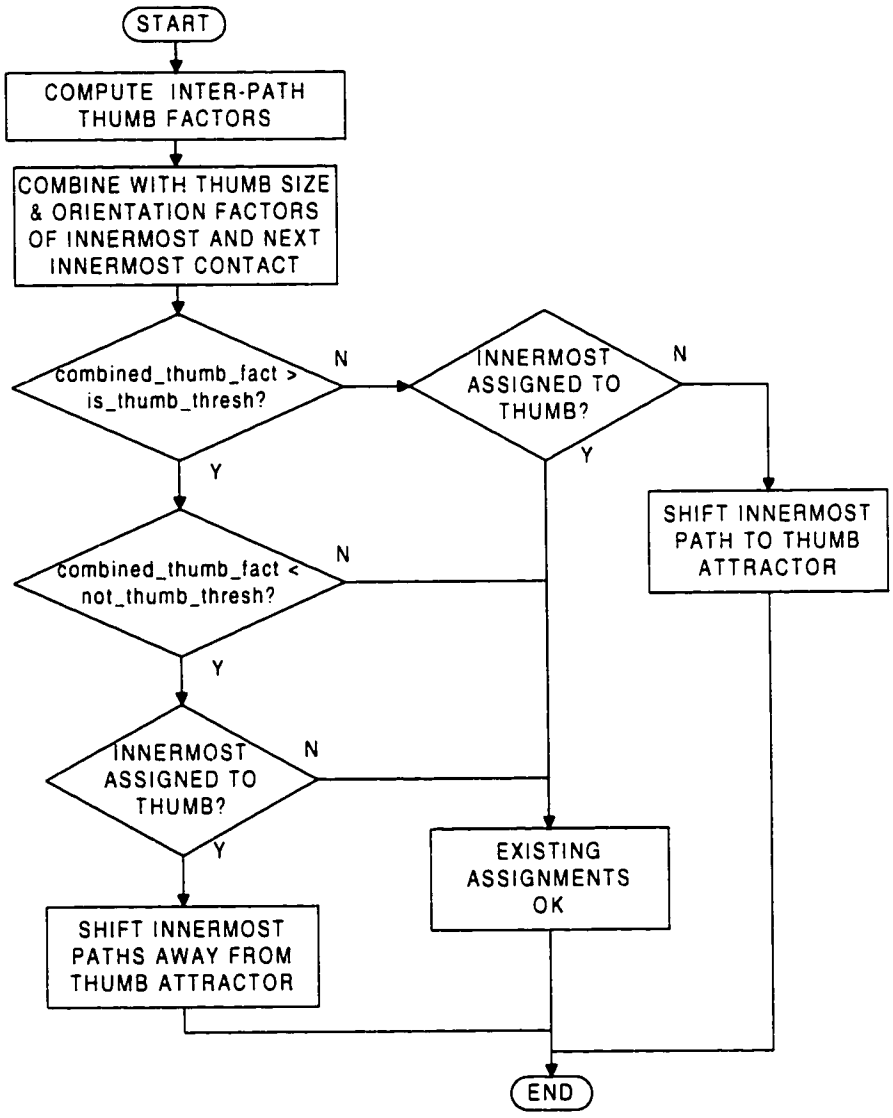


Figure 4.21: Flow chart of the thumb presence verification algorithm.

4.4.7.2 Inner Finger Angle Factor

Since the thumb rarely moves further forward than the fingertips except when the fingers are curled into a fist, the angle between the innermost and next innermost finger contacts can help indicate whether the innermost finger contact is the thumb. For the right hand the angle of the vector from the thumb to the index finger is most often 60°, though it ranges to 0° as the thumb moves forward and to 120° as the thumb adducts under the palm. This is reflected in the approximate plot of the inner angle factor in Figure 4.22, which peaks at 60° and approaches 0 toward 0°

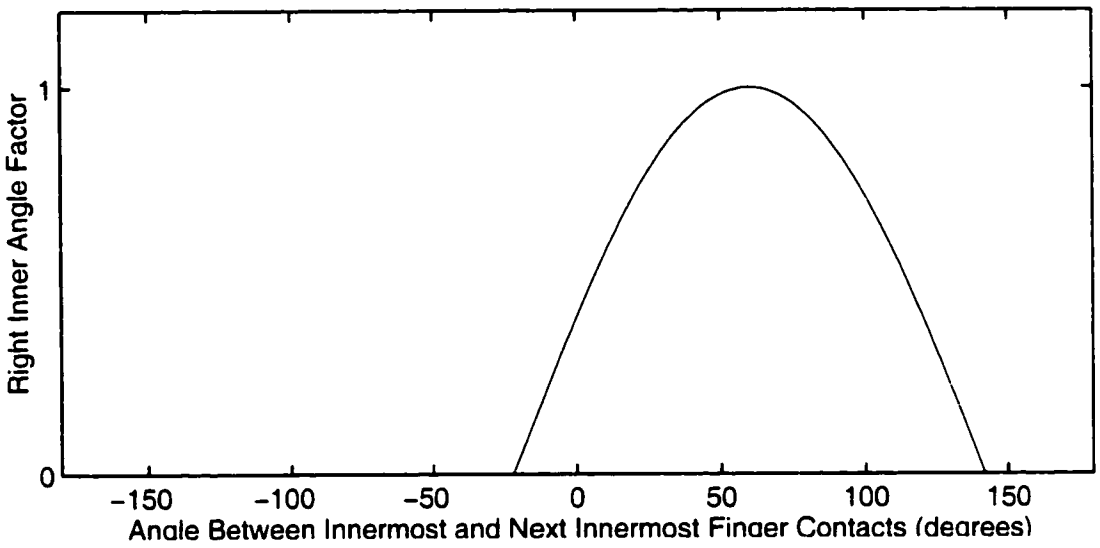


Figure 4.22: Right inner angle factor, *angle_fact*, versus the vector angle between the two innermost contacts identified as fingers.

and 120°. If the innermost finger contact is actually from the index fingertip, the measured angle between innermost and next innermost contact would probably be between 30° and minus 60°, producing a very small angle factor.

The inner separation and angle factors are highly discriminating of neutral thumb postures, but users often exceed the above cited separation and angle ranges when performing hand scaling or rotation gestures. For instance, during an anti-pinch gesture, the thumb may start pinched against the index or middle fingertip.

but then the thumb and fingertip slide away from one another. This causes the inner separation factor to be relatively small at the start of the gesture. Similarly, the thumb-index angle can also exceed the range expected by the inner angle factor at the beginning or end of hand rotation gestures, wherein the fingers rotate as if turning a screw. To compensate, the inner separation and angle factors are fuzzy OR'ed with expansion and rotation factors which are selective for symmetric finger scalings or rotations centered on a point between the thumb and fingertips.

4.4.7.3 Thumb-Fingertip Expansion Factor

When defined by the following approximate equation, the expansion factor peaks as the innermost and outermost finger contacts slide at approximately the same speed and in opposite directions, parallel to the vector between them:

$$\begin{aligned} expansion\_factor[n] \;\approx\; & -\sqrt{FI_{speed}[n] \times FO_{speed}[n]} \\ & \times \cos\left(FI_{dir}[n] - \angle(FI[n], FO[n])\right) \\ & \times \cos\left(FO_{dir}[n] - \angle(FI[n], FO[n])\right) \qquad (4.46) \\ clipped\_expansion\_fact[n] \;=\; & \max\left(0, expansion\_factor[n]\right) \qquad (4.47) \end{aligned}$$

where $\angle(FI[n], FO[n])$ is the angle between the fingers:

$$\angle(FI[n], FO[n]) = \arctan\left(\frac{FI_y[n] - FO_y[n]}{FI_x[n] - FO_x[n]}\right) \qquad (4.48)$$

Translational motions of both fingers in the same direction produce negative factor values which are clipped to zero by the max operation. Computing the geometric rather than arithmetic mean of the innermost and outermost speeds aids selectivity by producing a large expansion factor only when speeds of both contacts are high.

4.4.7.4 Thumb-Fingertip Rotation Factor

The rotation factor must also be very selective. If the rotation factor was simply proportional to changes in the angle between innermost and outermost finger,

it would erroneously grow in response to asymmetries in finger motion such as when the innermost finger starts translating downward while the outermost contact is stationary. To be more selective, the rotation factor must favor symmetric rotation about an imaginary pivot between the thumb and fingertips. The approximate rotation factor equation below peaks as the innermost and outermost finger move in opposite directions, but in this case the contacts should move perpendicularly to the vector between them:

$$\begin{aligned} rotation\_factor[n] \approx & -\sqrt{FI_{speed}[n] \times FO_{speed}[n]} \\ & \times \sin\left(FI_{dir}[n] - \angle(FI[n], FO[n])\right) \\ & \times \sin\left(FO_{dir}[n] - \angle(FI[n], FO[n])\right) \end{aligned} \quad (4.49)$$

$$clipped\_rotation\_fact[n] = \max(0, rotation\_factor[n]) \quad (4.50)$$

Since motions which maximize this rotation factor are easy to perform between the opposable thumb and another finger but difficult to perform between two fingertips, the rotation factor is a robust indicator of thumb presence.

4.4.7.5 Combining and Testing the Thumb Factors

The following expression essentially ORs these inter-contact factors with the innermost and next innermost contacts' thumb features:

$$\begin{aligned} combined\_thumb\_factor[n] \approx & \; clipped\_expansion\_fact[n] \\ & + clipped\_rotation\_fact[n] \\ & + inner\_separation\_factor[n] \times angle\_factor[n] \\ & \times (FI_{worient}/FN_{worient}) \\ & \times (FI_{wthumb\_size}/FN_{wthumb\_size}) \end{aligned} \quad (4.51)$$

The feature weighting ratios of this expression attempt to compare the features of the innermost contact to current features of the next innermost contact, which is

already known to be a fingertip. If the innermost contact is also a fingertip its features should be similar to the next innermost, causing the ratios to remain near one. However, thumb-like features on the innermost contact will cause the ratios to be large.

The action taken by the thumb verification module (Figure 4.21) depends on tests of *combined_thumb_factor*[n] against two thresholds, producing three cases:

1. If *combined_thumb_factor*[n] exceeds the high threshold, the innermost contact is definitely a thumb. If the assignment algorithm has not already put the innermost contact with the thumb attractor, the assignment algorithm must be overridden. Thumb verification shifts the innermost contact's assignment inward on the attractor ring to the thumb attractor.

2. If *combined_thumb_factor*[n] is between the low and high thresholds, the thumb verification test is ambiguous. The identification of the innermost contact made by the assignment algorithm is left unchanged since the assignment algorithm takes into account hand position estimate clues, but thumb verification does not.

3. If *combined_thumb_factor*[n] is less than the low threshold, the innermost contact is definitely *not* the thumb. For this conclusion to be reached, the expansion and rotation velocity factors must essentially be zero, the innermost and next innermost contact sizes and orientations must match, and either the inner angle must be near horizontal or the inner separation must be less than 2.5 cm. If the assignment algorithm has put the innermost contact with the thumb, thumb verification overrides it by shifting the innermost contact outward to the index finger attractor. Non-innermost contacts may need to be shifted outward as well to make the index finger attractor available.

Like the Voronoi cells of the assignment algorithm, the thumb verification expression (Equation 4.51) and thresholds establish clear cutoffs for thumb identity. Instead of utilizing hand position estimates, thumb verification imposes stricter tests on inter-contact velocity, angles, and separations. Since Equation 4.51 adds or essentially ORs the various feature measurements, thumb verification is most difficult when only one of the features is discriminating. Figure 4.23 plots the inner separation and angle cutoffs when the velocity factors are zero and the size and orientation ratios are one, providing no discriminating information. Likewise, for the expansion or rotation factors acting alone to surpass the high threshold and trigger identification of the innermost contact as the thumb, thumb-finger motions must be properly symmetric and exceed speeds of 1.5 cm/sec. For size ratios alone to trigger identification of the innermost as the thumb, the innermost contact must be at least twice as large as the next innermost. When several features act in combination, they need not be as strong as cited here to force identification of the innermost as the thumb.

4.4.8 Ratcheting Identification Accuracy

The quality of the constraints available for finger identification fluctuates as palm pressures change, fingers lift off, or more hand parts touch down on the surface. Running the assignment algorithm from scratch after segmenting each image could discard accurate assignments made when the hands started in a neutral posture or when more hand parts were touching the surface. Therefore the assignment and thumb verification algorithms for a hand are only executed for images in which the total hand proximity is increasing or when a touchdown has recently been attributed to the hand. Reassignment is *not* triggered by finger liftoff. This prevents degradation of identifications upon finger liftoff such as the dummy contact propagation upon pinky liftoff in Figure sequence 4.11–4.13. This also prevents erroneous