COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2014; **25**:293–302
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.1590

SPECIAL ISSUE PAPER

Real-time motion data annotation via action string

Tian Qi[1], Jun Xiao[1]*, Yueting Zhuang[1], Hanzhi Zhang[1], Xiaosong Yang[2], Jianjun Zhang[2] and Yinfu Feng[1]

[1] Institute of Artificial Intelligence, University of Zhejiang, 38 Zheda Road, Hangzhou, Zhejiang 310027, China
[2] National Centre for Computer Animation, Bournemouth University, Poole, Dorset BH12 5BB, UK

ABSTRACT

Even though there is an explosive growth of motion capture data, there is still a lack of efficient and reliable methods to automatically annotate all the motions in a database. Moreover, because of the popularity of mocap devices in home entertainment systems, real-time human motion annotation or recognition becomes more and more imperative. This paper presents a new motion annotation method that achieves both the aforementioned two targets at the same time. It uses a probabilistic pose feature based on the Gaussian Mixture Model to represent each pose. After training a clustered pose feature model, a motion clip could be represented as an action string. Then, a dynamic programming-based string matching method is introduced to compare the differences between action strings. Finally, in order to achieve the real-time target, we construct a hierarchical action string structure to quickly label each given action string. The experimental results demonstrate the efficacy and efficiency of our method. Copyright © 2014 John Wiley & Sons, Ltd.

KEYWORDS

motion annotation; action recognition; GMM pose feature; action string; string matching

***Correspondence**

Jun Xiao, Institute of Artificial Intelligence, Zhejiang University, 38 Zheda Road, Hangzhou, Zhejiang 310027, China.
E-mail: junx@cs.zju.edu.cn

1. INTRODUCTION

With the increasing popularity of mocap devices, using pre-captured real human motions to create motion sequences for virtual characters has become a standard procedure for animation production. A lot of research has been undertaken in recent years, such as motion synthesis [1–3], motion editing [4,5], and motion retargeting [6]. These techniques allow animators to specify what actions should be performed during animation and how it should be edited to fit the new virtual environment. To find a 'perfect' candidate from a motion database, a thorough and reliable annotation of all the stored motions is of great importance. Even though there is an explosive growth of freely available motion databases, there is still a lack of efficient methods that can automate the annotation process without manual intervention.

Because of the innovation of low-cost and portable motion capture systems, such as XSens [7] and Kinect [8], mocap data have been widely used in many games and home entertainment systems. The system needs to understand what kind of action the user has just performed and give a reasonable response immediately to the user. This poses a further challenge to motion annotation techniques that are required to be fully automatic for new motions and be able to finish the motion acquisition and recognition in real time.

In this paper, we present a new method to annotate motions automatically and efficiently. Figure 1 shows the framework of our system. First, a probabilistic pose feature is defined to represent each pose in the database. Then, each pose feature is indexed and translated into a 'character', each motion clip is turned into an 'action string'. In the third step, a dynamic programming-based string matching method is applied to compare and annotate those strings. Finally, to meet the real-time requirement, a hierarchical action string structure is constructed to search and label action strings.

As the main contribution of this paper, we propose a novel online annotation procedure to label unknown motion data in real-time. The key idea is to treat a human action as a string, which is inspired by the work of Liu *et al*. [9]. In our work, a new pose feature is proposed, and an effective string matching method is introduced to make it more robust. In addition, the action string structure is constructed based on a bottom-up hierarchical clustering algorithm to label a given action string in real-time.

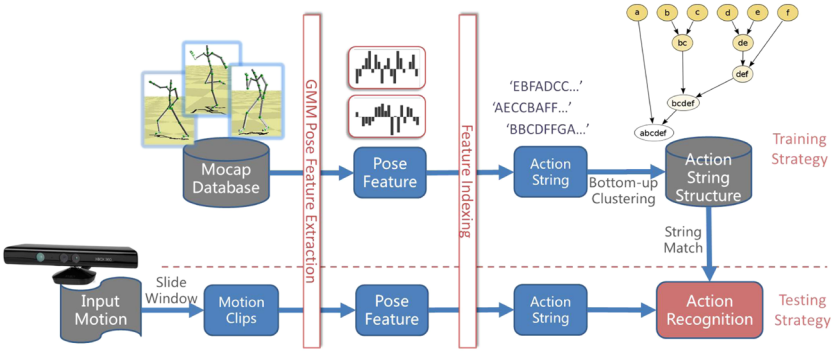


Figure 1. The flowchart of our motion annotation method. The training process is shown at the top, and the annotation process is shown at the bottom.

The remainder of this paper is organized as follows. In Section 2, the related work is discussed. The Gaussian Mixture Model (GMM) pose feature and action string are described in Sections 3 and 4. The experimental results are given in Section 5, and finally, the conclusion and future work are discussed in Section 6.

2. RELATED WORK

Human action recognition and annotation techniques have been widely used in many areas such as video surveillance, motion synthesis, and interactive entertainment. A great deal of research has been carried out in the past two decades. It can be traced back to a very early tradition problem, human motion recognition from monocular videos [10]. Because the recognition precision is limited by 2D videos, background subtraction remains a troublesome problem.

With the popularization of depth cameras, recent action recognition research [11–14] has been focused on depth maps. Li *et al*. [11] first used RGB-D sensors for human action recognition. In their work, a bag of 3D points is efficiently sampled by action graph, and a human posture model is constructed for recognition. Ni *et al*. [12] proposed a depth-layered multi-channel spatio-temporal interest points framework where spatio-temporal interest points are divided into multiple depth-layered channels and pooled correspondingly to incorporate the spatial information. Also, a 3D motion history images scheme is proposed in the same paper to recognize the human action. Because the captured depth images contains a lot of noise from both the clothing of the performer and the background, it is still very challenging to annotate motion clips directly from the depth map.

With the popularity of mocap devices in home entertainment systems, many research works and even commercial products have been developed to extract the human skeletons from a depth map. This results in a new research direction for motion annotation to recognize motions from mocap data. In this category, there are two crucial problems: how to predict human skeleton (or get motion data) from a depth map and how to recognize human action from skeleton movement in real-time. In recent years, many research works [15–17] have been aiming to the first problem, and the predicted skeleton becomes more and more robust. For example, Shotton *et al*. [17] proposed a novel depth image feature to represent each pixel, and randomized decision forests is used for training a model from a large database. After the classification of every pixel within the human body, a skeleton could be recovered from the labeled body parts. Because the extracted skeleton movement becomes more and more reliable and clean, in this paper, we will focus on the second problem, real-time human action recognition based on mocap data, assuming that the human skeleton could be well generated in real-time from a depth map.

Compared with monocular video and depth maps, mocap data are more structured and less noisy. This ensures that mocap-based motion annotation produces more accurate result. Forbes and Fiume [18] employed weighted principal component analysis (PCA) to distinguish the different importance of different skeleton nodes and presented a search algorithm to find similar motions. Meng *et al*. [19] introduced the frequently occurring temporal motion patterns (motion motifs). In their work, the motif discovery problem is translated into finding continuous paths in a matching trellis, and a tree-growing method is introduced to search for the continuous paths. However, these methods [9,18–20] can only recognize human actions from a pure motion clip (i.e., with only one type of motion) rather than an unsegmented or cross-category motion. Muller and Roder proposed a concept of motion template (MT) [21], by which a class of logically related motions can be represented by an explicit interpretable matrix using a set of Boolean geometric feature. Then an annotation procedure [22] based on MT is presented to segment and annotate motion data by comparing it with the available MTs. The training process of MT could tolerate temporal variance by using dynamic time warping (DTW), but if the training data in the same motion class have different cycles, MT will not be well trained. Moreover, both the training and comparing process of MT is time-consuming. Liu *et al*. [9] proposed a method to translate a given motion into a string. In their work, PCA is applied directly to all

motion poses in the database to reduce the dimensions, and a piecewise-linear model is generated via a divisive clustering method. Action recognition is then converted into a string matching problem. However, because there is no pose feature extracted, the clustering model could only answer the yes-or-no question, not giving a probability. To improve the distinction capability, we propose a GMM-based probabilistic pose feature before the 'action string' is generated.

3. GAUSSIAN MIXTURE MODEL POSE FEATURE

3.1. Feature Model Training

Motion data consist of the 3D positions of the joints of the subject frame by frame. A motion clip s with m frames can be represented as the set of its poses $s = \{f_1, f_2, \ldots, f_m\}$, where each frame f_i contains the x, y, z coordinates of all skeleton nodes. Intuitively, a motion could be described by some representative poses, which we call the key-poses. We refer to the method in [23], the key-poses are extracted from each motion class, after the key-frame selection. However, it is too complicated for our real-time application. So, based on their method, we proposed a compact approach to extract key-poses from all the poses in a motion database directly. From Figure 2, we can see that the key-poses extracted from the same motion class are very similar.

To use these key-poses, we need to build a suitable model based on the distribution probability of all poses in database. We assume that there are N poses in database and J key-poses extracted from them. Each key-pose could be described by a single Gaussian distribution $\mathcal{N}(\mu_j, \Sigma_j), j = 1, 2, \ldots, J$. The GMM is adopted, and the initial prior of each Gaussian distribution is set to be the same, $P(q_j|\Theta) = 1/J$, where q_j represents each Gaussian model and Θ is the parameter set. Then the probability of each data point $x_n (n = 1, 2, \ldots, N)$ belongs to $q_j, P(q_j|x_n, \Theta)$, can be calculated as follows:

$$\begin{aligned} P(q_j|x_n, \Theta) &= \frac{P(q_j|\Theta) \cdot p(x_n|q_j, \Theta)}{p(x_n|\Theta)} \\ &= \frac{P(q_j|\Theta) \cdot p(x_n|\mu_j, \Sigma_j)}{\sum_j P(q_j|\Theta) \cdot p(x_n|\mu_j, \Sigma_j)} \end{aligned} \tag{1}$$

To maximize the likelihood function $\Theta^* = argmax_\Theta L(X|\Theta)$, the expectation–maximization algorithm is adopted to solve the optimization problem, and the parameters (mean values μ_j, variances Σ_j and weights $P(q_j|\Theta)$) are updated as follows:

$$\mu_j^{(new)} = \frac{\sum_{n=1}^{N} x_n P(q_j|x_n, \Theta)}{\sum_{n=1}^{N} P(q_j|x_n, \Theta)} \tag{2}$$

Let $\bar{\mu}_j = \left(x_n - \mu_j^{(new)}\right)$,

$$\Sigma_j^{(new)} = \frac{\sum_{n=1}^{N} P(q_j|x_n, \Theta) \cdot \bar{\mu}_j \cdot \bar{\mu}_j^T}{\sum_{n=1}^{N} P(q_j|x_n, \Theta)} \tag{3}$$

$$P\left(q_j^{(new)} \middle| \Theta^{(new)}\right) = \frac{1}{N} \sum_{n=1}^{N} P(q_j|x_n, \Theta) \tag{4}$$

Equation 1–4 are repeated until convergence, and the key-pose model is completely trained.

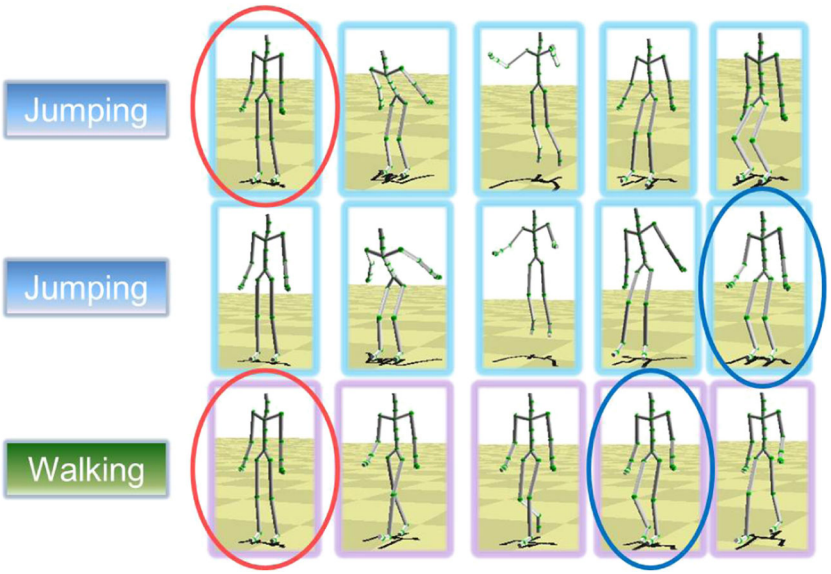


Figure 2. Examples of key-poses in three motion clips, where the first two are jumping and the last one is walking. It is evident that for the same kind of motions, almost all the corresponding key-poses are similar, but for different motion classes, only limited key-poses (two pairs of the circled) are similar.

3.2. Feature Calculation

As the key-pose model is well trained, the mean values, $\mu_j (j = 1, \ldots, J)$ of each Gaussian distribution are the key-poses extracted from the motion database. The training process is unsupervised. Given an unknown pose, the GMM-based key-pose model could tell the probabilities of each key-pose it may belong to. So the pose feature t could be represented by the probabilities p_j that is calculated by each $\mathcal{N}(\mu_j, \Sigma_j), j = 1, 2, \ldots, J$:

$$t = [p_1, p_2, \ldots, p_J] \tag{5}$$

4. ACTION STRING

4.1. Action String Generation

As explained before, the main idea of this paper is to use the 'action string', where a pose is treated as a 'character' in a 'string' of the human action. As our pose feature is high dimensional, a top-down hierarchical clustering method [24] is introduced to index it as a 'character'.

The clusters are considered as the leaf nodes in the hierarchical clustering tree. Given a user-specified error tolerance ϵ, the $l1$ distance between any two data points within a cluster must be less than ϵ. If not, the cluster is split into two clusters using k-means clustering algorithm, and the centroid position of the original cluster is saved to construct the decision tree for searching. After the aforementioned process, the decision tree can serve as the indexing model of the pose feature. Assuming that there are K clusters generated and the centroid of each cluster is $C_k, k = 1, 2, \ldots, K$, a pose could be represented by the ID of the cluster it belongs to. Each ID can be treated as a 'character', and they make up our 'action string' vocabulary $\mathbf{V} = \{1, 2, \ldots, K\}$. In addition, the action strings can be down-sampled at a fixed rate, so the string length will be changed into a suitable value.

4.2. String Matching

As human actions have been translated into action strings, a string matching algorithm is imperative to calculate the distance or similarity between two actions. Considering the special requirement from motion annotation, the string matching method should meet three targets.

(1) If one is a sub-string of the other, they must be matched. For example, the distance between '12345' and '345' is 0 or tend to be 0. In our annotation strategy, a sliding window scheme is applied for online recognition. The action string of motion data in a sliding window may not describe a complete action, but it will probably be a sub-string of a certain action.
(2) If the speed of two motions is a little different, their action strings should still be matched. For example, the distance between '112233' and '111223' is 0 or at least less than or equal to 2. As we know, the time warping problem is a crucial point in motion data recognition, because the speed of motions in the same class is never exactly the same. However, a bit difference of speed or sampling rate may cause a large variance in action string matching. Therefore, this is a very difficult point.
(3) The algorithm should tolerant small noise and missing data. For example, the distance between '12345' and '1345' or '15345' should not be greater than 1. Up to now, the motion capture devices and skeleton recovery algorithms are not robust enough, so that the generated motion data may contain small random noise or lack of data points.

We have surveyed many state-of-the art string matching algorithms [25], only a few algorithms based on dynamic programming could satisfy the three basic requirements mentioned earlier simultaneously. In addition, considering that the practical meaning of each 'character' is the ID of feature cluster, we hope that the distance between two nearby clusters is less than two faraway clusters. So we add a weight matrix as the distance function in the string matching algorithm.

Given the centroid of each feature clusters $C_k, k = 1, 2, \ldots, K$ and the vocabulary of 'character's $\mathbf{V} = \{1, 2, \ldots, K\}$, a symmetric weight matrix $\mathbf{W}$ of size $K \times K$ is calculated by the $l1$ distance of every pair of feature cluster centers. Then, for the two given strings A_1 and A_2, we first exchange them if $A_1 > A_2$. Assuming the lengths of A_1 and A_2 are n and m, a dynamic programming matrix $\mathbf{D}$ of size $(n + 1) \times (m + 1)$ is constructed. We initialize it by

$$\begin{cases} \mathbf{D}_{i,0} = \mathbf{W}_{A_1(i),A_2(1)}, & i = 1, 2, \ldots, n \\ 0, & \text{elsewhere} \end{cases} \tag{6}$$

Next, the algorithm processes the action string character by character. At each new character P_j, its column vector is updated by

$$\mathbf{D}_{i,j} = \min(\mathbf{D}_{i-1,j-1}, \mathbf{D}_{i-1,j}, \mathbf{D}_{i,j-1}) + \mathbf{W}_{A_1(i),A_2(j)} \tag{7}$$

for every $j = 1, 2, \ldots, m$. At last, the bottom-right element $\mathbf{D}_{n,m}$ of the dynamic programming matrix $\mathbf{D}$ is the distance between string A_1 and A_2. The time complexity of this algorithm is $O(mn)$.

For example, given the vocabulary $\mathbf{V} = \{1, 2, 3, 4, 5\}$, the weight matrix $\mathbf{W}$, and the two strings $A_1 = \{3, 3, 4, 5, 5\}$ and $A_2 = \{1, 1, 2, 2, 3, 3, 4, 4, 5, 5\}$, the calculation result is shown in Figure 3. The dynamic programming matrix $\mathbf{D}$ is calculated by the algorithm described earlier, and the path to the final result is shown as the bold entries.

DOI: 10.1002/cav

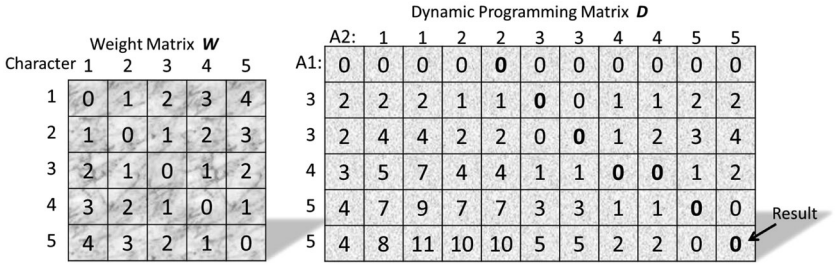


Figure 3. An example of the string matching algorithm.

4.3. Hierarchical Action String Structure

Based on the string matching method, we can retrieve the motion clips by calculating their similarities. Then, the K nearest neighbor algorithm is used to classify the input motion. However, if the database is very large, calculating the similarity with each motion in database is time-consuming, which is inappropriate for real-time annotation applications. So, in this paper, a bottom-up hierarchical action string structure is used to search and label an unknown action string in real-time.

An agglomerative strategy is applied to build the hierarchical structure of action string clusters, which iteratively merge the nearest two clusters into a combined one until the distance between all cluster centers falls out of a threshold σ. In detail, the process of building the hierarchy is shown in the following text.

(1) Initialization: for all the N motion clips in database, each corresponding action string $A_i, i = 1, 2, \ldots N$, starts in one cluster L_{A_i}, which contains only A_i itself. And its centroid position T_{A_i} is set to be A_i.
(2) Calculate the distance matrix $\mathbf{M}$, where $\mathbf{M}_{i,j} = stringMatch\left(T_{A_i}, T_{A_j}\right)$.
(3) Find the minimal distance in $\mathbf{M}$ and the corresponding two strings A_{k1} and A_{k2}.
(4) If $\mathbf{M}_{k1,k2} > \sigma$, return all clusters and their centroid positions. The algorithm terminates. Otherwise, merge the two clusters $L_{A_{k1}}$ and $L_{A_{k2}}$ into a new cluster L'.
(5) Choose the new centroid position T' of the new cluster L'. $\forall A_i \in L'$, calculate $D_i = max(stringMatch(A_i, A_j)), \forall A_j \in L'$, and $i' = argmin_i D_i$, then $T' = A_{i'}$.
(6) Update the distance matrix $\mathbf{M}$ as procedure 2. The distance within the same cluster is set to be $+\infty$.
(7) Repeat procedures 3–6, until end condition.

Based on the hierarchical action string structure, the motions in database can be represented by a number of selected action strings. Given an input action string, only the selected strings will be compared with it. Hence, the time spent on string matching will be negligible. Moreover, for denoising, the clusters containing only one action string will be discarded.

5. EXPERIMENTAL RESULT

Our method was implemented in MATLAB, and all experiments were executed on a computer with an Intel Core i5 2400 3.1 GHz and 4 GB of RAM.

5.1. Motion Data Recognition

In this part of the experiment, we use well-labeled motion data from HDM05 [26], a commonly used public mocap database, to prove the efficacy of our method in motion recognition. The original well-segmented motion cuts contain 130 motion classes, but many of them are very close (e.g., 'walk2StepsLstart' and 'walk2StepsRstart'). We manually combined those classes into 25 basic motion classes as the dataset in our experiment. A half of the total 2345 motion clips are served as training data, and the others are used as testing data.

In our experiments, we have adopted the empirical values for the parameters $J = 40, \epsilon = 1$ and $\sigma = 6$. We first compare the recognition accuracy with and without (only use PCA to reduce the dimensionality) our GMM pose feature. Because the DTW algorithm is one of the most frequently used method in motion data matching, we also compare the DTW algorithm with ours, as shown in Figure 4. It is clear that the combination of GMM pose feature and string matching gives the best performance compared with other combinations.

Next, we compare our method with two other algorithms (Liu *et al.* [9] and MT [21]). As shown in Figure 5, our method outperforms the other two in terms of the accuracy of motion recognition. In the work of Liu *et al.* [9], the cluster transition trajectory of each motion is used for indexing and searching the motion database. However, there is no pose feature extracted from the original motion data. This is why its recognition accuracy is lower than ours. When it comes to MT, we think it is abnormal for that terrible result. By analyzing the training process of MT, we found that if the training data in the same class has different cycles (e.g., 'walk2StepsLstart' and 'walk4StepsLstart'), the MT could not be well trained. For a fair comparison, we recut the original 130 motion classes into a new 80 classes, where the same action with different cycles is separated and the experiment is redone on the new dataset. At this time, MT outperforms ours by about 15%

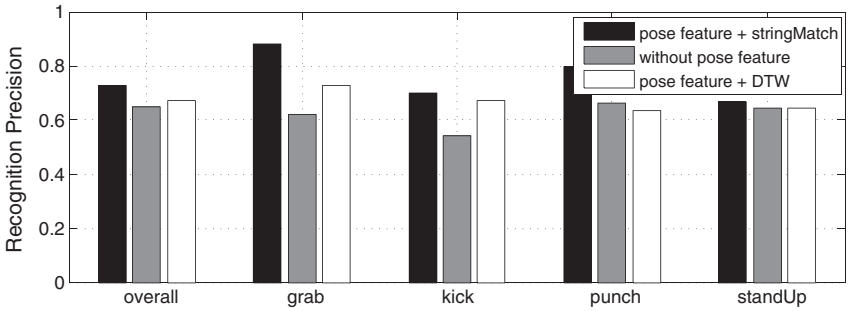


Figure 4. The comparison among our method (pose feature + string matching algorithm), our method without GMM pose feature, and our pose feature with another string matching algorithm dynamic time warping.

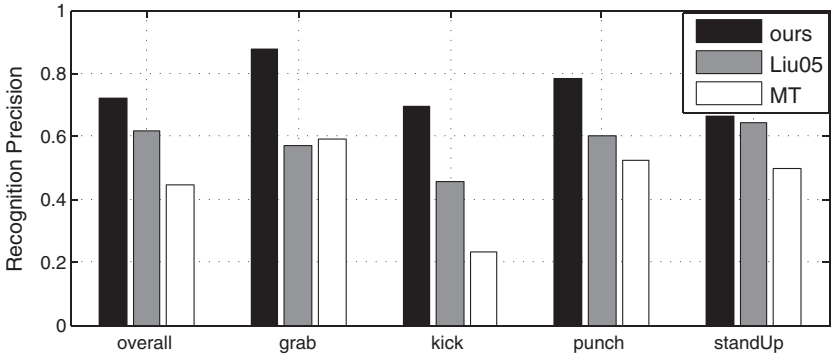


Figure 5. The comparison of our method, Liu05, and MT.

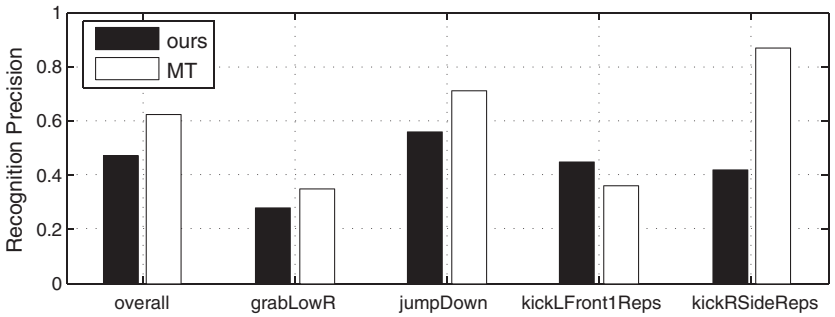


Figure 6. The comparison is redone on a new dataset, which is favorable for MT. The different performance shows the big limitation of MT on training data.

in recognition accuracy (shown in Figure 6). However, because of this special requirement on the training set, it significantly limits the applications of the MT method in practice.

Finally, the time consumed in each step of our method is shown in Table I. In the testing phase, the total time for a motion clip is about 150 ms. If the sliding window scheme is applied, it is therefore fast enough for real-time interactive systems. The time consumed in training models is also reasonable, making our method work well for the automatic annotation of large motion data repository.

5.2. Real-time Human Action Annotation

For real-time annotation, a sliding window scheme is introduced with a window size of 2 s and a step length of 0.5 s. The database we used for training is the same as before, consisting of 25 motion classes from HDM05 database. Some cross-category motion clips by new actors are recorded with Kinect as the unlabelled and unsegmented input motions. Given such a motion, the sliding window scheme is applied to cut it into short pieces in real-time, and then for each piece, our method is used to

DOI: 10.1002/cav

Table I. Time consumed for each step of our method.

	Training (s)	Testing stage (ms)
Gaussian Mixture Model pose feature model construction	1217	—
Feature calculation	27.45 (total)	26.4 (per clip)
Feature indexing model construction	33.86	—
Action string calculation	0.53 (total)	0.52 (per clip)
Action string structure construction	341.3	—
Action string recognition	—	123.2 (per clip)
Total	1620	150.1 (per clip)

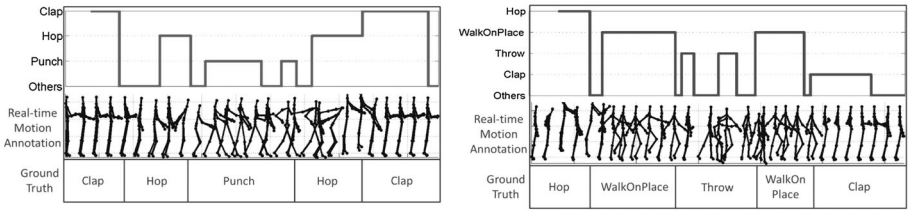


Figure 7. The online human action annotation results of two different motion clips.

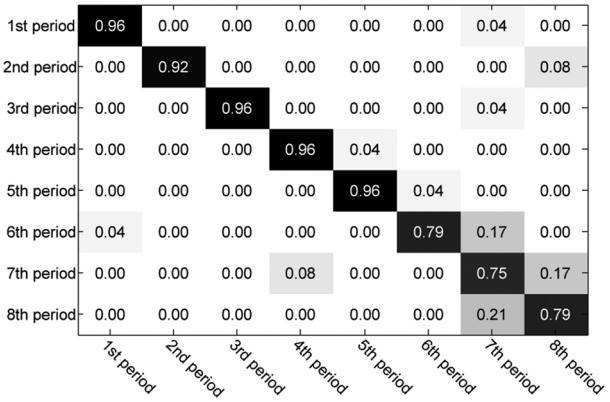


Figure 8. The confusion matrix of our method for specific motions including eight different classes.

return a label. In Figure 7, our annotation result is shown at the top and the ground truth at the bottom. It can be seen that our method is very robust in action recognition and segmentation. More details will be given in our demo videos.

5.3. Human Action Recognition for Other Type of Motions

The experiments earlier are based on an open database, which prove the efficacy and real-time performance of our method. To demonstrate the feasibility of our method for practical applications, we further challenge our method with a very complicated motion set from real life. The new data set consists of eight different motion classes from the Chinese 8th Broadcast Exercises (a motion including eight different periods) by six actors, which is recorded by Kinect (the detailed motions are shown in demo). Half of the motions are served as training data, and the others are testing data. The experimental result (in Figure 8) shows the confusion matrix of recognition precision rate. The precision rate is over 90% for the first five periods. It is relatively low (still over 75%) in No. 6–8 periods because our input mocap data are not very stable from fast motions. Besides, many poses in No. 8 period is very similar to No. 7 period, so they are comparatively hard to recognize. In general, our method maintains a high recognition precision rate for this practical problem. For other types of human motions, as long as there is a complete training database, our method will perform well.

6. CONCLUSION

In this paper, we have presented a novel motion annotation method that can automatically recognize motion classes in real-time. We first calculate a GMM-based pose feature to represent each pose. Each motion clip is then translated into an action string by clustering the pose

features. By introducing a string matching method, we are able to calculate the distance between action strings. As the final step, we construct a hierarchical action string structure to search and label action strings in real-time. The experiments have demonstrated that our method is robust and immune to time warping problems. Apart from motion annotation, our method can also be used to evaluate and score human motions by comparing the captured motion with the 'perfect motion' in training set. It can be used for sports training, patient rehabitation, and many other applications.

In our current system, the dimensionality of the GMM model (the number of key-poses) J is a crucial variable that influences the performance of annotation. In the future work, we will try to optimize the pose feature to enhance the stability of our method. For recognition, the string matching algorithm can also be improved to compare combined action strings.

ACKNOWLEDGEMENTS

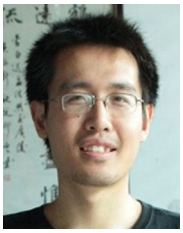
This research is supported by the National High Technology Research and Development Program (2012AA011502), the National Key Technology R&D Program (2013BAH59F00), the Zhejiang Provincial Natural Science Foundation of China (LY13F020001), the Fundamental Research Funds for the Central Universities (2014FZA5013), Zhejiang Province Public Technology Applied Research Projects (No. 2014C33090) and partially supported by the grant of the Sino-UK Higher Education Research Partnership for PhD Studies Project funded by the Department of Business, Innovation and Skills of the British Government and Ministry of Education of China.

REFERENCES

1. Heck R, Gleicher M. Parametric motion graphs. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, Seattle, USA, 2007; 129–136.
2. Jain S, Ye Y, Liu C. Optimization-based interactive motion synthesis. *ACM Transactions on Graphics (TOG)* 2009; **28**: 1–12.
3. Kovar L, Gleicher M, Pighin F. Motion graphs. *ACM transactions on graphics (TOG)* 2002; **21**: 473–482.
4. Gleicher M. Motion editing with spacetime constraints. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, Providence, USA, 1997; 139–148.
5. Min J, Liu H, Chai J. Synthesis and editing of personalized stylistic human motion. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, Bethesda, USA, 2010; 39–46.
6. Hecker C, Raabe B, Enslow R, DeWeese J, Maynard J, Prooijen K. Real-time motion retargeting to highly varied user-created morphologies. *ACM Transactions on Graphics (TOG)* 2008; **27**(3): 27.
7. Xsens, 2013. Available from: http://www.xsens.com.
8. Microsoft. Kinect 2013. Available from: http://www.xbox.com/en-US/kinect.
9. Liu G, Zhang J, Wang W, McMillan L. A system for analyzing and indexing human-motion databases. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, Baltimore, USA, 2005; 924–926.
10. Poppe R. A survey on vision-based human action recognition. *Image and Vision Computing* 2010; **28**(6): 976–990.
11. Li W, Zhang Z, Liu Z. Action recognition based on a bag of 3D points. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, San Francisco, USA, 2010; 9–14.
12. Ni B, Wang G, Moulin P. RGBD-HuDaAct: a color-depth video database for human daily activity recognition. In *Computer Vision Workshops (ICCV Workshops)*, Barcelona, 2011; 1147–1153.
13. Wang J, Liu Z, Wu Y, Yuan J. Mining actionlet ensemble for action recognition with depth cameras. In *Computer Vision and Pattern Recognition (CVPR)*, Providence, USA, 2012; 1290–1297.
14. Yang X, Zhang C, Tian Y. Recognizing actions using depth motion maps-based histograms of oriented gradients. In *Proceedings of the 20th ACM International Conference on Multimedia*, New York, USA, 2012; 1057–1060.
15. Baak A, Muller M, Bharaj G, Seidel H-P, Theobalt C. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Consumer Depth Cameras for Computer Vision*. Springer: London, 2013; 71–98.
16. Shen W, Deng K, Bai X, Leyvand T, Guo B, Tu Z. Exemplar-based human action pose correction and tagging. In *Computer Vision and Pattern Recognition (CVPR)*, Providence, USA, 2012; 1784–1791.
17. Shotton J, Fitzgibbon A, Cook M, Sharp T, Finocchio M, Moore R, Kipman A, Blake A. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition (CVPR)*, Providence, USA, 2011; 1297–1304.
18. Forbes K, Fiume E. An efficient search algorithm for motion data using weighted PCA. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Los Angeles, USA, 2005; 67–76.
19. Meng J, Yuan J, Hans M, Wu Y. Mining motifs from human motion. In *Eurographics 2008 Short Papers*, Grete, Greece, 2008; 71–74.

20. Vieira AW, Lewiner T, Schwartz WR, Campos M. Distance matrices as invariant features for classifying mocap data. In *International Conference on Pattern Recognition (ICPR)*, Tsukuba, JP, 2012; 2934–2937.
21. Muller M, Roder T. Motion templates for automatic classification and retrieval of motion capture data. In *Proceedings of the 2006 ACM SIGGRAPH/ Eurographics Symposium on Computer Animation*, Vienna, Austria, 2006; 137–146.
22. Muller M, Baak A, Seidel HP. Efficient and robust annotation of motion capture data. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, New York, USA, 2009; 17–26.
23. Qi T, Feng Y, Xiao J, Zhuang Y, Yang X, Zhang J. A semantic feature for human motion retrieval. *Computer Annimation and Virtual Worlds* 2013; **24**(3-4): 399–407.
24. Defays D. An efficient algorithm for a complete link method. *The Computer Journal (British Computer Society)* 1977; **20**(4): 364C–366.
25. Navarro G. A guided tour to approximate string matching. *ACM Computing Surveys (CSUR)* 2001; **33**(1): 31–88.
26. Müller M, Röder T, Clausen M, Eberhardt B, Krüger B, Weber A. Documentation mocap database HDM05. *Technical Reports CG-2007-2*, Universität Bonn, Bonn, Germany, June 2007.

AUTHORS' BIOGRAPHIES



Tian Qi obtained his Bachelor's degree from the Department of Computer Science at Zhejiang University (ZJU, 2010), majoring in Computer Science and Engineering. Now, he is a third-year Ph.D. student majoring in Computer Science and Technology at ZJU. His research fields include computer animation and computer vision.

Jun Xiao obtained his Ph.D. degree from the College of Computer Science at ZJU (2007), majoring in Computer Science and Technology. Now, he is an Associate Professor of the College of Computer Science at ZJU, working in the Microsoft Visual Perception Lab of ZJU. His research fields include computer animation, digital entertainment and multimedia retrieval.

Yueting Zhuang received his Ph.D. degree from the Department of Computer Science at ZJU (1998). From 1997 to 1998, he worked as a visitor at the Department of Computer Science and Beckman Institute of University of Illinois at Urbana-Champaign. He is now a Professor and Vice Dean of the College of Computer Science at ZJU. His current research interests are computer animation, multimedia analysis, digital entertainment and digital library technology.

Hanzhi Zhang obtained his Bachelor's degree from the Department of Software Engineering and Microelectronics at Northwestern Polytechnical University (2012), majoring in Software Engineering. Now, he is a first-year Master's student majoring in Computer Science and Technology at ZJU. His research fields include computer animation and computer vision.

Xiaosong Yang is a Senior Lecturer in the National Centre for Computer Animation, The Media School, Bournemouth University, UK. He received his Bachelor's (1993) and Master's degrees (1996) in Computer Science from ZJU (China) and Ph.D. (2000) in Computing Mechanics from Dalian University of Technology (China). He worked as a Postdoc (2000-2002) in the Department of Computer Science and Technology of Tsinghua University for 2 years and as a Research Assistant (2001-2002) at the 'Virtual Reality, Visualization and Imaging Research Centre' of the Chinese University of Hong Kong. His research interests include 3D modeling, animation, real-time rendering, virtual reality, virtual surgery simulation and computer-aided design.

Jianjun Zhang is currently a Professor of Computer Graphics at the National Centre for Computer Animation, Bournemouth University, UK, where he leads the Computer Animation Research Centre. He is also a cofounder of UK's Centre for Digital Entertainment, which received an initial funding of over £6m from the Engineering and Physical Sciences Research Council. His research focuses on a number of topics relating to 3D virtual human modeling, animation and simulation, including geometric modeling, rigging and skinning, motion synthesis, deformation and physics-based simulation.

Yinfu Feng received his B.S. in Information Security from the University of Electronic Science and Technology of China, Chengdu, China, in July 2009. Now, he is a fourth-year Ph.D. student majoring in Computer Science and Technology at ZJU. His current research interests include multimedia analysis and retrieval, computer vision and machine learning.

Comp. Anim. Virtual Worlds 2014; **25**:293–302 © 2014 John Wiley & Sons, Ltd.
DOI: 10.1002/cav