Example-Based Automatic Music-Driven Conventional Dance Motion Synthesis

Rukun Fan, Songhua Xu, and Weidong Geng

Abstract—We introduce a novel method for synthesizing dance motions that follow the emotions and contents of a piece of input music. Our method employs a learning-based approach to model the music to motion mapping relationship embodied in example dance motions along with those motions' accompanying background music. A key step in our method is to train a music to motion matching quality rating function through learning the music to motion mapping relationship exhibited in synchronized music and dance motion data, which were captured from professional human dance performance. To generate an optimal sequence of dance motion segments to match with a piece of input music, we introduce a constraint-based dynamic programming procedure. This procedure considers both music to motion matching quality and visual smoothness of a resultant dance motion sequence. We also introduce a two-way evaluation strategy, coupled with a GPU-based implementation, through which we can execute the dynamic programming process in parallel, resulting in significant speedup. To evaluate the effectiveness of our method, we quantitatively compare the dance motions synthesized by our method with motion synthesis results by several peer methods, by using the motions captured from professional human dancers' performance as the gold standard. We also conducted several medium-scale user studies to explore how perceptually our dance motion synthesis method can outperform existing methods in synthesizing dance motions to match with a piece of input music. These user studies produced very positive results on our music-driven dance motion synthesis experiments for several Asian dance genres, confirming the advantages of our method.

Index Terms—Dance motion and music mapping relationship, music-driven dance motion synthesis, learning-based dance motion synthesis.

1 Introduction

Dancing [1] to music is a highly appreciated artistic skill of human beings. Many people enjoy moving their bodies improvisationally while listening to music. In comparison, professional choreographers dance with rhythms and gestures to carefully match the rhythm and content of the background music, aiming at delivering the same kind of emotions as conveyed by the music. Because of this high expectation on the synchronization between dance motions and the accompanying music, motions in professional dance performance are usually thoughtfully designed prior to their live stage performance. The task of designing such dance motions is often noted as dance notation or choreography, which demands much expertise and talent. In reality, to plan dance motion for a new piece of music, choreographers rarely create the entire dance motion sequence from scratches. Rather, they often tend to reuse dance motion segments that have been carefully planned in the past for similar music pieces. Through reusing these previously successful dance motion case examples, the task of designing dance motions can be much more efficiently accomplished. Witnessing such practices of choreographers in reality, in this paper, we introduce an algorithmic method capable of synthesizing dance motions for an input piece of music through a learning-based approach. Fig. 1 shows some sample dance motions automatically generated by our method for the same piece of input music but with virtual characters dancing in different genres. It is noted that our study on automatically synthesizing quality dance motions to match with the background music has wide applications in multimedia and digital entertainment—for many low end to middle end computer animation and video game applications, our system can automatically generate dance motions according to the input music to replace the conventional labor intensive and tedious manual dance motion authoring process.

To automatically synthesize dance motions according to the input music, we need to resolve several computational challenges. First, we need to identify a compact set of salient motion and music features which can reliably reveal motion and music characteristics. Successful extraction of these features can facilitate accurate content analysis of the music and dance motion data. Given these discriminative features, our second challenge is to build a realistic computational model to capture the inherent music to motion mapping relationship exhibited in professional dance performance. Lastly, once the music to motion mapping relationship is captured, we need to apply the modeled relationship to optimally synthesize a dance motion sequence that best matches the input music. Unlike most of the previous work, such as [20], [1], [34], which used much manual work to first select reliable motion and music features, and then to build the music to motion mapping relationship model based on

1. Rukun Fan and Songhua Xu contributed equally to this work.

- *R. Fan is with the Department of Computer Science, The University of North Carolina at Chapel Hill, Chapel Hill, NC 27599. E-mail: fanrk@cs.unc.edu.*
- *S. Xu is with the Oak Ridge National Laboratory, One Bethel Valley Road, Oak Ridge, TN 37831. E-mail: xus1@ornl.gov.*
- *W. Geng is with the College of Computer Science, Zhejiang University (Yuquan Campus), Hangzhou 310027, P.R. China. E-mail: gengwd@zju.edu.cn.*

Manuscript received 6 May 2010; revised 23 Feb. 2011; accepted 2 Mar. 2011; published online 12 Apr. 2011.
Recommended for acceptance by R. Boulic.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2010-05-0096.
Digital Object Identifier no. 10.1109/TVCG.2011.73.

1077-2626/11/$26.00 © 2011 IEEE Published by the IEEE Computer Society

Fig. 1. Virtual characters dancing in different genres to follow the same piece of input music. From left to right, we show snapshots of dance motions synthesized by our method for the same moment in the music piece for characters dancing in the Chinese, Tibetan, Uighur, and Mongolian dance genres, respectively.

the manually identified features, in this paper, we adopt a machine learning-based approach to systematically select the most reliable and characteristic motion and music features to model the interrelationships between dance motions and their accompanying music. We also follow a learning-based approach to automatically establish the most accurate music to motion mapping model for each dance genre. Benefited from the learning nature of our method, we can automate both the feature selection process and the music to motion mapping relationship establishment process without involving any empirical knowledge-based manual work. However, developing a system implementation that includes a most comprehensive set of motion and music features to attain the highest end system performance is simply not our main purpose in this work since the prototype system we implemented is only for proof-of-concept.

Overall, our music-driven dance motion synthesis method consists of a training phase and a generation phase, whose architecture is overviewed in Fig. 2. In the training phase, sample dance motions with their synchronized background music are first captured from professional human dance performance. These sample dance data are then segmented into smaller pairs of synchronized motion and music segments (short for "motion-music pair"). The resultant motion-music pairs are used to estimate the music to motion matching quality through a boosted learning-based procedure. Since all the candidate music to motion correlation coefficients constitute a very high dimensional space, we first extract an optimal subset of these correlation coefficients for efficient and effective music to motion matching quality estimation. Given this reduced set of music and motion correlation coefficients, we optimally train a rating function which can evaluate how well a candidate motion segment matches with a given input music segment. In this training stage, we also construct a motion graph to efficiently find smoothly transiting motion segments. In the generation phase, our system first segments an input music piece into smaller music segments according to the rhythm of the music. We then adopt a two-way dynamic programming procedure to identify an optimal sequence of motion segments from the candidate motion collection, which could best match with the input music piece. The advantage of this two-way dynamic programming procedure is that it is suited for parallel execution. This feature especially benefits dance motion synthesis for a piece of long music. During the dynamic programming process, our motion synthesis algorithm considers both the quality of matching of a motion-music pair, which is evaluated by the music to motion matching quality rating function we trained, and the transition

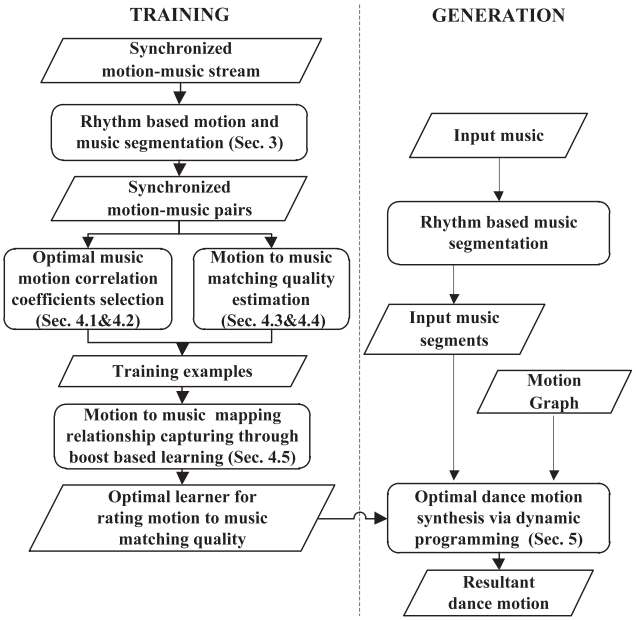


Fig. 2. System architecture.

smoothness in the motion synthesis result between every pair of adjacent motion segments, which is ensured by the motion graph. Lastly, we apply motion concatenation, warping, and blending to further refine the visual quality of the synthesized dance motion.

The remainder of this paper is organized as follows: Section 2 surveys some most related work to our study here. Section 3 describes our automatic salient music and motion feature extraction process. Section 4 explains how we capture the music to motion mapping relationship via a learning-based approach. Section 5 introduces how we generate an optimal dance motion sequence according to the input music via a dynamic programming procedure. Section 6 shows some selected experimental results. Finally, we conclude the paper in Section 7.

2 Related Work

There exists abundant work relating to our music-driven dance motion synthesis studies in this paper. Bregler et al. [3] introduced an automatic method for editing facial motions in videos. Their work implements an acoustic feature-driven mouth motion editing function. Cardle et al. [7] presented a framework for synchronizing music to motion by locally modifying motions using perceptual music clues. Lee and Lee [20] used a dynamic programming approach to synchronize animation with its background music by scaling both the music and the motion parts. Shiratori et al. [34] synthesized dance motions according to the rhythm and intensity of the input music. Kim et al. [5] designed a matching process by considering the correspondence of the relative changes in both the music and the motion feature spaces as well as the correlations between musical and motion features. Fan et al. [43] suggested a rhythm-based motion-music matching model for synthesizing dance animation. In all above works, the music to motion relationship is manually specified. Chen and Li [8] designed

a procedural animation system by changing the tempo, exaggeration, and speed of Chinese lion dance. Neff et al. [26] presented a system for animating humanoid characters using correlation maps. Their system allows animation authoring through conventional 2D mouse and keyboard. Hsu et al. [13] introduced a method for controlling human motion through an example-based approach, where the mapping between human motion control command and the target motion is defined following an example-based approach. Ren et al. [4] applied an Adaboost-based algorithm to learn a mapping relationship between silhouettes captured from three video cameras to 3D human motions. Given the mapping relationship, they successfully transformed 2D silhouette signals into 3D human motions.

One of the key problems in motion synthesis is to identify a sequence of smoothly transiting motion segments from the candidate motion collection to generate visually continuous motion synthesis result. For this purpose, motion graph has been proposed as an efficient solution [17], [21], [2]. Kim et al. [16] modified the traditional motion graph design by additionally considering the kinematic continuity and behavioral continuity between motion segments during the graph construction process. Zhao and Safonova [40] proposed a well-connected motion graph to facilitate the generation of well-connected motions with smooth transitions. Li et al. [22] introduced a novel technique called motion texture for synthesizing complex human-figure motions which are statistically similar to the originally captured motion data. Brand and Hertzmann [6], Grochow et al. [11], and Hsu et al. [14] all studied the motion style transfer problem which involves intensive motion feature analysis and synthesis, both closely related to our study here.

Another major problem in our dance motion synthesis study in this paper is to extract most characteristic music and dance motion features for establishing a quality music and dance motion mapping model. A general approach to identifying a sequence of motion segments to match with the input music is to first extract the motion and music features, respectively, and then for each music segment, select the best matching motion segment according to the music to motion mapping model. For the first problem of salient motion and music feature identification, most existing work manually choose the salient motion and music features according to empirical knowledge, e.g., [20], [1]. For the second problem of motion and music mapping relationship modeling, it is also usually handled manually. For example, the work by Shiratori et al. [34] constructs the relationship using the assumption that the rhythm and intensity of dance motions shall be synchronized to that of music. Unlike these manual efforts for motion and music feature identification and mapping relationship modeling, one of the most similar work to our study here is the learning-based dance motion generation system proposed by Oore and Akiyama [28]. They introduced a neural network-based learning approach to capture the relationship between motion and music features according to training examples. However, different from our method, the motion and music features in their method are all manually chosen; and their method can only generate dance motions for arms. Another piece of closely related work to our study here is the learning-based dance motion generation system proposed by Ofli et al. [42]. Following a HMM-based approach, they proposed a mapping from music measures to dance figures to represent the correlations between dance motions and music. Different from our method, their method requires manual labeling over a collection of dance figures in advance, which overall follows a supervised learning approach. In contrast, our method is fully automatic, capable of learning music-motion relationships without any human intervention. Such a feature of our algorithm makes our approach well suited for fully automatically handling new dance genres, as long as there are some decent amount of training data available. As a result, our algorithm is easy to set up and well suited for scaling up.

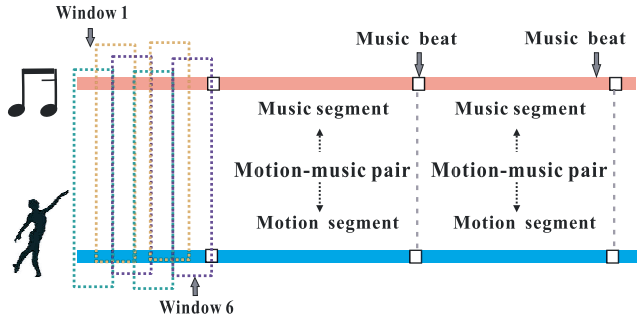


Fig. 3. Partitioning synchronized motion and music data into synchronized motion-music pairs. The motion and music division positions happen at the music beat points. Each motion-music pair is further divided into six overlapping windows.

3 Extracting Music and Motion Features

3.1 Rhythm-Based Music and Motion Segmentation

The primitive elements used in our dance motion synthesis are music and motion segments, which typically last between half to one second. Carrying out our synthesis on this granularity rather than working with the entire motion sequence for a whole piece of music can significantly improve the number of reusable motion elements. Inspired by the studies by Kim et al. [16], which showed that most dance motions exhibit some kind of rhythmic patterns, we use rhythm as a common feature between music and dance motions to produce synchronized motion-music pairs. The resultant motion-music pairs constitute the training set for our learning-based music-driven dance motion synthesis pipeline. To comprehensively capture the mapping relationship between motion and music for a dance genre, we need to extract a set of revealing music and motion features. In this paper, we define these features on the music and motion segment level.

In our current system implementation, we employed the music beat tracking method proposed in [9] to segment a sequence of synchronized motion and music data into a collection of motion-music pairs. We choose this method due to its computation efficiency in locating beat positions as well as its decent performance. Applying their method, motion and music signals are segmented at the music beat positions (see Fig. 3). Since the motion and music data are synchronized, we hence assume it suffices to detect the beat positions of music alone in the segmentation process. To capture the dynamic evolution of the motion and music, we further divide each motion-music pair uniformly into six windows. Each window lasts 2/7 of the length of a motion-music pair. Considering that the primitive elements typically last between half to one second, we choose six windows to obtain

enough details of music data without incurring too high computational overhead. Fifty percent overlapping between every pair of adjacent windows within the six windows is proposed in prior work in audio analysis [45] (see Fig. 3).

It should be noted that it is also easy to reconfigure our system to adopt a larger time step as our analysis primitive because our algorithm accepts any granularity. However, using a larger time step for analysis may miss some fine details of motion and/or music, hence running the risk of losing potential informative and insightful clues that reveal the intricate internal relationships between dance motions and music. For different dancing genres, the optimal choice of analysis granularity is certainly likely to vary. We assume segmentation by music beat is a reasonable choice as rhythm provides the natural link between motion and music, which has been widely advocated by many previous methods, e.g., [1], [16], [34]. Among the five typical Asian dance genres we experimented with in this paper, a beat usually contains $3 \sim 4$ salient movement poses, within which period a dancer can typically finish a basic sequence of dance actions, such as turning around (see our demo video, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2011.73). Here, we define salient movement pose frames as those frames where the dancer moves his or her body more intensely than in other frames.

3.2 Extracting Music Features

Prior studies have suggested a large number of candidate features for characterizing both motion and music data. However, exhaustively including all these features will incur too high computational overhead to our prototype system. Hence, we selected a subset of most salient motion and music features according to recommendations by prior work in the area, e.g., [30], [19], [16], [28]. In this paper, we adopt the following eight basic music features widely used in audio analysis: amplitude envelope, spectrum, cepstrum, spectrum histogram, periodicity histogram, fluctuation pattern, root-mean-square energy, and low-energy-rate. The definition and application of these features are jointly provided by [30] and [19]. For each of the first six types of basic music features in the above, we further extract the following six statistical features: zero-crossing rate [46], centroid [47], spread [48], kurtosis [48], flatness [49], and entropy [50] according to audio features over all frames in a window. For the latter two types of music features, each gives a scalar feature value for a music segment. These statistical features are frequently used in the audio analysis literature [30]. Overall, this results in a music feature vector with a total dimensionality of $6 \times 6 + 2 = 38$, which is defined over a window's period. We normalize each of these features to the range of 0 and 1 by dividing by the maximum value of the corresponding feature for all the music pieces used in our experiments. We represent the music feature vector for music segment $\mathbf{A}_i$ as $\mathbf{F}_i^a$, and the music feature vector for the kth window in the music segment $\mathbf{A}_i$ as $\mathbf{F}_{i,k}^a$.

3.3 Extracting Motion Features

In prior studies, Oore and Akiyama [28] used the distances of hands from the centroid of human body to characterize arm motions. Kim et al. [16] used the angular velocity of the joints for motion rhythm analysis. In theory, we can use either angular velocities or translational velocities; in practice, we chose to use joint velocities since they facilitate more convenient posture control for motion synthesis [16]. Inspired by both work [28], [16], we extend their motion feature definitions to cover a set of key joints in our human motion model. Our extended motion features cover the following four groups of joint positions: right and left shoulders, right and left elbows, right and left hips, right and left knees. The importance of these joint positions in characterizing human body motions has been intensively studied in [21] and [37], respectively. In addition, we also include the following three joint positions: left and right ankles, and a joint representing the hips, due to their popular inclusion in human body models. For each such joint, at every motion frame, we extract the distance from the joint to the body centroid as well as the angular velocity of the joint as two types of motion features for the joint. For the latter feature of angular joint velocity, we use the norm of the joint velocity as the feature value. Thus, each frame of our dance motion is associated with $11 \times 2 = 22$ motion feature values. Based on these individual frames' motion feature values, we can further extract each feature's mean, median, variance, and also the mean, median, and variance of the feature's first order forward finite difference. Therefore, for either one of the two motion features, we obtain six statistic values. Since there are 11 selected joints, each joint is associated with two features, and each feature produces six statistic values, this leads to a total of $11 \times 2 \times 6 = 132$ feature values for dance motions in a window. For each of these 132 motion features, we also normalize the feature value to the range of 0 and 1. We organize these motion features for the motion segment $\mathbf{M}_i$ as $\mathbf{F}_i^m$ and also organize motion features for the kth window of the motion segment $\mathbf{M}_i$ as the motion feature vector $\mathbf{F}_{i,k}^m$.

Our motion feature extraction method can derive most of the motion features used in previous methods. For instance, Kim et al. [16] extracted the moments of rapid direction changes in a dance motion as the key features for characterizing postures of articulated figures. Their features are essentially the variance of the angular velocity of a joint, which has been covered in our extracted features. Lee et al. [20] calculated the acceleration of the foot joints as their motion features. Such features can also be derived by applying first order forward finite difference on the foot angular velocity, which have also been included in our motion feature set. Shiratori et al. [34] derived their motion features using Laban's weight effort component [18]. Their features can be approximately derived through linearly summing up the angular velocities of all the key joints, as currently handled in our method.

4 Capturing Music to Motion Mapping Relationship

Once a collection of training examples in terms of synchronized motion-music pairs have been prepared, we can capture the motion to music mapping relationship reflected by these examples through a learning-based approach.

Mathematically, we attempt to train a rating function $S(\mathbf{M}_i, \mathbf{A}_j)$ which can evaluate the matching quality for an arbitrary pair of synchronized motion-music pair $(\mathbf{M}_i, \mathbf{A}_j)$ in terms of their matching error. Here $\mathbf{M}_i$ denotes the ith motion segment, $\mathbf{A}_j$ denotes the jth music segment. Function $S(,)$ is a real value function with range [0,1]. The smaller $S(\mathbf{M}_i, \mathbf{A}_j)$ is, the better the motion segment $\mathbf{M}_i$ is considered in its matching with the music segment $\mathbf{A}_j$. Below we will look at how to optimally train such a rating function for a specific dance genre according to the learning examples.

4.1 Extracting Music to Motion Correlation Coefficients from Training Examples

In our method, each synchronized motion-music pair defines a learning example, which is represented as $(\mathbf{M}_i, \mathbf{A}_j) = \{\mathbf{F}^m_{i,1}, \ldots, \mathbf{F}^m_{i,6}, \mathbf{F}^a_{j,1}, \ldots, \mathbf{F}^a_{j,6}\}$, forming a matrix. This matrix notation means that the overall feature set of a motion segment, F^m_i, is composed of feature sets of six corresponding motion windows in the segment. We use the same notation for F^a_j. Recall that there are always six windows in a motion/music segment. To better capture the correlation between the synchronized motion and music segment pair, we derive the linear correlation coefficients between motion and music features since it is an effective and widely used method for revealing correlation between two variables. Let $\mathbf{F}^m_{i,k}(p)$ be the pth feature value in the kth window of the ith motion segment $\mathbf{M}_i$, and $\mathbf{F}^a_{j,k}(q)$ be the qth feature value in the kth window of the jth music segment $\mathbf{A}_j$. Also let $\mathbf{X}$ denote the random variable whose observations are the motion feature values $\mathbf{F}^m_{i,1}(p), \ldots, \mathbf{F}^m_{i,6}(p)$, and $\mathbf{Y}$ denote the random variable whose observations are the music feature values $\mathbf{F}^a_{j,1}(q), \ldots, \mathbf{F}^a_{j,6}(q)$. Given a training example $(\mathbf{M}_i, \mathbf{A}_j)$, and for each individual motion feature p over $\mathbf{M}_i$ and music feature q over $\mathbf{A}_j$, we can derive the correlation coefficient between them as: $C(p,q) = \frac{E[(\mathbf{X}-\mu_\mathbf{X})(\mathbf{Y}-\mu_\mathbf{Y})]}{\delta_\mathbf{X}\delta_\mathbf{Y}}$, where $\mu_\mathbf{X}$, $\mu_\mathbf{Y}$, $\delta_\mathbf{X}$, and $\delta_\mathbf{Y}$ are the mathematical expectations and standard deviations of $\mathbf{X}$ and $\mathbf{Y}$, respectively; $E()$ computes the mathematical expectation.

As explained in Section 3, for each synchronized motion-music pair, we extract 38 music features and 132 motion features. Deriving the correlation between every pair of these features would produce an overall of $38 \times 132 = 5{,}016$ correlation coefficients, which can be represented as a correlation matrix.

4.2 Optimally Selecting Music-Motion Correlation Coefficients

Each of the above 5,016 correlation coefficients may suggest a clue on music to motion mapping relationship. However, taking into account all of them will incur a forbiddingly expensive computing cost (see Fig. 4). In addition, some of the correlation coefficients may not help reliably indicate the motion to music mapping relationship or could even introduce additional noise to the music to motion relationship capturing process. Therefore, we employ an optimal feature selection procedure to identify a representative subset of coefficients to reduce the overhead and also to avoid the possible unreliability caused by unrelated coefficients when establishing our music to motion mapping relationship.

In our method, we adopted the feature selection method proposed by Peng et al. [32] which uses the minimal-redundancy-maximal-relevance criterion to select a compact set of most essential features. Since their original method is designed for features with discrete values, to apply their method, we additionally introduced an entropy measure-based discretization procedure [25] to convert the continuous valued music to motion correlation coefficients into discrete values before applying Peng et al.'s feature selection method.

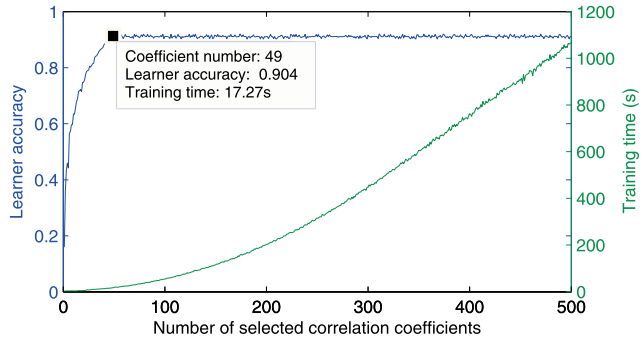


Fig. 4. Relationship between learner accuracy, training time, and number of selected correlation coefficients for a Korean dance motion synthesis task. The total length of the Korean motion-music pairs used in training is 73.4 seconds. The black point corresponds to the optimally selected number of correlation coefficients. Here the learner accuracy is measured according to (9).

Fig. 4 illustrates the influence of the number of correlation coefficients selected through the above process on the learner accuracy and the total training time required. This figure shows that selecting a small portion of those most salient correlation coefficients is sufficient to obtain a satisfying learning accuracy, which also saves tremendous amount of training time. In our system, we choose the number of the correlation coefficients according to the leaner accuracy for each dance genre. The termination condition is that when we admit seven additional correlation coefficients into the selected feature set, the relative learner accuracy should increase by at least 3 percent. We start from an empty feature set and keep including new features into the set until the above condition cannot be met. Here the learner accuracy is defined as the accuracy of the boosted version of the best performing learning method, which will be explained in Section 4.5.

4.3 Preparing Training Examples

To collect positive training examples, all the sample dance motion-music pairs captured from professional human dancers' performance would naturally qualify. To collect negative training examples, among all the captured motion segments, we randomly select one segment and pair it with a randomly selected music segment. Note that in this random motion and music segments pairing process, both the motion and music segments shall be of the same genre. That is, no Korean dance motion segment shall be paired with a Chinese music segment. We impose this constraint because the motion and music relationship model we intend to construct is dance genre specific. For each motion-music pair prepared this way, we then assign the music to motion matching error score as the label of the training example pair. The value range of the label is between 0 and 1. The smaller the label value is, the better matched the music and motion segments are. For the

motion and music pairs resulting from original professional dancer's performance, the label value is always set to 0.

Requesting human annotators to come up with a consistent manual labeling over the matching quality of all the training motion-music pairs is nontrivial and difficult. Thus we introduce the following motion similarity-based approach to derive the music to motion matching errors for labeling training examples. The assumption we adopted here is that dancers are prone to perform similar dance motions while listening to similar music. This assumption can be intuitively validated by observation in real-world dance learning scenarios: the coach often corrects postures of student dancers to make their dance postures closely follow or resemble the instructor's "standard" dance postures. The more similarly a student dancer can imitate the instructor's dance movements and postures, the faster and better a learner the student is considered as. This assumption is also quantitatively validated through a formal user study, to be reported in Section 6.3.2.

Mathematically, given a pair of music to motion matching pair $(\mathbf{A}_i, \mathbf{M}_i)$, captured from the dance performance by a professional human dancer, we always assume its music to motion matching error is 0. For a new dance motion $\mathbf{M}_j$, we will measure its distance to $\mathbf{M}_i$. The larger the distance between $\mathbf{M}_j$ and $\mathbf{M}_i$ is, the less well matched we assume the pair of $\mathbf{A}_i$ and $\mathbf{M}_j$ is. Now the problem of estimating the matching quality between a pair of music and motion segments is reduced to evaluating the distance between two motion segments. For this purpose, we introduce the function $Dist_m(\mathbf{M}_i, \mathbf{M}_j)$ [15] to measure the pairwise motion distance between two motion segments $\mathbf{M}_i$ and $\mathbf{M}_j$, whose definition will be introduced shortly in Section 4.4. Given $Dist_m(\mathbf{M}_i, \mathbf{M}_j)$, we can estimate the music to motion matching error score as follows:

$$L(\mathbf{A}_i, \mathbf{M}_j) \triangleq 1 - \exp(-Dist_m(\mathbf{M}_i, \mathbf{M}_j)). \quad (1)$$

4.4 Measuring Pairwise Motion Distance

Measuring the distance between two motion segments is the key for judging the quality of a dance motion synthesis algorithm. As mentioned earlier, in our system, we also use this motion distance measurement to derive labels for the training data set. To calculate the distance between two motion segments, we first calculate the distance between two motion frames. The latter is done using the method proposed in [21], which takes into account both the positions and velocities of the key joints in a human body. The weights associated with each joint are optimally tuned using the method proposed in [37]. Once the frame level posture distance is defined, we can use the method described in [15] to compute the distance between two motion segments based on the difference in body postures of all the individual frames. This method uses dynamic time warping during its motion distance measurement, which has the advantage of supporting motion distance comparison between motion segments of different lengths. Adopting their motion distance measurement method, for any pair of motion segments $\mathbf{M}_i, \mathbf{M}_j$, we can compute their distance as $Dist_m(\mathbf{M}_i, \mathbf{M}_j)$.

For some dance genres, a dancer might make a left-right symmetric counterpart movement during her dance performance. These are likely to be far apart in terms of motion distance and so would be determined as a poor fit by the above method, whereas they are in fact both good fit for the input music. To address this problem, we additionally introduce the following processing: suppose $Dist_o(\mathbf{M}_i, \mathbf{M}_j)$ denotes the original distance between motion segments $\mathbf{M}_i$ and $\mathbf{M}_j$, and $Dist_s(\mathbf{M}_i, \mathbf{M}_j)$ denotes the distance between $\mathbf{M}_i$ and the left-right symmetric motion of $\mathbf{M}_j$. We then take the minimum of $Dist_o(\mathbf{M}_i, \mathbf{M}_j)$ and $Dist_s(\mathbf{M}_i, \mathbf{M}_j)$ as the eventual value for $Dist_m(\mathbf{M}_i, \mathbf{M}_j)$.

TABLE 1
Seven Base Learners Used to Capture the Music to Motion Mapping Relationship

Name	Meaning
linear	Linear model with optional ridge regression [12]
perceptron	Nørgaard's perceptron, trained with Levenberg-Marquart [27]
vicinal	K-nearest-neighbor regression with adaptive metric [23]
lssvm	Johan Suykens' least-square SVM toolbox [36]
ares	Friedman's MARS algorithm [10]
mpmr	Thomas Strohmann's mimimax probability machine regression [35]
rbf	Mark Orr's radial basis function [29]

4.5 Capturing Music to Motion Mapping Relationship

Once we have prepared all the learning examples on the motion to music mapping relationship for a specific dance genre, we can use these examples to optimally train a learner to capture the mapping relationship. The number of correlation coefficients used for characterizing each training sample is determined by the procedure explained in Section 4.1. We use seven base learners for the learning task, which are listed in Table 1. In our experiments, we use the implementations of these learning models offered by the open-source software toolbox, ENTOOL [38]. Once these base learners are trained, we further apply the AdaBoost algorithm [33] to boost the learning performance through the optimal combination of these base learners. Since the classical AdaBoost algorithm only deals with learning examples with binary labels, we adopt the extended AdaBoost learning scheme proposed by Zhu et al. [41] to deal with the real valued learning examples in our case. The learning accuracy is measured using (2) and evaluated with the ten folded cross validation strategy

$$Accuracy \triangleq 1 - \frac{1}{T}\sum_{i=1}^{T} |S(\mathbf{A}_i, \mathbf{M}_{r_i}) - L(\mathbf{A}_i, \mathbf{M}_{r_i})|, \quad (2)$$

where $(\mathbf{A}_i, \mathbf{M}_{r_i})$ is an arbitrary testing example; $S(\mathbf{A}_i, \mathbf{M}_{r_i})$ is the music to motion matching error score predicted by our trained learner, which is introduced at the beginning of Section 4; $L(\mathbf{A}_i, \mathbf{M}_{r_i})$ is the groundtruth music to motion matching error of the pair $(\mathbf{A}_i, \mathbf{M}_{r_i})$ according to (1). Here T is the total number of testing examples for a dance genre. Table 2 gives the key statistics on the learning data used in our experiment for each dance genre. Table 3 reports the learning accuracy of each of the seven base learners, along with the boosted learner for every dance genre. From these experiment results, we can clearly see that our learning

TABLE 2
Example Dance Data Used in Our Training Process

Genre	Korean	Tibetan	Uighur	Dai	Chinese	Mongolian
No. of Songs	5	4	4	4	5	5
No. of Pairs	194	172	168	133	247	151
Duration (s)	163.2	146.6	146.1	126.5	171.8	170.4
Std (Duration)	0.33	0.34	0.16	0.25	0.23	0.29
Std (Pairs)	7.6	9.5	11.8	8.7	9.9	8.1

The second, third, and fourth rows list the number of songs, total number of motion-music pairs, and their lengths in seconds in these training data, respectively. We further list the standard deviation of "Duration" in the fifth row and standard deviation of "No. of Pairs in each song" in the sixth row.

TABLE 3
Learning Accuracy for Capturing the Music to Motion Mapping Relationship for Six Dance Genres

Genre	Korean	Tibetan	Uighur	Dai	Chinese	Mongolian
linear	0.897	0.892	0.837	0.918	0.891	0.863
perceptron	0.902	0.911	0.902	0.911	0.894	0.849
vicinal	0.892	0.905	0.789	0.898	0.841	0.857
lssvm	0.899	0.910	**0.915**	0.911	0.905	0.835
ares	0.781	0.878	0.874	0.881	0.851	0.798
mpmr	**0.904**	0.913	0.897	**0.920**	0.903	0.888
rbf	0.901	**0.915**	0.896	0.868	**0.910**	**0.892**
AdaBoost	0.935	0.930	0.929	0.935	0.937	0.929

Here, we show the performance of our seven base learners as well as the combined learning method enhanced by the Adaboost strategy for capturing the music to motion matching relationship for Korean, Tibetan, Uighur, Dai, Chinese, and Mongolian dances, respectively. Here the learning accuracy is measured according to (2).

method with the boosted learning scheme can satisfyingly capture the numeric relationships between motion and music as reflected from the training data.

5 Music-Driven Dance Motion Synthesis

5.1 Preprocessing

To prepare our pipeline for dance motion synthesis, we first construct a motion graph containing all the available candidate motion segments. We applied the motion graph construction method proposed by Zhao and Safonova [40] because it can generate a graph with good connectivity and smooth transitions. We use the metric proposed in [37] for computing transition cost between a pair of adjacent motion segments. This metric considers both the weighted differences between joint angles and joint velocities, which also carries a weighting parameter balancing the measurement over the two types of joint differences. To acquire candidate motion segments for constructing the motion graph, dance motions with synchronized music are segmented according to music rhythm analysis [9]; dance motions without accompanying music are segmented through the motion rhythm analysis [16]. The benefit to include motion segments without accompanying music is to have access to a richer collection of motion segments for constructing a well-connected motion graph. Restricting our motion graph only to motion segments with synchronized music will greatly reduce the scope of candidate motion choices during motion synthesis process. In our constructed motion graph, each motion segment represented in the graph is also labeled with the genre of the dance motion. This additional node property in the motion graph allows either full traversal across the graph or partial traversal only over those nodes corresponding to a certain dance genre. We introduce this property because in our work we assume the music to motion mapping relationship is genre specific. Hence, genre-based node selection or filtering is often useful.

5.2 Objective Function for Music-Driven Dance Motion Synthesis

To generate dance motion for an input music piece, we consider two objectives, i.e., the quality of matching between dance motions and the input music and the motion transition smoothness. Accordingly, the objective function we constructed for guiding the music-driven dance motion synthesis consists of two parts, each dedicated to one of the above two objectives. This objective function leads us to conjecture that if we can predetermine those music segments that could only match with a unique motion segment, we can then divide the whole input music piece into multiple parts at the positions of these highly restricted music segments. Such an operation will allow us to synthesize dance motions for each part of the input music independently and in parallel, earning significant speedup of our algorithm. We will come back to this point shortly.

Formally, given a part of input music piece which contains n music segments $\{\mathbf{A}_1, \ldots, \mathbf{A}_n\}$, our goal is to find an optimal sequence of motion segments $\{\mathbf{M}_{u_1}, \ldots, \mathbf{M}_{u_n}\}$ where $\mathbf{M}_{u_i}$ is the motion segment corresponding to the music segment $\mathbf{A}_i$. In particular, the starting and ending motion segments, $\mathbf{M}_{u_1}$ and $\mathbf{M}_{u_n}$, may have been predetermined. Our goal is to minimize the following objective function:

$$F(n, u_1, u_2) \triangleq \sum_{j=1}^{n} R(j, u_j) + \gamma \sum_{j=2}^{n} T(u_{j-1}, u_j). \qquad (3)$$

In the above equation, $R(j, u_j)$ is the simplified notation of $S(\mathbf{A}_j, \mathbf{M}_{u_j})$ introduced at the beginning of Section 4.5, which measures the matching error between the music segment $\mathbf{A}_j$ and the motion segment $\mathbf{M}_{u_j}$; $T(u_{j-1}, u_j)$ is the transition cost from the motion segment $\mathbf{M}_{u_{j-1}}$ to the segment $\mathbf{M}_{u_j}$, whose value can be obtained by querying the motion graph. γ is a modulation parameter balancing the two considerations, which can be either empirically tuned by users or automatically acquired according to the training data. A typical setting for γ as used in all our experiments is 0.6.

5.3 Minimizing the Objective Function via Parallel Dynamic Programming

Assuming there are w candidate motion segments in total for a music-driven dance motion synthesis task. The input music consists of n music segments. To find an optimal motion sequence to match the input music using brutal force exhaustive search will run into a space of w^n candidate solutions. To avoid the exponential search space, we thus turn to dynamic programming to find the optimal solution. Further, we also introduce a two-way dynamic programming framework to allow parallel execution of the dynamic programming procedure. As mentioned earlier, in some cases, the whole input music piece can be divided into several parts, where each part can be separately analyzed to

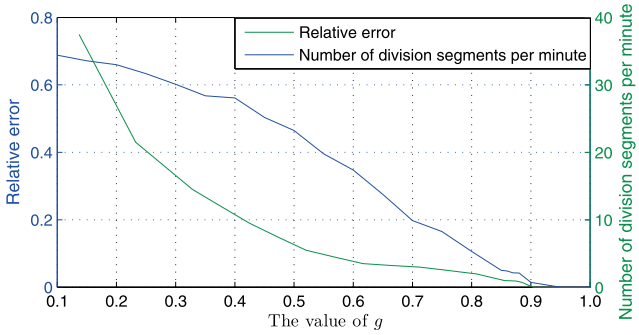


Fig. 5. The relationship between the g value, the relative error of our algorithm, and the number of division segments. The data come from analyzing 30 randomly chosen songs which last 920 seconds in total. The relative error is calculated as the absolute error of synthesized result using two-way dynamic programming divided by the absolute error of synthesized result using traditional dynamic programming. The absolute error is measured according to (9).

generate its accompanying dance motion. Ideally, if a music segment can only match a unique motion segment, dividing the music piece at the music segment will not affect the solution quality of the dynamic programming procedure at all. To optimally divide the music piece, we try to find those music segments which have a high likelihood to correspond to a unique motion segment. Once the starting and ending motion segments for a music piece or subpiece are known, we can apply our two-way dynamic programming procedure to find optimal dance motion sequence for each music subpiece in parallel.

To minimize the objective function, our algorithm consists of four stages:

1. we first precompute the matching quality scores for all the music-motion pairs;
2. we then divide the input music piece into multiple parts by identifying those music segments which likely only correspond to a unique motion segment;
3. after that, we perform a two-way dynamic programming procedure to find the optimal motion sequence for each part of the input music; and
4. finally, we synthesize a smooth motion sequence for the whole input music piece based on all the identified motion segments.

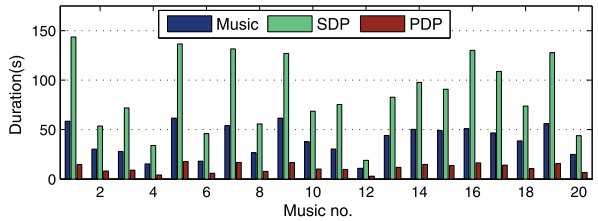
5.3.1 Dividing Music Piece into Multiple Parts

To divide the input music piece into multiple segments, we attempt to identify those music segments that have a high likelihood to match with a unique dance motion segment. We call such music segments the *division segments*. The criterion we adopted for identifying such division segments is that among all the candidate motion segments for a music segment, there exists a motion segment whose motion-music matching score is significantly higher than all the other candidate motion segments. Mathematically, for a music segment, we denote the absolute difference between the best music-motion matching score and the second best music-motion matching score as g. Note that the value range for a music-motion matching score is between 0 and 1. Fig. 5 shows the relationship between the value of g, the number of division segments identified under the specific g value,

TABLE 4
Runtime Breakdown in Different Stages of the Serial Dynamic Programming Implementation (Short for "SDP") and the Parallel Dynamic Programming Implementation (Short for "PDP") of Our Algorithm

Method	g-value	Stage 1	Stage 2	Stage 3	Stage 4	Time	Error
SDP	-	273.6	0.7	10.8	2.2	287.3	-
PDP	0.89	28.8	0.7	10.8	2.2	42.5	0.00
PDP	0.88	28.8	0.7	7.2	2.2	38.9	0.02
PDP	0.87	28.8	0.7	5.7	2.2	37.4	0.04
PDP	0.86	28.8	0.7	5.5	2.2	37.2	0.05
PDP	0.85	28.8	0.7	5.4	2.2	37.1	0.05
PDP	0.80	28.8	0.7	3.6	2.2	35.3	0.10
PDP	0.75	28.8	0.7	3.1	2.2	34.8	0.16
PDP	0.70	28.8	0.7	2.7	2.2	34.4	0.21
PDP	0.65	28.8	0.7	2.6	2.2	34.3	0.28

(a) Run time breakdown for different stages of a dance motion synthesis experiment, where the input music piece lasts for 117 seconds.



(b) Bar graph of run time comparison for 20 dance motion synthesis experiments. In all the experiments, the g-value is set to 0.86. We report the length of input music used in each experiment, along with the overall computation time by the SDP and PDP versions of our algorithm implementation respectively.

In this experiment, a total of 537 candidate motion segments are involved. The "Error" metric is the relative error of dance motion synthesis, as used in Fig. 5. We analyze the performance of our algorithm under different g-values.

and the relative error of our dance motion synthesis algorithm. In all our dance motion synthesis experiments reported in this paper, we empirically set the g value as 0.86.

Also at this step, we calculate all the matching scores between every pair of dance motion segment and input music segment. These precomputed music-motion pair matching scores are very useful for accelerating the dynamic programming process. Considering the independence between each motion-music pair, we make use of GPU programming toolkit CUDA to accelerate the coefficients calculation process. The calculation of matching quality score for each motion-music pair is assigned to a GPU thread. In our current system implementation, we use the GTX 285 Graphics card, which allows 23,040 active threads simultaneously. This means we can compute the matching quality scores for up to 23,040 music-motion pairs all at once. In the experiment reported in Table 4, the total number of candidate motion segments is 537 and the input music contains 100 segments. Hence, we need to evaluate matching quality scores for 53,700 music-motion pairs. This can be done using only three runs of our GPU-based evaluation procedure. It is easy to see such a GPU-based parallel music-motion matching quality score evaluation approach brings a major time saving, which is confirmed by the timing numbers reported in Table 4.

5.3.2 Two-Way Dynamic Programming Procedure

We assume that the first and the last motion segments for this music piece have been prechosen as $\mathbf{M}_{u_{k0}}$ and $\mathbf{M}_{u_{k1}}$, respectively. We then introduce two auxiliary functions $H_F(i, u_j, u_f)$ and $H_B(i, u_j, u_b)$: $H_F(i, u_j, u_f)$ is the forward

objective function, which denotes the minimum cost for the first to the ith music segments matching with i motion segments, with the constraint that the first and last motion segments must be $\mathbf{M}_{u_f}$ and $\mathbf{M}_{u_j}$, respectively. $H_B(i, u_j, u_b)$ is the backward objective function, which denotes the minimum cost of matching the nth (the last music segment) to the ith music segments with $n - i + 1$ motion segments, with the constraint that the starting and ending motion segments must be $\mathbf{M}_{u_j}$ and $\mathbf{M}_{u_b}$, respectively. $H_F(i, u_j, u_f)$ searches from the first motion segment $\mathbf{M}_{u_f}$ forward to an intermediate motion segment $\mathbf{M}_{u_j}$; $H_B(i, u_j, u_b)$ searches from the ending motion segment $\mathbf{M}_{u_b}$ backward to an intermediate motion segment $\mathbf{M}_{u_j}$.

Let $F_{\min}(n, u_{k_0}, u_{k_1})$ be the minimum value of the objective function $F(n, u_{k_0}, u_{k_1})$ in (3) when all the n music segments have been optimally matched with n corresponding motion segments, with the first and last motion segments fixed to be u_{k_0} and u_{k_1}, respectively. Now we can formulate the following iterative equations, in the form of a dynamic programming procedure, to find the value of $F_{\min}(n, u_{k_0}, u_{k_1})$

$$F_{\min}(n, u_{k_0}, u_{k_1}) = \min\{H_F(\eta, u_j, u_{k_0}) + H_B(\eta, u_j, u_{k_1}) \mid j = 1, \ldots, w; \eta = 1, \ldots, n\}, \tag{4}$$

$$H_F(1, u_j, u_{k_0}) = \begin{cases} R(1, u_j) & \text{if } j = k_0 \\ +\infty & \text{else} \end{cases} \quad (j = 1, \ldots, w), \tag{5}$$

$$H_F(i+1, u_j, u_{k_0}) = R(i+1, u_j) + \min\{(H_F(i, u_t, u_{k_0}) + \gamma T(u_t, u_j) \mid t = 1, \ldots, w\} \quad (i = 1, \ldots, n-1; j = 1, \ldots, w), \tag{6}$$

$$H_B(n, u_j, u_{k_1}) = \begin{cases} R(n, u_j) & \text{if } j = k_1 \\ +\infty & \text{else} \end{cases} \quad (j = 1, \ldots, w), \tag{7}$$

$$H_B(i-1, u_j, u_{k_1}) = R(i-1, u_j) + \min\{(H_B(i, u_t, u_{k_1}) + \gamma T(u_t, u_j) \mid t = 1, \ldots, w\} \quad (i = n, \ldots, 2; j = 1, \ldots, w). \tag{8}$$

Here, the function $R(,)$ and $T(,)$ have been previously introduced in Section 5.2. Equations (5) and (6) represent the forward searching process; and (7), (8) represent the backward searching process. Equation (5) initializes the base case for the forward searching process; while (7) initializes the base case for the backward searching process. Overall, the forward and backward searching processes in our two-way dynamic programming procedure meet at the ηth music segment. We test every possible η to find the optimal value for (4). The time complexity of our two-way dynamic programming process is $O(n \times w^2)$. Given that we have precomputed and stored the music-motion matching quality scores for all the possible music-motion pairs, the actual dynamic programming procedure runs very efficiently (see Table 4).

Note that if the ending motion segment, $\mathbf{M}_{u_{k1}}$, could not be predetermined, we will use the traditional one-way dynamic programming process to solve the problem by ignoring the backward search part, i.e., by ignoring the function $H_B(,,)$ in (4). This is because our problem degenerates to the traditional one-way dynamic programming scenario under this circumstance, in which case $H_B(,,)$ would have no effect on the formula. Then we can solve the degenerated problem using the answer retrieval technique of classical dynamic programming procedure.

Currently, our method generates an optimal solution without the trouble of any user involvement. We do recognize the importance of giving professional users the choice of low level control to allow them to achieve some specific desired artistic effects. Thanks to our two-way dynamic programming process, our system does support arbitrary level of user involvement: each time, when a user wants a particular moment of music to be associated with a particular dance pose or motion segment, the user can manually specify this requirement as a constraint. We call such a moment a user specified music to motion matching point, short for "user specified point." An end user can introduce as many user specified points as needed. Given all the user input, our algorithm can naturally apply its two-way dynamic programming process to synthesize motions for music segments between two adjacent user specified points.

5.3.3 Result Motion Synthesis

Once the optimal sequence $\{\mathbf{M}_{u_1}, \ldots, \mathbf{M}_{u_n}\}$ has been identified, we can synthesize the entire dance motion for the input music by concatenating all these motion segments according to their order in the sequence. After that, we carried out two additional postprocessing operations to further improve the visual quality of the synthesized dance motion: first, for every identified motion segment $\mathbf{M}_{u_i}$, whose length is not exactly the same as the length of its corresponding music segment $\mathbf{A}_i$, we would "scale" the motion segment to align with the music segment via motion warping [39]. After that, we apply motion blending [31] between every pair of adjacent motion segments in the synthesized result to make the overall dance motion generation result more smooth looking.

As previously mentioned, our music-driven dance motion synthesis algorithm consists of four stages, including music-motion matching quality score calculation, division music segments identification, optimal motion segment finding via two-way dynamic programming, and dance motion synthesis. The first stage is executed on GPU; while the rest stages are all executed on CPU. The match quality calculation stage evaluates the music-motion matching quality scores for all the music-motion pairs. Given the precomputed dance-motion matching quality scores, we can quickly identify those division segments from all the music segments with a high confidence. After that, the optimal motion segment finding stage performs the two-way dynamic programming procedure to identify an optimal sequence of dance motions to match with the input music. Finally, the dance motion synthesis stage generates a smooth motion sequence based on all the motion segments selected in the dynamic programming process. We compared the execution efficiency of our algorithm between its serial implementation (short for "SDP") and its parallel implementation (short for "PDP"), both coded in C++. In the PDP version, we use GTX 285 Graphics card and Intel Core i7 Duo 2.66 GHz CPU. The number of GPU threads we

used in the first stage is the maximum number of parallel threads supported by our graphics card, which is 23,040. The number of CPU threads we used in the two-way dynamic programming stage is the same as the number of cores inside the CPU, which is four in our setting. We present more details about our experiments in Table 4, from which we can see that the parallel version performs significantly faster than the serial implementation.

6 Experimental Results

6.1 Experiment Setup

We employed an optical 3D motion capture device to record example dance motions by professional human dancers along with their accompanying music. The professional human dance motions used in our experiments are recorded in the standard BVH format with 60 degrees of freedoms (DOF), which includes 24 joints where each joint has 1 to 3 DOFs depending on the visual importance of the joint in the dance performance. The example dance motions were captured at the rate of 30 frames per second. The accompanying music is input in the mono wave format with the sampling rate of 44.1 KHz. In our current experiments, we construct a motion graph containing 952 nodes in total, where each node corresponds to a motion segment. The overall duration of all the motion segments represented by our motion graph lasts 643 seconds. Example dance motions of six different dance genres are used during the training process, whose key statistics are reported in Table 2. It should be noted that not all the motion segments used in our training phase are included in the constructed motion graph as we discarded those motion segments with poor connectivity with the rest motions. In all these experiments, we used a desktop PC with an Intel Core i7 Duo 2.66 GHz CPU, 4 GB memory, and an NVIDIA GTX285 graphics card.

6.2 Comparison with Motion Data Captured from Professional Dance Performance

To evaluate the overall performance of our method, we compare the dance motions synthesized using our method and those by three other peer methods. Two comparison methods are proposed in [34] (short for "SHI") and [5] (short for "KIM"), respectively. To our best knowledge, KIM and SHI are the most recent dance motion synthesis work that most closely relate to our study here. We also implemented another baseline algorithm for comparison purpose, which is called the smooth motion synthesis method (short for "SMS"). SMS only considers the smoothness between motion transitions and ignores the music to motion matching quality during its dance motion synthesis process. We implement SMS as follows: each time when we need to identify a motion segment from the candidate motion collection to match a given music segment, we first find the top 1 percent of all the candidate motion segments which achieve the smoothest transition with the proceeding motion segment. We then randomly pick one of them to match with the current music segment. The initial postures of the human body in all the four methods are taken from the motion captured data of human dance performance, which always correspond to the posture where the dancer stands straight and still with his/her arms and legs rest vertically downward.

6.2.1 Quantitative Comparison with the Captured Human Dance Performance Data

To objectively evaluate the accuracy of our motion synthesis results, we introduce the following dance motion synthesis error metric to measure the difference between the synthesized dance motion and the motion captured data from professional human dance performance for the same music piece. For a sequence of input music segments, $\mathbf{A} = \{\mathbf{A}_1, \ldots, \mathbf{A}_n\}$, we denote its corresponding motion segment sequence in the captured human dance performance as $\{\mathbf{M}_1, \ldots, \mathbf{M}_n\}$. We can then define the following dance motion synthesis error score W on the matching quality between a synthesized dance motion sequence $\{\mathbf{M}_{u_1}, \ldots, \mathbf{M}_{u_n}\}$ with the input music sequence $\mathbf{A}$ through measuring the average motion distance between the sequence of synthesized motion segments $\{\mathbf{M}_{u_1}, \ldots, \mathbf{M}_{u_n}\}$ with the sequence of captured motion segments in the human dance performance $\{\mathbf{M}_1, \ldots, \mathbf{M}_n\}$, i.e.,

$$W = \sum_{k=1}^{n} Dist_m(\mathbf{M}_k, \mathbf{M}_{u_k})/n, \tag{9}$$

where $Dist_m(\mathbf{M}_k, \mathbf{M}_{u_k})$ is the distance between the two motion segments $\mathbf{M}_k$ and $\mathbf{M}_{u_k}$, which is introduced toward the end of Section 4.4.

To construct the testing data set, for each dance genre, we include five new songs outside the training data set. With this testing set, we compare the performance of the KIM method, SHI method, SMS method, and our method in terms of their dance motion synthesis errors using the metric (9). The results are reported in Table 5, from which we can clearly see that our method significantly outperforms the "KIM" method, "SHI" method, and also the baseline algorithm "SMS." We also illustrates these synthesis errors in a relative sense in Table 5b.

We also visually compare the synthesized dance motions with original human dance motions for the same input music piece. The results are reported in the Appendix of this paper, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2011.73. In all our comparison experiments, no part of the input music is included in our training data set.

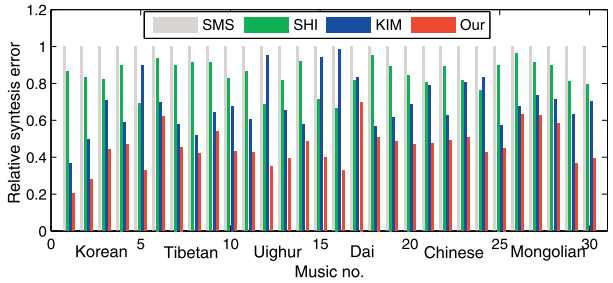
6.3 User Studies

To more objectively evaluate our method, we also carried out several medium-scale user studies by recruiting 42 fourth year undergraduate students from the digital media department of our university. Half of the participants are female and half are male. All the participants are senior grade students in our university's digital media department. Each of them had finished at least one course on digital music processing and another course on computer animation. The common education background of these participants is likely the main factor leading to the nice agreement among their individual grading results. Thus, to ensure that the participants of our user studies are indeed able to tell whether a motion segment and a music pair are

TABLE 5
Comparison between Synthesis Error Scores for 30 Randomly Selected Songs (Excluding All the Songs and Music Segments Used in Our Training Process) Using the KIM Method, SHI Method, SMS Method, and Our Method, Respectively

Korean						Tibetan					
#	Length	KIM	SHI	SMS	Our	#	Length	KIM	SHI	SMS	Our
1	24.8	1.09	2.56	2.97	0.60	6	6.93	0.75	1.01	1.08	0.67
2	13.5	0.85	1.43	1.72	0.48	7	3.35	0.50	0.77	0.86	0.39
3	22.1	2.91	3.39	4.12	1.83	8	8.62	0.21	0.37	0.40	0.17
4	15.2	1.20	1.83	2.03	0.95	9	5.37	0.67	0.95	1.04	0.56
5	23.3	21.2	17.0	24.6	8.04	10	9.70	2.03	2.48	3.00	1.29
Uighur						Dai					
#	Length	KIM	SHI	SMS	Our	#	Length	KIM	SHI	SMS	Our
11	15.4	4.23	6.05	7.01	2.97	16	8.15	5.25	3.55	5.32	1.74
12	6.83	0.82	0.59	0.86	0.30	17	5.03	0.50	0.49	0.62	0.43
13	12.9	3.32	4.14	5.06	2.00	18	5.07	0.41	0.69	0.73	0.37
14	5.55	8.31	13.2	14.4	6.94	19	7.85	0.22	0.32	0.36	0.17
15	23.7	2.18	1.65	2.31	0.92	20	5.22	0.51	0.63	0.74	0.35
Chinese						Mongolian					
#	Length	KIM	SHI	SMS	Our	#	Length	KIM	SHI	SMS	Our
21	17.9	1.85	1.89	2.35	1.12	26	31.8	12.3	17.5	18.2	11.5
22	15.1	0.71	1.02	1.14	0.56	27	32.8	3.67	4.56	4.99	3.12
23	10.7	10.6	10.7	13.1	6.68	28	23.3	1.78	2.24	2.49	1.45
24	37.6	5.84	5.33	6.99	2.99	29	29.3	2.77	3.54	4.37	1.59
25	8.42	0.76	1.20	1.33	0.60	30	12.7	0.64	0.73	0.92	0.36

(a) Synthesis error scores for 30 randomly selected songs using the error metric defined in (9). The first two columns show the song index number and length in seconds for each song used in our experiment respectively. We also illustrate these synthesis errors in a relative sense at (b).



(b) Relative dance motion synthesis errors for 30 randomly selected songs reported in (a). Here the relative error for a song is calculated as its absolute synthesis error as reported in (a) divided by the maximum dance motion synthesis error for this song among the four methods.

synchronized in their contents, we first conducted a participant capability screening test.

6.3.1 Participant Capability Screening Test

In our survey participant capability screening test, we first randomly picked six well-matched motion-music pairs from professional human dancers' performance data. And then we intentionally unsynchronized the motion-music pair by displacing the music part for half a beat, i.e., by advancing or delaying the music segment for half a beat with respect to the motion part. Here, we use the method proposed in [9] to detect a beat. Each original synchronized motion-music pair and its corresponding displaced unsynchronized pair are organized as a group. All together, we prepared six groups. For each group, every participant of our study was asked to identify between the two motion-music pairs in the group which one is the synchronized one. Only the participants who can successfully identify all the authentic motion-music pairs in the six groups were admitted in our actual user study. By doing this, we can exclude those candidates without a good sense to tell whether a music clip and a motion segment is well matched in terms of their contents and rhythm. We carefully filter out these unqualified subject candidates because including them in our user survey would only incur misleading noisy results. In our actual user study process, 30 out of the 42 students (17 male and 13 female students) passed our above test and were invited to participate in our actual user studies.

6.3.2 Validation of Our Basic Assumption

In our user study, we first validated one of our basic assumptions introduced in Section 4.3 for preparing the training examples of our learning-based approach—"given a pair of well-matched motion and music segments, the more we vary the motion part, the less well matched it is between the original music segment and the varied new motion segment." To explore the validity of this assumption, we conducted the following survey to compare the music-motion matching scores manually labeled by our human subjects and the matching quality scores generated using our algorithm using the above assumption. More concretely, for each dance genre out of six dance genres currently studied in our work, we randomly selected five training samples generated using the method described in Section 4.3 based on our above assumption. This produced 30 testing examples in total. For each example, we asked our participants to rate whether the motion part is well matched with its corresponding music part using a five-level Likert scale, (i.e., 4: strongly agree; 3: agree; 2: neither agree nor disagree; 1: disagree; 0: strongly disagree). A higher matching score indicates a better matching quality. For each of our 30 testing examples, we averaged the music to motion matching quality score manually provided by our 30 participants. Our testing examples covered all five types of dance genres and are all randomly chosen. We then compared these averaged scores with the ones generated by our algorithm through (1). Noticing that (1) gives the music to motion matching error score L ranging from 0 to 1, we thus derive its corresponding five-level music to motion matching quality score by uniformly discretizing the value of $1 - L$ into five levels. Both the human reviewers' manual scores and our algorithm's automatic scores are listed in Fig. 6. By comparing these two kinds of scores, we can clearly see that under most situations, our algorithm's results agree very well with human reviewers' manual rating scores. This validates our above basic assumption for collecting training examples in our method. It also shows that the music to motion matching quality scores rated by our trained learners are very close to the opinions of our human subjects.

6.3.3 Effectiveness of Our Approach

We also conducted a user study to explore whether our proposed new music-driven dance motion synthesis method can generate dance motions expressing better characteristics of an input music piece than other methods. This study includes testing of two assumptions:

- *Assumption* 1. Dance motion synthesized by our method better matches the input music in terms of their content characteristics and emotions.
- *Assumption* 2. Dance motion synthesized by our method along with the background music achieves a

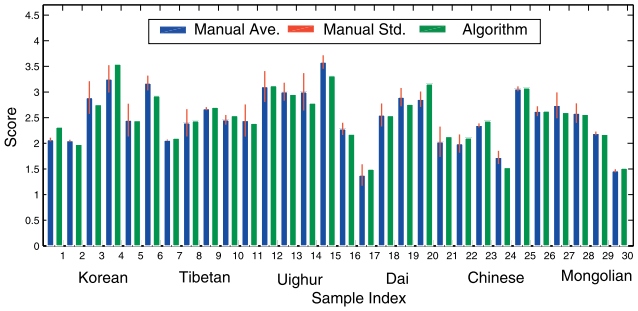
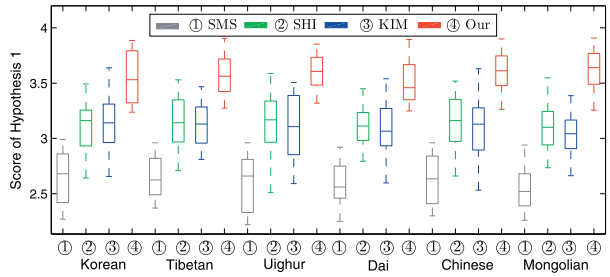


Fig. 6. Bar graph illustration between manual rating scores and algorithm generated scores on the motion-music matching quality for 30 randomly selected motion-music pairs.

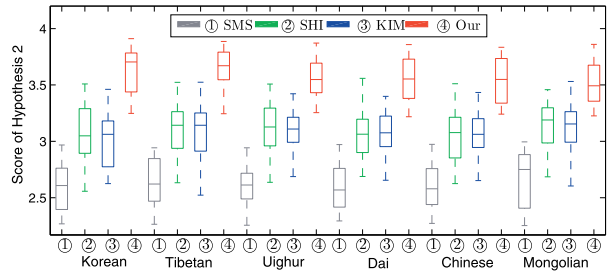
better overall impression and artistic effect on the audience.

For each participant, we show him/her six groups of dance videos. These six video groups, respectively, correspond to Korean, Tibetan, Uighur, Dai, Chinese, and Mongolian dance genres. Each group contains four dance videos, which are dance motions generated by the KIM, SHI, SMS, and our method, respectively, for the same input music. In our user survey, we asked each participant to answer two questions to test the above two assumptions through providing their answers using the five-level Likert scale. A higher score denotes a better user evaluation result. The performance of our subjects are reported in Fig. 7. During the user evaluation process, the actual method used to generate each dance motion video is kept confidential from the participants. We also analyzed the significance of our results using the analysis of variance method (short for "ANOVA"), which is also reported in Fig. 7. From these user survey results we can clearly see that there is no significant performance gap between the SHI method and the KIM method. This is probably because the music to motion relationships were manually specified in both approaches through best human efforts. According to the survey results on assumption 1 as reported in Fig. 7, the group opinions of our subjects clearly indicate that our method is capable of synthesizing dance motions better matching the input music in terms of their mutual content and emotions. Also, according to the survey results on assumption 2 as reported in Fig. 7, our subject group's opinions collectively indicate that the human dance motions synthesized by our method along with the background music can achieve a better overall visual effect and artistic impression than counterpart results produced by the other three peer methods.

To explore whether the participants in our studies are simply judging the motion-music synchronization quality or whether they are just judging the motion synthesis quality itself, we conducted another survey test in which participants were asked to score the overall impression of the dance motions synthesized by different methods but without hearing the accompanying music. In this new user survey, we show the dance clips used in the previous survey to our subjects; but without playing the background music this time. We conducted this new user survey ten days after conducting the original user survey when the accompanying music was played. The result is



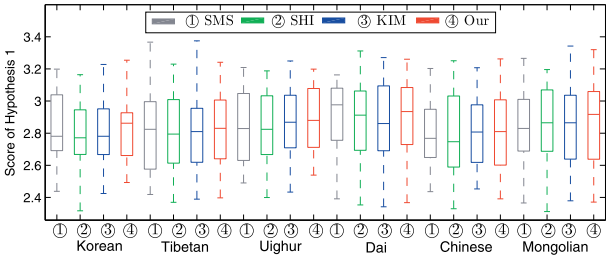
(a) Box plot of user study results on testing assumption 1. For each dance genre, we show user rating scores on the quality of dance motions generated by the SMS, SHI, KIM methods and our method respectively. In each box plot, the central line, the bottom and top edges of the box correspond to the median, 25th and 75th percentiles of the user survey data respectively. The bottom and top extended lines correspond to the minimum and maximum user survey data respectively.



(b) Box plot of user study results on testing assumption 2.

Fig. 7. User study results for testing two assumptions on our algorithm's effectiveness in comparison with three peer methods.

reported in Fig. 8. From these results we can see that between all the four methods, the overall user rating scores are fairly close to each other, meaning that the motion synthesis parts of the four methods in our implementation are quite comparable in generating smoothly transiting dance motions. This implies that, without considering the music to motion synchronization factor, there is no bias toward our method in terms of the quality of the synthesized dance motions. When compared with the survey results reported in Fig. 7, it can now be clearly seen that the advantage of our method in generating perceptually more impressive dance motions along with the background music comes principally from the quality of the synchronization between motion and music. In other words, the participants were truly judging the motion-music matching quality when conducting the user survey for testing assumption 2. We also notice that when including the background music, the overall perceived dance performance effect score has been increased for the SHI, KIM and our method, which suggests that by including the background music, the three methods can all produce better overall perceptual effect. However, for the SMS method, the overall impression score is reduced when including the background music due to its naive motion segment selection approach. In the appendix of this paper, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2011.73 we also reported the result of an additional user study,



(a) Box plot of user study results on testing assumption 2. For each dance genre, we show user rating scores on the quality of dance motions generated by the SMS, SHI, KIM methods and our method respectively. In each box plot, the central line, the bottom and top edges of the box correspond to the median, 25th and 75th percentiles of the user survey data respectively. The bottom and top extended lines correspond to the minimum and maximum user survey data respectively.

Fig. 8. User study results on testing assumption 2 when the background music is muted.

where some authentic human dance motion identification tests were conducted, to compare the performance of different dance motion synthesis methods, in terms of the resultant motions' matching quality with the input music. We further analyzed how participants in our user study based their reporting of emotional and aesthetic factors by asking each of them to provide a self-report, which are listed in the appendix of our paper, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2011.73.

7 Conclusion and Discussions

In this paper, we introduce a novel method for synthesizing dance motions which are closely synchronized with the input music via a learning-based approach. The key of our music-driven dance motion synthesis method is to model the music to motion mapping relationship exhibited in the sample dance motions along with their accompanying background music. Such sample data are captured from dance performance by professional choreographers. To establish the music to motion mapping relationship, we first identify a compact subset of music to motion correlation coefficients which can efficiently indicate the correlation relationship between dance motions and their accompanying music in the sample dance data. Once the subset of most reliable music to motion correlation coefficients is identified, we can train a music to motion matching quality rating function based on these correlation coefficients. Such a rating function is developed for an individual dance genre through learning examples of synchronized sample music to motion pairs of the genre. Given the matching quality rating function for a particular dance genre, we can apply the function to evaluate how well a candidate motion segment agrees with a music segment assuming the dancer is performing the corresponding dance genre. Finally, to generate an optimal sequence of dance motion segments to match with a piece of input music, we introduce a constraint-based dynamic programming procedure. This procedure considers both the motion to music matching quality as estimated by the music to motion matching quality rating function and the visual smoothness in the resultant dance motion sequence. We also introduced a two-way dynamic programming procedure, allowing parallel evaluation of the dynamic programming process, which is not possible with traditional dynamic programming techniques. This parallel feature results in significant speedup. We also implemented our dynamic programming procedure using GPU-based hardware acceleration. The above two parallel efforts enable our algorithm to be capable of synthesizing dance motions in real time.

The main advantage of our new dance motion synthesis method is that it generates dance motion according to the input music using automatically modeled music to motion mapping relationship according to the training data. Compared with prior methods [5] and [34] where the music to motion relationship is manually specified, our learning-based approach can more comprehensively and precisely capture the mapping relationship for each type of dance genre without heavy manual work, yet with optimized performance. Such automation in capturing the music to motion mapping relationship enables our method to be easily applicable for new dance genres when a decent amount of training data of human dance performance of the genre is available. Internal to our learning-based procedure for capturing music to motion mapping relationship, we utilize an AdaBoost-based scheme to boost the learning capabilities of several base learners for optimized performance in modeling the music to motion mapping relationship. Experimental results have successfully demonstrated that our new learning-based dance motion synthesis approach can generate dance motions for the input music with significant improvement over previous methods, which can be witnessed through both objective comparison with the motion captured data of professional human dancers' performance and also via several medium scale user studies we conducted. We also collected a self-reported feedback survey by practicing animators to explore the effectiveness of our implemented prototype system for dance motion generation, which can be found in the appendix of our paper, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2011.73.

To better understand the novelty and our algorithmic contribution in this work, we would also like to compare our method with several closely related approaches proposed in prior studies. In the work of [21], nine visual features are extracted from human body silhouettes, which are then used to retrieve motions from motion database for animation generation. It is noted these features are all manually selected, which is different from our automatic feature selection process in this work. Similarly, in the work of [20], [1], [34], [43], a set of motion and music features are first manually selected for constructing a music to motion mapping relationship model, the process of which is also manually done. Different from their practice, in our work, we apply machine learning techniques to automatically select the most reliable and characteristic motion and music features and also to automatically establish the most

accurate music to motion mapping model for each dance genre individually in an optimized way.

Compared with the work of [4], The mapping relationship between silhouette and human motion as studied in [4] is very different from the motion and music mapping relationship studied in our work. Since the 2D silhouette is a projected view of the 3D human motion, its relationship with respect to the human motion is much more natural to establish than the relationship between background music and human dance motion, because the latter relationship straddles across two medias. Also, the music content space has a much higher dimensionality than that of the 2D silhouette space. This makes our task of establishing a reliable relationship between music and human motion computationally a much harder problem, due to the curse of dimensionality.

We speculate the experimental success of our music-driven dance motion synthesis algorithm over the five traditional dancing genres as reported in this paper is mainly due to the fact that these dance genres exhibit well-recognized performance rules and convention that dancers generally follow in their personal practice. Such performance rules and convention are developed across time. Probably only those most stable and representative dance performance patterns that embody unique emotional and aesthetic characteristics of a traditional dance genre are reserved during the evolution of the genre through history; other less stylistic performance elements have been lost in time. Our above assumption can be intuitively verified by observing that all traditional dance genres demonstrate certain conventional dance patterns of their own. For example, Mongolian dance always tends to exercise violent movements, matching well with typical Mongolian music that is often loud, sonorous, and in high tone. In this case, our proposed method works satisfactorily in identifying motions for dance genres where the dance movement and background music coherently exhibit strong stylistic performance convention. Consequently, genres such as break dance that do not carry the above property are beyond the synthesis capability of our algorithm. Despite this limit, it is worth noticing that even for break dance, not any arbitrary motion segment can be used to compose its dance performance. Typically, break dancers do not perform soft and gentle dance motions as frequently exercised in ball dance and ballet. In this sense, there is already a prescreening and prior selection process that filters out a large number of candidate motion segments unsuitable for break dance, which to some extent ensures the style consistence between music and motions of break dance. Within the remaining subset of candidate motion segments, which are largely restricted in terms of movement and posture, random matching may suffice to generate satisfying solutions. From this perspective, our algorithm will only consider typically exercised break dance motion segments during the music-driven dance motion synthesis process. Hence, even for break dance, our algorithm is still partially useful for automatically generating matching dance motions. We acknowledge the importance of many additional salient music and motion features, which are not currently utilized in our system. However, it would be very expensive and unnecessary to exhaustively include all the available features to build a prototype system for demonstrating the effectiveness of our algorithm. Hence, in this paper, we only use some most popularly adopted music and motion features in our prototype system for proof of principle. In the future, we would also like to explore music theory that relates to human interpretation of emotional content in music, as well as the characteristics of western tonal music [44], for building a more elaborate and theoretically grounded music-motion mapping relationship model.

Acknowledgments

This work was partly supported by NSFC 60633070, 60773183 and 60903132, National 863 High-Tech Program (Grant no: 2006AA01Z313 and 2006AA01Z335), and National Key Technology R&D Program of China (Grant no: 2007BAH11B02 and 2007BAH11B03). It is also supported by NCET-07-0743, and PCSIRT 0652. S. Xu performed this research as a Eugene P. Wigner Fellow and staff member at the Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the US Department of Energy under Contract DE-AC05-00OR22725.

References

[1] G. Alankus, A. Bayazit, and O. Bayazit, "Automated Motion Synthesis for Virtual Choreography," *J. Computer Animation and Virtual Worlds,* vol. 16, no. 3/4, pp. 259-271, 2005.

[2] O. Arikan and D. Forsyth, "Interactive Motion Generation from Examples," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 483-490, 2002.

[3] C. Bregler, M. Covell, and M. Slaney, "Video Rewrite: Driving Visual Speech with Audio," *Proc. ACM SIGGRAPH '97,* pp. 353-360, 1997.

[4] L. Ren, G. Shakhnarovich, J. Hodgins, H. Pfister, and P. Viola, "Learning Silhouette Features for Control of Human Motion," *Proc. ACM SIGGRAPH '04,* 2004.

[5] J. Kim, H. Fouad, J. Sibert, and J. Hahn, "Perceptually Motivated Automatic Dance Motion Generation for Music," *Computer Animation and Virtual Worlds,* vol. 20, no. 2/3, pp. 375-384, 2009.

[6] M. Brand and A. Hertzmann, "Style Machines," *Proc. ACM SIGGRAPH '00,* pp. 183-192, 2000.

[7] M. Cardle, L. Barthe, S. Brooks, and P. Robinson, "Music-Driven Motion Editing: Local Motion Transformations Guided by Music Analysis," *Proc. 20th UK Conf. Eurographics (EGUK '02),* pp. 38-44, 2002.

[8] J. Chen and T. Li, "Rhythmic Character Animation: Interactive Chinese Lion Dance," *Proc. ACM SIGGRAPH '05,* 2005.

[9] D. Ellis, "Beat Tracking by Dynamic Programming," *J. New Music Research,* vol. 36, no. 1, pp. 51-60, 2007.

[10] J. Friedman, "Fast Mars," Dept. of Statistics, Technical Report LCS110, Stanford Univ., 1993.

[11] K. Grochow, S.L. Martin, A. Hertzmann, and Z. Popović, "Style-Based Inverse Kinematics," *Proc. ACM SIGGRAPH '04,* pp. 522-531, 2004.

[12] A. Hoerl and R. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *J. Technometrics,* vol. 42, no. 1, pp. 80-86, 2000.

[13] E. Hsu, S. Gentry, and J. Popović, "Example-Based Control of Human Motion," *Proc. Symp. Computer Animation,* pp. 69-77, 2004.

[14] E. Hsu, K. Pulli, and J. Popović, "Style Translation for Human Motion," *Proc. ACM SIGGRAPH '05,* pp. 1082-1089, 2005.

[15] E. Keogh and C. Ratanamahatana, "Exact Indexing of Dynamic Time Warping," *Knowledge and Information Systems,* vol. 7, no. 3, pp. 358-386, 2005.

[16] T. Kim, S. Park, and S. Shin, "Rhythmic-Motion Synthesis Based on Motion-Beat Analysis," *ACM Trans. Graphics,* vol. 22, no. 3, pp. 392-401, 2003.

[17] L. Kovar, M. Gleicher, and F. Pighin, "Motion Graphs," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 473-482, 2002.

[18] R. Laban and L. Ullmann, "The Mastery of Movement," 1971.

[19] O. Lartillot and P. Toiviainen, "MIR in Matlab (II): A Toolbox for Musical Feature Extraction from Audio," *Proc. Int'l Conf. Music Information Retrieval (ISMIR '07),* pp. 237-244, 2007.
[20] H. Lee and I. Lee, "Automatic Synchronization of Background Music and Motion in Computer Animation," *Computer Graphics Forum,* vol. 24, pp. 353-361, 2005.
[21] J. Lee, J. Chai, P. Reitsma, J. Hodgins, and N. Pollard, "Interactive Control of Avatars Animated with Human Motion Data," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 491-500, 2002.
[22] Y. Li, T. Wang, and H. Shum, "Motion Texture: A Two-Level Statistical Model for Character Motion Synthesis," *Proc. ACM SIGGRAPH '02,* pp. 465-472, 2002.
[23] M. Maltamo and A. Kangas, "Methods Based on k-Nearest Neighbor Regression in the Prediction of Basal Area Diameter Distribution," *Canadian J. Forest Research,* vol. 28, no.8, pp. 1107-1115, 1998.
[24] E. Moulines and F. Charpentier, "Pitch-Synchronous Waveform Processing Techniques for Text-to-Speech Synthesis Using Diphones," *Speech Comm.,* vol. 9, no. 5/6, pp. 453-467, 1990.
[25] P. Nardiello, F. Sebastiani, and A. Sperduti, "Discretizing Continuous Attributes in AdaBoost for Text Categorization," *Proc. 25th European Conf. IR Research (ECIR '03),* pp. 320-334, 2003.
[26] M. Neff, I. Albrecht, and H. Seidel, "Layered Performance Animation with Correlation Maps," *Computer Graphics Forum,* vol. 26, no. 3, pp. 675-684, 2007.
[27] M. Nørgaard, *Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook.* Springer, 2000.
[28] S. Oore and Y. Akiyama, "Learning to Synthesize Arm Music to Motion By Example," *Proc. Int'l Conf. Central Europe on Computer Graphics Visualization and Computer Vision (WSCG '06),* 2006.
[29] M. Orr, "Introduction to Radial Basis Function Networks," technical report, Inst. for Adaptive and Neural Computation, Edinburgh Univ., 1996.
[30] E. Pampalk, "A Matlab Toolbox to Compute Music Similarity from Audio," *Proc. Fifth Int'l Conf. Music Information Retrieval (ISMIR '04),* pp. 254-257, 2004.
[31] S. Park, H. Shin, and S. Shin, "On-Line Locomotion Generation Based on Motion Blending," *Proc. Symp. Computer Animation,* pp. 105-111, 2002.
[32] H. Peng, F. Long, and C. Ding, "Feature Selection Based on Mutual Information: Criteria of Max-Dependency Max-Relevance, and Min-Redundancy," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 8, pp. 1226-1238, Aug. 2005.
[33] R. Schapire, "The Boosting Approach to Machine Learning: An Overview," Lecture Notes In Statistics-New York-Springer Verlag, pp. 149-172, 2003.
[34] T. Shiratori, A. Nakazawa, and K. Ikeuchi, "Dancing-to-Music Character Animation," *Computer Graphics Forum,* vol. 25, pp. 449-458, 2006.
[35] T. Strohmann and G. Grudic, "A Formulation for Minimax Probability Machine Regression," *Proc. Advances in Neural Information Processing Systems,* pp. 785-792, 2003.
[36] J. Suykens and J. Vandewalle, "Least Squares Support Vector Machine Classifiers," *Neural Processing Letters,* vol. 9, no. 3, pp. 293-300, 1999.
[37] J. Wang and B. Bodenheimer, "An Evaluation of a Cost Metric for Selecting Transitions between Motion Segments," *Proc. Symp. Computer Animation,* pp. 232-238, 2003.
[38] J. Wichard and C. Merkwirth, "ENTOOL—A Matlab Toolbox for Ensemble Modeling," http://www.j-wichard.de/entool, 2007.
[39] A. Witkin and Z. Popovic, "Motion Warping," *Proc. ACM SIGGRAPH '05,* pp. 105-108, 1995.
[40] L. Zhao and A. Safonova, "Achieving Good Connectivity in Motion Graphs," *Proc. Symp. Computer Animation,* 2008.
[41] J. Zhu, S. Rosset, H. Zou, and T. Hastie, "Multi-Class Adaboost," technical report, Stanford Univ., 2005.
[42] F. Ofli, E. Erzin, Y. Yemez, and A.M. Tekalp, "Multi-Modal Analysis of Dance Performances for Music-Driven Choreography Synthesis," *Proc. IEEE Int'l Conf. Acoustics Speech and Signal Processing (ICASSP '10),* 2010.
[43] R. Fan, J. Fu, S. Cheng, X. Zhang, and W. Geng, "Rhythm Based Motion-Music Matching Model," *J. Computer-Aided Design and Computer Graphics,* vol. 22, pp. 990-996, 2010.
[44] D. Cooke, *The Language of Music.* Oxford Univ. Press, 2010.
[45] M. Goto, "An Audio-Based Real-Time Beat Tracking System for Music with or without Drum-Sounds," *J. New Music Research,* vol. 30, pp. 159-171, 2001.
[46] M.J. Carey and E.S. Parris, and H. Lloyd-Thomas, "A Comparison of Features for Speech, Music Discrimination," *Proc. Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '99),* 1999.
[47] D. Liu and L. Lu, and H.J. Zhang, "Automatic Mood Detection from Acoustic Music Data," *Proc. Int'l Conf. Music Information Retrieval (ISMIR '03),* 2003.
[48] D. Liu and L. Lu, and H.J. Zhang, "Phase-Based Note Onset Detection for Music Signals," *Proc. Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP '03),* 2003.
[49] O. Izmirli, "Using a Spectral Flatness Based Feature for Audio Segmentation and Retrieval," *Proc. Int'l Conf. Music Information Retrieval (ISMIR '00),* 2000.
[50] L. Knopoff and W. Hutchinson, "Entropy as a Measure of Style: The Influence of Sample Length," *J. Music Theory,* vol. 27, pp. 75-97, 1983.

Rukun Fan received the MSc degree from State Key Lab of CAD&CG, Zhejiang University, China, in 2010. Currently, he is a doctoral student of computer science, The University of North Carolina at Chapel Hill. His research interests include computer animation, computer music, and motion tracking.

Songhua Xu received the PhD degree in computer science from Yale University. He is a researcher and a Wigner Fellow at Oak Ridge National Laboratory, US Department of Energy. His research interests include knowledge retrieval and management, web search, innovative applications of AI, intelligent systems, biomedical informatics, human-computer interaction, and computer art. He is an honorary assistant professor in the Computer Science Department at the University of Hong Kong.

Weidong Geng received the BSc degree from the Computer Science Department in Nanjing University, China, in 1989, and an MSc degree from the Computer Science Department of National University of Defense Technology in 1992. In 1995, he received the PhD degree from the Computer Science and Engineering Department of Zhejiang University, China. Currently, is a professor of College of Computer Science, Zhejiang University, China. From 1995 to 2000, he was in Zhejiang University, where he took charge of a number of projects about CAD/CG, and intelligent systems. He joined Fraunhofer Institute for Media Communication (former GMD.IMK), Germany, as a research scientist in 2000. In 2002, he worked in Multimedia Innovation Center, The Hong Kong Polytechnic University, Hong Kong. Since 2003, he has been working in State Key Laboratory of CAD&CG, Zhejiang University, and his current research focuses on computer aided design, computer animation, perceptual user interface, interactive media, and digital entertainment.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.