A genetic algorithm approach to human motion capture data segmentation

Na Lv1,2, Yan Huang1*, Zhiquan Feng2 and Jingliang Peng1

1 School of Computer Science and Technology, Shandong University, Jinan 250101, Shandong Province, China
2 Department of Information Science and Technology, University of Jinan, Jinan 250022, Shandong Province, China

ABSTRACT

In this paper, we propose a novel genetic algorithm approach to human motion capture (MoCap) data segmentation. For a given MoCap sequence, it constructs a symbolic representation through unsupervised sparse learning, detects the candidate segmenting points to the sequence, models the selection/deselection of each candidate with a gene, and employs the genetic algorithm to find the optimal solution. To the best of our knowledge, we for the first time introduce the genetic algorithm and the sparse learning technique to the problem of MoCap data segmentation, leading to excellent segmentation performance as experimentally demonstrated. Copyright © 2014 John Wiley & Sons, Ltd.

KEYWORDS
motion capture; segmentation; genetic algorithm; sparse learning; symbolic representation

*Correspondence
Yan Huang, School of Computer Science and Technology, Shandong University, Jinan 250101, Shandong Province, China.
E-mail: yan.h@sdu.edu.cn

1. INTRODUCTION

Three-dimensional (3D) human animation has been widely used in filming, gaming, virtual reality, and many other fields. It is a challenging task to realistically and efficiently produce 3D human animations, and for that purpose, motion capture (MoCap) has been prevalently used.

A vast volume of human motion data has been produced by motion capture systems. In particular, long human motion sequences are being increasingly produced, mainly because of two reasons: (i) frequently, long motion sequences are demanded to capture complex behaviors and natural transitions between different behaviors; and (ii) actresses (actors) often feel it more natural and easy to act long sequences than separate short ones. With an increasing volume of MoCap data at hand, it becomes crucially challenging to effectively index, retrieve, and reutilize these data, and as the first step, it is critical to divide a long sequence accurately into short clips of simple behavioral units. Correspondingly, automatic human MoCap data segmentation has attracted significant research attention in recent years.

In this work, we propose a novel scheme for human MoCap data segmentation, which leads to superior segmentation accuracy and computational efficiency. Novelties of the proposed work mainly reside in the following aspects:

- Sparse learning-based motion representation and analysis. Given an input human motion sequence, we conduct sparse learning to extract a dictionary of most representative postures and convert the original sequence to a symbolic sequence, that is, a sequence of indices into the dictionary. On this symbolic sequence, string analysis is further made to identify simple behavioral patterns and correspondingly a set of candidate segmenting points.

- Genetic modeling and optimization. From the set of candidate segmenting points, we determine the optimal subset that produces the best segmentation. Instead of basing our method on local transitional characteristics, we model selection/deselection of each candidate with a gene and perform global heuristic search using the genetic algorithm to reach the optimal solution.

2. RELATED WORK

Kinematic features, such as velocity and acceleration, are used in early algorithms (e.g., [1,2]) for motion sequence segmentation. However, these kinematic features reflect only low-level motion characteristics of a single body part and do not work directly on segmenting complex motions of the whole human body system.
Some methods find similarity between frames through clustering or dimensionality reduction and, on the basis of which, classify the motion frames and make the segmentation correspondingly. Manifold learning [3] and nonlinear dimensionality reduction [4] are used to classify motion data. Jenkins and Mataric [5] used the Isomap algorithm for spatio-temporal dimensionality reduction to extract structure-indicative features and conduct the kinematic centroid segmentation by finding the local maximum distance of the centroid of a kinematic substructure between two frames. Zhou et al. [6,7] proposed two temporal clustering methods, namely aligned cluster analysis (ACA) and hierarchical aligned cluster analysis (HACA), respectively, which both are based on kernel k-means method. In general, with this type of methods, frames of the same class may not be temporally consecutive, and therefore, sub-optimal or inaccurate segmentation may be produced.

Time series analysis approaches to motion sequence segmentation are proposed by Barbić et al. [8], including the principal component analysis (PCA), the probabilistic PCA, and the Gaussian mixture model approaches. They are based on the observation that simple motions exhibit lower dimensionality than complex ones. The PCA approach finds the segmenting points when the projection error on the optimal r-dimensional hyperplane begins to increase rapidly for a fixed r; the probabilistic PCA approach estimates the distribution of the motion data and finds the segmenting points when the distribution significantly changes; the Gaussian mixture model approach segments the sequence when two consecutive sets of frames belong to different Gaussian distributions. These methods work well for simple motion sequences but not for sequences with complex motions.

Another class of algorithms employs machine learning techniques. Bouchard and Badler [9] constructed a Laban movement analysis effort classifier by creating a set of neural networks. Li et al. [10] used the feature vectors extracted by singular value decompositions and used multi-class support vector machine classifiers to segment the motion stream. With this class of algorithms, efforts are needed for training, and the run-time segmentation performance is highly dependent on the training data set.

Recently, string segmentation methods have been used for MoCap data segmentation. The methods of Yang et al. [11] and Lan et al. [12], and our method all fall into this category. Yang et al. converted the raw motion sequence to a motion string, identified from its static and repetitive patterns, and manually labeled the behavior meaning of each identified pattern to make a behavior segmentation. Lan et al. obtained the motion vocabulary through key pose extraction and clustering analysis, transformed the Mocap data into motion documents, used latent Dirichlet allocation (LDA) to discover motion topics, and found the segmentation points by detecting significant changes in the motion topic distribution.

3. PROPOSED METHOD

Our target is an automatic, generic, simple, and effective method for human MoCap data segmentation. Mathematical formulation of the problem and detailed description of the proposed method are provided in the following subsections.

3.1. Problem Formulation

We take as input a human motion sequence, $F = \{f_i | i = 1, 2, \ldots, n\}$, where $f_i$ is the i-th frame. For any sequence of segmenting points, $S = \{s_j | 1 \leq s_j \leq n, j = 1, 2, \ldots, n_s\}$, we can segment the motion sequence into $n_s + 1$ segments. Denoting $s_0 = 0$ and $s_{n+1} = n$, the resultant sequence of segments is denoted as $S = \{S_k | 0 \leq k \leq n_s\}$ where $S_k = \{f_i | s_k + 1 \leq i \leq s_{k+1}\}$. Defining a metric for the segmentation quality, $Q(S)$, for $S$, our goal is to find an optimal segmenting point sequence, $S$, that leads to the maximum quality measure, $Q(S)$.

Regarding the data format, each human MoCap sequence stores the skeletal structure and its joints’ rotational movements. All bones are organized in a hierarchical structure. For each frame, the root node contains the position and orientation of the whole body, while each of the other contains a joint’s rotation relative to its parent.

Because the whole body’s positions and orientations have less significance in posture change detection while bringing the issue of registration between MoCap sequences, we only take the nonroot joints’ rotations into our raw motion frame representation. Assuming that there are $q$ nonroot joints and each joint’s relative rotation is stored as three Euler angles, each frame, $f_i$, is represented by a matrix, $m_i \in \mathbb{R}^{3q \times 1}$, and the whole motion sequence, $F$, by a matrix, $M \in \mathbb{R}^{3q \times n}$.

3.2. Method Overview

We propose a genetic algorithm approach to human MoCap data segmentation, which provides a generic framework while allowing for flexibility in underlying data representation and fitness function definition.

For an input human motion sequence, we first learn from it a dictionary of representative postures and convert each motion frame to an index into the dictionary. All these indices form a symbolic representation of the raw MoCap data, providing a high-level abstraction of the input data. Following the fact that cyclic occurrences of a posture sequence usually correspond to a semantical behavioral unit, we identify cyclic symbolic patterns to determine a set of candidate segmenting points. It is noteworthy that the symbolic representation simplifies the fitness computation and the candidate segmenting point set, usually a small subset of the whole set of time instances, significantly reduces the search space.

Thereafter, we select an optimal subset from the candidate segmenting point set using the genetic algorithm.
of its sparse nature, the whole Mocap sequence data contain much spatial and temporal redundancy [13]. Because

3.3.1. Sparse Learning.

the weight part of the most representative postures, while we employ sparse learning techniques for that purpose.

3.3.2. Motion String Construction.

The previously described hard clustering algorithm produces a matrix $D$ of $K$ clusters’ representative vectors, $D = [d_1, d_2, \ldots, d_K]$. Each original data sample, $f_i, i \in [1,n]$, belongs to the $j_i$-th cluster or, in other words, can be approximated with one atom (for hard clustering) or more than one atom (for fuzzy clustering).

Formally put, a hard clustering algorithm produces a matrix $D$ of $K$ clusters’ representative vectors, $D = [d_1, d_2, \ldots, d_K]$. Each original data sample, $f_i, i \in [1,n]$, belongs to the $j_i$-th cluster or, in other words, can be approximated with one atom (for hard clustering) or more than one atom (for fuzzy clustering).

Exact solution of sparsest representations is proved to be an NP-hard problem [14]. Over the past few decades, many algorithms for approximate solution have been proposed, such as the matching pursuit (MP) [15] and the orthogonal MP (OMP) [16] algorithms. The MP [15] and the OMP [16] algorithms assume that the dictionary is known. However, in our problem, we aim to learn the dictionary and derive the sparse representation at the same time. For our purpose, we use the K-SVD [17] method. It iteratively alternates between two stages: (i) computing the sparse coding coefficients for the data based on the current dictionary by a pursuit algorithm; and (ii) updating the dictionary by singular value decomposition to achieve a better fitting to the data. Details of the K-SVD algorithm are shown in Algorithm 1.

3.3.3. Symbolic Representation

Similar to Yang et al. [11], we convert the input motion data into a symbolic sequence, or called motion string. Yang et al. [11] adopted spectral clustering to extract a set of representatives, and each original data sample can be approximated with one atom (for hard clustering) or more than one atom (for fuzzy clustering).

The raw Mocap data are of high dimensionality and contain much spatial and temporal redundancy [13]. Because

3.3. Symbolic Representation

specifically, we use a gene sequence to model one solution, with a “1”-gene meaning selection or a “0”-gene meaning nonselection of the corresponding candidate segmenting point. Importantly, we define a fitness function for each solution to measure the quality of segmentation. The fitness function is defined in a way to favor simple motions and long repetitive patterns in each segment.

It is worth noting that the genetic algorithm sets up a generic framework, and other data representation schemes and fitness functions can also be designed to achieve different effects and meet various application requirements.

A genetic algorithm approach to motion segmentation

**Algorithm 1 K-SVD Algorithm**

<table>
<thead>
<tr>
<th>Input:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix $M \in R^{3q \times n}$, $L \in \mathbb{N}^+$, $K \in \mathbb{N}^+$,</td>
</tr>
<tr>
<td>$L$ is the number of non-zero elements in each column,</td>
</tr>
<tr>
<td>$K$ is the number of dictionary atoms;</td>
</tr>
<tr>
<td>Output:</td>
</tr>
<tr>
<td>the dictionary $D \in R^{3q \times K}$,</td>
</tr>
<tr>
<td>the weight matrix $W \in R^{K \times n}$;</td>
</tr>
<tr>
<td>1: Initialize $D$;</td>
</tr>
<tr>
<td>2: repeat</td>
</tr>
<tr>
<td>3: /* Sparse coding stage */</td>
</tr>
<tr>
<td>4: Use OMP [16] to get $W$ by solving Equation (1);</td>
</tr>
<tr>
<td>5: /* Dictionary learning stage */</td>
</tr>
<tr>
<td>6: for each column $k \in [1,K]$ in $D$ do</td>
</tr>
<tr>
<td>7: $w_k = {i \mid 1 \leq i \leq n, W_{ki} \neq 0}$;</td>
</tr>
<tr>
<td>8: Compute the overall representation error matrix $E_k = Y - \sum_{j \neq k} D_j W_j$;</td>
</tr>
<tr>
<td>9: Obtain $E_k^R$ by restricting $E_k$ to the columns in $w_k$</td>
</tr>
<tr>
<td>$E_k^R = (E_k)_{w_k}$;</td>
</tr>
<tr>
<td>10: Apply SVD decomposition $E_k^R = U \Delta V^T$;</td>
</tr>
<tr>
<td>11: Update dictionary $D_{wk} = U_{w_k}$;</td>
</tr>
<tr>
<td>12: Update weight matrix $W_{kwj} = \Delta_{11} V_{w_k}$;</td>
</tr>
<tr>
<td>13: end for</td>
</tr>
<tr>
<td>14: until convergence or an iteration number limit is reached</td>
</tr>
<tr>
<td>15: return $D$ and $W$</td>
</tr>
</tbody>
</table>

to compute the similarity between any two data samples to construct a similarity matrix.

### 3.4. Motion Pattern Analysis

We analyze the motion string in order to extract sub-strings with certain semantic meanings. As Yang et al. [11], we think that repetitive occurrences of symbolic patterns signify semantic behavioral units. Therefore, we identify segments in the motion string, each of which contains repetitively occurring symbolic patterns and potentially corresponds to a certain behavioral unit. When we conduct the final motion data segmentation, we hope to preserve those segments.

Specifically, we search for two types of sub-strings. The first type contains a continuous sequence of the same symbol (e.g., “aaaaa”), while the second type contains repetitive occurrences of the same sequence of symbols (e.g., “abcabcabcabc”). The symbol in the former case (e.g., “a”) is called a static pattern, and the symbol sequence in the latter case (e.g., “abc”) is called a repetitive pattern. Sub-strings of the former type signify a human keeping static or repeat the same simple motion for a period, while sub-strings of the latter type reflect the cyclic nature of some human motions.

Initially, we search in the motion string for consecutive occurrences of each static pattern whose length is above a threshold, $T$. Then, from the remaining part of the motion string, we search for consecutive occurrences of each repetitive pattern. When searching for repetitive patterns, we ignore the exact length of each character’s consecutive occurrences, observing that the same action may be made in different posture duration patterns even by the same person at different times. That is, we reduce both “aaaaaaaaabbbb” and “aaabbbbbbb” to “aab”. On the reduced motion string, we then employ the suffix array technique to retrieve the repetitive patterns and their occurrences and in turn, map these patterns and occurrences back to corresponding clips in the raw sequence.

### 3.5. Motion String Segmentation by Genetic Algorithm

#### 3.5.1. Genetic Algorithm Basics.

The genetic algorithm was invented by John Holland [18] and Kenneth DeJong [19]. As a global random search and optimization method, it mimics the natural selection process in the biological evolution system. A genetic algorithm works in iterations. First, it initializes (often randomly) the first generation of individuals, each expressed as a gene sequence. For each individual, its fitness is computed. Thereafter, a series of operators (selection, crossover, and mutation) are conducted: the individuals with the highest fitness are selected for crossover to generate children for the next generation, and mutation is conducted by randomly flipping genes in some sequences such that randomness is introduced to the next generation. After several generations of evolution, the individual with the highest fitness becomes an approximate optimal solution.

#### 3.5.2. Genetic Modeling of Motion String Segmentation.

Genetic algorithm is naturally fit for the motion string segmentation problem. For each motion string, if we use one bit, “1” or “0”, to indicate whether there is a segmentation point at each frame, each bit is called a gene and all these bits concatenated together will form a gene sequence. By selection and crossover, the best local segmentation results of one generation will be propagated to the next, leading to gradually improved global segmentation results.

Straightforwardly, a gene is used for each frame, and the initial generation is created in a completely random fashion. However, this will lead to an explosive search space, considering that even a short MoCap sequence easily contains tens of thousands of frames. Further, the thus-generated initial generation may be of low quality and contain invalid individuals, leading to slow convergence of the optimization process. For instance, a randomly generated individual may contain consecutive “1” bits indicating
a consecutive sequence of segmenting points, which is usually unreasonable.

Instead, we first identify a small set of candidate segmenting points and apply the genetic algorithm only on them. In other words, we use one gene, “1” or “0”, to indicate whether one candidate is taken as a final segmenting point in the solution, and all these bits are concatenated to form the gene sequence. The candidate segmenting point set is determined on the basis of the results of motion pattern analysis as described in Section 3.4. By doing this, we reduce the search space and improve the quality of the first generation at the same time.

3.5.3. Candidate Segmenting Points.

As described in Section 3.4, repetitive occurrences of patterns have been retrieved from the motion string, which potentially correspond to elementary behavioral units. We hope to preserve the integrity of those elementary behavioral units in the segmentation. Therefore, we define the candidate segmenting points in the following way.

Consecutive occurrences of each pattern, static or repetitive, form a motion clip that we call stable motion clip. All these stable motion clips will break the raw MoCap sequence (or the motion string, correspondingly) into segments, and a segment not corresponding to any stable motion clip forms an unstable motion clip. Finally, we record the starting frame number of all the motion clips except the first one and put them in an ascending order to form the candidate segmenting points.

3.5.4. The Fitness Function.

We design the fitness function to evaluate the quality of each segmentation so that the populations can evolve in the rising direction of the fitness to finally reach an optimal segmentation.

We mainly consider two aspects in evaluating the quality of a segmentation. First, we prefer to have simple motions in a segment, or in other words, we do not favor highly irregular and variant postures in a segment. The information entropy measures the amount of information, so we take the opposite entropy of the motion string in the fitness function. Second, we prefer to have cyclic motion patterns in a segment, which tend to convey meaningful behavioral units. Therefore, we design the fitness function, Q, of a motion string segmentation, S = {S_i | 1 ≤ i ≤ s}, as

\[ Q(S) = \sum_{i=1}^{s} \left( -E(S_i) + w \times \frac{L_r(S_i)}{L(S_i)} \right) \]  

(2)

where \( E(.) \), \( L_r(.) \), and \( L(.) \) return the entropy of a segment, the length of stable motion clips contained in a segment, and the total length of a segment, respectively. For instance, for \( S_1 = \text{“abefgkjd”} \), \( L(S_1) = 8 \) and \( L_r(S_1) = 0 \) because there is no stable motion clip in \( S_1 \); for \( S_2 = \text{“abcdaabdcd”} \), \( L(S_2) = 8 \) and \( L_r(S_2) = 8 \) because \( S_2 \) contains a stable motion clip, “abcdaabdcd” whose length is 8; for \( S_3 = \text{“aaaaaabcdefg”} \), \( L(S_3) = 14 \) and \( L_r(S_3) = 11 \) because \( S_3 \) contains stable motion clips, “aaaaa” and “efgefg” whose total length is 11. In Equation (2), w is a user-specifiable weight parameter that we empirically set to 0.5.

4. EXPERIMENTAL RESULTS

4.1. Data Set

We select eight long motion sequences from Carnegie Mellon University [20] database for experiment. They are trials 1~8 from subject 86, which we denote as sequences 1~8 in the following. The lengths of these sequences range from 4579 to 10,617 frames, and the average is 8733 frames. Each of these sequences contains several types of motions, such as walking, squatting, running, stretching, jumping, punching, knee kicking, drinking, and arm swinging.

4.2. Benchmarks and Settings

We experimentally compare our method with the HACA [7], ACA [6], and LDA [12]. Further, we make efforts in a coarse-grain comparison with Yang et al. [11], which is also based on the motion string representation, although it is hard for us to exactly reproduce their algorithm and results, especially because manual labeling is involved in their method. In addition, we compare different clustering methods in the same genetic algorithm framework to demonstrate the advantages of the sparse learning-based method over the K-means and the spectral clustering ones for our problem.

In order to obtain the ground-truth data for algorithm evaluation, we have invited five subjects and asked them to manually segment the eight motion sequences. Although the five subjects have made slightly different segmentation results on each motion sequence, we view all of their segmenting points as reasonable and retain them as the ground-truth reference.

We implement the proposed algorithm in C++ and run it on a computer with an Intel(R) (Dell, Jinan, China) Core(TM)2 Duo 2.93-GHz CPU and 4-GB memory to obtain the experimental data as reported in Section 4.3.

4.3. Results and Analysis

Experimental comparison of our method with the other ones are made in different aspects: segmenting points distribution, error statistics, and timing statistics. Further, we visually illustrate the segmentation results on selected motion sequences.

4.3.1. Segmenting Points Distribution.

We use a sparse learning-based method for the motion string construction. However, because Yang et al. [11] employed the spectral clustering technique to obtain the motion string, we also implement a similar spectral clustering process for the motion string construction for the
Figure 2. Candidate segmenting points distributions for spectral clustering and our method, final segmenting points distributions for our method and the ground truth.

Figure 3. Segmenting points distributions for different methods on the test data set.

purpose of comparison. On the motion string, both our work and the work of Yang et al. [11] extract consecutive occurrences of static or repetitive patterns to obtain a candidate segmentation in similar ways. Thereafter, Yang et al. [11] needed manual behavior labeling to guide the final segmentation, but the final segmentation may be quite consistent with the candidate one in many cases except that semantic labels are added. Differently, our method automatically optimizes the candidate segmentation through the genetic algorithm.

As shown in Figure 2, on two motion sequences, we plot the distributions of the candidate segmenting points obtained with the spectral clustering (SC initial), the sparse learning-based method (ours initial), the distributions of the final segmenting points obtained with our method (ours final), and the manually labeled ground truth (manual). From Figure 2, we see that the candidate segmenting points obtained with our method are already of good quality, which are further refined through the optimization. By contrast, the candidate segmenting points obtained with spectral clustering are more scattered and farther away from the ground truth.

Further, we compare our method with HACA [7], ACA [6], and LDA [12] and compare different clustering methods within the same genetic algorithm-based framework that we propose. The segmenting point distributions with these different methods on the eight test motion sequences are illustrated in Figure 3. For all the sequences, we consistently observe the best segmentation accuracy with our method.

4.3.2. Error Statistics.

For a given motion sequence of $n$ frames, denoting the ground-truth segmenting point set as $g = \{g_i|1 \leq g_i \leq n, j = 1, 2, \ldots, n\}$, and a segmenting point set resultant from any method as $r = \{r_i|1 \leq r_i \leq n, j = 1, 2, \ldots, n\}$, we need a metric to quantitatively measure the segmentation error of $r$. For that purpose, we employ the normalized symmetric Hausdorff distance, $e$, between $g$ and $r$ as detailed in the succeeding text.

The segmenting point sets, $g$ and $r$, can be viewed as two 1D point sets. In general, the Hausdorff distance from set $A$ to set $B$ is defined as

$$h(A, B) = \max_{a \in A} \min_{b \in B} d(a, b)$$  \hspace{1cm} (3)$$

where $d(a, b)$ measures the distance between $a$ and $b$. For our problem, the normalized symmetric Hausdorff distance

---

A genetic algorithm approach to motion segmentation

The normalized symmetric Hausdorff distance of each sequence

Figure 4. Error statistics for different methods on the eight motion sequences.

Table I. Timing statistics of different algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>ART (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-means-based GA</td>
<td>2.88</td>
</tr>
<tr>
<td>Spectral clustering-based GA</td>
<td>14.00</td>
</tr>
<tr>
<td>LDA [12]</td>
<td>30.8</td>
</tr>
<tr>
<td>ACA [6]</td>
<td>4.95</td>
</tr>
<tr>
<td>HACA [7]</td>
<td>0.94</td>
</tr>
<tr>
<td>Ours</td>
<td>4.05</td>
</tr>
</tbody>
</table>

ART, average running time; GA, genetic algorithm; LDA, latent Dirichlet allocation; ACA, aligned cluster analysis; HACA, hierarchical aligned cluster analysis.

\[ e(r, g) = \frac{1}{2n} (h(r, g) + h(g, r)) \] (4)

Following the formula in Equation (4), we compute the segmentation errors for different methods on the test motion sequences and plot the error statistics in Figure 4. From this figure, we again observe the clear advantage of our method in producing low segmentation errors.

4.3.3. Timing Statistics.

We measure the average running time of different algorithms on the eight test data sets, as shown in Table I. From this table, we see that spectral clustering-based genetic algorithm and LDA are relatively slow mainly because the similarity matrix construction and the comparison between each pose and each motion vocabulary pose are highly time consuming. ACA and HACA are much faster because they perform on the temporally reduced motion sequences. Our scheme goes between ACA and HACA in terms of running efficiency.

4.3.4. Selected Visual Results.

We show the detailed string segmentation process and results on three selected motion sequences.

Figure 5. Segmentation results of sequence 2.
A genetic algorithm approach to motion segmentation

N. Lv et al.

Figure 6. Segmentation results of sequence 3.

Figure 7. Segmentation results of sequence 6.

The first sequence is sequence 2 with 10,617 frames. We obtain the motion string 
\(\text{"a"}^{1047}\text{"b"}^{84}\text{"c"}^{142}\text{"d"}^{130}\text{"e"}^{154}\text{"f"}^{132}\ldots\text{"s"}\) 
after the sparse learning, where a superscript represents the number of consecutive occurrences. The static and repetitive patterns are then found by the method described in Section 3.4, which in turn leads to candidate segmenting points on the raw motion string. For sequence 2, the set of motion patterns detected is {\text{"a"}, \text{"bc"}^{3}, \text{"d"}, \text{"e"}, \text{"ab"}^{4}, \text{"abc"}^{4}, \text{"a"}, \text{"f"}, \text{"ghih"}^{3}, \text{"g"}, \text{"aj"}^{2}, \text{"a"}, \text{"s"}}, where a superscript again signifies the number of consecutive occurrences. The second motion sequence is sequence 3 with 8401 frames, for which we obtain 14 motion patterns, that is, \{\text{"a"}, \text{"b"}, \text{"ab"}, \text{"acde"}^{2}, \text{"a"}, \text{"eafg"}, \text{"ea"}^{3}, \text{"cgfba"}^{2}, \text{"cdgijfcbacgfd"}, \text{"ca"}^{2}, \text{"cbac"}, \text{"aklm"}^{5}, \text{"ak"}, \text{"a"}\}. The third motion sequence is sequence 6 with 9939 frames, for which we obtain 13 motion patterns, that is, \{\text{"a"}, \text{"bcbd"}^{3}, \text{"ba"}, \text{"befd"}^{3}, \text{"bg"}^{2}, \text{"b"}, \text{"gbabjklnmoibajklmpnobiakjklmnpnioim"}, \text{"bikc"}^{3}, \text{"bhi"}^{3}, \text{"bar"}^{2}, \text{"a"}, \text{"a"}\}. All the corresponding candidate segmenting points in the raw motion string then provide the basis for further optimization using the genetic algorithm.

The final segmentation results for the aforementioned sequences are respectively shown in Figures 5, 6, and 7. For each segment, a representative image is shown on the left, and the corresponding motion type, reduced motion sub-string, and starting–ending frame numbers are shown on the right. Note that the motion type is manually made but not detected by our algorithm, just to help the illustration.

5. CONCLUSION AND FUTURE WORK

In this work, we have proposed a novel genetic algorithm approach to human MoCap data segmentation. We first conduct sparse learning on the raw motion data to derive a dictionary of representative postures and convert the raw motion sequence into a symbolic sequence. Thereafter, on the symbolic sequence, we conduct string analysis to extract repetitively occurring patterns, producing a set of
candidate segmenting points. Finally, on the candidate segmenting points, the genetic algorithm is applied to obtain the optimal solution of segmentation. As experimentally demonstrated, the proposed scheme yields outstanding accuracy and efficiency of segmentation.

In the future, we will explore introducing other rules based on kinematic characteristics and statistical characteristics to further enhance the optimization process.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China (61303083, 61173079, U1035004, and U1201258), the Scientific Research Foundation for the Excellent Middle-Aged and Youth Scientists of Shandong Province of China (BS2011DX017), Shandong Provincial Natural Science Foundation, China (ZR2011FZ003, ZR2011FZ004), and the Program for New Century Excellent Talents in University (NCET) in China.

REFERENCES


AUTHORS’ BIOGRAPHIES

Na Lv was born in 1979 in Jinan, China. She received her BS and MS degrees in Computer Software and Theory from Shandong University in 2001 and 2004. Currently, she is a PhD student at the School of Computer Science and Technology, Shandong University since 2010, and a teacher in the Department of Information Science and Technology, Shandong University since 2010, and a teacher in the Department of Information Science and Technology, Shandong University since 2010, and a teacher in the Department of Information Science and Technology, Shandong University since 2010, and a teacher in the Department of Information Science and Technology, Shandong University since 2010, and a teacher in the Department of Information Science and Technology, Shandong University.
Science and Technology, University of Jinan, China since 2004. Her main research interests are in computer graphics, and motion capture data processing and analysis.

**Yan Huang** received the PhD degree in computer science from the University of California, Irvine in 2009, the MS degree in computer science from the University of California, Irvine in 2003, and the BS degree in computer science from Peking University in 1997. Currently, she is an associate professor in the School of Computer Science and Technology, Shandong University, China. Her research interest mainly resides in large-scale 3D data visualization and intelligent multimedia data analysis.

**Zhiquan Feng** is a professor of Jinan University. He obtained the master’s degree from Northwestern Polytechnical University, China in 1995, and PhD degree from Computer Science and Engineering Department, Shandong University in 2006. He has published more than 50 papers on international journals. His research interests include human hand tracking/recognition/interaction and human computer interaction.

**Jingliang Peng** received the PhD degree in electrical engineering from the University of Southern California in 2006, the BS and MS degrees in computer science from Peking University in 1997 and 2000, respectively. Currently, he is a professor in the School of Computer Science and Technology, Shandong University, China. His research interest mainly resides in digital geometry processing, 3D animation, and image/video analysis.