

Action Recognition using Motion Primitives and Probabilistic Edit Distance

P. Fihl, M.B. Holte, T.B. Moeslund, L. Reng

Laboratory of Computer Vision and Media Technology
Aalborg University, Denmark
Email: tbm@cvmt.dk

Abstract. In this paper we describe a recognition approach based on the notion of primitives. As opposed to recognizing actions based on temporal trajectories or temporal volumes, primitive-based recognition is based on representing a temporal sequence containing an action by only a few characteristic time instances. The human whereabouts at these instances are extracted by double difference images and represented by four features. In each frame the primitive, if any, that best explains the observed data is identified. This leads to a discrete recognition problem since a video sequence will be converted into a string containing a sequence of symbols, each representing a primitive. After pruning the string a probabilistic Edit Distance classifier is applied to identify which action best describes the pruned string. The approach is evaluated on five one-arm gestures and the recognition rate is 91.3%. This is concluded to be a promising result but also leaves room for further improvements.

1 Introduction

In the last decade the focus on automatic analysis of human motion has increased rapidly. This is evident by the number of workshops and special sessions at conferences and special journal issues devoted to this research field. Furthermore, the recent public interest in security issues has increased the interest from the funding agencies leading to even more research in this field. Whereas more and more robust solutions are seen within both tracking and pose estimation, the subfield of automatic recognition of actions and activities is still lacking. One reason being that this field is not only based on advances in signal processing but also in AI. A number of advanced approaches have, however, been reported. The current trend is not as much on first reconstructing the human and the pose of his/her limbs and *then* do the recognition on the joint angle data, but rather to do the recognition directly on the image data, e.g., silhouette data.

Yu *et al.* [19] extract silhouettes and unwrapped their contours. PCA is used to obtain a compact representation. A three-layer feed forward network is used to distinguish actions such as walking and running based on the trajectories in eigenspace. Yilmaz and Shah [17] use spatio-temporal volumes (STV) for action recognition. A person's 3D contour is projected into 2D over time and yields the STV. Differential geometric is used to extract features from the STV and action recognition is carried out as an object matching task by interpreting the STV as rigid 3D objects. Bobick and Davis [4] apply temporal templates based on motion energy images (MEI) and motion history images

(MHI). The MEI is a binary cumulative motion image. The MHI is an enhancement of the MEI where the pixel intensities are a function of the motion history at that pixel. Matching temporal templates is based on Hu moments. Related approaches are to use a 4D motion history volume based on the visual hull [16] or motion flow history [1].

Common for these approaches is that they represent an action by image data from all frames constituting the action, e.g., by a trajectory through some state-space or a spatio-temporal volume. This means that the methods in general require that the applied image information can be extracted reliably in every single frame. In some situations this will not be possible and therefore a different type of approach has been suggested. Here an action is divided into a number of smaller temporal sequences, for example movements [6], atomic movements [7], states [5], dynamic instants [13], exemplars [11], behaviour units [9], key-frames [8], and primitives [14]. The general idea is that approaches based on finding smaller units will be less sensitive compared to approaches based on an entire sequence of information.

For some approaches the union of the units represents the entire temporal sequence, whereas for other approaches the units represent only a subset of the original sequence. In Rao *et al.* [13] dynamic hand gestures are recognized by searching a trajectory in 3D space (x and y -position of the hand, and time) for certain dynamic instants. Gonzalez *et al.* [8] also look for key-frames for recognizing actions, like walking and running. Approaches where the entire trajectory (one action) is represented by a number of subsequences, are Barbic *et al.* [2] for full body motion, where probabilistic PCA is used for finding transitions between different behaviors, and Bettinger *et al.* [3] where likelihoods are used to separate a trajectory into sub-trajectories. These sub-trajectories are modeled by Gaussian distributions each corresponding to a temporal primitive.

In this paper we address action recognition using temporal instances (denoted primitives) that only represent a subset of the original sequence. That is, our aim is to recognize an action by recognizing only a few primitives as opposed to recognition based on the entire sequence (possibly divided into sub-trajectories). The actions that we focus on in this work are one-arm gestures, but the approach can with some modifications be generalized to body actions. Concretely we represent our primitives by four features extracted from a motion-image, yielding simple and yet powerful descriptors for our primitives. In each frame the primitive, if any, that best explains the observed data is identified. This leads to a discrete recognition problem since a video sequence will be converted into a string containing a sequence of symbols, each representing a primitive. After pruning the string a probabilistic Edit Distance classifier is applied to identify which action best describes the pruned string.

The paper is structured as follows. In section 2 we describe our features used to represent the primitives. In section 3 we recognize the actions by first recognizing the primitives and then the actions. In section 4 the approach is evaluated on a number of actions and in section 5 the approach is discussed.

2 Representation of Primitives

Our long term goal is for any given set of actions to be able to automatically find primitives that can be used to represent the actions independent of the viewing angle [14].

In this work, however, we work with arm gestures and assume the torso to be fronto-parallel.

For a given set of training sequences a set of primitives is defined. This set of primitives allows for a representative description of the different actions. Concretely the primitives are each a 3D body configuration and the nature of these primitives will depend on the actions to be recognized. The actual selection of the primitives is presented in section 4 after the test data has been introduced. This and the following section will therefore describe the general principles, whereas implementation details are left for section 4.

Instead of attempting to reconstruct the 2D/3D pose of the human and compare with a 2D/3D action database, we use local motion to describe the whereabouts of the person. This approach is motivated by the notion that image motion is often less sensitive compared to other cues, but yet a powerful cue for inferring information from a sequence of images [4]. The simplest type of local motion is a difference image. Even though this only provides crude information it has the benefit of being rather independent to illumination changes and clothing types and styles. Furthermore, no background model or person model is required. However, difference images suffer from "shadow effects" and we therefore apply double difference images, which are known to be more robust [18]. The idea is to use three successive images in order to create two difference images. These are thresholded and ANDed together. This ensures that only pixels that have changed in both difference images are included in the final output. In figure 1 the principle is illustrated. The effects of outliers and "holes" are addressed using morphology.

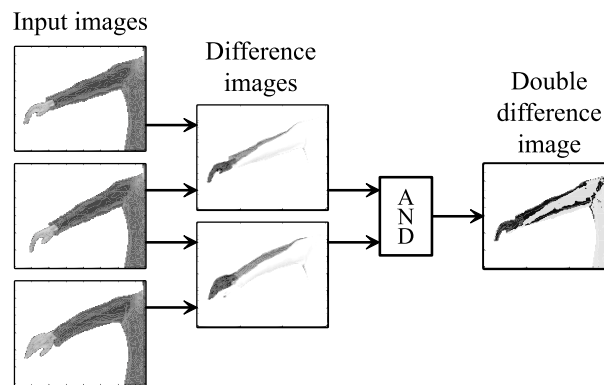


Fig. 1. Principle behind the double difference image. **Left:** Input images. **Middle:** Two (inverted) difference images. **Right:** Double difference image. The silhouette of the input (light gray) has been overlaid for clarification, hence the black pixels are the output.

When doing arm gestures the respond from the double difference image will roughly speaking be a "motion-cloud", which we model compactly by an ellipse. The length and

orientation of the axes of the ellipse are calculated from the Eigen-vectors and Eigen-values of the covariance matrix defined by the motion pixels.

We use four features to represent this cloud. In order to make the features independent of image size and the person's position in the image they are represented as ratios. Furthermore, they are defined with respect to a reference point currently defined as the center of gravity of the person (discussed further in section 5). The four features are illustrated in figure 2 and defined as:

1. The eccentricity of the motion cloud defined as the ration between the minor and major axes of the ellipse.

$$\text{Eccentricity} = \frac{\text{Minor axis length}}{\text{Major axis length}} \quad (1)$$

2. The orientation ϕ of the ellipse.
3. The minimum ratio r between the length of the major axis and the distance d from the reference point to the center of the ellipse.

$$r = \min \left(\frac{\text{Major axis length}}{d}, \frac{d}{\text{Major axis length}} \right) \quad (2)$$

4. The angle θ between the reference point and the center of the ellipse.

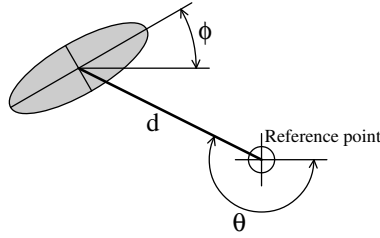


Fig. 2. An illustration of the four features used to describe the primitives.

3 Recognition of Actions

3.1 Recognition of Primitives

For a given set of actions a set of primitives is defined using the four features. To be able to recognize the primitives a Mahalanobis classifier is build by forming the covariance matrix for each primitive based on a set of representative examples. The four features are not equally important and therefore weighted in accordance with their importance. This yields the following classifier for recognizing a primitive at time, t :

$$\text{Primitive}(t) = \arg \min_i [(\mathbf{W} \cdot (\mathbf{f}_t - \mathbf{p}_i))^T \Pi_i^{-1} (\mathbf{W} \cdot (\mathbf{f}_t - \mathbf{p}_i))] \quad (3)$$

where \mathbf{f}_t is the feature vector estimated at time t , \mathbf{p}_i is the mean vector of the i th primitive, Π_i is the covariance matrix of the i th primitive, and \mathbf{W} contains the weights and are included as an element-wise multiplication.

The classification of a sequence can be viewed as a trajectory through the 4D feature space where, at each time-step, the closest primitive (in terms of Mahalanobis distance) is found. To reduce noise in this process we introduce a minimum Mahalanobis distance in order for a primitive to be considered in the first place. Furthermore, to reduce the flickering observed when the trajectory passes through a border region between two primitives we introduce a hysteresis threshold. It favors the primitive recognized in the preceding frame over all other primitives by modifying the individual distances. The classifier hereby obtains a "sticky" effect, which handles a large part of the flickering.

After processing a sequence the output will be a string with the same length as the sequence. An example is illustrated in equation 4. Each letter corresponds to a recognized primitive and \emptyset corresponds to time instances where no primitives are below the minimum required Mahalanobis distance. The string is pruned by first removing ' \emptyset 's, isolated instances, and then all repeated letters, see equation 5. A weight is generated to reflect the number of repeated letters (this is used below).

$$\text{String} = \{\emptyset, \emptyset, B, B, B, B, B, E, A, A, F, F, F, F, \emptyset, D, D, G, G, G, G, \emptyset\} \quad (4)$$

$$\text{String} = \{B, A, F, D, G\} \quad (5)$$

$$\text{Weights} = \{5, 2, 4, 2, 4\} \quad (6)$$

3.2 Recognition using Probabilistic Edit Distance

The result of recognizing the primitives is a string of letters referring to the known primitives. During a training phase a string representation of each action to be recognized is learned. The task is now to compare each of the learned actions (strings) with the detected string. Since the learned strings and the detected strings (possibly including errors!) will in general not have the same length, the standard pattern recognition methods will not suffice. We therefore apply the Edit Distance method [12], which can handle matching of strings of different lengths.

The edit distance is a well known method for comparing words or text strings, e.g., for spell-checking and plagiarism detection. It operates by measuring the distance between two strings in terms of the number of operations needed in order to transform one to the other. There are three possible operations: *insert* a letter from the other string, *delete* a letter, and *exchange* a letter by one from the other string. Whenever one of these operations is required in order to make the strings more similar, the score or distance is increased by one. The algorithm is illustrated in figure 3 where the strings *motions* and *octane* are compared.

The first step is initialization. The two strings are placed along the sides of the matrix, and increasing numbers are placed along the borders beside the strings. Hereafter

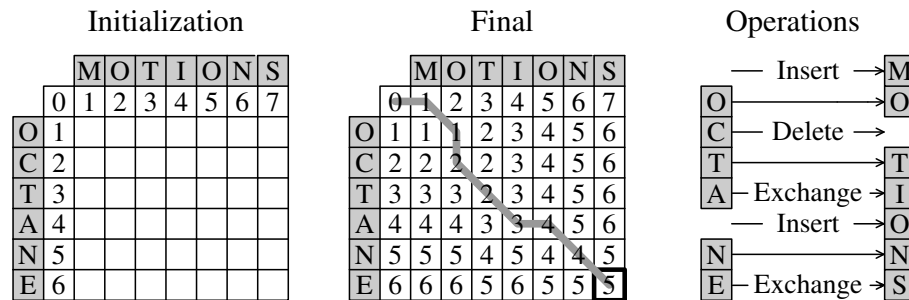


Fig. 3. Measuring the distance between two strings using edit distance.

the matrix is filled cell by cell by traversing one column at a time. Each cell is given the smallest value of the following four operations:

- Insert:** The value of the cell above + 1
- Delete:** The value of the cell to the left + 1
- Exchange:** The value of the cell up-left + 1
- No change:** The value of the cell up-left + 0. This is the case when the letters in question in the two strings are the same.

Using these rules the matrix is filled and the value found at the bottom right corner is the edit distance required in order to map one string into the other, i.e., the distance between the two strings. The actual sequence of operations can be found by back-tracing the matrix. Note that often more paths are possible.

When the strings representing the actions are of different lengths, the method tends to favor the shorter strings. Say we have detected the string $\{B, C, D\}$ and want to classify it as being one of the two actions: $\#1 = \{J, C, G\}$ and $\#2 = \{A, B, C, D, H\}$. The edit distance from the detected string to the action-strings will be two in both cases. However, it seems more likely that the correct interpretation is that the detected string comes from action #2 in a situation where the start and end has been corrupted by noise. In fact, 2 out of 3 of the primitives have to be changed for action #1 whereas only 2 out of 5 have to be changed for action #2. We therefore normalize the edit distance by dividing the output by the length of the action-string, yielding 0.67 for action #1 and 0.2 for action #2, i.e., action #2 is recognized.

The edit distance is a deterministic method but by changing the cost of each of the three operations with respect to likelihoods it becomes a probabilistic method¹. Concretely we apply the weights described above, see equation 6. These to some extent represent the likelihood of a certain primitive being correct. The higher the weight the more likely a primitive will be. We incorporate the weights into the edit distance method by increasing the score by the weight multiplied by β (a scaling factor) whenever a primitive is *deleted* or *exchanged*. The cost of *inserting* remains 1.

The above principle works for situations where the input sequence only contains one action (possibly corrupted by noise). In a real scenario, however, we will have

¹ This is related to the Weighted Edit Distance method, which however has fixed weights.

sequences which are potentially much longer than an action and which might include more actions after each other. The action recognition problem is therefore formulated as for each action to find the substring in the detected string, which has the minimum edit distance. The recognized action will then be the one of the substrings with the minimum distance. Denoting the start point and length of the substring, s and l , respectively, we recognize the action present in the detected string as:

$$\text{Action} = \arg \min_{k,s,l} PED(\Lambda, k, s, l) \quad (7)$$

where k index the different actions, Λ is the detected string, and $PED(\cdot)$ is the probabilistic edit distance.

4 Results

4.1 Test Setup

To evaluate our approach we use five arm gestures inspired by [10, 14], see figure 4. In order to get better insides to our novel recognition approach we apply semi-synthetic data in this work. Another reason for semi-synthetic data is that we need access to the 3D configurations of the test subjects when defining the primitives. Concretely we use a magnetic tracking system with four sensors to capture movements of the test subjects. The sensor placements are: one at the wrist, one at the elbow, one at the shoulder, and one at the upper torso (for reference). The hardware used is the Polhemus FastTrac [15] which gives a maximum sampling rate of 25Hz when using all four sensors. The data is converted into four Euler angles: three at the shoulder and one at the elbow in order to make the data invariant to body size. An action corresponds to a trajectory through a 4D space spanned by the Euler angles.

We use seven test subjects, who each perform each gesture 20 times. This leads to 840 synthetic sequences. We manual evaluate the sequences from three of the test subjects and find 10 primitives to describe the five different actions. The criteria for finding the primitives are 1) that they represent characteristic and representative 3D configurations, 2) that their projected 2D configurations contain a certain amount of fronto-parallel motion, and 3) that the primitives are used in the description of as many actions as possible, i.e., fewer primitives are required.

Based on the manually selected primitives we randomly choose 20 sequences for each primitive. The sequences are aligned temporally and the double difference images are calculated and represented by the four features, yielding a 4x4 covariance matrix for each primitive. The maximum Mahalanobis distance for primitive recognition is set to 40, the weighting of the features are $\{1, 2, 1, 2\}$, and $\beta = 1/8$. A string representation of each action is found and since the shortest string contains five primitives and the longest eight primitives, we only perform the probabilistic edit distance calculation for substrings having the lengths $\in [4, 16]$.

4.2 Tests

The tests are performed on the four test subjects not included in the training data. We randomly choose 23 sequences of each gesture, yielding 115 test sequences. For each

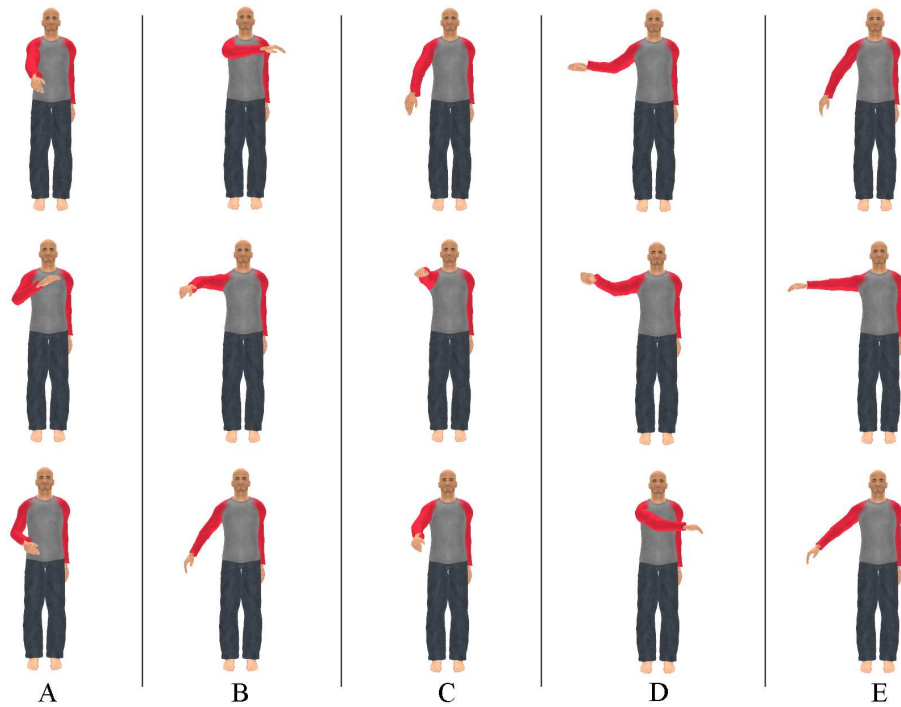


Fig. 4. Examples of images generated by Poser using real motion captured data. Each column shows samples from the five gestures. **A - Move closer:** A stretched arm is raised to a horizontal position pointing forward while the palm is pointing upwards. The hand is then drawn to the chest, and lowered down. **B - Move right:** Right hand is moved up in front of the left shoulder. The arm is then stretched while moved all the way to the right, and then lowered down. **C - Point forward:** A stretched arm is raised to a horizontal position pointing forward, and then lowered down. **D - Move left:** A stretched arm is raised to a horizontal position pointing right. The arm is then moved in front of the body ending at the right shoulder, and then lowered down. **E - Point right:** A stretched arm is raised to a horizontal position pointing right, and then lowered down.

sequence we add "noise" in both the beginning and end of the sequence. The noise is in the form of approximately half a sequence of a different gesture. This introduces the realistically problem of having no clear idea when an action commence and terminates.

In figure 5 a typical situation is shown for using the probabilistic edit distance to match a detected string with an action. The X-axis represents the frame number. The Y-axis represents the string length. The Z-axis represents the probabilistic edit distance - the smaller the better the match. One point on the surface, e.g., (4, 5, 1.2), corresponds to the distance 1.2 between a substring of the detected string, and an action-string. The substring has length 5 and starts at time instance #4. For this particular figure the best match is found for the substring of length 7 and starting at time instance #7. Its value is compared to the best matches for the other actions and the one with the smallest value

defines the recognized action for this sequence. The overall recognition rate is 91.3%. In figure 6 the confusion matrix for the results is shown.

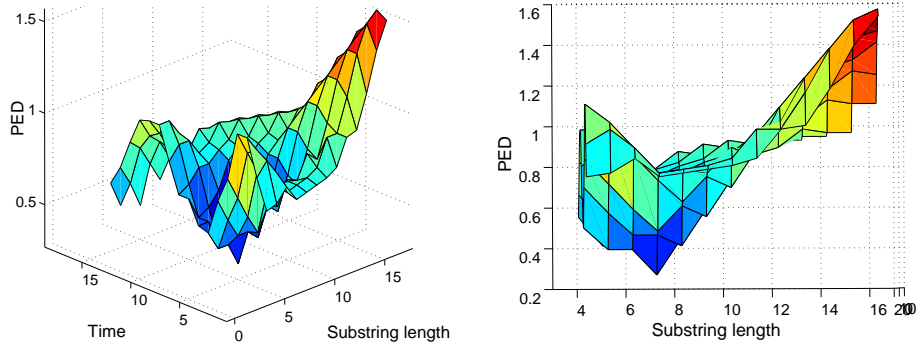


Fig. 5. An example of the probabilistic edit distance. See text for details.

	1	2	3	4	5
1. Point right	23				
2. Point forward		21			2
3. Move left	1		22		
4. Move right				23	
5. Move closer		7			16

Fig. 6. The confusion matrix for the recognition of the different actions.

5 Conclusion

In this paper we have presented an action recognition approach based on primitives as opposed to trajectories. Furthermore, we extract features from temporally local motion as opposed to background subtraction or another segmentation method relying on learned models and a relatively controlled environment. We hope this makes our approach less sensitive, but have still to prove so in a more comprehensive test.

The primitives used in this work are found manually. This turned out to be quite an effort due to the massive amount of data. Currently we are therefore working to automate this process [14].

The presented results showed that action #2 and #5 are sometimes confused. This problem might be solved by using better primitives - learned automatically. But this is not certain as the confusions are mainly due to the fact that 1) the two actions are very similar, and that 2) some test subjects did the same action rather differently. Seen from this point of view the recognition rate is quite good.

References

1. R.V. Babu and K.R. Ramakrishnan. Compressed domain human motion recognition using motion history information. In *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, Hong Kong, April 6-10, 2003.
2. J. Barbic, N.S. Pollard, J.K. Hodgins, C. Faloutsos, J-Y. Pan, and A. Safonova. Segmenting Motion Capture Data into Distinct Behaviors. In *Graphics Interface*, London, Ontario, Canada, May 17-19 2004.
3. F. Bettinger and T.F. Cootes. A Model of Facial Behaviour. In *IEEE International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, May 17 - 19 2004.
4. A. Bobick and J. Davis. The Recognition of Human Movement Using Temporal Templates. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.
5. A.F. Bobick and J. Davis. A Statebased Approach to the Representation and Recognition of Gestures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(12):1325 – 1337, 1997.
6. C. Bregler. Learning and Recognizing Human Dynamics in Video Sequences. In *Conference on Computer Vision and Pattern Recognition*, pages 568 – 574, San Juan, Puerto Rico, 1997.
7. L. Campbell and A. Bobick. Recognition of Human Body Motion Using Phase Space Constraints. In *International Conference on Computer Vision*, Cambridge, Massachusetts, 1995.
8. J. Gonzalez, J. Varona, F.X. Roca, and J.J. Villanueva. *aSpaces*: Action spaces for recognition and synthesis of human actions. In *AMDO*, pages 189–200, AMDO02, 2002.
9. O.C. Jenkins and M.J. Mataric. Deriving Action and Behavior Primitives from Human Motion Data. In *Proc. IEEE Int. Conf. on Intelligent Robots and Systems*, pages 2551–2556, Lausanne, Switzerland, Sept.30 – Oct.4, 2002.
10. A. Just and S. Marcel. HMM and IOHMM for the Recognition of Mono- and Bi-Manual 3D Hand Gestures. In *ICPR workshop on Visual Observation of Deictic Gestures (POINT-ING04)*, Cambridge, UK, August 2004.
11. A. Kale, N. Cuntoor, and R. Chellappa. A Framework for Activity-Specific Human Recognition. In *International Conference on Acoustics, Speech and Signal Processing*, Orlando, Florida, May 2002.
12. V.I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Doklady Akademii Nauk SSSR*, 163(4):845–848, 1965.
13. C. Rao, A. Yilmaz, and M. Shah. View-Invariant Representation and Recognition of Actions. *Journal of Computer Vision*, 50(2):55 – 63, 2002.
14. L. Reng, T.B. Moeslund, and E. Granum. Finding Motion Primitives in Human Body Gestures. In S. Gibet, N. Courty, and J.-F. Kamps, editors, *GW 2005*, number 3881 in LNAI, pages 133–144. Springer Berlin Heidelberg, 2006.
15. <http://polhemus.com/>, January 2006.
16. D. Weinberg, R. Ronfard, and E. Boyer. Motion History Volumes for Free Viewpoint Action Recognition. In *IEEE Int. Workshop on Modeling People and Human Interaction*, 2005.
17. A. Yilmaz and M. Shah. Actions Sketch: A Novel Action Representation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, San Diego, CA, June 20-25, 2005.
18. K. Yoshinari and M. Michihito. A Human Motion Estimation Method using 3-Successive Video Frames. In *Int. Conf. on Virtual Systems and Multimedia*, Gifu, Japan, 1996.
19. H. Yu, G.-M. Sun, W.-X. Song, and X. Li. Human Motion Recognition Based on Neural Networks. In *Int. Conf. on Communications, Circuits and Systems*, Hong Kong, May 2005.