

Handsignals Recognition From Video Using 3D Motion Capture Data

Tai-Peng Tian and Stan Sclaroff*
Computer Science Department
Boston University
Boston, MA 02215

Abstract

Hand signals are commonly used in applications such as giving instructions to a pilot for airplane takeoff or direction of a crane operator by a foreman on the ground. A new algorithm for recognizing hand signals from a single camera is proposed. Typically, tracked 2D feature positions of hand signals are matched to 2D training images. In contrast, our approach matches the 2D feature positions to an archive of 3D motion capture sequences. The method avoids explicit reconstruction of the 3D articulated motion from 2D image features. Instead, the matching between the 2D and 3D sequence is done by backprojecting the 3D motion capture data onto 2D. Experiments demonstrate the effectiveness of the approach in an example application: recognizing six classes of basketball referee hand signals in video.

1. Introduction

Hand signals are useful for non verbal communication when people are out of voice range or in a noisy environment. Such hand signals are commonly used in scenarios such as directing an airplane to the runway for takeoff or controlling the flow of traffic at a road junction. Hand signals are also used as an interface for instructing robots [18].

This paper describes a technique for recognizing hand signals from a sequence of tracked 2D feature positions obtained with a single uncalibrated camera. The sequence of 2D feature positions is assumed to be temporally segmented and the system tries to recognize the sequence by matching it against an archive of 3D motion capture data of hand signals.

The two key motivations for using 3D motion capture include ease in obtaining high quality motion capture data and greater accuracy in representing hand signals as compared to a 2D representation.

The matching algorithm between a 2D sequence and a 3D sequence lies at the heart of our recognition algorithm. Broadly speaking, the algorithm computes a dissimilarity

score between a 2D sequence and a 3D motion capture sequence in two steps. Firstly, the 3D motion capture data is projected to 2D and the optimal projection matrix is chosen by minimizing the back projection error between each frame of the 2D and 3D sequences. Secondly, a Dynamic Time Warping (DTW) algorithm is used to find the best alignment between the 2D and 3D sequence. In our implementation, the two steps are fused together to form a coherent matching algorithm. During recognition, the 3D motion capture data with the lowest matching score will be output as the recognition result.

This technique does not place any assumptions on the location or movement of the camera. The camera is allowed to move freely and such flexibility gives the algorithm the ability to compare 2D motion taken from an uncalibrated camera from any view point. Additionally, there is no need to reconstruct the 3D body motion from the tracked 2D features sequence during the matching process.

In this preliminary study, we assume that the sequence has been temporally segmented and the 2D feature points can be reliably tracked over the whole sequence. Based on this assumption, experiments are performed on synthesized test sequences and results are encouraging. Further experiments in Section 4 also show that the algorithm works well when feature points are missing or corrupted with noise.

2 Related Work

Typically, such an algorithm would be useful in gestural communication with intelligent vehicles such as directing an Unmanned Aerial Vehicle (UAV). Cohen worked on recognizing oscillatory motion [5] derived from frequently used aircraft runway commands. The two main types of oscillatory motion identified are circular and linear. Circular motions are those that form an ellipse on the image plane and linear are those that form a line in the image plane. The oscillatory motions are compactly described using a system of differential equations. Though elegant, a fair amount of trial and error is needed to handcraft models for hand signal.

2D-2D Matching: Template based methods typically build and store models of features extracted from the training video footage to be used for comparison during the

*This research was funded in part by NSF grants CNS-0202067, IIS-0208876 and IIS-0308213, and ONR grant N00014-03-1-0108.

recognition stage. Motion Templates [2] compress information from a temporal sequence into a single image. Efros, et al. [6] separate optical flow into four channels and use it as a feature for each frame. High recognition rates have been reported using such techniques but templates are usually view-dependent and require training video footage to be taken from multiple viewpoints in order to form a representative training set.

2D-3D Matching: A technique based on invariants [13] has been proposed for human motion recognition. The technique extracts 2D invariants from the video footage for matching with 3D invariants obtained from motion capture sequences. Classification is achieved by matching the 2D invariants against a set of 3D invariant features obtained from a motion capture library. Thus given a 2D invariant feature, the algorithm will try to solve for the corresponding 3D invariant feature. As it is a under-constrained problem, the solution is a quartic surface in 3D invariant feature space. Therefore recognition is reduced to a path-surface intersection problem with the 3D invariants from the motion capture library. Only results of classifying individual frames for running, walking and sitting sequences are reported. Moreover, the method does not address the time scaling problem between two motion sequences.

3D-3D Matching: Dynamic Time Warping is used in an animation retrieval system [9, 10]. Given a 3D motion data segment as a query, comparison between the query and an element in the database is done using a dynamic programming approach. Each frame of the motion data consists of surface points derived from the mesh representing the animation model. Taking into account that motion matching should be invariant to the location of the animation model on the ground as well as the direction it is facing, the dissimilarity function first computes the rotation and translation that aligns the two frames, and then takes the weighted Euclidean distance between corresponding points. The dissimilarity measure proposed in this paper shares a similar idea of finding the optimal transformation parameters to project the 3D database motion frame in order to match it with a 2D query frame. The main difference lies in the fact that the rotation and translation are represented in the projection matrix implicitly.

3. Aligning a 2D sequence with a 3D sequence

Alignment is usually a preprocessing step prior to computing the actual matching score between two sequences. The purpose of aligning two sequences is to handle the time scale difference between the two time sequences. For example, both sequences may be sampled from the same signal at different rates, thus there is a mismatch in the number of sample size. Natural variation in speed among peo-

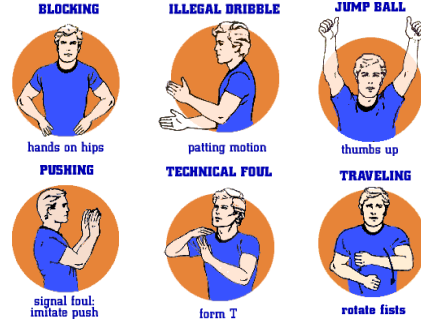


Figure 1: Hand signals used for the experiments. Images are obtained from NBA website at www.nba.com

ple is another probable cause. Thus alignment ensures that the points on one sequence are matched to correspondingly similar points on the other sequence.

A standard technique for aligning sequences is to use the Dynamic Time Warping algorithm (DTW). Traditionally the two sequences to be aligned are expressed in the same representation. For example, matching time series or aligning words to find the longest subsequence.

When dealing with a 2D pose sequence and a 3D pose sequence, we need to define a dissimilarity between the two. One way to make a comparison is to transform one signal into the same representation as the other. Such a transformation can be applied to the 2D pose sequence or the 3D pose sequence. A method of transforming the 2D pose sequence into 3D pose sequence is to reconstruct the 3D structure from the 2D pose sequence using techniques such as a pose from a single image [1, 16, 14]. Alternatively, techniques for monocular motion capture could be used to reconstruct a 3D pose sequence from the monocular 2D pose sequence [12, 7]. Both sets of approaches are inherently difficult as such a reconstruction is an ill-posed inverse problem.

An alternative explored in this paper is to back-project the 3D pose sequence onto the 2D pose sequence. Such a transformation is more appealing as the problem is well-posed, granted that a sufficient number of correspondences is known. Then the next issue is finding the best projection matrix to back-project the 3D pose onto 2D.

3.1 Matching a 2D pose with a 3D pose

Assuming that correspondences between the 3D points and 2D feature points are known, given a 3D pose \mathbf{p} with k joints

$$\mathbf{p} = (x_1, y_1, z_1, \dots, x_k, y_k, z_k)^T$$

and a 2D pose \mathbf{q} with k feature points,

$$\mathbf{q} = (u_1, v_1, \dots, u_k, v_k)^T$$

the goal is to find the projection matrix \mathbf{M} that minimizes the back-projection between \mathbf{p} and \mathbf{q} . Where

$$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix}.$$

One way of recovering the matrix \mathbf{M} is to minimize the least squares distance between the 2D points and the back-projected 3D points. A solution is given in Section 6.3 of [17] and a minimum of six correspondences is required in order to solve for the eleven independent entries in \mathbf{M} . For completeness the algorithm is briefly described here.

Since the correspondences between the 3D points \mathbf{p} and the 2D features \mathbf{q} are known, we can form the equation :

$$\mathbf{A}\mathbf{m} = 0,$$

with

$$\mathbf{A} = \begin{pmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 & -x_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 & -x_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_k & y_k & z_k & 1 & 0 & 0 & 0 & 0 & -u_kx_k & -u_ky_k & -u_kz_k & -x_k \\ 0 & 0 & 0 & 0 & x_k & y_k & z_k & 1 & -v_kx_k & -v_ky_k & -v_kz_k & -x_k \end{pmatrix}$$

and

$$\mathbf{m} = [m_{11}, m_{12}, \dots, m_{33}, m_{34}]^T.$$

The least squares solution to the above equation can be obtained via the Singular Value Decomposition (SVD). With $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$, \mathbf{m} corresponds to the column of \mathbf{V} with the smallest singular value found in \mathbf{A} .

The matrix \mathbf{M} is correct up to a scaling factor γ . To recover γ , it suffices to observe that

$$\kappa\gamma\sqrt{m_{31}^2 + m_{32}^2 + m_{33}^2} = \sqrt{r_{31}^2 + r_{32}^2 + r_{33}^2} = 1$$

where $\kappa \in \{1, -1\}$ denotes the sign and r_{ij} are the entries of the rotation required to align the 3D points with the 2D points. The value of κ can be derived from m_{34} using the relation

$$T_z = \kappa\gamma m_{34}$$

where $[T_x, T_y, T_z]^T$ is the origin of the world reference frame. The value of $[T_x, T_y, T_z]^T$ can always be chosen to be positive as it is user defined. In our experiments, the world reference frame is translated to the root of the skeleton, that is joint one in Figure 2.

After obtaining the correct scaling factor γ , the desired dissimilarity function is

$$D(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^k \left\| \mathcal{P} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} - \begin{pmatrix} u_i \\ v_i \end{pmatrix} \right\|$$

Where \mathcal{P} is the function that applies the projection matrix $\gamma\kappa\mathbf{M}$ to $[x_i, y_i, z_i]^T$ to obtain the corresponding 2D points.

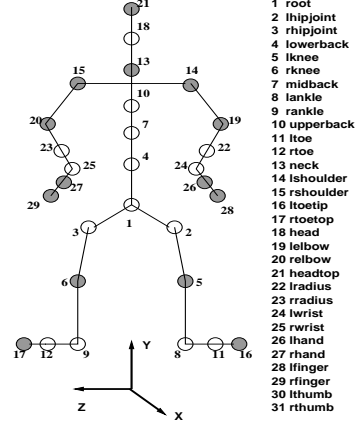


Figure 2: Feature positions for the 3D motion capture data.

3.2 Computing the alignment score

Given a 3D motion capture sequence

$$\mathbf{P} = \langle \mathbf{p}_1, \dots, \mathbf{p}_s \rangle$$

of length s and a sequence of 2D feature locations

$$\mathbf{Q} = \langle \mathbf{q}_1, \dots, \mathbf{q}_t \rangle$$

of length t , an alignment between the two sequences can be computed via the DTW algorithm. The score σ for the alignment is defined using the recurrence relation

$$\sigma(\mathbf{P}, \mathbf{Q}) = \begin{cases} 0 & \text{if } \text{len}(\mathbf{P}) = 0 \text{ or } \text{len}(\mathbf{Q}) = 0 \\ D(\mathbf{p}_1, \mathbf{q}_1) + \min \begin{cases} \sigma(\Downarrow\mathbf{P}, \mathbf{Q}), \\ \sigma(\mathbf{P}, \Downarrow\mathbf{Q}), \\ \sigma(\Downarrow\mathbf{P}, \Downarrow\mathbf{Q}) \end{cases} & \text{otherwise} \end{cases}$$

where

$$\begin{aligned} \text{len}(\langle \mathbf{p}_1, \dots, \mathbf{p}_s \rangle) &= s \\ \Downarrow \langle \mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_s \rangle &= \langle \mathbf{p}_2, \dots, \mathbf{p}_s \rangle \end{aligned}$$

An efficient algorithm to compute the σ is to use dynamic programming. The DTW algorithm proceeds by computing an $s + 1$ by $t + 1$ matrix E starting from the cell $E(s, t)$. Where

$$E(i, j) = \begin{cases} 0 & \text{if } i > s \text{ or } j > t \\ D(\mathbf{p}_i, \mathbf{q}_j) + \min \begin{cases} E(i + 1, j) \\ E(i, j + 1) \\ E(i + 1, j + 1) \end{cases} & \text{otherwise} \end{cases}$$

and the algorithm is summarized in Algorithm 1.

The matrix E is used to solve the recurrence relation in a bottom up manner. Ultimately, the entry $E(1, 1)$ will contain the score σ . It stores the cumulative dissimilarity value of the optimal alignment path between the two sequences. To prevent bias toward lower error for shorter sequences, σ is normalized by the length of the optimal path l . The average dissimilarity score $\hat{\sigma}$ is defined as

$$\hat{\sigma}(\mathbf{P}, \mathbf{Q}) = \sigma(\mathbf{P}, \mathbf{Q})/l.$$

Computation of the optimal path length l is described by the function *backtrack* in Algorithm 1.

For each query, the algorithm computes the dissimilarity function D $O(st)$ times. Each invocation of D makes a call to the SVD algorithm with complexity $O(\min\{2k(12)^2, (2k)^2k\})$. Ignoring constants the complexity is $O(k)$ for the SVD algorithm and the overall time complexity is $O(stk)$.

4 Experiments and Results

The motion capture data used in our experiments is obtained from the CMU Graphics lab [4]. It contains motion capture of nine different subjects performing basketball hand signals. Six different signals are chosen and manually segmented into the individual hand-signals. The hand-signals are depicted in Figure 3. Below is a breakdown of the six hand-signals :

1. *Blocking* and *Technical Foul* are static poses.
2. *Jumpball* and *Pushing* are non-cyclic but dynamic hand-signals
3. *Illegal Dribble* and *Traveling* are cyclic and dynamic hand-signals

Table 1 shows the distribution of the motion capture data for training and testing. From the nine subjects, four of them are used for training and the remaining five subjects are used for testing. The feature points used in the experiments are show in Figure 2.

The classifier used is the nearest neighbor classifier. Given a query sequence \mathbf{Q} , the classifier will compare \mathbf{Q} against all 3D motion segments in the database using the score $\hat{\sigma}$. The query \mathbf{Q} is assigned the same motion label as the lowest scoring 3D motion segment. The algorithm is summarized in Algorithm 1.

Currently using unoptimized Matlab code, it takes approximately 55 secs to compare a query sequence of length 40 frames against the entire database of 5170 frames. The machine used is a 2.2 GHz AMD Opteron Processor running Matlab 6.5.

Algorithm 1 Classifying a 2D features sequence \mathbf{Q} .

Input : Query $\mathbf{Q} = \langle \mathbf{q}_1, \dots, \mathbf{q}_t \rangle$ and database \mathcal{DB} containing 3D motion capture sequences

for each element $\mathbf{P} \in \mathcal{DB}$ **do**
 score[i] = $\hat{\sigma}(\mathbf{P}, \mathbf{Q})$
 label[i] = handsignal category of \mathbf{P}
end for
Return the label with the lowest score

Function $\hat{\sigma}(\langle \mathbf{p}_1, \dots, \mathbf{p}_s \rangle, \langle \mathbf{q}_1, \dots, \mathbf{q}_t \rangle)$

Initialize a matrix E of size $s + 1$ by $t + 1$ with zeros

Set row $s + 1$ and column $t + 1$ of E to ∞

$E(s + 1, t + 1) = 0$

for $i = s$ **downto** 1 **do**

for $j = t$ **downto** 1 **do**

$E(i, j) = \min\{E(i + 1, j), E(i, j + 1), E(i + 1, j + 1)\}$

$E(i, j) = E(i, j) + D(\mathbf{p}, \mathbf{q})$

end for

end for

$l = \text{backtrack}(E)$

Return $E(1, 1)/l$

Function : *backtrack*(E)

Let s and t be the number of rows and columns in E respectively.

$l = 0$

$r = 1, c = 1$

while not ($i = s$ and $j = t$) **do**

if $r = s$ **then**

$c = c + 1$

else if $c = t$ **then**

$r = r + 1$

else

 Find the minimum value among $E(r + 1, c)$, $E(r, c + 1)$ and $E(r + 1, c + 1)$. The variable r and c will take on the coordinates of the lowest value.

end if

$l = l + 1$

end while

Return l

4.1 Experiment 1: Using all feature positions

In the first experiment, all the feature positions in Figure 2 are used for testing the effectiveness of the algorithm. The test cases are synthesized from the five testing subjects E, F, G, H and I. The test subjects' motion capture data is projected to obtain a frontal view. The projected 2D feature positions are used as input for this experiment.

Results of the experiment are shown in Table 2. All the 2D query motions are correctly classified except for two misclassified *Illegal Dribble* sequences. For the first misclassification, the subject did not lift his hands high enough when executing the *Illegal Dribble* hand-signal. Therefore, from a frontal view, it was confused with the *Pushing* hand-signal. For the other query, it was wrongly matched to a

	Subj	Blocking	Illegal Dribble	JumpBall	Pushing	Technical Foul	Traveling
Train	A	0	2	2	2	2	2
	B	0	1	1	1	1	1
	C	0	2	2	2	2	2
	D	2	2	2	1	0	3
Total		2	7	7	6	5	8
Test	E	0	2	2	2	2	2
	F	0	2	2	2	2	2
	G	0	1	1	1	1	1
	H	0	1	1	1	1	1
	I	3	3	3	3	0	3
Total		3	9	9	9	6	9

Table 1: The distribution of motion capture samples used for training and testing in the experiments.

training subject who was not performing the full action of the *Jumpball* hand-signal. The training subject’s *Jumpball* hand-signal moved only a small distance instead of over the head and this lead to the confusion.

4.2 Experiment 2: Simulating a robust tracker

This experiment simulates the scenario when the 2D features could be robustly tracked. Such tracking results could come from tracking brightly colored markers on the subject; for example, using a robust estimator to fit features tracked over multiple frames with a motion model [11].

Given the true 2D position \mathbf{x} of a feature point on the image and the corresponding feature position \mathbf{y} obtained by using the tracker, the tracker error ϵ is defined as $\epsilon = \|\mathbf{x} - \mathbf{y}\|$. We further assume ϵ follows a Gaussian distribution where $\epsilon \sim \mathcal{N}(0, \alpha)$. The tracker is parameterized by its *error margin* α . For example with an error margin of α pixels, it implies that 95% of the time, the tracked position falls within a radius 2α of the true position.

In this experiment, the height of the person is scaled to one and zero mean Gaussian noise is added to each feature position to simulate tracked feature points. The standard deviation of the noise was varied between 0 and 0.1.

The recognition results are shown in Figure 3. The x-axis represents the error margin of the tracker and the y-axis denotes the overall recognition rate. The solid line curve represents how the overall recognition rate varies with the tracker’s error margin. The black dots represent the experiments conducted under the corresponding error margin.

	Blocking	Illegal Dribble	Jumpball	Pushing	Technical Foul	Traveling	Accuracy
Blocking	3	0	0	0	0	0	100%
Illegal Dribble	0	7	1	1	0	0	78%
Jumpball	0	0	9	0	0	0	100%
Pushing	0	0	0	9	0	0	100%
Technical Foul	0	0	0	0	6	0	100%
Traveling	0	0	0	0	0	9	100%
Overall Acc							96%

Table 2: Confusion Matrix results. Each row contains outcome of classifying queries drawn from the same hand-signal category. The diagonal entries are the number of correctly classified queries and the off-diagonal entries represent misclassification

Within the range 0 to 0.06 error margin, the overall accuracy drops slowly with a gentle slope. Beyond 0.06 standard deviation, the recognition declines at a higher rate. Thus in the context of tracking, it is desirable that the tracker performs with an error margin of at most 0.06 (in normalized coordinates), to obtain reasonable recognition rates of 96% to 70%. Analogously if the person’s height is 300 pixels tall, the error between the tracked feature points and true feature position should be less than 36 pixels, 95% of the time.

4.3 Experiment 3: Reduced number of feature positions

In a realistic scenario, not all the 31 feature positions in Figure 2 are trackable. Experiment 3 investigates the effect of decreasing feature positions on recognition rate. In a basketball game sports video, feature positions like 4,7 and 10 are on the spine. These positions may not be realistically trackable and hence they are dropped from Experiment 3. Similarly, positions 1, 2, 3, 8, 9, 11, 12, 18, 22, 23, 24 and 25 are also dropped. The number of feature positions is reduced from 31 to 14. These 14 feature positions are denoted by the shaded circles in Figure 2. As before, noise is also added to the 2D feature sequences to simulate tracking.

The experiment results are shown in Figure 3. The dashed line with triangles plots recognition rate versus noise level using the 14 shaded feature positions. The triangles denote the actual experiments performed and the curve is interpolated from these points.

To obtain a reasonable accuracy between 75 % and 90%, the tracker is expected to have an error margin less than 0.04 measured in normalized coordinates. Again, using the 300 pixels tall man as an example, the tracker should be within 24 pixels of the true position 95% of the time.

In the last experiment, the number of feature positions are further dropped from 14 to 10. The four points removed represent feature positions on the legs. This simulates the real world example where only the upper body of the referee is visible in the video. The recognition curve is shown as a dotted curve with crosses in Figure 3. Once noise is added, the experiment's overall accuracy falls below 30%. This experiment suggests that the upper body representation is not robust to noise. With the feature points on the legs removed, the upper body feature points are able to match more 3D poses by rotating about the spine, thus reducing the chances of finding the correct match. A possible solution may be to track more feature points on the arms, hands and face.

5 Conclusion and Future Work

A new technique for recognizing 2D motion using a 3D motion capture data archive is described in this paper. The algorithm has the advantage of viewpoint invariance as it automatically computes the optimal projection matrix between the 2D and 3D sequence as part of the matching process.

In this preliminary work, only synthesized data is used for experiments and results are encouraging as misclassification rates are reasonable. In the experiments, we have shown the behavior of the tracker with respect to different levels of noise. Simulating different real world conditions, the experiments relate the behavior of the tracker to the recognition rate of the algorithm.

The next step is to evaluate the recognition rate of the algorithm on real footage of basketball games. This will require modules for detection and tracking of the referee e.g. [8, 19]. Given overall position of the referee obtained via person tracking, body feature locations could be tracked [3]. Reliability of body feature tracking could be further improved through the use of skin-color based hand tracking, as well as a face detection/tracking module [15].

References

- [1] C. Barron and I. A. Kakadiaris. Estimating anthropometry and pose from a single uncalibrated image. *CVIU*, 81:269–284, 2001.
- [2] A. Bobick and J. Davis. The representation and recognition of action using temporal templates. *PAMI*, 23(3):257–267, 2001.
- [3] T.-J. Cham and J. M. Rehg. A multiple hypothesis approach to figure tracking. In *CVPR*, pages 239–245, 1999.
- [4] CMU. Graphics lab motion capture database, <http://mocap.cs.cmu.edu>.
- [5] C. J. Cohen. *Dynamical System Representation, Generation, and Recognition of Basic Oscillatory Motion Gestures, and*

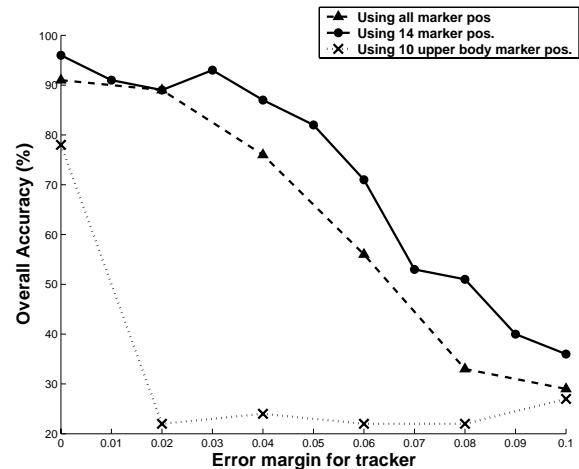


Figure 3: Classifier performance with respect to increasing noise.

Applications for the Control of Actuated Mechanism. PhD thesis, University of Michigan, 1996.

- [6] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Proc. ICCV*, pages 726–733, 2003.
- [7] N. R. Howe, M. E. Leventon, and W. T. Freeman. Bayesian reconstruction of 3d human motion from single-camera video. In *NIPS*, volume 12, page view 820, 1999.
- [8] S. Ioffe and D. A. Forsyth. Probabilistic methods for finding people. *IJCV*, 43(1):46–68, 2001.
- [9] L. Kovar and M. Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Transactions on Graphics*, 23(3):559–568, 2004.
- [10] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. *ACM Transactions on Graphics*, 21(3):473–482, 2002.
- [11] V. Lepetit, A. Shahrokhni, and P. Fua. Robust data association for online applications. In *CVPR*, pages 281–288, 2003.
- [12] D. Liebowitz and S. Carlsson. Uncalibrated motion capture exploiting articulated structure constraints. In *Proc. ICCV*, pages 230–237, 2001.
- [13] V. Parameswaran and R. Chellappa. View invariants for human action recognition. In *CVPR*, pages 619–619, 2003.
- [14] V. Parameswaran and R. Chellappa. View independent human body pose estimation from a single perspective image. In *CVPR*, pages 16–22, 2004.
- [15] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *PAMI*, 20(1):23–28, 1998.
- [16] C. J. Taylor. Reconstruction of articulated objects from point correspondences in a single uncalibrated image. *CVIU*, 80(3):349–363, 2000.
- [17] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.
- [18] S. Waldherr, S. Thrun, and R. Romero. A gesture-based interface for human-robot interaction. *Autonomous Robots*, 9(2):151–173, 2000.
- [19] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland. Pfunder: Real-time tracking of the human body. *PAMI*, 19(7):780–785, July 1997.