Motion Retrieval Based on Kinetic Features in Large Motion Database

Tianyu Huang
School of Software
Beijing Institute of Technology
Haidian, Beijing, China, 100081
huangtianyu@bit.edu.cn

Haiying Liu
School of Software
Beijing Institute of Technology
Haidian, Beijing, China, 100081
liuhaiying@bit.edu.cn

Gangyi Ding
School of Software
Beijing Institute of Technology
Haidian, Beijing, China, 100081
dgy@bit.edu.cn

ABSTRACT

Considering the increasing collections of motion capture data, motion retrieval in large motion databases is gaining in importance. In this paper, we introduce kinetic interval features describing the movement trend of motions. In our approach, motion files are decomposed into kinetic intervals. For each joint in a kinetic interval, we define the kinetic interval features as the parameters of parametric arc equations computed by fitting joints trajectories. By extracting these features, we are able to lower the dimensionality and reconstruct the motions. Multilayer index tree is used to accelerate the searching process and a candidate list of motion data is generated for matching. To find both logically and numerically similar motions, we propose a two-level similarity matching based on kinetic interval features, which can also speed up the matching process. Experiments are performed on several variants of HDM05 and CMU motion databases proving that the approach can achieve accurate and fast motion retrieval in large motion databases.

Categories and Subject Descriptors

I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism – *animation*; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – *retrieval models.*

General Terms

Algorithms, Management, Performance, Experimentation.

Keywords

Motion capture data, motion retrieval, kinetic interval, multilayer indexing.

1. INTRODUCTION

Motion capture technology has been widely used for capturing character motions in many applications such as performance animation, virtual reality, video games and so on [1, 2]. An increasingly large amount of motion capture data has been recorded, and some large motion databases [3, 4] have emerged in the past few years. In order to reuse the recorded data better, an efficient approach of motion retrieval in large motion databases is still a challenge problem.

Several factors impact motion retrieval efficiency: motion expression, matching algorithm, indexing structure, search space and more. Motion expression is an important basis of constructing motion databases. Usually we abstract motion features to represent the characteristics of a motion sequence. However, using overly abstract descriptions of motion speeds up the searching procedure at the expense of information loss in retrieval results. By providing properly representative features of motion content and a multi-resolution searching mechanism in this paper, we can achieve better efficiency and accuracy in retrieval results for large-scale motion databases.

Some prior works have shown potentially effective retrieval methods. One popular method of motion expression is the numerical method [2], but it is inefficient on large motion databases because of the high-dimensional motion data. Researchers have proposed various dimension reduction methods, like principal component analysis (PCA) [5] and singular value decomposition (SVD) [6], to decrease the dimensionality of motion data which usually causes feature information loss. Some other researchers focus on content-based motion expression [7]. However, this technique requires user interaction to specify the content-based features of the query for each retrieval.

In this paper, we introduce a kinetic interval feature of motion data. A kinetic interval is a set of kinetic features describing the movement trend of a motion sequence. For each joint of a continuous motion interval, we compute the parametric arc equations by fitting joint trajectories in the same movement trends. The kinetic interval features are defined as the parameters of these parametric equations. Using kinetic intervals, high-dimensional motion capture data can be decomposed as low-dimensional kinetic features. The original motion data is easily reconstructed from this low-dimensional form.

To speed up retrievals in the large-scale motion database, matching and indexing method are also considered in this paper. A two-level matching method based on kinetic interval features is adopted. We get a rough result by Derivative Dynamic Time Warping (DDTW)[21] algorithm with mean values of kinetic intervals in the first level of the algorithm. Then matching results will be refined by interpolating re-sampled the kinetic interval features in the second level. Indexing structure and search space are also major aspects of fast motion retrievals. A k-means based iterative method is used to construct the multilayer clustered index tree. We find that constructing index trees with kinetic intervals is too time consuming as the scale of motion database increases. To solve this problem, a Gaussian-based method is used to roughly segment motion sequences. After constructing the index tree with segments, we can locate the query in a coarse range of motion files by pruning of the index tree nodes. Then, we use the kinetic intervals for precise matching.

Our contributions can be summarized as follows:

1) We propose a method to interpret motion features as kinetic interval features to lower the data dimensionality and it is re-constructible.
2) A two-level similarity matching based on kinetic interval features is introduced to speed up the matching process. Similarity can be matched both numerically and logically.
3) We propose a pipeline of motion retrieval for large motion databases.

The rest of this paper is organized as follows. Section 2 summarizes the related work on motion retrieval. In Section 3, we introduce the kinetic interval method to represent high-dimensional motion data in a lower dimensional way. In Section 4, similarity matching based on kinetic intervals is presented. Section 5 gives the pre-computing and indexing of the motion database construction. The whole process of motion retrieval pipeline is described in Section 6. Section 7 offers a experiment results of our methods, and we make a conclusion in Section 8.

2. RELATED WORK

Motion preprocessing and retrieval are key issues of motion synthesis and reuse in applications, which have been studied for over a decade. Most solutions are based on the framework proposed by Faloutsos et al. [8]. The main idea of the framework is about efficient pruning of searching space and the low dimensional approximation extraction. However, their framework cannot work well with logically similar motions. In recent years, many researchers have improved this framework or even proposed new methods, such as semantics-based motion retrieval [7, 9] and example-based human motion retrieval [10].

Various ways have been pursued to transform original high-dimensional human motion data to a lower expression. Chuanjun Li et al. [6] proposed a method which can extract the motion features by exploiting the SVD, but it fails in finding similar sub-body motions in varied whole-body motions. Wang et al. [11] reduced the high-dimensional motion data by a curve simplification algorithm. They devise an EigenDis distance measure, use k-means to cluster motions, and then apply APP method to compress the motion database in a lossless manner. Forbes et al. [5] proposed a weighted PCA method to extract motion features, but as a result the non-linear motion data cannot be well-represented. Chuan Sun et al. [12] used a low-rank subspace decomposition method to reduce motion dimensions.

Several index methods have been proposed to organize the motion database efficiently. Yasuhiko Sakamoto et al. [13] used the method of motion map based on SOM, but it fails in distinguishing and retrieving motions containing the same movements at different velocities. Muller et al. [9] transformed the motion retrieval problem into a binary matching problem by proposing an efficient semantics-based motion retrieval method. However, the retrieval results rely on the quality of the geometry feature relationship in the query motion. Huang et al. [14] presented a geometric feature coding method to build index of the database. Deng et al. [10] decomposed motions into body parts and extracted common motion patterns for each part. A motion clip can be represented as a string of pattern indices, and so motion retrieval becomes simple string matching. Björn Krüger et al. [15] proposed a method for local and global similarity searches by kd-tree based local neighborhood searches. Yi Lin [16] segmented and clustered geometric features automatically into an index tree, and this method can find logically similar motions.

There are also several works about matching method used to automatically search for similar motions to the query. Since the motion data is kind of signal, researchers usually use DTW [5, 17] or uniform scaling [18] for motion matching. Keogh et al. [19] presented an indexing method based on lower bounding for retrieving in large motion database. Kovar et al. [20] proposed the pre-computing “match webs” to describe potential matches for each motion segment in the data set, but while dealing with large motion databases, building the “match webs” for every possible pair of motions may not work. Muller et al. [7] introduced a fault-tolerant retrieval method based on fuzzy hits and mismatches.

Two main similarity measures are numerical similarity and logical similarity. Numerical similarity methods like Böhm et al. [2] determined similarity through direct numerical comparison by comparing a computed distance against a user-supplied threshold. Logical similarity methods like Muller et al. [9] identified the logical similar motions by the binary geometric features and the “motion templates”.

In summary, prior works in motion expression are not effective enough in reconstructing motion data after dimension reduction or compression. If the compression procedure is not lossless, additional error will be introduced. We solve this problem by presenting a kinetic interval method to compress and reconstruct the motion data while maintaining the lossless property. Both similarity match methods mentioned above have their strengths and weaknesses. In our approach, we combine the two similarity matching methods. A numerical DDTW-based method is used for rough similarity matching, and then a logical kinetic interval-based method is used to refine the matching results. The efficiency of matching is also improved by this strategy.

3. KINETIC INTERVAL FEATURES

In this section, we introduce a method to compress high-dimensional motion capture data into low-dimensional sequences by using kinetic intervals. A kinetic interval feature f_{KI} is defined as a set of parameters of parametric equations which describe the motion trajectories of a joint.

In the human skeleton, a limb between each joint and its parent joint is a lever. The parent joint is the fulcrum, and the joint moves around an axis that passes through the center of its parent joint. So we can say the range of motion (ROM) of each joint satisfies the sphere constraint. The movement trajectories of each joint can be represented as several arcs along the sphere surface in its parent joint’s coordinate. In such a coordinate, the joint moves around a sphere S where the center is the parent joint and the radius is the length of the limb. Since an arc is the intersection of a sphere and a plane, the arc trajectory of a joint can be computed by the parameters of the sphere and the plane. We assume the joint moves along different arcs if it behaves movements with different semantic. The expression of joints’ movement arcs, shown in figure1, is a kind of geometric representations, which guarantees numerical-logical similarity while matching in motion database.

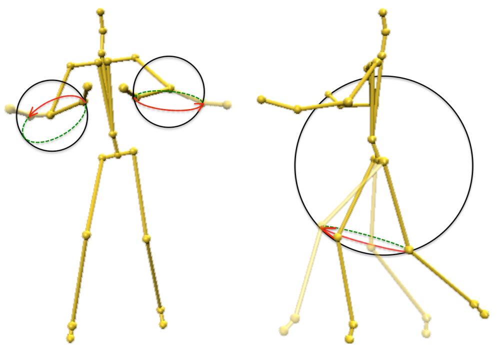


Figure 1. Joints' movement arcs of human motion data.

Assume that the position of the j-th joint in the i-th frame which is in the parent's coordinate is $p_{i,j}=[x_{i,j},y_{i,j},z_{i,j}]$ and $p_{i,j}$ satisfies the constraint of equation (1) where r_j is the radius:

$$S(x_{i,j},y_{i,j},z_{i,j})=x_{i,j}^2+y_{i,j}^2+z_{i,j}^2-r_j^2=0 \qquad (1)$$

A joint moves along an arc under consistent force action in its parent's coordinate. This arc is the intersection of a plane and a sphere. The joints' positions in a consecutive sequence of frames with the same semantics lie on the same plane, so $p_{i,j}$ satisfies the constraint of (2) as well.

$$P(p_{i,j})=\hat{n}\cdot(p_{i,j}-q)=0 \qquad (2)$$

Here, $\hat{n}$ denotes the normal of the plane, and q is an arbitrary point on this plane. Then, we can fit the plane and its normal $\hat{n}$ to all those $p_{i,j}$ of the kinetic interval by using the Levenberg Marquardt method.

Assume that the plane P can be represented as $Ax+By+Cz+D=0$, $\hat{n}=[A,B,C]$ denotes the normal of the plane P, the basis vectors of P are $\hat{u}=[B,-A,0]$ and $\hat{v}=\hat{n}\times\hat{u}$. The center c of the arc is the intersection of plane P and a line $L(c)$:

$$L(c)=O+\lambda\hat{n} \qquad (3)$$

where $O=[0,0,0]$ and λ is an arbitrary value. The radius R of the arc can be computed as $R=\sqrt{r_j^2-|c|^2}$ where r_j is the radius of the sphere S of the j-th joint. And the arc can be represented as:

$$f_{KI}(t)=c+R(\hat{u}\cos t+\hat{v}\sin t) \quad t\in(0,2\pi] \qquad (4)$$

Assume p_{start} and p_{end} denote the position of the beginning and the end frames of the kinetic interval. Now, we get all the elements of kinetic interval feature $f_{KI}=\{u,v,c,R,p_{start},p_{end}\}$.

Figure 2 illustrates the wrist joint trajectory represented by a kinetic interval feature. In this figure, the wrist joint moves from J_{wrist} to J'_{wrist} along the green dashed line in the joint space of its parent J_{elbow}. The parametric equation of the kinetic interval can be obtained by fitting the trajectory, and $\hat{u}$, $\hat{v}$, c and R are variables of the parametric equation.

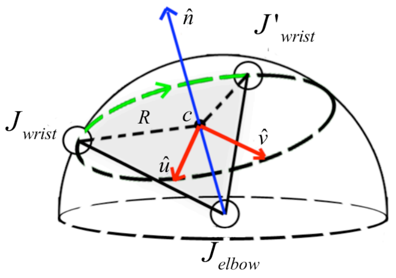


Figure 2. Illustration of wrist joint trajectory represented by kinetic interval feature.

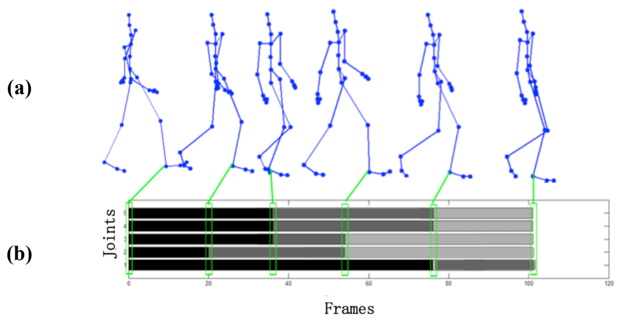


Figure 3. Walking motion decomposed by kinetic interval method. (a): Poses of a motion at frame position 0, 20, 37, 56, 78 and 100(30Hz). (b): Feature matrix of the walking motion used in (a). The gray gradient means the alteration of different planes of joints. The green blocks depict the kinetic interval segmentation.

The start and the end of the arc can be computed by fitting a period of the trajectory to a plane P using least square method. The fitting error $E(p,P)$ can be computed as:

$$E(p,P)=\left(\sum_{i=\text{start}}^{N}\frac{|A\cdot x_{i,j}+B\cdot y_{i,j}+C\cdot z_{i,j}+D|}{\sqrt{A^2+B^2+C^2}}\right)\Big/N \qquad (5)$$

where N is the number of points from the start point to the current point. When $E(p,P)<\tau$, where τ is a threshold, the current point belongs to the arc; When $E(p,P)\geq\tau$, we take the current point as the start point of the next arc. All the f_{KI} s in a kinetic interval should have the same frames. When a kinetic interval is segmented from the motion sequences, we resize all the f_{KI} s as long as the shortest one and computing the next arc for all joints with the same start frame. The process is shown in figure 3.

4. SIMILARITY MATCHING BASED ON KINETIC FEATURES

To speed up the matching process, we propose a two-level matching method based on the kinetic interval features. In the first level, we match each kinetic interval of the two motion files to get rough results by using DDTW algorithm. Unlike DTW, DDTW considers the first derivative of the sequences instead of the data points' values and works well when features in one sequence are slightly higher or lower than their corresponding features in another sequence. The matching results will be refined in the second level by interpolation between kinetic intervals based on kinetic interval features. The two-level matching method can improve the matching speed with a time requirement of $O(m/K * n/L)$, where K is the average length of kinetic intervals of an m-frame motion file, and L is the average length of kinetic intervals of an n-frame motion file.

Assume that $M = [Mean_1, \ldots, Mean_m]$ is the mean set of a motion file with m kinetic intervals, and $Mean_i$ denotes the mean of motion data in the i-th kinetic interval. While using DDTW, the distance D can be computed as:

$$D_x[Mean_i] = \\ \left((Mean_i - Mean_{i-1}) + (Mean_{i+1} - Mean_{i-1})/2\right)/2 \quad 1 < i < m \tag{6}$$

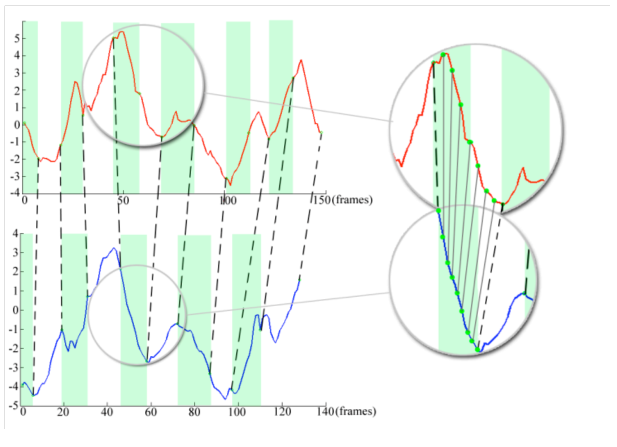
$$D = \left(D_x[Mean_i] - D_x[Mean_i']\right)^2 \; 1 < i < m, 1 < j < n \tag{7}$$

where $D_x[Mean_i]$ is the derivative of $Mean_i$.

In the second level, we introduce the refining process with interpolation based on kinetic interval feature. First, the variable of kinetic interval features f_{KI_i} varies from $\Omega_i = [f_{KI}^{-1}(p_{start}), f_{KI}^{-1}(p_{end})]$. And the inverse function of (5) $f_{KI}^{-1}(p)$ can be represented as:

$$f_{KI}^{-1}(p) = \left(\hat{u}(p-c) \pm \hat{v}\sqrt{R^2 - (p-c)^2}\right)/R \tag{8}$$

Assume that there are M and N frames in the two matched kinetic intervals KI_i and KI_j. We take $\min(N, M)$ as the grain size of the resampling. The sample set S_i can be obtained by uniformly sampling the range of $f(t), t \in \Omega_i$. We can get the sample set S_j of f_{KIj} from the other kinetic interval KI_j in the same way.



(a) (b)

Figure 4. Running exercise motion matching by kinetic interval method. (a) shows the matching result of two kinetic intervals sequences. (b) indicates the process of the interpolation between two kinetic intervals.

We then interpolate the samples in S_i and S_j one-by-one using weighted Euclidean distance with a user defined ω for the joint. Considering the different speeds of motion sequences, we use the ratio of N and M to show the differences between the speeds.

Finally, we modify the matching result by

$$Sim' = Sim \cdot w_d + w_i \cdot \frac{N}{M} \cdot \sum_{s=1}^{J^n} \sum_{t=1}^{m} \left((S_i - S_j) \cdot \omega_s\right) \tag{9}$$

Sim is the roughly matching result we calculated in the first level; w_d and w_i are user-defined parameters, ω_s is the weight of the s-th joint, N and M are the length of kinetic intervals and $m = \min(N, M)$, and J^n is the number of joints.

One example of matching by by kinetic interval method is shown in figure 4. Figure 4(a) shows the matching result of two kinetic intervals sequences. Figure 4(b) indicates the process of the interpolation between two kinetic intervals.

5. MOTION DATBASE CONSTRUCTION

There are two procedures in constructing motion database, pre-computing and indexing. In the pre-computing process, we use Gaussian-based segmentation method to decompose motion sequences. In indexing, an iterative procedure is used for constructing the multilayer clustered index tree from the segments. This structure can generate a candidate list of a coarse range of motion files by pruning of the index tree nodes.

Gaussian model can describe characteristics of human motion. In pre-computing process, we use Gaussian-based method to segment the motion sequences. The centroid of human body C_i at the i-th frame can be computed by

$$C_i = \left(\sum_{j=0}^{J^n} (x_{i,j} + y_{i,j} + z_{i,j})\right)/J^n \tag{10}$$

For each frame, where $p_{i,j} = [x_{i,j}, y_{i,j}, z_{i,j}]$ denotes the position of each joint in each frame, and J^n denotes the number of the joints. The mean μ of the motion sequence and standard deviation σ of motion sequence can be computed as:

$$\mu = \sum_{i=f_{begin}}^{f_{end}} (C_i) \Big/ (f_{end} - f_{begin}) \tag{11}$$

$$\sigma = \sqrt{\sum_{i=f_{begin}}^{f_{end}} (C_i - \mu)^2} \tag{12}$$

where f_{end} denotes the number of the processing frame and f_{begin} denotes the starting frame number of the current motion sequence segment. the $(f_{end}+1)$-th frame satisfies the current Gaussian model, if the following inequality holds.

$$C_{f_{end}+1} - \mu < range \;\; \sigma \tag{13}$$

where $C_{f_{end}+1}$ denotes $(f_{end}+1)$-th frame motion data, $range$ is a user parameter which means the deviations from μ.

If $f_{end} - f_{begin} < \varepsilon$, for some value epsilon, we combine the current segment with the previous segment because it contains less information to express a simple action. The time requirement of this method is $O(n)$. Our experiments show that when ε is 30, the result of segmentation is desirable for retrieval.

In Indexing, multilayer clustered index tree is applied. We cluster μ of all the motion segments into several clusters. Then if there are more than k elements in this cluster we cluster again until there are less than k elements. Finally, we construct an index tree for all the motion segments. Each node of the index tree denotes a category of motion. As the layer goes deeper, the precision of classification is higher. The time requirement of the whole process is $O(n*k*t)$ where k is the number of desired clusters and t is the number of iterations.

6. PIPELINE AND RETRIEVAL

Figure 5 visualizes the pipeline of our motion retrieval approach. Preprocessing motion data and constructing the index tree are offline procedures. Step 1-3 is offline processes, and Step 4-7 is real-time retrieval processes.

Given a human motion repository,

Step 1: Decompose the motion files into kinetic interval features.

Step 2: Segment motion files by Gaussian-based method for the indexing process.

Step 3: Cluster the segments iteratively to build up an index tree.

Given a query motion file,

Step 4: Decompose the query file into kinetic intervals and segments.

Step 5: Retrieve similar segments from the index tree.

Step 6: Generate a candidate list of similar motion files.

Step 7: Compute the similarity of the kinetic intervals between these candidate motion files and the query, and get the matching results.

Given a query motion file, we first decompose it into segments $S=\{seg_1, seg_2, seg_3, ..., seg_n\}$ by using the method in Section 5. For

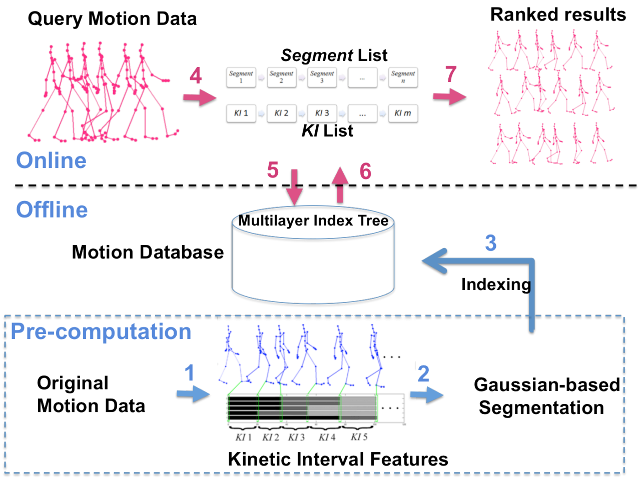


Figure 5. Pipeline of motion retrieval based on kinetic features and multilayer index tree. Step 1-3 is offline processes. Step 4-7 is real-time retrieval processes.

each segment seg_i, $i \in [1, n]$, we can get a set of segments S_{result} by searching in the index tree. The motion files containing S_{result} will be added to a candidate list. Then, in order to match the file precisely, we decompose the query motion file by using kinetic interval method and extract the kinetic interval features. We match each interval of the query motion file with every interval of the motion files in candidate list by using the two-level matching method mentioned in Section 4. Also, we adopt the feature sets F_E^{15}, F_E^{30}, F_E^{39} devised by Björn Krüger et. in [15] to discuss whether they can help us to reduce time overhead.

7. RESULTS AND DISCUSSION

To prove the applicability of the proposed method, we conducted several experiments. The presented algorithm has been implemented in Matlab. All experiments were executed on a computer with an Intel Core T7500 and 2GB of RAM.

7.1 Performance of Matching Method Based on Kinetic Features

We evaluate the proposed method on the HDM05 database by analyzing the performances on five pairs of motion files of various lengths. The comparisons are made between our proposed method ($DD_{msm}+I_{msi}$), $DD_{msm}+DD_{msi}$ and DDTW. DD_{msm} denotes the method of matching $M=[Mean_1, \ldots, Mean_m]$ and $M'=[Mean_1', \ldots, Mean_m']$ by DDTW algorithm. I_{msi} denotes the method of linear interpolation between two resampled kinetic intervals. DD_{msi} denotes the method of matching two kinetic intervals by DDTW[21] algorithm. The time requirement of all those DDTW methods are $O(m*n)$. The matching time results are shown in Table 1.

Comparing the results between $DD_{msm}+DD_{msi}$ and DDTW, we can see that the time cost is lower after segmenting the motion file into kinetic intervals. Linear interpolation between resampled motion data can improve the performance by comparing the results between $DD_{msm}+I_{msi}$ and $DD_{msm}+DD_{msi}$. Therefore, $DD_{msm}+I_{msi}$ performs well in our motion matching tasks.

7.2 Performance Analysis on Motion Retrieval

To evaluate the performance of our motion retrieval method, we run the experiments on several variations of CMU [3] motion database of different scales with different segmentation methods.

Table 2 shows the pre-computing time, matching time, indexing time, searching time of $M_{KI}+M_{GS}$, M_{GS} and M_{KI}. $M_{KI}+M_{GS}$ is our proposed method. M_{GS} is segmenting the motion sequences only using the Gaussian-based motion sequence segmentation method and M_{KI} is segmenting the motion sequences only using kinetic interval method, and visualized in Figure 6. The query is an 800-frame length motion file. From the experiment results, we can see that the $M_{KI}+M_{GS}$ can significantly reduce the retrieval time, so our method can meet the performance on real-time applications. The reduction is even more significant when the size of the database is large. Unlike the retrieval time, we do not have to be strict with the offline preprocessing time for segmenting and indexing; 2036 seconds is tolerable for such a huge database.

Table 1. Matching time (in seconds) for five pairs of motion files (in frames) with different methods.

Motion file pairs	$DD_{msm}+I_{msi}$	$DD_{msm}+DD_{msi}$	DDTW
Pair (500 ,500)	0.0692	0.2394	8.6635
Pair (500 ,1000)	0.1456	0.4505	16.3010
Pair (1000,1000)	0.2371	0.6988	28.3173
Pair (1000,2000)	0.4617	1.4352	54.0013
Pair (2000,2000)	0.9360	2.7331	205.7091

Table 2. The detailed time (in seconds) of retrieval, indexing and segmenting in different databases (in frames).

DB Size	**Time**	$M_{KI}+M_{GS}$	M_{GS}	M_{KI}
5762	Pre-computing Time	7.72	0.03	7.69
	Indexing Time	0.10	0.10	1.69
	Retrieval Time	0.04	0.14	2.42
25125	Pre-computing Time	32.40	0.20	32.21
	Indexing Time	0.30	0.30	3.16
	Retrieval Time	0.17	0.78	2.52
100190	Pre-computing Time	131.02	0.68	130.34
	Indexing Time	0.54	0.54	16.30
	Retrieval Time	0.25	2.86	2.60
1273227	Pre-computing Time	2036.11	12.85	2023.26
	Indexing Time	8.14	8.14	890.44
	Retrieval Time	0.59	10.39	2.84

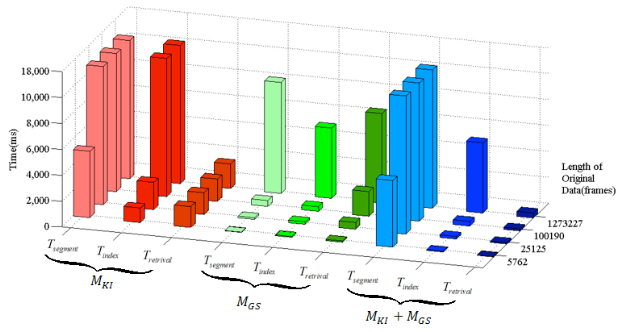


Figure 6. The cost in retrieval, indexing and segmenting of our proposed method compared to the other two in four motion databases with different sizes.

Kinetic interval feature is not only a low-dimension expression of original motion data, but also a kind of compression representation. It records time-space information of motion data. In retrieval experiments, the matching results are similar numerically and logically. The motion categories do not affect the retrieval accuracy.

7.3 Performance on Different Feature Sets

We conducted this experiment to evaluate performance on different feature sets. The feature sets are defined by Björn Krüger et al. [15], F_E^{15} consists of the positions of 4 end-effectors and head; F_E^{30} consists positions of 4 end-effectors and head, as well

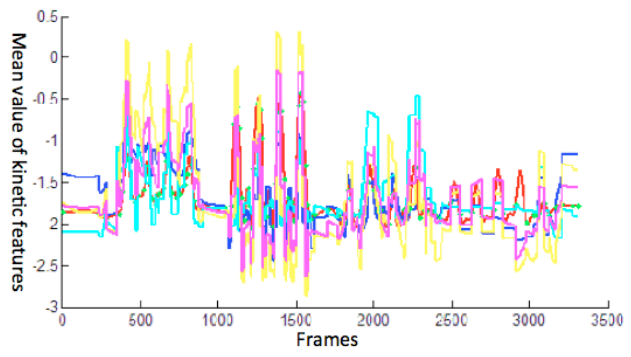
Table 3. Average computation times (in seconds) for segmenting, indexing, matching using various feature sets on motion of four CMU databases (in frames) with different sizes

DB Size	**Feature Sets**	**Pre-computing Time**	**Indexing Time**	**Matching Time**
5762	F_E^{15}	2.6520	0.1092	0.0208
	F_E^{30}	4.5084	0.1092	0.0210
	F_E^{39}	5.6940	0.1248	0.0215
	F_E^{all}	7.7237	0.1092	0.0225
25125	F_E^{15}	10.9045	0.3432	0.0936
	F_E^{30}	18.7513	0.1560	0.0960
	F_E^{39}	23.7746	0.1716	0.0998
	F_E^{all}	32.4042	0.2028	0.1009
100190	F_E^{15}	41.9487	0.6084	0.3533
	F_E^{30}	76.0349	0.6552	0.3550
	F_E^{39}	96.5334	0.6396	0.3571
	F_E^{all}	131.0273	1.0920	0.3581
1273227	F_E^{15}	514.0723	9.1261	2.8584
	F_E^{30}	1044.1354	45.1467	2.8721
	F_E^{39}	1394.7034	66.6124	2.8891
	F_E^{all}	2036.1132	89.8878	2.8972

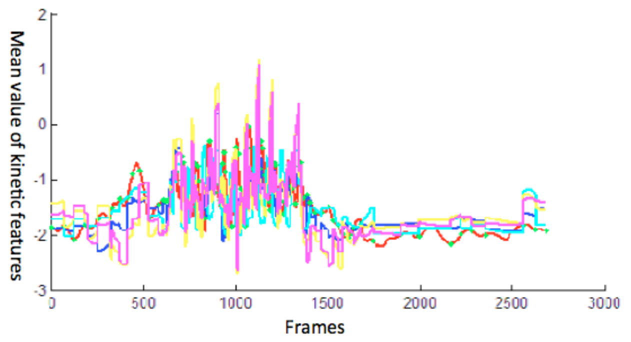
as the 5 positions of the elbows, knees and one chest joint (30 dimensions); F_E^{39} has all features of F_E^{30} and adds position of the shoulders and one lower-back joint (39 dimensions). The efficiency and usability of F_E^{15}, F_E^{30} and F_E^{39} has already been proved, especially for F_E^{15}. We use these feature sets and full feature set F_E^{all} consisting all the joints to compute the segmenting, indexing, and matching time on databases with different sizes to speed up the offline procedure and evaluate their performances on our approach. In Table 3, segmenting time is the sum of kinetic interval segmenting time and Gaussian-based segmenting time, indexing time is the time for constructing the index tree, and matching time is the sum of time for matching processes between query motion file and all the motion files in the candidate list. In addition, there are more than 20000 frames in the candidate list and 1273227 frames in the entire database.

In Table 3, the computation times shows that as the number of features gets smaller, the segmenting and indexing process can be more efficient, especially for F_E^{15}.

Whereas lower dimensional feature sets make an improvement on offline procedure efficiency, we still need to keep the accuracy of the motion retrieval. We synthesize motion sequence by using the parameter equation of kinetic interval with different feature sets. Qualitative comparison between original motion sequence and calculated motion sequence has been made with the aims of experimenting variations of movement trend in different feature sets.



(a)



(b)

Figure 7. Kinetic intervals of different motion categories, (a) stylized walking motion, (b) dancing motion. The horizontal axis is frames of motion file, and the longitudinal axis is mean value of kinetic interval features. The red curves are the raw motion data, the blue ones are F_E^{all} ,the light blue ones are F_E^{15} ,the yellow ones are F_E^{39} ,the purple ones are F_E^{all} .

In Figure 7, the curves with low-dimensional feature sets can keep the feature of the original motion well. As Table 3 has already proved the efficiency of low-dimensional feature sets, it is reasonable to use low-dimensional feature sets while the accuracy requirement is not very strict.

8. CONCLUSION

In this paper, we proposed kinetic interval features which transform the motion expression from geometric space to a kinetic parameter space, at the same time they speed up the matching process. We use multi-layer clustered index tree to organize the motion databases. Given a query motion, The numerical and logical similar results can then be efficiently computed by two-level matching method. A numerical DDTW-based method is used for rough similarity matching, and then a logical kinetic interval-based method is used to refine the matching results. To evaluate the performance and usability of our approach, we tested our approach in a series of databases with different sizes. By analyzing the results, we found that the kinetic interval method has a significant improvement in the performance of matching process, and also has potential for applications of reconstructing motion from relatively few markers.

The current approach does not consider data compression for the repeated motion data or cycle data. If make such improvement for repeated and cycle motion data, the database construction and retrieval could be much more efficient. Also user study can be applied for retrieval results, which could gradually improve the accuracy for a motion database.

9. ACKNOWLEDGMENTS

Our thanks to HDM Stuttgart and CMU motion lab for sharing their motion data files. We also thank all anonymous references for constructive and valuable comments.

This research is supported by National Natural Science Foundation of China (61202243) and Doctoral Fund of Ministry of Education of China (20091101120032).

10. REFERENCES

[1] Rosenhahn B., Klette R., and Metaxas D. 2007. *Human Motion Understanding, Modeling, Capture, and Animation. Springer-Verlag Press*, Rio de Janeiro, Brazil.

[2] Christian B., Stefan B. and Daniel A. K. 2001. Searching in High-dimensional Spaces Index Structures for Improving the Performance of Multimedia Databases. *ACM Computing Surveys* 33, 3 (September 2001), 322–373. DOI= http://doi.acm.org/10.1145/502807.502809.

[3] CMU. Carnegie-Mellon Mocap Database. http://mocap.cs.cmu.edu.

[4] HDM05. Hochschule der Medien Database. http://www.mpi-inf.mpg.de/resources/HDM05/.

[5] Forbes K. and Fiume E. 2005. An Efficient Search Algorithm for Motion Data Using weighted PCA. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics* (Los Angeles, CA, USA, July,2005). SCA '05. ACM, New York, NY, 67-76. DOI= http://doi.acm.org/10.1145/1073368.1073377.

[6] Chuanjun L., Gaurav P., Siqing Z. and Prabhakaran B. 2004. Indexing of variable length multi-attribute motion data. In *Proceedings of the 2nd ACM international workshop on Multimedia databases* (Washington D.C., USA, November 8 - 13, 2004). MMDB '04. ACM, New York, NY, 75-84. DOI= http://doi.acm.org/10.1145/1032604.1032617.

[7] Meinard M., Tido R. and Michael C. 2005. Efficient Content-Based Retrieval of Motion Capture Data. *ACM Trans.on Graphics* 24, 3 (July 2005), 677–685. DOI= http://doi.acm.org/10.1145/1073204.1073247.

[8] Christos F., Ranganathan M. and Yannis M. 1994. Fast Subsequence Matching in Time-Seies Databases. In *Proceedings of the 1994 ACM SIGMOD international* (Minneapolis, Minnesota, USA, May 24 - 27, 1994). SIGMOD '94. ACM, New York, NY, 419-429. DOI= http://doi.acm.org/10.1145/191839.191925.

[9] Meinard M. and Tido R. 2006. 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, September 2 - 4, 2006). SCA '06. ACM, New York, NY, 137-146.

[10] Zhigang D., Qin G. and Qing L. 2009. Perceptually Consistent Example-based Human Motion Retrieval. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games* (Boston, MA, USA, February 27 - March 1, 2009).

I3D '09. ACM, New York, NY, 191-198. DOI= http://doi.acm.org/10.1145/1507149.1507181.

[11] Pengjie W., Rynson W.H., Mingmin Z., Jiang W., Haiyu S. and Zhigeng P. 2011. A real-time database architecture for motion capture data. In *Proceedings of the 19th ACM international conference on Multimedia* (Scottsdale, Arizona, USA, Nov. 28 - Dec. 1, 2011). MM '11. ACM, New York, NY, 1337-1340. DOI= http://doi.acm.org/10.1145/2072298.2072008.

[12] Chuan S., Imran J. and Hassan F. 2011. Motion Retrieval Using Low-Rank Subspace Decomposition of Motion Volume. *Computer Graphics Forum*, 30,7 (September 2011), 1953-1962.

[13] Yasuhiko S., Shigeru K. and Toyohisa K. 2004. Motion Map: Image-based Retrieval and Segmentation of Motion Data. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Grenoble, France, August 27 - 29, 2004). SCA '04. ACM, New York, NY, 259-266. DOI= http://doi.acm.org/10.1145/1028523.1028557.

[14] Tianyu H., Feng-xia L., Shouyi Z. and Xiangchen L. Motion Retrieval Method Based on Geometric Feature Coding. *Journal of System Simulation*, 18, 10 (Oct. 2006), 2767-2773.

[15] Bjorn K., Jochen T., Andreas W. and Arno Z. 2010. Fast Local and Global Similarity Searches in Large Motion Capture Databases. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Madrid, Spain, July 02 - 04, 2010). SCA '10. ACM, New York, NY, 1-10.

[16] Yi L.. 2006. Efficient Human Motion Retrieval in Large Databases. In *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia* (Kuala Lumpur, Malaysia, November 28 - December 02, 2006). GRAPHITE '06. ACM, New York, NY, 31-37. DOI= http://doi.acm.org/10.1145/1174429.1174434.

[17] Chihyi C., Shihpi C., Mingyang W., Shinine Y. and Hsinchih L. 2008. Content-Based Retrieval for Human Motion Data. *Journal of Visual Communication and Image Representation*, 15, 3 (September, 2004), 446-466. DOI= http://dx.doi.org/10.1016/j.jvcir.2004.04.004.

[18] Eamonn K., Themistoklis P., Victor B. Z., Dimitrios G. and Marc C. 2004. Indexing Large Human-Motion Databases. In *Proceedings of the Thirtieth international conference on Very Large Data Bases* (Toronto, Canada, August 31 - September 3 2004). VLDB '04. Morgan Kaufmann Publishers, San Fransisco, CA, 780-791.

[19] Eamonn K. 2002. Exact indexing of dynamic time warping. In *Proceedings of the 28th international conference on Very Large Data Bases* (Hong Kong, China, August 20 - 23, 2002). VLDB '02. Morgan Kaufmann Publishers, San Fransisco, CA, 406-417.

[20] Kovar, L. and Gleicher, M. 2004. Automated extraction and parameterization of motions in large data sets. *ACM Trans.on Graphics* 23, 3 (August 2004), 559–568. DOI= http://doi.acm.org/10.1145/1015706.1015760.

[21] Eamonn J. K. and Michael J. P. 2001. Derivative Dynamic Time Warping. In *Proceedings of the First SIAM International Conference on Data Mining* (Chicago, IL, USA, April 5 - 7 2001).