M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 487
Work originally completed April 30, 1998

Recognizing Movement using Motion Histograms

James W. Davis
MIT Media Laboratory, 20 Ames Steet
Cambridge, MA 02139
jdavis@media.mit.edu

Abstract

In this paper, we present a real-time computer vision approach to recognizing human movements based on patterns of motion. The underlying representation is a Motion History Image (MHI) which is characterized by multiple histograms of the local motion orientations. The approach is adapted to accommodate movements of different durations by using a simple iterative technique. Quantitative results are given showing discrimination between different human movements using the approach. An extension addressing occlusion and distractor motion is also presented within this framework.

1 Introduction

The recognition of human motion and action using computer vision has widespread interest ranging from surveillance applications to entertainment systems. Being able to recognize the presence of human motion is desirable because every little change or movement in the environment may not be consequential. Monitoring applications, for instance, may wish to signal only when a person is seen in a particular area (perhaps inside a dangerous or secure area). Thus only those motions belonging to human activity are of importance. In the entertainment domain, an increasing interest in "people watching" systems is growing. Here the systems watch for the gestures made by participants which control/drive the program or interaction [7, 3]. Thus one requirement (or demand) of such machine vision systems is their ability to perform in real-time. It would not be of much use for the monitoring system to report that a person entered into a dangerous area an hour after the fact. Also, systems incorporating human gestures for input must recognize and respond quickly to the user without noticeable lag to give a sense of immersion and actual control. The quickness of response is paramount.

In this paper, we present a real-time computer vision approach to recognizing human movements. In earlier work [2], we described a representation of movement known as a Motion History Image (MHI). The MHI is a compact representation of temporal movement and is simple to compute. In this paper, we present a new method for recognizing movement which relies on localized regions of motion derived from the MHI. By gathering and matching multiple, overlapping histograms of the motion orientations from the MHI, we offer a real-time solution to recognizing various human movements.

The remainder of this paper first examines the related research for which this work has been based (Section 2). Next we present the approach of using motion histograms (Section 3). This section is sub-divided into discussions on the underlying representation (Section 3.1), the calculation of the motion orientations (Section 3.2), and the histogram generation (Section 3.3). We then present a simple recognition method (Section 4) along with some quantitative results (Section 4.2). A method for handling variable-length movements is also described (Section 4.3). We then address the notions of occlusion and distractor motions within an extension of this framework (Section 5). Lastly, we conclude with a brief summary of the approach (Section 6).

2 Related work

In previous work [2], we presented a real-time approach for representing and recognizing simple human movements. The motivation for the approach was based on how easily people can recognize common human movements (like sitting) from low-resolution imagery where the image features are basically not perceivable, thus showing the importance of the motion information. The approach relies on "patterns of motion" rather than structural features as the representation for identifying various human movements. In that method, there is a collapsing of the space-time volume into a single 2-D template form, where the representation still perceptually captures the essence of the movement and its temporal structure. This template is referred to as a Motion History Image (MHI). The MHI can be compared to many famous stroboscopic images [5, 1] and comic strip panels showing character motion, where time is

collapsed and represented in a single static frame. For recognition, seven higher-order moments are used as global shape descriptors and energy localizers for the motion template. These descriptors are then statistically matched to stored examples of different movements. Though this method has shown promising results, the main problem is with the recognition approach, where the discrimination between motion templates is based upon *global* properties and therefore is susceptible to region-based errors such as the addition or removal of motion. Another limitation is that the recognition was token (label) based, where it cannot yield much information other than recognition matches (i.e. it cannot report that a lot of "up" motion is happening in a particular area). We therefore wish to develop better methods of representation and recognition which can account for various motion *regions* around the body by retaining and analyzing a more localized form of the motion. In this paper, we develop such a method using multiple, overlapping histograms of the MHI motion orientations.

The most closely related work to our motion histogram approach is that of Freeman and Roth's work on recognizing hand gestures from orientation histograms [6]. In their approach, they use a single histogram of edge orientations of the user's hand to recognize various static gestures. Though they address dynamic gesture, the problem of finding the start and stop times of a gesture was not considered. Thus all input sequences to their system were fixed-length, and the resultant representation was basically a concatenation of individual orientation histograms from each image in the sequence. Their method is simple, fast, and robust against certain illumination changes. Our method in this paper recasts their orientation histogram approach to become "motion orientation" histograms, where the directions of motion are accumulated in a histogram format and used for recognition. By using the motion template representation from [2] to generate the motion information and using a simple iterative matching technique, we can account for various length movements while still retaining real-time performance.

3 Motion histograms

The method for generating the motion histograms is developed by extracting directional motion information from the movement sequence's MHI representation. We then cluster this motion information into overlapping histogram regions to more locally represent the movement.

3.1 Motion history images

We use the MHI representation described in [2] as the basis for the motion histograms. Currently, we generate the motion between frames by differencing successive binary silhouette images of the person. The reason for this is two-fold. First, we believe that strict optical flow methods are still too brittle for real imagery of people moving (due to noise, shadows, textures, and rate of movement) and generally computationally taxing (i.e. not real-time)[1]. Image differencing continues to be a fairly robust method for cheaply locating the *presence* of motion. One of the main problems with image differencing though (as opposed to optical flow) is that one cannot tell the magnitude or direction of the motion, only its presence. Thus it is hard to remove spurious unwanted motion purely from the differencing result. But as we will show, the *accumulation* of image differences can yield directional motion information. The second reason for differencing silhouettes corresponds to the fact that much of the clothing texture frequently signals unwanted motion, which can cause problems when using motion for recognition. For this reason, we chose to extract the silhouette form of the person (thus masking the clothing texture). A side effect of using silhouettes is that no motion inside of the silhouette can be seen. For example, a camera facing the person would not show the hands moving "in front of" the body in the silhouette. One possibility to help overcome this problem is to use multiple cameras (the approach here easily extends to multiple views). Therefore, image differences (we used the union of differences at both normal and low resolutions) show only boundary motion of the silhouettes, but still yield quite useful motion information for many movements. To acquire the full-body silhouette of the person, we developed a robust and precise real-time silhouette extraction process based on spectral selectivity [4].

To generate the MHI for the movement, we weight and layer the successive silhouette image differences. In the MHI, each pixel value is a function of the temporal history of motion at that point from all the frames in the movement sequence. We currently use a simple replacement and decay operator based on time-stamping (the previous method was based on frames rather than time):

$$MHI(x,y) = \begin{cases} \tau & \text{if current motion at (x,y)} \\ 0 & \text{else if } MHI(x,y) < (\tau - \delta) \end{cases}$$

[1] The overall approach outlined in the paper though is general enough to also be used on optical flow data if desired.

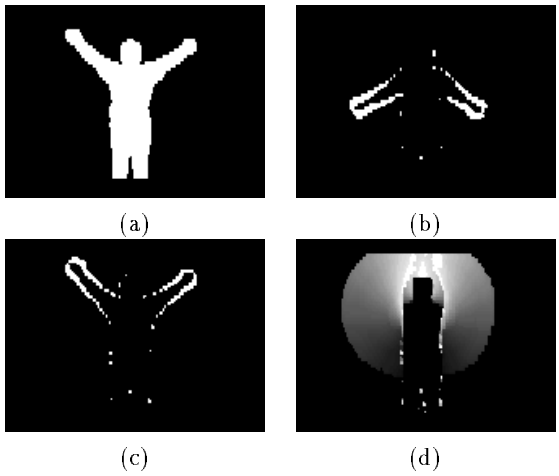

Figure 1: Generation of MHI for raising arms movement. (a) Sample silhouette of the person with their arms raised near the end of the movement. (b) A difference of silhouettes early in the sequence. (c) A difference of silhouettes later in the sequence. (d) Resulting MHI of layered silhouette differences (normalized for display).

where τ is the current time-stamp, and δ is the maximum time duration constant. The time-stamps allow for an easier port of the system between various faster and slower platforms (time is constant where frame rate is not). The above function is called to update the MHI each time a new image difference result is calculated. By linearly normalizing the MHI time-stamps to values between 0 and 255, we can see that the more recently moving pixels are brighter than pixels belonging to older motion. The result of the above process is shown in Figure 1 for the movement of raising both of the arms.

3.2 Gradient of motion

The MHI layers the silhouette differences in such a way that motion from the silhouette boundary can be perceived in the gradient of the MHI. This is very similar to the concept of normal flow. Notice that as the arms are raised up in Figure 1(d), that the intensity fading (from dark to light) gives the impression of motion in the direction of the arm movement. It can be said that the MHI "visually encodes" some motion information from the silhouette boundary. We see the *direction* of movement clearly, but the magnitude is not as accessible. Our goal is to use this directional motion information for recognition.

The local gradient orientations of the MHI directly show the direction of the silhouette boundary movement. Therefore, we can convolve classic gradient masks with the MHI to extract the directional motion information. For this work, we union the convolution at two resolutions (the original and a lower resolution) to handle more widespread gradients (due to differing speeds of movement). Sobel gradient masks were used for the convolution:

$$F_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, F_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

With the gradient vectors calculated, it is a simple matter to get the gradient orientation for a pixel by:

$$\theta = \arctan \frac{F_y}{F_x}.$$

We must be careful when calculating the gradient information because it is only valid at particular locations within the MHI. The boundaries of the MHI should not be used because non-moving (zero valued) pixels would be included in the gradient calculation, thus corrupting the result. Only MHI *interior* motion pixels should be examined. Additionally, we must not use gradients of MHI pixels which have a too low or too high contrast in their local neighborhood. A small contrast does not give a reliable measure of the gradient direction, and a large contrast signifies a large temporal disparity between pixels, which makes the directional information biased and un-usable. The results of the motion orientation using gradient masks is shown in Figure 2.

3.3 Histogram hierarchy

Previously in [2], we performed recognition on the MHI using a set of global moment-based features. Though these moments are excellent shape descriptors and energy localizers, they are still global computations and do not describe the motion characteristics in different regions around the body (e.g. "a lot of upward motion in the left-side of the body"). Here, we present a more characterizing and local approach for representing the motion information.

A simple means of localizing the motion for recognition is to separately pay attention to different regions around the body. One way of doing this is to divide the MHI into various regions (or windows) and then characterize each region. A method of characterization is to use a histogram of the motion orientations for a region. Thus we can divide the MHI motion pattern into regions each being represented by a histogram of local motion orientation. We define the center point of the window configuration based upon the centroid of the current silhouette of the person, and also define the boundaries (or extent) of the regions from a bounding box over the MHI motion pixels. Thus the windowing can adapt to the location of the person and the size of

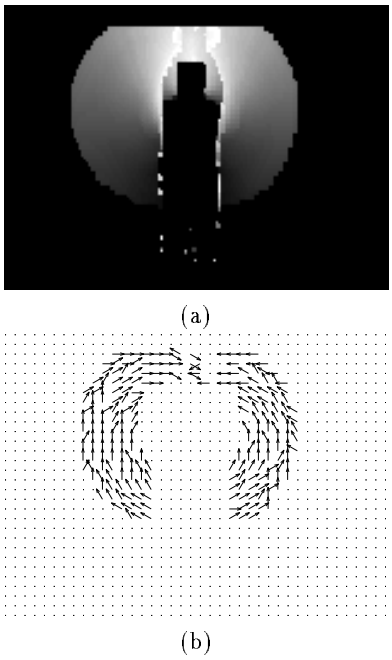

Figure 2: Directions of motion from MHI gradients. (a) MHI for raising arms movement. (b) Result of convolving gradient masks with the MHI. The gradient directions show the approximate motion of the arms.

their recent movement. The window placement yields X-Y translation invariance and the boundary helps in achieving scale invariance during recognition.

One possible set of histogram windows is shown in Figure 3. This set of nine overlapping regions basically divides the body into windows covering the whole body, left-side, right-side, upper, lower, and four surrounding quadrants. So instead of having just one large window, we have a hierarchy of additional support windows to help characterize the motion.

To generate the histograms for these window regions, we first quantize the gradient directions from the MHI into multiples of 30 degrees, resulting in histograms with 12 bins each (mainly for speed during recognition). Since the number of bins is a relatively small, we chose not smooth the histogram as done in [6]. To handle changes in scale between different sized people (or in location of depth), we need to normalize these histograms with respect to some measure of the person or motion. One possibility is to normalize each histogram by the number of entries in that histogram (e.g. normalizing for a probability distribution). This approach is highly subjective to problems arising from only small motions present in a window. A better method is to normalize each window by the sum of *all* the motion orientation

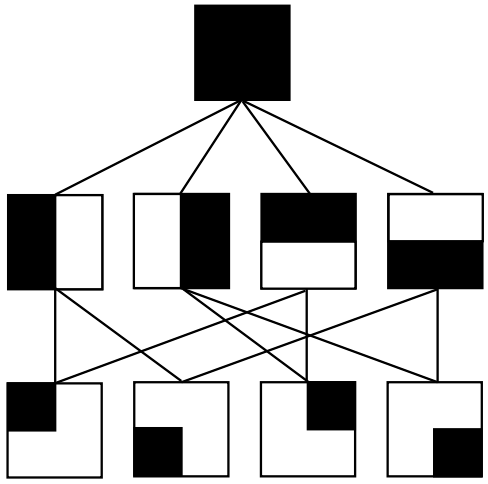

Figure 3: Overlapping windows for generating motion histograms. The dark areas represent areas which are included in that window's histogram; white areas are ignored. The first window (top window) covers the entire motion region within the MHI. The windows below cover progressively smaller regions of the motion.

pixels found in the gradient map (also the number of entries in the overall window histogram #1). We note that the total number of motion pixels should be greater than some minimum threshold to be effective. The result of this normalization is that the histograms are no longer relatively sensitive to small motions.

Figure 4 shows the nine overlapping, normalized histograms for the *left-arm-up* movement. Notice the large response for window #3 which registers the left motion. The concentration of entries around bins 7-10 (180 - 90 degrees, respectively) shows that there is much upward and sideways motion on the left side of the body. The histogram for the opposite side of the body (window #2) hardly shows any motion. Having a collection of histograms also lets us employ, if desired, the motion orientations directly to get a finer sense of how motion occurred within the different regions around the person (rather than just the presence of motion, or a labeled movement). For example, the localized directions of motion may be useful for interaction or control mapping. We now present a simple recognition method which uses these motion histograms to recognize various body movements.

4 Recognition

The result of generating motion histograms for the body movement is a collection of nine, 12-bucket histograms. There are many possible ways of using this data for recognition. The simplest approach, and the one taken

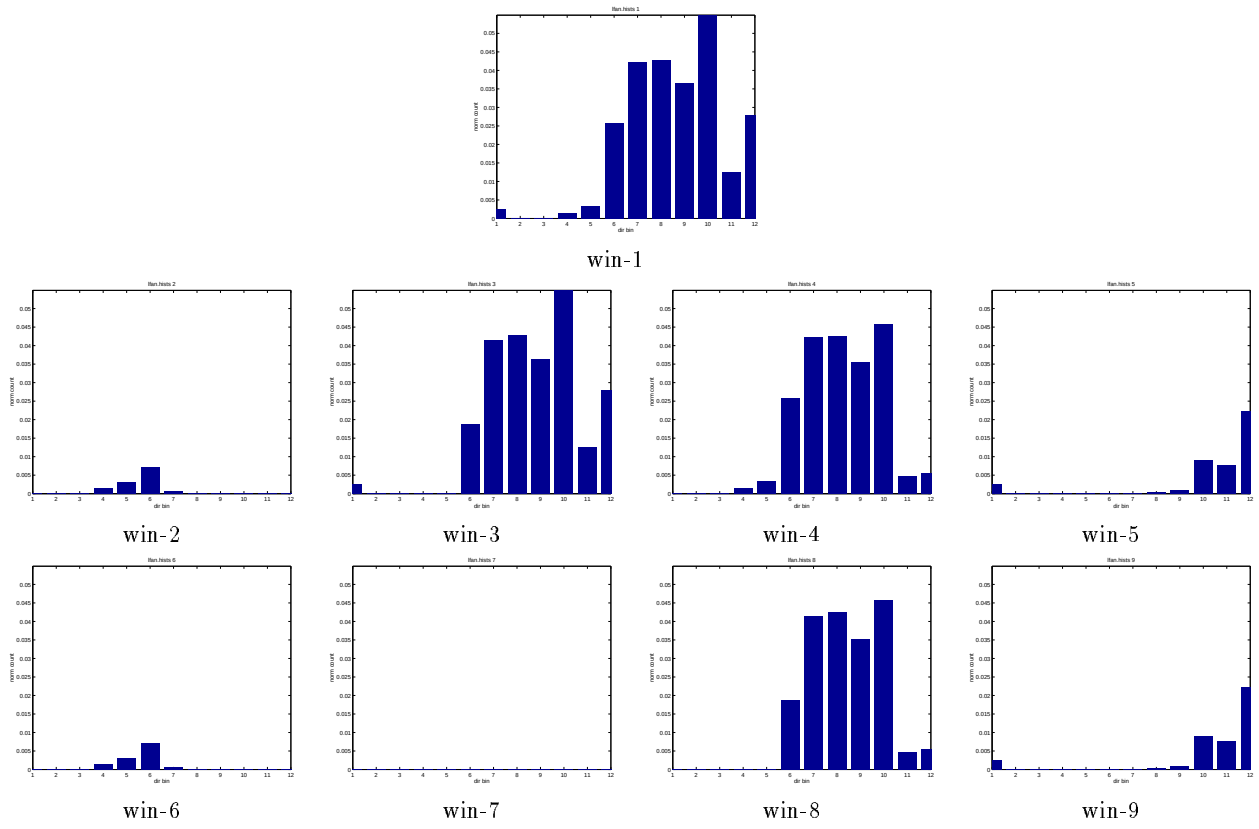

Figure 4: Motion histograms for raising the left arm movement. Most of the motion is localized to the left (win-3) and top side (win-4, win-8) of the body. The clustering of directions around bins 9 and 10 (approx. 90 degrees) shows the upward motion, and the clustering around bins 7 and 8 (approx. 180 degrees) shows the left side motion.

here, is to concatenate the histograms into a single column vector (108 x 1) and use the Euclidean distance between an input and stored model vector as a measure of closeness for recognition. An effect of the histogram normalization method (based on the total amount of motion) is that histograms with larger amounts of motion end up being more heavily weighted in the Euclidean distance than histograms with smaller amounts of motion. Intuitively, we believe that this is a desirable effect (and not problematic) because a tiny motion region should not carry as much weight as a larger more encompassing (or expressing) area.

4.1 Movement model

To generate a model for a particular movement, we gather multiple examples of a person (or people) performing the movement. The motion histograms for each move are generated and stored (in vector form) when each example is completed (by manual selection during training). For a simple model of this movement, a set of mean motion histograms is formed by averaging together the histograms (in vector form) of the training data. The training examples are then used to find the mean and variance of the Euclidean distance from the training vectors to the newly generated mean vector. By collecting a mean and variance measure of the Euclidean distance using multiple training examples, we can select a recognition threshold based on the variability of the data measures from training. We could have examined variances *within* each of the histograms and used these measurements as weighting factors in a new distance metric (e.g. *weighted* Euclidean distance), but for simplicity we chose to measure only the change in the *global* Euclidean distance. The vector mean, distance mean, and distance variance of the training motion histograms are stored as the model for that particular movement. This process is repeated for each of the different movements that are required to model.

4.2 Matching

As for matching new input, we simply calculate the Euclidean distance between the input motion histogram

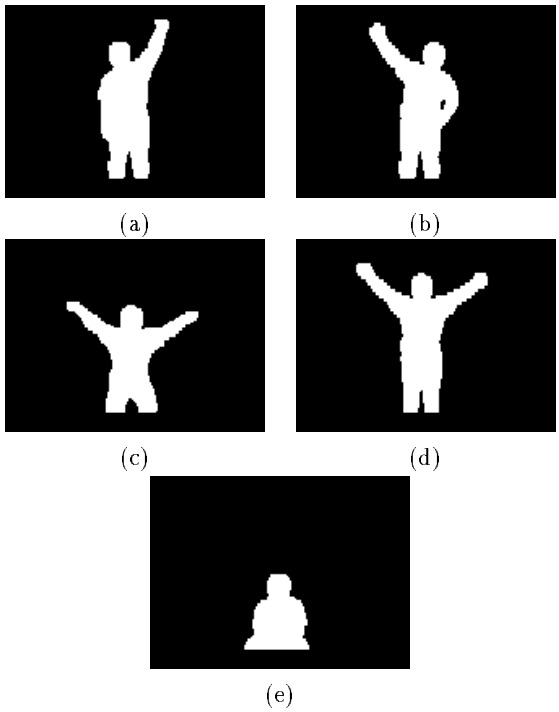

Figure 5: Silhouette key-frames for the movements to recognize. (a) left-arm-fan-up. (b) right-arm-fan-up. (c) squat-with-two-arm-fan-up. (d) two-arm-fan-up. (e) crouch-down.

vector and the model mean vector. Using the model's distance mean and variance, we then calculate the Mahalanobis distance [8] for the new vector's Euclidean measure. This gives a measure of how many standard deviations the input's Euclidean distance is away from the model statistics. We can threshold this value based on statistics to declare if a match was found. This process can be easily repeated to seek a match against all the stored movement models without much computational expense. Table 1 shows the Mahalanobis distances for a set of new input examples (which were not used in training) matched against the stored models (silhouette key-frames for the different movements are shown in Figure 5). The training process used in generating Table 1 only employed ten training examples for each model, which is generally too small of a training set to gather good variances, but sufficient enough to show the discrimination power of the approach. The table shows the correct classification with the Mahalanobis distances being considerably smaller for the correct target as opposed to the other movements. We see that this recognition method clearly discriminates between this set of moves using only the location and direction of motion from the histograms.

	lfan	*rfan*	*angel*	*fan*	*crouch*
T_1	**5.99**	179.27	119.63	190.83	289.94
T_2	58.98	**2.93**	105.73	137.35	230.33
T_3	98.18	82.95	**8.80**	97.45	217.49
T_4	59.89	42.11	54.14	**10.24**	265.74
T_5	176.77	135.29	138.28	396.34	**3.83**

Table 1: Matching results. Entry [i,j] reports the Mahalanobis distance between test input i (T_i) and model j. The models in this example are T_1 = left-arm-fan-up, T_2 = right-arm-fan-up, T_3 = squat-with-two-arm-fan-up, T_4 = two-arm-fan-up, and T_5 = crouch-down. The bold entries highlight the minimum distances for each input.

4.3 Variable length movements

Since a set of movements to recognize most likely contains gestures of different time lengths (durations), we need a recognition mechanism that can handle variable length movements. The main problem to overcome is that we basically need to generate an MHI for each model movement to recognize, or more precisely, for each model movement that has a different time duration.

If we know a priori the minimum and maximum durations of all the movements to recognize, then *all* recognizable movements have a duration *within* that time window. With the simple replacement and decay operation used in generating the MHI (which generates an MHI for a specific time duration), we have an inexpensive iterative means of achieving multiple simulated MHIs from only a single MHI. We begin by always generating an MHI with the time duration constant δ being the maximum duration found from all the movements to recognize. Thus the MHI is generated for the movement(s) which have the longest time duration. During the recognition process, we can iteratively lower the time duration constant (from the maximum to the minimum) for the MHI which thus thresholds and removes older motions; we then look for a match with the new simulated MHI (of a smaller time duration) and its updated motion histograms (updated by removing those values deleted from the MHI); the process is repeated until the time duration reaches the minimum value for the recognizable movements. This method progressively removes older motion from the MHI (and histograms) in such a way that all possible MHIs (and histograms) that could have been generated *within* the movement duration window (maximum and minimum movement times) are in fact quickly created for examination of a match during the iteration phase. The newly created MHIs are not approximations to the true MHIs, but

are the genuine re-creations. This approach can also be coded in such a way that information is only *updated* (removed) rather than being totally recalculated (e.g. it is too time-consuming to recalculate *all* the motion gradients during the iteration process). This makes for a faster algorithm.

4.4 Computational specifics

The entire matching process results in real-time performance of speeds greater than 20 Hz on-average for an SGI R10000 O2 platform with no special computer hardware. Algorithm specifics for the above speed include:

- Image resolution of 160 x 120.
- Fast silhouette extraction [4].
- Five model movements to match against. (Having more models will not significantly effect the speed.)
- A minimum/maximum time duration window of [1.0, 2.5] seconds for iterative matching ($\delta = 2.5$).
- Iterative step size removing time-stamped motion in 0.067 second (15 Hz) intervals. (Minimum step size is bounded by the digitizing rate, typically 30 Hz.)

5 Extensions for occlusions and distractor motions

With the above methodology for localizing and recognizing motion information, we also have an opportunity to address occlusions and distractor motions. Though the current matching technique is simple and does not directly model or explain any form of occlusions or large amount of noise motion, the windowing process does offer a framework for handling such problems. In both occlusion and distractor motion cases, we want to remove those troublesome regions from the matching function. We will mainly discuss occlusions here, but it is clearly extendible to distractor motions as well.

For simplicity, let's consider an example occlusion contained and bound by one of the four motion histogram quadrants (the four quadrants are shown in the bottom row of Figure 3). This occlusion propagates back through the hierarchy and results in four of the total nine motion histograms containing this occlusion region, and therefore almost half of the motion histograms are corrupted from the occlusion. With the current matching process, this would certainly cause problems, being that each motion histogram contributes to the overall match (though smaller windows can give less weight, due to the current normalization method).

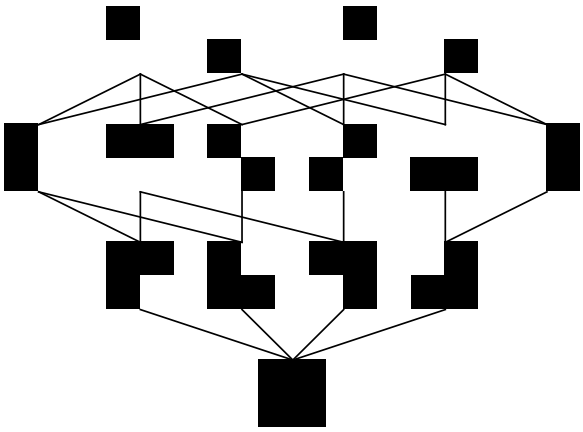

Figure 6: New motion window regions generated by combining the four primary windows. The dark areas represent areas which are included in that window's histogram; white areas are ignored.

What is more desired is to be able to remove this occlusion region from its membership in the window hierarchy, and to match based on some function of the remaining regions. Thus the windowing hierarchy as previously shown may not necessarily be the most appropriate for being able to resist or discount the occluded data.

By considering a new bottom-up hierarchy from the four quadrants, we can derive a different set of windows where the possible regions of occlusion can be explicitly modeled and discounted. Given four "primary" windows (the four quadrants), we can generate combinatorially fifteen combination windows

$$\binom{4}{1} + \binom{4}{2} + \binom{4}{3} + \binom{4}{4} = 2^4 - 1 = 15$$

which are shown in Figure 6. If smaller primary windows were used, many more window combinations would be generated. This set has no constraints on continuity of the original primary windows. If we did impose a continuity, say 4-way connectedness, then there would only be thirteen windows (fifteen minus the two diagonal only pairs). Other constraints could also be imposed. For example, in addition to the primaries, requiring a combination of only even numbers of primary windows along with 4-way connectedness generates the original set of windows shown in Figure 3. Given this new bottom-up window set, we now have additional region information (e.g. 15 windows instead of only 9, including cross-connected regions) and explicit occlusion regions (or regions to discount) modeled into the window set.

Having a multiplicity of motion windows does not directly solve the occlusion problem, though. A ques-

tion of *how* to discount the regions must be addressed. Consider a movement where one primary window is occluded. Then seven out of fifteen windows will be corrupted by propagation of that region through the hierarchy, but the remaining eight windows still encode a large variety of valid locations. We need to retain these valid regions and remove the occluded regions. One possibility is to collect model statistics for each window histogram from training data and use this information to compute *plausibility* for the new input on a window-by-window basis. A question of normalization follows. Recall that normalization of the histograms is needed with the current method to achieve scale invariance during matching. Normalization also allows a simple Euclidean measure to be used for matching. Previously, all histograms in the hierarchy were normalized by the largest overall motion window (encompassing the entire motions of the body). This will no longer be applicable if we perform a verification on the windows in a window-by-window fashion, because the largest window will contain the occlusion if one exists, and thus flaw the overall normalization. Therefore, we need a new means of normalization for the motion histograms. We could easily revert to self-normalization by the amount of motion in each window, but as previously stated, this becomes problematic when the size of the window or size of the motion becomes small. A method more closely related to the previous overall normalization approach is to use the plausible input window with the most motion to normalize all the remaining plausible sub-windows. To "verify" the plausibility, we might be able to use the individual window counts to "roughly" match for plausibility, then later normalize based on the largest, plausible histogram. Other means of verification are possible, which may include looking at the maximum direction of motion, or spread of the motion information.

This process will work only if we retain the original model histograms (un-normalized) and re-normalize to them to match the test input normalization. The motion histograms for the input are individually verified against the target model. The largest, most encompassing input histogram which is verified is then used to re-normalize the remaining verified histograms (the model is also normalized using that corresponding histogram in its collection). Once the normalization process is complete, a finer recognition method using only the valid histograms can be used to determine a match (as in the Euclidean match described above). This is a simple computation at virtually no expense during recognition. The process can easily be re-applied to all target models for recognition against multiple movements.

When matching against several models, we must also consider the case where the input matches well on a small number of verified regions of move A and matches not quite as well on a larger set of verified regions of move B. Here there is a tradeoff between precision of a window match and the number of verified windows. There are several issues involved in making this decision. We could simply state that the model with the most "verified" windows is the best match, because to verify means to be acceptable given some training measurements. But on the other hand, a solid match on few windows may be more *descriptive* of the actual activity, given that the verification method will most likely be loose in its acceptance criteria. It is not clear which method of recognition is more desirable, but perhaps the goal and context of the situation may determine which method is preferable.

6 Summary

In this paper, we presented a real-time computer vision approach to recognizing human movements based on patterns of motion. The underlying representation is a Motion History Image (MHI) which is characterized by multiple, overlapping histograms of the motion orientations. These histograms separate and localize regions of motion for a better description of the movement. Quantitative results show that the method can easily discriminate between different human movements, and is extendible to variable length motions. Occlusion and distractor motions are also addressable within this framework.

References

[1] Braun, M. *Picturing time: The work of Etienne-Jules Marey (1830 - 1904).* University of Chicago Press, 1992.

[2] Davis, J. and A. Bobick. The representation and recognition of human movement using temporal templates. In *Proc. Comp. Vis. and Pattern Rec.*, pages 928–934, June 1997.

[3] Davis, J. and A. Bobick. Virtual PAT: a virtual personal aerobics trainer. MIT Media Lab Perceptual Computing Group Technical Report No. 436, MIT, 1997.

[4] Davis, J. and A. Bobick. SIDEshow: A silhouette-based interactive dual-screen environment. MIT Media Lab Perceptual Computing Group Technical Report No. 457, MIT, 1998.

[5] Edgerton, H. and J. Killian. *Moments of vision: the stroboscopic revolution in photography.* MIT Press, 1979.

[6] Freeman, W., and M. Roth. Orientation histogram for hand gesture recognition. In *Int'l Workshop on Automatic Face- and Gesture-Recognition*, 1995.

[7] Freeman, W., Tanaka, K., Ohta, J., and K. Kyuma. Computer vision for computer games. In *Int'l Workshop on Automatic Face- and Gesture-Recognition*, 1996.

[8] Therrien, C. *Decision Estimation and Classification.* John Wiley and Sons, Inc., 1989.