Motion Retrieval Based on Energy Morphing

Gary K.L. Tam[1] Qingzheng Zheng[1] Mark Corbyn[2] Rynson W.H. Lau[1]
Department of Computer Science, University of Durham, United Kingdom
{g.k.l.tam, qingzheng.zheng, rynson.lau}@durham.ac.uk[1] markcorbyn@lycos.co.uk[2]

Abstract

Matching and retrieval of motion sequences has become an important research area in recent years, due to the increasing availability and popularity of motion capture data. The main challenge in matching two motion sequences is the diversity of the captured motions, including variable length, local shifting, local and global scaling. Most existing methods employ Dynamic Time Warping (DTW) or Uniform Scaling to handle these problems. In this paper, we propose a novel content-based method for matching of this human motion captured data. We convert the matching problem of motion capture data into a transportation problem. To solve this problem efficiently, we employ Earth Mover's Distance (EMD) as the matching framework. To penalize any strayed matching, we provide a ground distance that works similar to Sakoe-Chiba band of DTW. Empirical results obtained are encouraging.

1. Introduction

In recent years, computer animation has become very popular due to the increasing importance of it in many applications. It is used not only to produce animated films, but also in computer games, simulations such as crowd and environmental simulations, and virtual environments. In computer animation, we are particularly interested in human motion as human is the species that we most frequently encounter in our daily life. As such, we can easily recognize any subtle changes of human motion. Hence, there is a high demand to produce very realistic motion to model human movement. In interactive applications, such as computer games and virtual environments, there is an additional requirement – the motion must be computed or obtained within a very short period of time in order to maintain the interactivity of the application.

There are a number of methods developed to produce human motion data. A well-known method is called motion capture (MoCap). In this method, motion sensors are attached to various parts of a human actor. As the human actor performs some specific movement, the computer records the position of each sensor at each time frame. This creates a series of motion frames, known as time-series. With this method, real human motions can be captured and used to drive the movement of virtual characters. We may also store these motion data as motion files to form a motion database. With the motion capture devices becoming more widely available, large motion databases start to appear [6]. By joining several time-series together, more complex human motions can be created [14, 15]. However, as the number of motions in the database grows, it becomes difficult to select an appropriate motion that satisfies certain requirements. As such, motion retrieval has become one of the major research focuses in motion capture animation in recent years.

To design a reliable motion retrieval method, there are many challenges to address. First, a human model contains many joints and human animation consists of many time series, each representing the motion of a single joint. A method should be fast enough to enable interactive retrieval of matched motions despite the size of the database. Second, similar motions may still be differed in many ways, such as length, local shifting, local and global scaling, which affect the matching accuracy. While most existing methods use DTW or Uniform Scaling to solve the matching problem, we consider another method to handle this problem in this paper.

We have observed that although motion sequences may consist of local shifting, local and global scaling, they are rather similar in shape. To match these sequences, we propose to model the matching problem as a transportation problem. In other words, we consider the matching problem as the calculation of the minimum amount of energy required to morph one motion to another. An efficient and accurate method to calculate such morphing is called Earth Mover's Distance (EMD). EMD has been extensively used in matching and retrieval of multimedia data, such as images [22] and 3D geometry models [24]. However, because EMD allows morphing from any frame of a motion to any frame of another motion, strayed matching may result. To solve this issue, we propose a ground distance that penalizes strayed matching. Our experimental results show that the proposed method is promising.

The rest of this paper is organized as follows. Section 2 summarizes existing motion matching and retrieval methods. Section 3 presents our method in detail. Section 4 evaluates the performance of our method through a number of experiments. Section 5 evaluates the experimental results and Section 6 briefly concludes this paper.

2. Related Work

Motion retrieval research is still relatively new compared to retrieval research of other multimedia data. There are only a few motion retrieval methods in the literature. A number of them are extended from or strongly related to existing audio retrieval methods. A good example is the Dynamic Time Warping (DTW) method. DTW is a signal processing technique used to find a non-linear alignment between two signals, while minimizing the error between them. Many audio and speech processing algorithms as well as music retrieval methods apply DTW extensively. As motion data can be considered as multi-attribute time-series, applying DTW on motion retrieval is straightforward. Many motion retrieval systems use DTW as the similarity measure [14, 2, 18, 5, 11].

However, DTW has low efficiency. In addition, as motion capture data consists of many parameters and attributes, dimension reduction methods are typically employed to reduce computational complexity. GEMINI framework, proposed by [10], was one of the earliest examples. It first extracts a low-dimensional approximation for each time-series in the database. Then a distance metric is defined to lower bound the approximation. Usually these low-dimensional approximations are stored in a spatial data structure, like R-Tree, for fast retrieval. There are many dimension reduction methods, which all adapt similar framework. Notable examples include Fourier transform [1], wavelet transform [3], average values in adjacent windows [4] and bounding boxes [25]. In order for DTW to support indexing, [12] proposes the bounding envelopes, similar to the GEMINI framework. [13] further suggests that similar motions which are differed by uniform scaling cannot be matched by DTW because DTW can only handle subtle local differences after the best globally scaled match has been found [8]. Hence, [13] proposes an algorithm which is based on uniform scaling to match the query to those globally scaled candidate motions. However, based on the authors' understanding, in order to handle motions that contain both local and global differences, one needs to apply DTW and Uniform Scaling separately, significantly increasing the computational cost.

Recently, a geometry based method is proposed [19] by extracting boolean features from geometric relationships of motion data. These qualitative features are compact and descriptive. [20] extends the work by introducing a linear-space indexing structure, fuzzy queries and adaptive fuzzy hits. Although these features are descriptive, they require textual descriptions to be used as queries, and users need to define specific geometric features for comparison. Subsequent work [21] considers the method for the creation of motion templates. [7] provides a GUI that eliminates the need for textual description. On the other hand, [17] proposes to segment and cluster geometric features automatically into an indexing tree, and a matching algorithm based on peak points. However, in general, these methods concern finding logically similar motions but not a close match which is essential for concatenating two motions.

Our method also considers dimensionality reduction. However, we extract features based on the Douglas-Peucker algorithm, which was originally proposed as a line simplification method to reduce number of points in data representation [9]. Our method differs from all previous methods in that we do not necessarily require a fixed number of dimensions. This results in the advantage that our method can best preserve the original motion. This is particularly good for joints such as hands, where vigorous motions are expected and we may use more features to represent them. On the other hand, for those stationary joints such as head, our method extracts only a few features and thus reduces computational cost.

In this paper, we are interested in finding motions that are entirely similar to a given query. While a typical motion retrieval method may only handle local shifting, local scaling or global scaling, we try to tackle the same problem from another point of view. We convert the matching into a transportation problem. We compute the minimum energy to morph one motion sequence to another using EMD. As this paper focuses on accuracy analysis, we compare mainly with DTW and Uniform Scaling method. Though we have not implemented any indexing scheme, extending our method to support indexing can be easily achieved because our distance function is a metric [24].

3. Method Overview

Suppose we want to determine the similarity between two motion sequences,

$$Q = q_1, q_2, \ldots, q_N$$

$$C = c_1, c_2, \ldots, c_M$$

where Q is the query motion of length N and C is the candidate motion of length M. It is important that the

similarity method considers each feature of Q and C when comparing the data, where the features are extracted from Q (or C) and are used to represent Q (or C). Obviously, different similarity methods use different features for comparison. For example, Euclidean distance compares the distance of the joint position (or angle) in each frame of Q and C. The number of features is therefore equal to the number of frames. This implies a very high dimensionality of the feature space. Hence, it is favorable to reduce the dimensionality of this space and hence the computational complexity. On the other hand, we also need to minimize the data loss during this dimensionality-reduction process. As we reduce the dimensionality of the feature space, we must exploit properties of the data to facilitate this procedure.

We propose a new approach to solving the human motion retrieval problem. To reduce the dimensionality of the feature space, we apply the Douglas-Peucker curve trimming algorithm. This reduction technique gives a very tight approximation of the time-series data for each joint rotation. By considering each Douglas-Peucker point as pieces of mass and holes in space, we can effectively apply the EMD method to the point sets. The EMD value returned provides a similarity score between the query and candidate. Applying this technique to every joint of the human skeleton, the summation of the values provides a score for the overall skeleton animation. In our implementation, we weight the joint scores according to their perceptual importance. For example, the arm rotations receive a higher weighting than the finger rotations since they are generally perceptually more important.

3.1. Feature Extraction

In reducing the dimensionality of the features (i.e., frames) from N in n features, $n < N$, we apply the Douglas-Peucker (D-P) algorithm to the motion data. In human motion data, the joint rotation value from one frame is generally similar to its predecessor and successor frames. This property of motion data allows the application of D-P approximation.

3.1.1. The D-P Algorithm

Commonly used in data compression, the D-P algorithm is a recursive procedure that represents time-series data using a series of straight lines. Given a time-series Q and a value ε, the D-P algorithm begins with a single line with end points at q_1 and q_N. For each frame in Q, D-P computes the distance from the line, storing the maximum distance of any point from a line. If the maximum distance is greater than ε, from the line, the line is split into two lines and the D-P algorithm is then called on the two line segments produced. In this manner, the algorithm recursively splits the line until there are no frames greater than ε from some line. Figure 1 illustrates this operation.

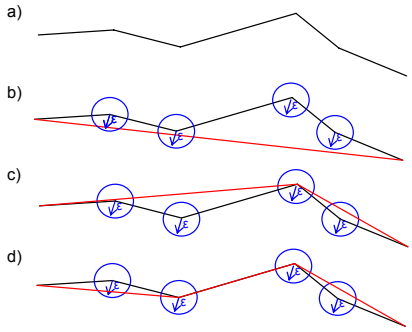

Figure 1. Applying the recursive D-P operation on time-series data.

3.1.2. Feature Representation

To apply the D-P algorithm here, each line segment representing the motion of a joint in time contains a start point and an end point. Each point (or *feature* in our method) contains an x, y and z rotation value, along with a t value indicating the time position of that point, i.e., the frame number, and an m value indicating the number of frames it encapsulates (or the *mass* as will be explained in Section 3.2). We define the i-th feature as follows:

$$<x_i, y_i, z_i, t_i, m_i>$$

Hence the feature set of one joint motion becomes:

$$F = \{ <x_1, y_1, z_1, t_1, m_1>, \ \dots \ , <x_n, y_n, z_n, t_n, m_n>\}$$

Note that the D-P algorithm is a lossy compression algorithm and therefore some data loss is unavoidable. The accuracy of the approximated data depends on the number of features used to approximate it, and thus the threshold ε.

3.2. Feature Extraction

For feature matching, we apply the EMD method on the features extracted by the D-P algorithm. To the authors' knowledge, the EMD method has never been applied to human motion matching before, although it has been applied to image retrieval [22] as well as model retrieval [24] with considerable success. By applying the EMD to the extracted features of each joint in the human body, the summation of results for each joint provides an overall dissimilarity score for the entire skeleton. As mentioned earlier, we weight each joint according to its perceptual importance. Hence, given the extracted features of the query, F_q, and that of the candidate, F_c, the dissimilarity score is computed as follows:

$$D_{EMD}(F_q, F_c) = \sum_{j=1}^{|J|} W_j \cdot \mathrm{EMD}(F_{q,j}, F_{c,j}) \quad (1)$$

where W_j is the weight of joint $j \in J$.

3.2.1. The EMD Method

The EMD method is based on the Transportation Problem. Consider representing the features of the query as pieces of mass in space and the features of the candidate as holes in space. The concept behind the EMD method is to calculate the minimum energy required to move the pieces of mass (or earth) to the holes to completely fill the holes.

Transporting a heavy piece of mass a long distance generally requires more energy than transporting it a short distance. Therefore it is beneficial to find a solution to transport the pieces as short a distance as possible. The energy required to move a piece of mass to a hole is called the *flow*. The total energy to move all pieces of mass is called the *total flow*. In the EMD we try to find an optimal total flow.

More formally, the EMD method computes the minimum energy to move a set of masses, P, to a set of holes, Q, as follows:

$$EMD(P,Q) = \frac{\sum_{i=1}^{m} \sum_{k=1}^{n} d_{ik} f_{ik}}{\sum_{i=1}^{m} \sum_{k=1}^{n} f_{ik}} \quad (2)$$

where d_{ik} is the energy required to transport one unit of mass from feature i of the query to feature k of the candidate. f_{ik} is the amount of flow transported from feature i of the query to feature k of the candidate.

To calculate d_{ik}, a Ground Distance metric is required. For each feature i of the query and feature k of the candidate, we compute a Ground Distance value. All the computed values are stored in a *cost matrix*. Figure 2 shows an example of this operation. After the cost matrix has been produced, a flow matrix is computed containing the flow from feature i of the query to feature k of the candidate for all i and k. This flow matrix is iteratively optimized to find the optimal total flow.

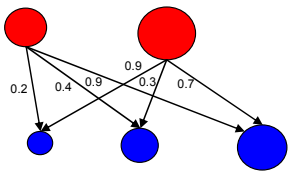

Figure 2. Example *ground distance* between each pair of features, producing the cost matrix.

3.2.2. Integration into EMD

In the EMD framework, we are allowed to specify the *mass* of each feature. It is a value that describes how important the feature point is. The larger this value, the higher the importance. To explain it in physical terms, we can indicate the mass of a feature as the size of a circle as shown in Figure 2. In our implementation, the mass is stored as one of the components in a feature, $F[m_i]$, as described in Section 3.1.2.

To compute the ground distance, d_{ik}, to transport one unit of mass from one location to another, we consider the Euclidean distance between each dimension of a feature point, x, y, z and t as follows:

$$d_{ik} = \sqrt{\begin{aligned}&(F_q[x_i] - F_c[x_k])^2 + (F_q[y_i] - F_c[y_k])^2 \\ &+ (F_q[z_i] - F_c[z_k])^2 + (F_q[t_i] - F_c[t_k])^2\end{aligned}} \quad (3)$$

where F_q and F_c are the features extracted by the D-P algorithm from the query and the candidate, respectively, for one joint.

Using the cost matrix constructed, the optimization of the flow matrix may help determine the similarity between the query and the candidate. For example, if a query contains a section where a subject raises his hand and then lowers it, and the candidate also contains similar motion, then there would be a high flow between these features requiring a low amount of energy. On the contrary, if the candidate does not contain this motion, the features of the query must be transported to some feature(s) of the candidate using a high amount of energy. This results in a higher overall energy transfer, indicating a poor similarity between the two motions.

3.2.3. Comparison to the Sakoe-Chiba Band

In our method, we penalize the transfer of a piece of mass to a hole with a different start frame t. This produces a similar effect as the Sakoe-Chiba band, which is widely used in Dynamic Time Warping.

There are two important properties in applying the Sakoe-Chiba band in DTW. First, it prevents frame matching from straying too far from the diagonal of the matrix, i.e., matching frames which lie outside the Sakoe-Chiba band. Second, it can provide a significant speed improvement. This is achieved by restricting the frame matching computation process to only those frames that lie within the Sakoe-Chiba band [23].

By considering the t value in our method we *restrict* frame matching to within the diagonal region. This is similar to the first property of the Sakoe-Chiba band. However, while the Sakoe-Chiba band implements a sharp cut-off point beyond which frames cannot be matched, the weighting in our Ground Distance provides a gradual restriction (Figure 3). As a result, our method does not have the speed

improvement as the Sakoe-Chiba band does because our method needs to compute the entire cost matrix.

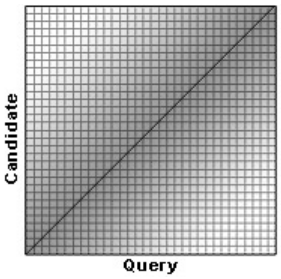

Figure 3. Penalizing straying from the diagonal, similar to that of the Sakoe-Chiba band.

4. Experiments and Results

To evaluate the performance of the proposed matching method on different motion clips, we discuss some of the experiments that we have conducted. We have constructed a small motion database from 115 different motion clips. We categorize the 115 motions into 5 motion groups, climbing, jumping, running, sword playing and walking. All the experiments presented here are performed on a PC with a Pentium 4 3GHz CPU and 1GB RAM. The motion files are downloaded from CMU [6].

The motion clips typically contain more than one action within each clip. To obtain more accurate performance results, we manually break each of the clips down into basic motion clips with a single action. The basic motion clips have different lengths varying from 150 to 600 frames. We down sample all the motion clips into 128 frames while ensuring that the down sampled motion clips do not have apparent artifacts. In our experiments, DTW and our method use only these down sampled motion clips as input. However, in order to allow scaling in Uniform Scaling, we use the basic motion clips as input and we take the first 128 frames of the basic motion clips as the query for scale computation. This is due to the fact that down sampling may produce the effect of scaling. Our objective in the experiments is to find the most similar motion clips within the motion database. For comparison, we have implemented Dynamic Time Warping and the Uniform Scaling method. In our DTW implementation, we use a Sakoe-Chiba Band of width r=10% of the basic motion to control local shifting and scaling. In Uniform Scaling, we use a maximum scaling factor of 1.2 to search for the best suitable scale. For time comparison, we also apply bounding envelopes for fast pruning of irrelevant motions.

4.1. Performance on Motion Discrimination

In the first experiment, we compare the retrieval performance of the three methods, DTW, Uniform Scaling and our method, using the similarity matrix. To generate the matrix, we first compute the similarity score between every motion pair in our database. We then normalize the results with the maximum and minimum of the corresponding matrices to show the contrasts. The darker the color, the more similar the two motions are.

Figure 4 shows the similarity matrices of the three methods. To improve readability, we have clustered the similarity results according to the five motion groups. They are labeled as well as highlighted with five different colored squares in the diagrams. From the similarity matrices, we have the following observations:

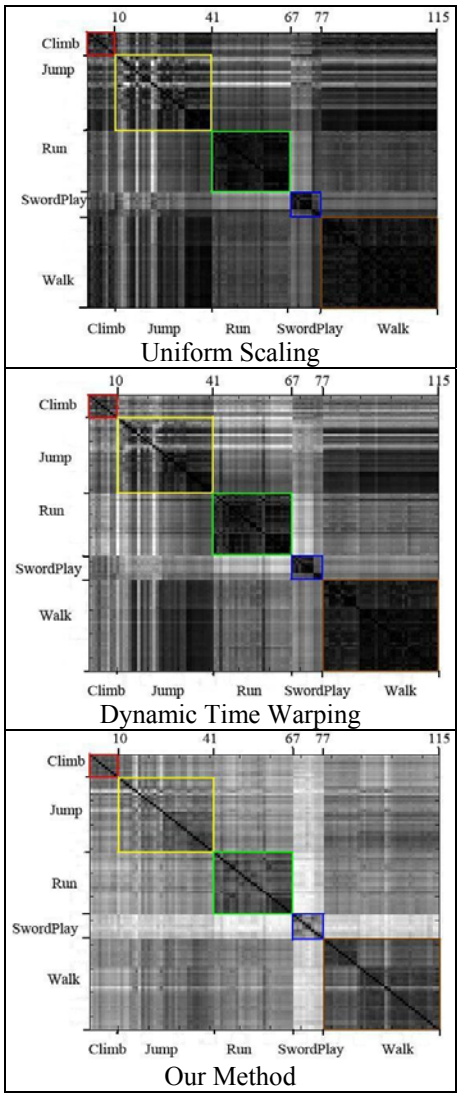

Figure 4. Similarity matrix of three different methods.

1. The diagonal lines of the three matrices give the darkest color. This means all three methods perform well in identifying the same motion.
2. In general, the results within the five colored squares show dark color for the three matrices. This means that all three methods are able to give high similarity scores for similar motions.
3. Our method also gives a larger similarity contrast than DTW and Uniform Scaling when comparing two motions from different groups, as the similarity matrix of our method generally gives lighter colors outside the five square regions. This may suggest that our method can distinguish different motion groups relatively easier with high contrast.

4.2. Performance on Motion Retrieval

In the second experiment, we compare the retrieval performance of the three methods using the average precision and recall graph (Figure 5). This graph is generated by taking each of the motions in the database as query, searching similar motions from the same database and averaging all the precision and recall values. The precision and recall values are computed as follows:

$$\text{precision} = \frac{\left|\{\textit{relevant objects}\} \cap \{\textit{retrieved objects}\}\right|}{\left|\{\textit{retrieved objects}\}\right|}$$

$$\text{recall} = \frac{\left|\{\textit{relevant objects}\} \cap \{\textit{retrieved objects}\}\right|}{\left|\{\textit{relevant objects}\}\right|}$$

Figure 5 shows our precision and recall results. From the diagram, our method performs much better than the other two methods because our method always gives a better precision for any given recall, especially when the recall is above 0.1. This finding confirms our similarity analysis that our method can distinguish dissimilar motions. Figure 6 plots the precision and recall for each of the motions in our database.

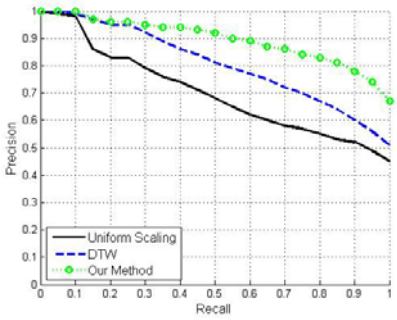

Figure 5. Average Precision Recall.

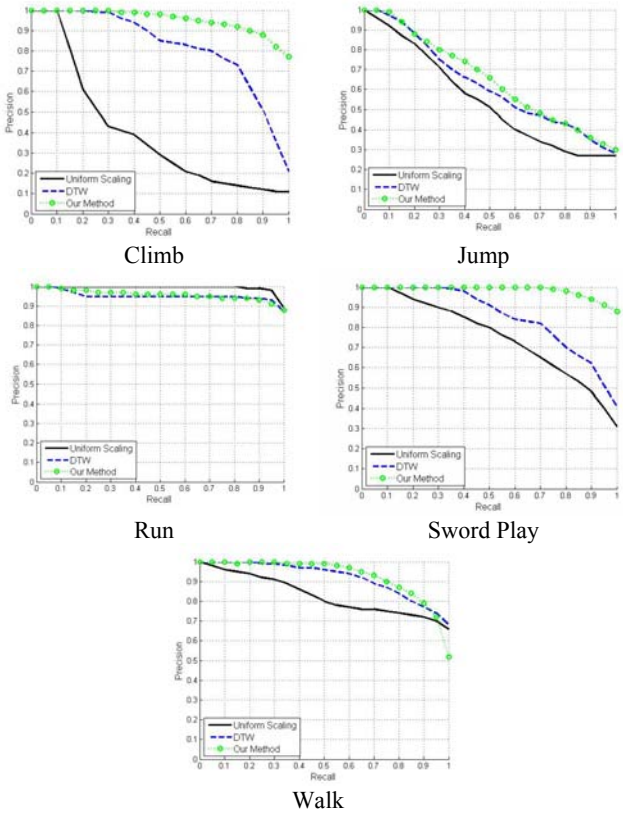

Figure 6. Individual Precision Recall.

From Figure 6, we can see that all three methods perform very well in the run motion, whilst our method performs better in the climb, jump, sword play and walk motions. All three methods have similar good performance in the run motion may suggest that all these run motions are highly similar to each other within the group and are very distinctive from motions of other groups. This also accounts for the high precision (>=0.9) at high recall (1.0).

Currently we can only perform precision and recall analysis on five groups of motion data because most data available at [6] consist of many different kinds of motions and cannot be manually classified into a single meaningful group.

4.3. Speed Comparison and Complexity Analysis

In the third experiment, we would like to compare the performance of the three methods based on speed and computational complexity. In Uniform Scaling, we try to find the best scaled match between the query and the candidate. So, the time complexity is $O(p*(M-N))$, where p, M and N represent the lengths of a scaled time series, the candidate and the query, respectively. The time complexity of DTW is roughly $O(MN)$.

The time complexity of EMD used in our method is harder to analyze because it is based on the simplex algorithm. However, according to our earlier analysis [24], if the algorithm is modeled as a bipartite matching problem, the complexity is $O(n^3 logn)$, where n is the number of features. Based on this time complexity, it may appear that DTW would perform better than EMD method. However, our experimental results as shown in Table 1 reveal that EMD actually performs better than DTW. This is because DTW computes all the motion frames (128 frames) but EMD, by applying the D-P algorithm, involves a far smaller number of features. From our experiments, the number of features used in the EMD algorithm varies from 2 to 30. This explains why the computation time consumed by EMD is far less than DTW method.

To explain why n varies so much in the EMD method, we may consider human movement. Different parts of the human body have different ranges of movement when performing different actions; some joints may vary a lot during a particular action while others may move very little. For example, during walking, most of the time the human's head will not change much while the feet may move continuously. This causes n to vary from joint to joint and from motion type to motion type.

To extract the feature points with the D-P algorithm, we need to define a threshold ε (Section 3.1.1) as a condition to determine if a line should be subdivided. If this threshold is set to be too large, the number of features produced may be too small. On the other hand, if it is set to be too small, then the number of features produced may be too large, with a lot of unnecessary feature points. In our experiment, we determine this value empirically and set it as 10.

Table 1. Feature matching time for the three methods.

	Uniform Scaling	DTW	EMD
Climb(10 files)	11.5 s	9.8s	6.0s
Jump (31 files)	104.4 s	124.9s	12.8s
Run (26 files)	242.9 s	230.8s	11.8s
SwordPlay (10 files)	49.3 s	59.2s	2.0s
Walk (38 files)	220.8 s	359.0s	18.3s

5. Evaluation

From our experimental results, we can see that our method performs relatively better than DTW and Uniform Scaling in terms of accuracy and speed. There may be two reasons that can explain such performance improvements. First, DTW can only handle local scaling and shifting while the Uniform Scaling method can only handle global scaling during the motion matching process. On the other hand, our method, as suggested by the experiments results, may be able to handle local shifting, local and global scaling simultaneously because our method depends on an energy morphing technique. Since our data are real motion data, they may contain all of the problems within a single motion, which DTW and Uniform Scaling fail to solve by themselves.

Second, our database contains motion clips of significantly different lengths, i.e., different numbers of frames. As we use the settings defined in the original experiments in the corresponding papers (e.g., DTW alignment range r=10% and Uniform Scaling maximum scaling factor = 1.2), they may not find a best match in our motion data.

However, our method also has its limitations. Currently, because we consider joint matching, frame synchronization of joints is not considered. In addition, although our method discourages strayed matching, it cannot prevent it. Further analysis is needed to verify the importance of frame synchronization to motion similarity search.

6. Conclusions and Future work

In conclusion, we have introduced a novel and efficient method for retrieving human motion data. Unlike other approaches, our method applies the Douglas-Peucker (D-P) algorithm to extract features according to the movement of the motion. As the number of features is not fixed, our method can preserve more vigorous motion by using more features. Whilst for relatively stationary joints, it reduces the number of features to reduce the computation time. To analyze the similarity between two motions, we model it as a transportation problem, and apply the Earth Mover's Distance (EMD) method as the matching framework. Our experiments show encouraging results as compared to the Dynamic Time Warping and Uniform Scaling methods.

As a future work, we are currently considering further method to speed up the matching process. Since our ground distance function is a metric, by applying a special spatial structure like Vantage Point tree, our method can support indexing [24]. This will further strengthen our retrieval speed and practical use on motion retrieval. To further stress the importance of t in our ground distance function, we are also considering a better way to prevent possible strayed matching.

7. References

[1] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient Similarity Search In Sequence Databases," *Proc. Foundations of Data Organization and Algorithms*, pp. 69-84, 1993.

[2] M. Cardle, M. Vlachos, S. Brooks, E. Keogh, and D. Gunopulos, "Fast Motion Capture Matching with Replicated Motion Editing," *ACM SIGGRAPH Sketches and Applications*, 2003.

[3] K. Chan and A. Fu, "Efficient Time Series Matching by Wavelets," *Proc. ICDE*, 1999.

[4] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani, "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases," *ACM Trans. Database Systems*, **27**(2):188-228, 2002.

[5] C. Chiu, S. Chao, M. Wu, S. Yang, and H. Lin, "Content-based Retrieval for Human Motion Data," *Journal of Visual Communication and Image Representation*, **15**(3):446-466, 2004.

[6] Graphics Lab, "Motion Capture Database," *Carnegie Mellon University*, http://mocap.cs.cmu.edu/

[7] B. Demuth, T. Röder, M. Müller, and B. Eberhardt, "An Information Retrieval System for Motion Capture Data," *Proc. European Conf. on Information Retrieval*, pp. 373-384, **LNCS 3936**, 2006.

[8] R. Dannenberg, W. Birmingham, G. Tzanetakis, C. Meek, N. Hu, and B. Pardio, "The MUSART Testbed for Query-by-Humming Evaluation," *Proc. Int'l Conf. on Music Information Retrieval*, 2003.

[9] D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required to represent a line or its caricature," *The Canadian Cartographer*, **10**(2):112-122, 1973.

[10] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast Subsequence Matching in Time-Series Databases," *Proc. ACM SIGMOD*, pp. 419-429, 1994.

[11] K. Forbes and E. Fiume, "An Efficient Search Algorithm for Motion Data using Weighted PCA," *Proc. ACM SCA*, pp. 67-76, 2005.

[12] E. Keogh, "Exact Indexing of Dynamic Time Warping," *Proc. VLDB*, pp. 406-417, 2002.

[13] E. Keogh, T. Palpanas, V. Zordan, D. Gunopulos, and M. Cardle, "Indexing Large Human-Motion Databases," *Proc. VLDB*, pp. 780-791, 2004.

[14] L. Kovar, M. Gleicher, and F. Pighin, "Motion Graphs," *Proc. ACM SIGGRAPH*, pp. 473-482, 2002.

[15] J. Lee, J. Chai, P. Reitsma, J. Hodgins, and N. Pollard, "Interactive Control of Avatars Animated with Human Motion Data," *Proc. ACM SIGGRAPH*, pp. 491-500, 2002.

[16] C. Li, P. Kulkarni, and B. Prabhakaran, "Motion Stream Segmentation and Recognition by Classification," *Multimedia Tools and Applications*, **35**(1), 2007.

[17] Y. Lin, "Efficient Human Motion Retrieval in Large Databases," *Proc. ACM GRAPHITE*, pp. 31-37, 2006.

[18] F. Liu, Y. Zhuang, F. Wu, and Y. Pan, "3D Motion Retrieval with Motion Index Tree," *Computer Vision and Image Understanding*, **92**(2-3):265-284, 2003.

[19] M. Müller, T. Röder, and M. Clausen, "Efficient Content-Based Retrieval of Motion Capture Data," *Proc. ACM SIGGRAPH*, 2005.

[20] M. Müller, T. Röder, and M. Clausen, "Efficient Indexing And Retrieval of Motion Capture Data Based on Adaptive Segmentation," *Proc. Int'l Workshop on Content-Based Multimedia Indexing*, 2005.

[21] M. Müller and T. Röder, "Motion Templates for Automatic Classification and Retrieval of Motion Capture Data," *Proc. ACM SCA*, 2006.

[22] Y. Rubner, C. Tomasi, and L. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval," *Int'l Journal of Computer Vision*, **40**(2):99-121, 2000.

[23] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, **26**(1):43-49, 1978.

[24] G.K.L. Tam and R.W.H. Lau, "Deformable Model Retrieval Based on Topological and Geometric Signatures," *IEEE Trans. on Visualization and Computer Graphics*, **13**(3):470-482, 2007.

[25] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. Keogh, "Indexing multi-dimensional time-series with support for multiple distance measures," *Proc. ACM SIGKDD*, pp. 216-225, 2003.