Vis Comput
DOI 10.1007/s00371-016-1308-y

ORIGINAL ARTICLE

Motion normalization method based on an inverted pendulum model for clustering

Taekhee Lee[1] · **Daeun Kang**[2] · **Taesoo Kwon**[2]

Abstract In many creative industries, such as the animation, movie, and game industries, artists often make good use of motion data to create their works by retrieving a particular motion from motion-capture data and reusing it. A large database of human motion is difficult to use unless the motion data are organized according to the type of motion. Although there have been many results for clustering motion capture data, many variations in the motion data complicate the clustering of data by making one type of motion numerically similar to other types of motions. To improve the motion clustering performance, we present a novel physically based motion normalization method that reduces ambiguous elements of motions, so that motions that have different semantics can be differentiated. The normalized motion data generated by our method can be used as input to existing clustering algorithms and improves the results.

Keywords Physics-based method · Motion capture and reuse · Motion clustering

1 Introduction

In many industries such as the animation, movie, and game industries, technicians and artists often retrieve a particular motion from a large database and use it when they animate characters. However, finding and searching through raw motion data for a specific motion is difficult and tedious for users, because the source databases include many different types of motion and their variations. Our research is motivated by the idea that the users' working efficiency levels would greatly improve if the motion data are well organized.

To organize a motion data, it is often divided into many short motion segments so that each piece contains one primitive motion, and then the segments are classified into multiple groups. For example, one classified group contains walking segments, while another group contains running segments. Motion classification is often performed manually to best suit the intention of users. However, manual classification is a tedious work, especially when the given motion dataset is large. To reduce the burden of users, several automatic motion organization techniques have been developed that divide a dateset into many motion segments and classify motion segments. However, those techniques often do not provide sufficient accuracy due to the variations and ambiguities in the motions. For example, walking with a wide stride can easily be miscategorized as running by a clustering algorithm, and slow running can be miscategorized as walking. Also, turning toward the left makes the body lean toward the left, and the leaning of the upper body produces additional ambiguities when attempting to distinguish whether the character is walking or running while it is turning. In short, there are often ambiguous cases in which similar motions appear numerically different and/or different motions appear numerically similar.

The ambiguity problem makes the motion sorting problem challenging. There are some unnecessary variations of motion parameters, such as the stride, speed, and leaning angle, and these variations disturb the accurate classification of motion segments. For more accurate motion sorting,

✉ Taesoo Kwon
taesoo@hanyang.ac.kr

Taekhee Lee
watersp@gmail.com

Daeun Kang
teddysiah@gmail.com

[1] FXGear, Seoul, South Korea

[2] Hanyang University, Seoul, South Korea

Published online: 01 September 2016

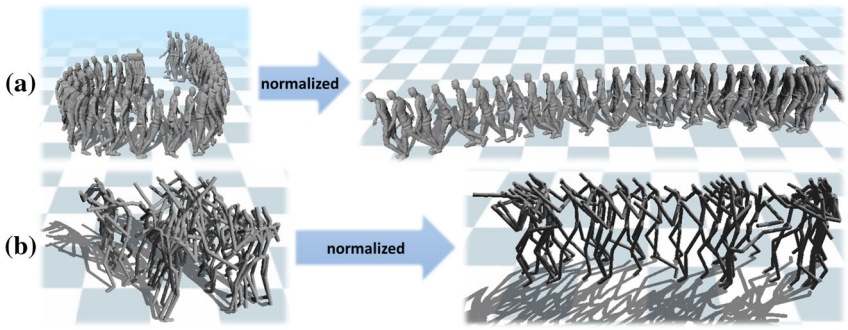

Fig. 1 Our method normalizes motion data by reducing unnecessary motion variations. A locomotion dataset is normalized in (**a**), and a kick-boxing data is normalized in (**b**)

each motion segment needs to be refined by reducing unnecessary variations and emphasizing the necessary elements of motion information; we call this process "normalization" (Fig. 1). Recently, many studies have attempted to analyze the features of human motion to determine which elements are necessary and which are not [9,20,27,31]. However, these previous works commonly are limited in that the underlying dynamics in human motion data, an important source of variations, is ignored. To address this limitation, we suggest a novel physically based normalization method.

There are two representative automatic motion classification techniques: supervised and unsupervised. Although our normalization method can be used for both techniques. In this study, we focus on the case of unsupervised learning, i.e., clustering. For supervised classification, the user needs to prepare a labeled dataset in advance. Then, the system learns discriminative models from the features of each motion type and use the models for classifying new unlabeled datasets. The input of the classification system is a sequence of unannotated motion segments, and the output is a sequence of labeled motion segments. Although supervised learning techniques can achieve good accuracies relatively easily, preparing a labeled training dataset is tedious and inconvenient.

On the other hand, clustering does not require any particular input except for a sequence of motion segments. The output of clustering is groups of similar segments with the same motion type. After clustering, the user can look at the segments in each cluster to label a cluster with a name and discard a few outliers in the cluster. Despite the convenience of clustering, the quality of clustering results highly depends on the feature extraction method, the distance metric, and the clustering algorithm that are in use. The main contribution of this work is a novel normalization method for motion clustering. When the normalized motion data are input into clustering algorithms, much higher clustering accuracies are achieved compared to when the original motion data are input, as demonstrated in Sect. 7. Further, our normalization method can be used with any existing methods for feature extraction and distance measurement.

2 Related works

The temporal segmentation and grouping of motion data by semantically meaningful behavior have been challenging research topics in relation to several applications, such as interactive motion synthesis and content-based motion retrieval systems.

2.1 Applications

Many statistical motion blending approaches have been developed due to their efficiency and controllability [5,13,28–30,35,37,39]. These approaches generated continuous variations of motion accurately by blending examples. Most of these require that all motion segments in one group represent a logically identical action.

Recently, as the handling of large amounts of motion data has become possible, content-based motion retrieval systems have also become increasingly popular. Kovar et al. [20] presented a parameterizing method for extracting logically related motions from motion data. Kapadia et al. [18] proposed a flexible and efficient method that searches for arbitrarily complex motion from a large motion database using a tree-based data structure. In addition, a sketch-based retrieval system has been developed [6].

2.2 Motion segmentation and classification

Though there have been many related studies, the segmentation of motion data and the classification of motion segments remain as challenging topics. Jenkins et al. [16] suggested a method that automatically derives the vocabularies of parameterized primitive motion segments from motion data. Barbic et al. [2] presented a probabilistic approach that uses principal component analysis with a Gaussian mixture model. Kwon et al. [23] made use of the concept of the center of mass (COM) to generate natural motion during motion segmentation. Müller et al. [31–33] proposed an efficient indexing method for 3D human motion data. They designed a query structure to record motion information on both a whole body and a sub-body, identifying

logically similar motions with different appearances. Zhou et al. [45] proposed a method which segments time-series data into multiple segments, such that each one belongs to one of the resulting clusters. This method was efficiently optimized with a coordinate descent strategy and dynamic programming. Our method can be used in combination with any existing segmentation and classification methods. We aim to complement earlier results by improving the overall performance of the systems.

2.3 Feature vectors and distance metrics

There are a number of studies which are closely related to ours in that they tried to improve the underlying pose comparison metric. López-méndez et al. [27] made use of a supervised learning scheme in their research, proposing a method that learns which elements make two human motions different from each other. Deng et al. [9] presented a motion retrieval method that ranks similar motion sequences when a query motion is given as input. They conducted comparative user studies to evaluate the accuracy and perceptual consistency of the proposed metric. Other researchers have concentrated on discovering parameters to be normalized to improve the invariance properties. Gao et al. [11] suggested a very simple and efficient normalization method that normalizes the entire motion data in the database, such that it has the same skeleton length. Other researchers investigated video-based human activity recognition using a novel feature descriptor [3,10,15,43]. Wang et al. represented low-level features of human motion by means of geometric invariance, employing an extended random transform, called the R transform [42].

2.4 Motion editing

The proposed normalization method can be seen as a motion-editing technique in that an original motion data is modified by controlling some elements of the motion, such as turning direction and leaning angle. Many motion-editing techniques have been suggested that also modifies a motion moving along a curved path. Kovar et al. [19] proposed a data structure called registration curves to improve the result quality of existing motion blending methods. Shin et al. [38] suggested a method to correct a previously edited motion to improve the plausibility. Unlike our method, these methods are not physically based and do not modify the leaning angle of the character to match the acceleration of the character.

There haven many physically based motion-editing schemes that generate motions based on footprints. Girard proposed a method that generates a motion which fits the given footprint set [12]. Van de Panne proposed a method that generates animated locomotion based on an explicit footstep plan [34]. A single compact optimization problem was formulated while following the laws of physics. Chung and Hahn proposed a procedural system that generates walking animation along the given footprints [7]. Other physics-based and kinematic controllers that make characters move very closely to given footprints have also been developed [4,8,36,40,44].

Our method is different from these schemes, in that we normalize the given motion data by decreasing the motion variation caused by different acceleration profiles. These variations are modeled using a physical model known as an inverted pendulum on a cart (IPC). This physical model has been commonly used to generate balance feedback in motion-control systems [17,41]. The IPC model is also commonly used for simulating biped motions physically, such as walking, running, and diverse variations of these types of motions [1,14,22,24]. To the best of our knowledge, our method is the first approach to normalize motion data by employing an IPC model. By normalizing the variations while following the laws of physics, a reduction in the clustering errors is realized by our proposed method.

3 System overview

Figure 2 shows the system overview of our normalization method. Our system takes an original motion dataset as input and produces the normalized dataset as output. Our normalization system consists of four sub-systems: a pendulum motion estimator, a normalizer, a footstep location genera-

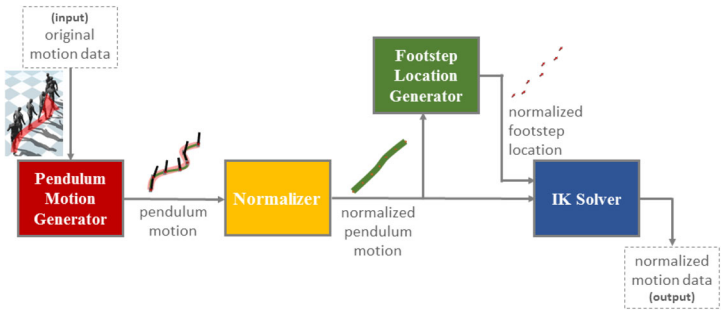

Fig. 2 Overview of the normalization process. The pendulum motion estimator extracts the pendulum motion from the original human motion to compactly represent the route and the overall leaning angle in a physically plausible manner

tor, and an inverse kinematics (IK) solver. First, the pendulum motion estimator initially defines the route along which the character moves. In this process, the position of the character and the overall leaning angle of the character body are measured. The measured data are provided as output in the form of a pendulum motion. Second, the output of the pendulum motion estimator is sent to the normalizer and it normalizes the given pendulum motion while adhering to the laws of physics. As the normalizer reduces ambiguous elements, the output pendulum motion shows more straight moving trajectory than the input pendulum motion. After passing through the normalizer, the normalized pendulum motion is sent to both the footstep location generator and the IK solver. After that, the footstep location generator plans a set of footstep locations based on the given normalized pendulum motion. As the input data are already normalized, the output of the footstep location generator is also normalized. Finally, the normalized footstep location is sent to the IK solver, and the IK solver receives it while also receiving the normalized pendulum motion from the normalizer. The IK solver rebuilds a new normalized human motion based on the two inputs as well as the input human motion. After the entire process, the normalized human motion is used as the input for existing clustering algorithms to show improved clustering accuracy.

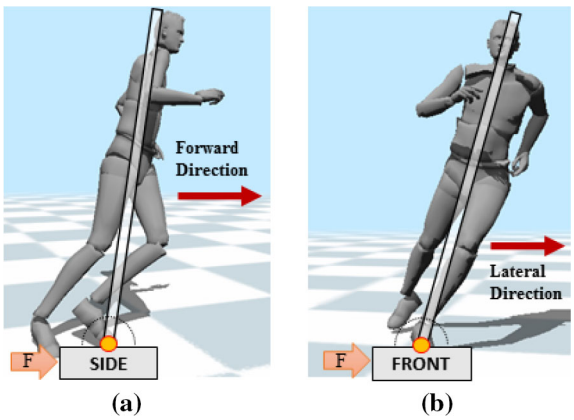

Fig. 3 Two IPCs for one human character. One IPC represents the leaning angle along the forward direction, and the other one is for the lateral direction

4 Estimating the pendulum motion

In this section, we describe how our system estimates the pendulum motion from the given original motion data. The pendulum motion estimator, the first sub-system of our method, extracts the necessary elements such as the route and the overall leaning angle from the original human motion. To extract the necessary information, we adopted an IPC model and used it to represent the character's overall movement.

Figure 3 shows the structure of the IPC model adopted in our method. An IPC model is a simple model that has only two joints. One of the two joints is a passive rotational joint which is located at the base of the pendulum, and the other is an active translational joint which moves backward and forward to balance the pendulum. We used two IPCs to represent one human character. One IPC controls the leaning angle along the forward direction and the other one is for the lateral direction. To handle the IPC model in a way that it moves in accordance with the input human motion, we utilized a pendulum control method described in a previous work [22]. As we modified the previous method slightly for our purpose, we describe the overall process briefly while focusing on the differences.

Figure 4 shows how the pendulum motion estimator operates. Our system initially generates the initial pendulum motion and extracts its COM trajectory. Then, the optimizer compares the pendulum COM trajectory with the COM trajectory extracted from the input human motion. To minimize the difference between the two COM trajectories, the optimizer modifies some of the control parameters of the pendulum, and the pendulum motion is modified in

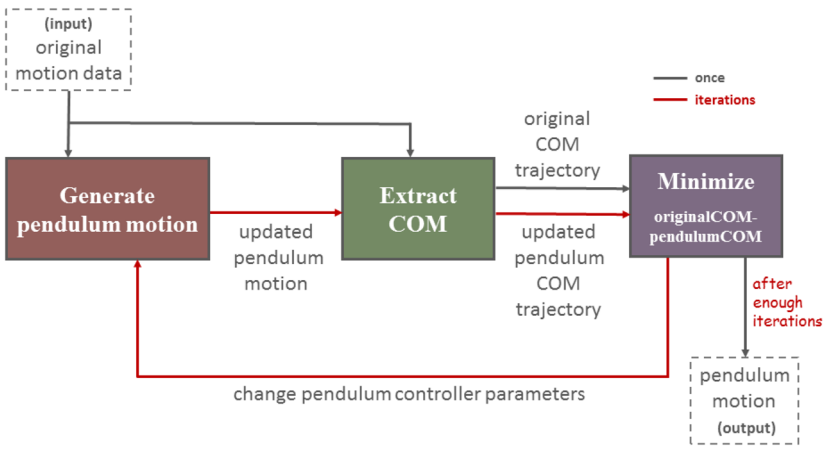

Fig. 4 Pendulum motion estimation: a pendulum motion is estimated by minimizing the difference between the pendulum COM trajectory and the COM trajectory extracted from the input human motion

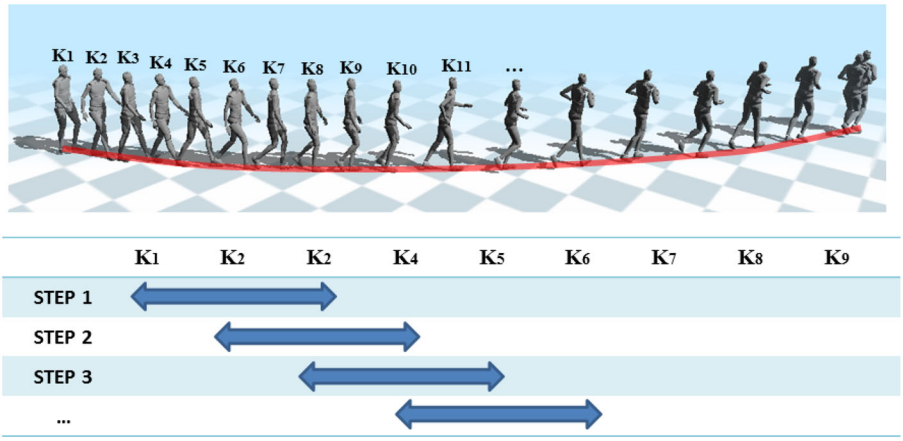

Fig. 5 Multi-step optimization process. For efficient and robust optimization, the space–time optimization problem is divided into multiple overlapping subproblems

the process. This process marked with red lines in Fig. 4 is repeated until the iteration converges. The resulting pendulum motion well represents the overall movement of the character observed from the original human motion, but using much less DOFs.

To define the initial pendulum motion, the initial value of the pendulum's forward direction and the desired velocity need to be determined. The forward direction is simply calculated from the pelvis orientation of the original human motion. Then, our system sets key frames at a uniform rate (5 Hz in our experiments) and specifies a desired velocity of the pendulum at each key frame. The initial value of the pendulum's desired velocity $\hat{\dot{x}}_s$ at each key frame is calculated using the following equation:

$$\hat{\dot{x}}_s = \dot{x}_{\mathrm{COM}} + \alpha \ddot{x}_{\mathrm{COM}}, \tag{1}$$

where $\dot{x}_{\mathrm{COM}}$ is the velocity of the human COM, and $\ddot{x}_{\mathrm{COM}}$ is the acceleration of the human COM obtained from the original data. α is a constant value denoting the weight of the acceleration. We found that the initial pendulum motion closely matches the input human motion when $\alpha = 10$. The initial pendulum motion based on these initial parameter values follows the original motion closely at the beginning, but the error between the initial trajectory and the original human motion becomes larger as the frame goes on due to the accumulated error.

To minimize the difference between the pendulum COM trajectory and the COM trajectory from the original human motion, we employed an optimization method that minimizes the horizontal distance between the pendulum COM and the human COM. The objective function for optimization is defined as shown below and works in a conjugate gradient solver,

$$\{\hat{\dot{x}}_{\mathrm{s}}\} = \arg\min \sum_{i \le N} \left\| \mathrm{project}(x_i^{\mathrm{cCOM}} - x_i^{\mathrm{pCOM}}) \right\|^2. \tag{2}$$

In Eq. 2, $\{\hat{\dot{x}}_{\mathrm{s}}\}$ is a set of the desired velocity values at each key frame; x_i^{cCOM} is the position of the human COM at frame i; and x_i^{pCOM} is the position of the pendulum COM at frame i. As we operate the IPC model considering only horizontal forces, the vertical components are ignored in our method. Repeating the optimization process, the initial pendulum motion, which shows an increasing amount of error, is modified and becomes the final pendulum motion, which is quite close to the original motion.

In our experiment, however, we found that solving the optimization problem for the entire pendulum motion at once, corresponding to all input data (a sequence of many motion segments), can lead to the local minima, as the dimension of the search space becomes too high when the motion data are large in terms of time. To settle this local minima problem, we employed a bottom-up approach and designed the optimization process that has multiple steps, as illustrated in Fig. 5.

During the first step, the pendulum's state at the first key frame and the first three continuous key frames $K \in \{K_1, K_2, K_3\}$ are optimized at once. Then, at the n-th step, three continuous key frames $K \in \{K_n, K_{n+1}, K_{n+2}\}$ are optimized, and the best optimization solution is used as the initial solution for optimizing the next key frames $K \in \{K_{n+1}, K_{n+2}, K_{n+3}\}$ at the $(n+1)$ step. The pendulum's initial state is optimized in the first step while it is not in other steps, because the error between the generated trajectory and the original motion strongly depends on the pendulum's initial state. At each step of the optimization process, the optimizer calculates the gradient of the objective function (Eq. 2) using a finite difference method numerically.

The pendulum motion estimator computes the pendulum motion, which includes the necessary elements of motion information from the original human motion data. As the pendulum motion is obtained based on the original human motion data, it is close to the original human motion. To perform motion sorting well, the pendulum motion is sent to the normalizer, the second sub-system.

5 Normalizing pendulum motion

In this section, we describe how our system normalizes the given pendulum motion. When the original motion data are given as input to our system, the pendulum motion generator takes the input and defines the pendulum motion based on it. The generated pendulum motion is sent to the normalizer, the second sub-system of our method, and normalizes the given trajectory by minimizing the ambiguous elements, allowing important elements to be emphasized.

We found that there are two elements that often make motions ambiguous: the forward speed and the turning speed. For example, the computer often misrecognizes fast walking as running and slow running as walking. However, if we normalize the forward speed, walking and running become distinguishable from each other, because the stylistic differences are emphasized. Similarly, walking along a curved path is often erroneously classified into the same group as running along a similar curve. In this case, if we normalize the turning speed, walking and running become easier to distinguish regardless of the route.

The normalization process is performed by clamping the values of the desired velocity and the turning speed parameters.

$$\hat{\dot{\mathbf{x}}}_f \leftarrow \text{clamp}(\hat{\dot{\mathbf{x}}}_f, \alpha), \hat{\dot{\mathbf{x}}}_l \leftarrow \text{clamp}(\hat{\dot{\mathbf{x}}}_l, \beta), \hat{\dot{\theta}} \leftarrow \text{clamp}(\hat{\dot{\theta}}, \gamma). \tag{3}$$

In Eq. 3, $\hat{\dot{\mathbf{x}}}_f$ is the desired velocity in the forward direction; $\hat{\dot{\mathbf{x}}}_l$ is the desired velocity in the lateral direction; $\hat{\dot{\theta}}$ is the desired turning speed; and α, β, and γ are the clamping coefficients as obtained in experiments.

After clamping the parameters, the input pendulum motion is modified based on the clamped parameters. The redefined, i.e., normalized, pendulum motion is the output of the normalizer; it is sent to the footstep location generator and the IK solver.

6 Generating the footstep location

In this section, we describe how our system calculates the location of the character's feet based on the given original motion data and the normalized pendulum motion. The footstep location generator, the third sub-system of our method, determines where each foot should be located when the character moves along the normalized route. Our system defines the footstep location for the left and right foot independently using the same algorithm.

Before calculating the footstep location, it is necessary to detect whether each foot is contacting the ground or not at each frame of the original motion data. To detect the contact state, we employed the contact state detection method suggested in an earlier work [26]. The basic idea of this method is that a foot is contacting the ground when it maintains its position lower than a threshold and its velocity close to zero for several frames. In our method, we call the duration while the foot is touching the ground the "support phase" and the duration while the foot is moving in the air the "swinging phase." After tuning the threshold parameters carefully, the detection method works well in our experiments, except for a few missing footsteps. These missing footsteps can make the resulting motion appear as though the foot is slightly slipping for a moment, but it rarely affects the motion sorting accuracy.

To calculate the footstep location based on the normalized pendulum motion and the contact state of the two feet from the original data, we employed a footstep planning method described in a previous work [22]. As we modified the previous method for our method only slightly, we describe the overall process briefly while focusing on the differences.

We first encode the captured feet trajectories relative to the original pendulum motion. For this purpose, we define a trajectory of the reference coordinate frame for each foot, and the actual trajectory of the foot is encoded locally to the trajectory of the reference coordinate frame. The local feet coordinates are later decoded on top of the normalized pendulum trajectory for motion normalization.

The trajectory of the reference coordinate frame is obtained using only a pendulum trajectory and a sequence of contact states of a foot as follows. During the support phase, the reference coordinate frame maintains its position at its desired location. During the swing phase, the coordinate frame follows an arc at ground level between the foot's positions at the previous support moment and the next support moment. For notational simplicity, we define an operator denoted as $\{a, k\}$, where $0 \leq a \leq 1$ is the support phase and k denotes the foot the system is working on, left or right. This operator converts a phase into a frame number, as $\{a, k\} = f_k + a(l_k - f_k)$, where f_k and l_k are, respectively, the first frame and the last frame of the current support phase a. Then, the normalized coordinate for the desired position of the support foot at phase a, $\mathbf{L}_{\{a,k\}}$ is defined as $\mathbf{L}_{\{a,k\}} = \text{shear}(\mathbf{x}^{\text{pend}}_{\{0.5,k\}}, \mathbf{q}^{\text{pend}}_{\{0.5,k\}})$, where $\mathbf{x}^{\text{pend}}_{\{0.5,k\}}$ is the pendulum's position and $\mathbf{q}^{\text{pend}}_{\{0.5,k\}}$ is the pendulum's leaning angle at the middle frame of the current support phase $\{0.5, k\}$. In addition, shear$(\cdot)$ denotes a transformation matrix that shears the vertical axis y to the pendulum axis, which is defined according to $\mathbf{q}^{\text{pend}}_{\{0.5,k\}}$. The position of the coordinate frame is set to $\mathbf{x}^{\text{pend}}_{\{0.5,k\}}$ during the support phase with the pendulum configuration at $\{0.5, k\}$.

In the swinging phases, the position of the foot is encoded using a coordinate frame that interpolates the nearby coordinate frames of the support phases. During

each swinging phase, the foot swings in the air, drawing an arc that approximates the pendulum's curved trajectory. The normalized coordinate for the desired position of the swinging foot at phase s, $\mathbf{L}_{\{s,k\}}$ is defined as $\mathbf{L}_{\{s,k\}} = \text{shear}(\text{arc}(s, \mathbf{x}^{\text{pend}}_{\{-0.5,k\}}, \mathbf{x}^{\text{pend}}_{\{1.5,k\}}), \text{slerp}(s, \mathbf{q}^{\text{pend}}_{\{-0.5,k\}}, \mathbf{q}^{\text{pend}}_{\{1.5,k\}}))$, where $\mathbf{x}^{\text{pend}}_{\{-0.5,k\}}$ and $\mathbf{x}^{\text{pend}}_{\{1.5,k\}}$ are, respectively, the pendulum's positions at the middle frame of the previous support phase and at the middle frame of the next support phase. The radius of the arc is determined by least-square fitting of the arc to the pendulum COM trajectory projected onto the ground (refer to [35] for more details).

The footstep location generator determines how two feet of the character would move along the normalized trajectory. The output—the normalized footstep location—is sent to the IK solver and is used to rebuild the normalized full-body motion from the normalized pendulum motion and the original full-body motion. We employed the analytic IK solver suggested in earlier research [25]. The final normalized motion data can be used for motion clustering in place of the original motion data to improve the accuracy of motion clustering.

7 Motion clustering

To demonstrate the effectiveness of our normalization method in clustering, we designed our test cases as follows. We used two motion-capture datasets. The first dataset, termed the locomotion dataset, was sampled at a rate of 30 Hz. It is a 5-min-long sequence of motion segments which describe a human's basic locomotion. The other dataset, termed the kickboxing dataset, was sampled at a rate of 60 Hz. It is a 15-min-long sequence of motion segments which describe Muay Thai motions.

7.1 Preparing ground truth data

These two motion-capture datasets were precisely segmented based on the local minima of the center of the mass height profile in advance. The segments were classified by first using semi-automatic classifiers in [21,23], and then by manually fixing incorrect segmentations or labels. The locomotion dataset was classified into eight clusters that define eight different types of locomotion as summarized in Table 1. The kickboxing segments were grouped into 73 clusters using a multi-aspect classification scheme. We defined seven aspects which were moving, jumping, left-kicking, right-kicking, left-punching, right-punching and reacting. Also, for each aspect, we defined several categories which describe the characteristics of each motion according to the aspect. For example, for both the left- and right-kicking aspects, we defined up, down, and others classes. Also, for both the left- and right-punching aspects, we defined forward, backward, and others classes. A cluster is defined by a combination of the classes of all the aspects. For example, a motion segment that contains no specific movement is classified as others with respect to all aspects. Also, there can be a segment which shows both right-kick-up and left-punch-fwd at the same time. As this dataset was previously used in [21], the reader can refer to this work to see the defined motion types of Muay Thai motion in more detail. Preparing the ground truth data took more than 10 h even for an experienced user, because the user had to look through the entire motion using multiple passes to ensure satisfactory results.

Table 1 Eight different types of locomotion and the names of each type for clustering locomotion dataset

Type name	Motion type
LDR	Walking, starting with the left foot
RDF	Walking, starting with the right foot
FLF	Running, starting with the left foot
FRF	Running, starting with the right foot
LDRF	Transition from walk to run, leaping with the left foot
RDLF	Transition from walk to run, leaping with the right foot
FRDL	Transition from run to walk, landing with the left foot
FLDR	Transition from run to walk, landing with the right foot

7.2 Clustering algorithms

After creating the ground truth data, we normalized the two datasets with our method and then segmenting the datasets into many short motion segments using the same segmentation time instances with the one used for the ground truth result to make the comparison fair. Several existing automatic segmentation algorithms such as the one in [2,21] can be used instead. After the segmentation process, we extract a feature vector for each motion segment using a method similar to that in [21]. For every motion segment, ten poses are sampled at equally spaced time instances, including the start and end timing of the motion segment, and each pose is represented as a tuple of joint positions. A feature vector is obtained by concatenating all the joint positions at all the time instances. To represent the feature vector in a coordinate invariant manner, the joint positions are represented in a local coordinate frame defined by the forward-facing direction of the character projected to the ground and the vertical axis (y axis). The extracted feature vectors are used to run the clustering algorithms.

We ran several conventional clustering algorithms using the original and normalized motion segments as inputs. Four conventional clustering algorithms were adopted: a k-mean

clustering, a Gaussian mixture model (GMM) clustering, and agglomerative clustering algorithms with a mean-linkage (Agglo1) and a max-linkage (Agglo2). K-mean and GMM clustering algorithms have random characteristics; hence, we performed 20 trials of clustering with these two algorithms and averaged all of the clustering accuracies of the trials. In addition, for the GMM clustering algorithm, we used diagonal or tied covariance matrices, as the feature space has high-dimensional spaces and there are too few segments in some clusters. Finally, we evaluated the accuracy of each clustering result by comparing it with the ground truth result.

We measured the clustering accuracy of the two kinds of original and normalized datasets using the four aforementioned conventional clustering algorithms and the results are summarized in Figs. 6 and 7. To measure the accuracy, we used the ground truth data. Specifically, a cluster is labeled as the most common motion type of the segments in the cluster. Then, the accuracy is calculated by counting the number of motion segments, of which the motion type matches the label of its cluster and divided this number by the total number of segments. Figure 6 shows the accuracy using the locomotion dataset, and Fig. 7 shows the accuracy using the kickboxing dataset. The solid lines in the graphs represent the clustering accuracy using the normalized datasets, and the dotted lines represent the accuracy using the original datasets. Figures 8 and 9 show confusion matrices obtained by clustering with the Agglo2 clustering algorithm. We calculated the clustering accuracy by dividing the sum of the diagonal elements of the matrix, which is equal to the number of correctly clustered segments, by the total number of clustered segments. Our test was performed with a PC equipped with an Intel i5 processor with 8GB of memory.

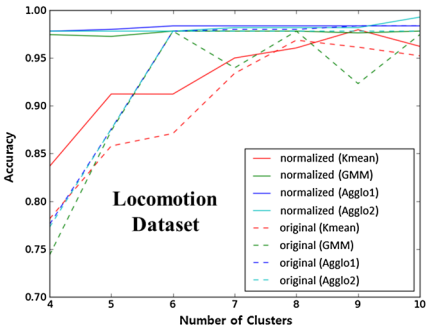

Fig. 6 Comparison of the measured clustering accuracy using the original and normalized locomotion datasets using four conventional clustering algorithms

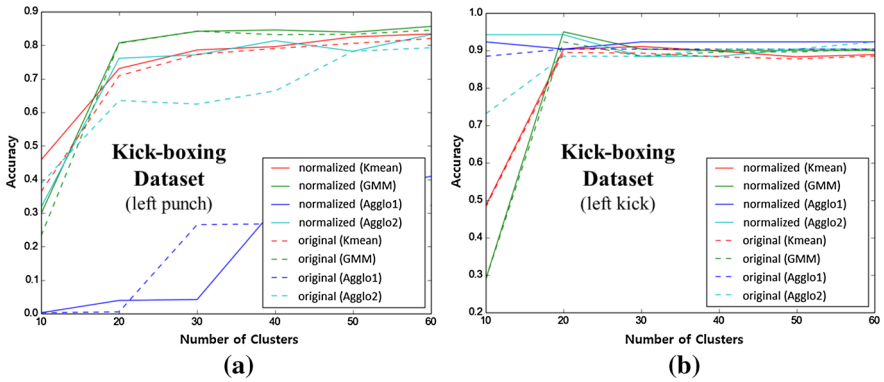

Fig. 7 Comparison of the measured accuracy levels when clustering the original and normalized kickboxing datasets using four conventional clustering algorithms. **a** When clustering the left-punching motion segments and **b** when clustering the left-kicking motion segments

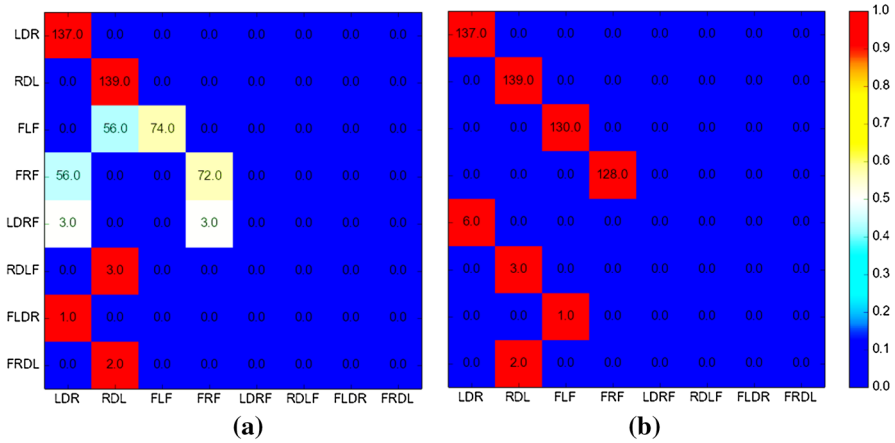

Fig. 8 Confusion matrices obtained by clustering the original (**a**) and the normalized (**b**) locomotion dataset into four clusters using Agglo2

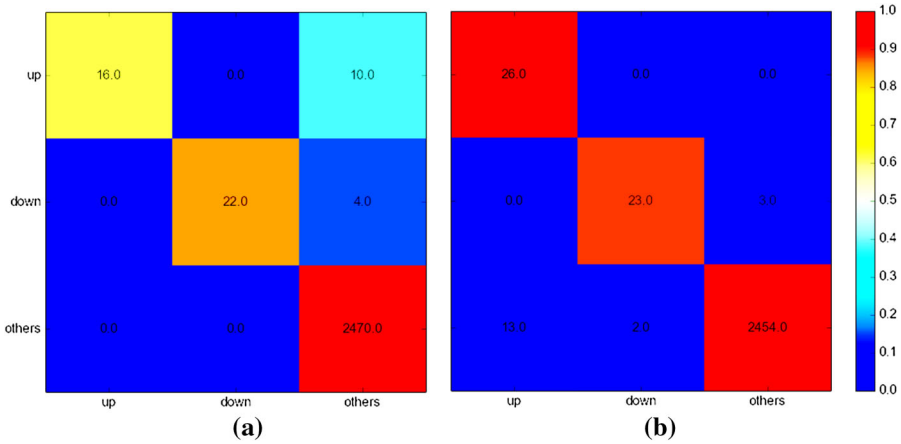

Fig. 9 Confusion matrices obtained by clustering the original (**a**) and the normalized (**b**) left-kicking motion segments in the kick-boxing dataset into ten clusters using Agglo2

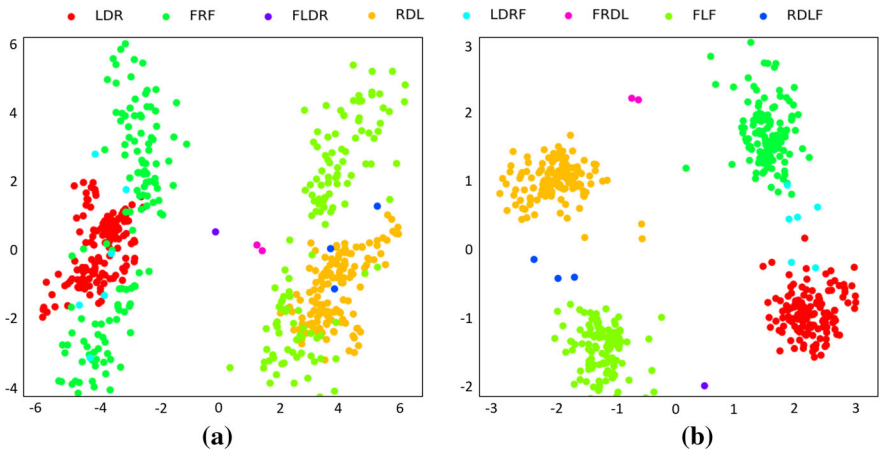

Fig. 10 Scatter plot graphs of the original (**a**) and the normalized (**b**) motion segments in the locomotion dataset

7.3 Locomotion dataset

In Fig. 6, all clustering accuracy levels on the normalized dataset consistently remain higher than 97 % regardless of the number of cluster changes. Also, the figure shows that the accuracy levels when clustering the original and the normalized datasets have a general tendency to increase as the number of clusters increases.

Note that the number of clusters of the ground truth result is eight, which is equal to the number of predefined motion types. Thus, if we use fewer number of clusters than eight, more than two different types need to be grouped into one cluster. Therefore, it is impossible to achieve 100 % accuracy when the number of clusters is lower than the number of predefined motion types. In contrast, theoretically the accuracy becomes close to 100 % as the number of clusters increases, which is why the measure accuracy levels generally increase as the number of clusters increases. Our normalizing method achieves much higher accuracy compared to the original dataset, especially when the number of clusters is low. This implies that users can browse the motion segments by types more easily, because small number of clusters can be used for most practical purposes.

Figure 8 shows 8 × 8 confusion matrices obtained by clustering the original (Fig. 8a) and the normalized (Fig. 8b) locomotion dataset into four clusters. We clustered motion segments into four clusters because it was enough to separate all the walking segments from the running segments. The figure shows 100 % clustering accuracy for walking and running motion segments, and 97.8 % accuracy when transition segments were included. In other words, transition segments (LDRF, RDLF, FLDR, RFDL) were incorrectly clustered into either walking or running clusters (LDR, RDL, FLF, FRF). Not only Agglo2, but also other clustering algorithms showed similarly high degrees of accuracy when normalized datasets were used.

To describe why normalized motion leads to better clustering accuracy, we employed scatter plot graphs. To create the graph, we initially chose the two basis vectors for the feature space which have the largest degrees of variation and set those two vectors as two-dimensional principal axes using the principal component analysis (PCA). Then, we projected the feature vectors onto the 2D space spanned by the two principal axes for plotting. Thus, the scatter plot graphs approximately indicate the distance between segments, which strongly affect the clustering result. Figure 10

shows scatter plot graphs of the original (a) and the normalized (b) motion segments in the locomotion dataset, where each color in the plots represents a motion type. In case of the original dataset, green and bright green plots are widely scattered and overlapped with other motion types in the graph, and thus undistinguishable motion types can arise during the clustering process. However, in the normalized dataset, all of the plots in the graph are sectioned by color and the overlapped area is clearly reduced.

7.4 Kickboxing dataset

In Fig. 7, the measured accuracy levels of the kickboxing dataset do not show such a high and increasing tendency akin to that shown in Fig. 6. Figure 7a shows the clustering accuracy measured using the left-punch aspect, and Fig. 7b shows the clustering accuracy measured using the left-kick aspect. In Fig. 7a, the accuracy levels of when clustering the normalized dataset with k-mean, GMM, and Agglo2 are higher than when clustering the original dataset, but they do not maintain a consistently high degree as the number of clusters increases.

Figure 9 shows 3×3 confusion matrices obtained by clustering the original (9a) and the normalized (9b) motion segments in the kickboxing dataset into ten clusters. In both figures, horizontal axes indicate ground truth clustering result, and vertical axes indicate actual clustering results. As shown in Fig. 9b, all segments including right kicks, moves, reacts and punches are labeled as *others* under the lkick aspect, and thus the number of segments labeled *others* dominates the number of segments labeled *up* or *down*. Therefore, to calculate a meaningful level of clustering accuracy, we measured the clustering accuracy using only the left-punch segments and left-kicking segments in Fig. 9b. In other words, we computed the clustering accuracy using the upper-left 2×2 sub-block matrix of the confusion matrix. As shown in Fig. 7, our normalization method is effective for the kickboxing dataset, especially when the number of clusters is low (20).

8 Conclusion

We present a novel physics-based normalization method to improve the accuracy of various clustering methods. Our method uses an inverted pendulum on a cart model as a generative model for the normalization process, and it works for any footstep-driven motion as long as the contact state of the two feet is known. Our method complements existing conventional clustering methods by increasing the accuracy in most test cases, especially when the number of clusters is small.

We found that there are a few limitations of our method. Our normalization method dramatically improved the clustering accuracy when the locomotion dataset was used as input, but there was less of an increase in the accuracy when unique motions were included in the input, such as kicking and/or punching. We believe that this is related to the limited expression capability of the inverted pendulum model. For example, our method may not work well for motions involving many contacts, such as rolling on the ground. Also, diving, cartwheel motions, and broad jumping motions are not well represented by the pendulum model.

In the future, we would like to improve our method by incorporating more generalized physical models with more degrees of freedom. In this way, a wider range of variations that is within the expression capability of the physical model could be normalized. However, questions of which variables in the motion data need to be preserved during normalization and which variables need to be normalized would require further study. Other applications such as physics-based motion segmentation, annotation and retrieval would also be interesting future works.

Acknowledgments This work was supported by the research fund of Hanyang University (HY-201200000000616).

References

1. Agrawal, S., Shen, S., van de Panne, M.: Diverse motion variations for physics-based character animation. In: Proceedings of the 2013 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (2013)
2. Barbič, J., Safonova, A., Pan, J.Y., Faloutsos, C., Hodgins, J.K., Pollard, N.S.: Segmenting motion capture data into distinct behaviors. In: Proceedings of the 2004 Graphics Interface Conference, pp. 185–194 (2004)
3. Barnachon, M., Bouakaz, S., Boufama, B., Guillou, E.: Ongoing human action recognition with motion capture. Pattern Recognit. **47**(1), 238–247 (2014)
4. van Basten, B.J.H., Stüvel, S.A., Egges, A.: A hybrid interpolation scheme for footprint-driven walking synthesis. In: Proceedings of Graphics Interface 2011, GI '11, pp. 9–16 (2011)
5. Beaudoin, P., Coros, S., van de Panne, M., Poulin, P.: Motion-motif graphs. In: Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 117–126 (2008)
6. Choi, M.G., Yang, K., Igarashi, T., Mitani, J., Lee, J.: Retrieval and visualization of human motion data via stick figures. Comput. Graph. Forum **31**(7–1), 2057–2065 (2012)
7. Chung, S.K., Hahn, J.K.: Animation of human walking in virtual environments. In: Proceedings of the Computer Animation, CA '99, pp. 4– (1999)
8. Coros, S., Beaudoin, P., Yin, K.K., van de Pann, M.: Synthesis of constrained walking skills. ACM Trans. Graph. **27**(5), 113:1–113:9 (2008)
9. Deng, Z., Gu, Q., Li, Q.: Perceptually consistent example-based human motion retrieval. In: Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games, pp. 191–198 (2009)

10. Feng, Y., Ji, M., Xiao, J., Yang, X., Zhang, J.J., Zhuang, Y., Li, X.: Mining spatial-temporal patterns and structural sparsity for human motion data denoising. IEEE Trans. Cybern. **45**(12), 2693–2706 (2015)
11. Gao, Y., Ma, L., Chen, Z., Wu, X.: Motion normalization. In: Tao, J., Tan, T., Picard, R. (Eds.) Affective Computing and Intelligent Interaction, *Lecture Notes in Computer Science*, vol. 3784. Springer, Berlin pp. 95–101 (2005)
12. Girard, M.: Interactive design of computer-animated animal motion. IEEE Comput. Graph. Appl. **7**(6), 39–51 (1987)
13. Heck, R., Gleicher, M.: Parametric motion graphs. In: Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games, pp. 129–136 (2007)
14. Hodgins, J.K., Wooten, W.L., Brogan, D.C., O'Brien, J.F.: Animating human athletics. In: SIGGRAPH, pp. 71–78 (1995)
15. Hou, J., Chau, L., Magnenat-Thalmann, N., He, Y.: Human motion capture data tailored transform coding. IEEE Trans. Vis. Comput. Graph. **21**(7), 848–859 (2015)
16. Jenkins, O.C., Mataric, M.J.: Automated derivation of behavior vocabularies for autonomous humanoid motion. In: Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 225–232 (2003)
17. Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Biped walking pattern generation by using preview control of zero-moment point. In: Proceedings of the 2003 IEEE international Conference on Robotics and Automation, pp. 1620–1626 (2003)
18. Kapadia, M., Chiang, I.K., Thomas, T., Badler, N.I., Kider Jr., J.T.: Efficient motion retrieval in large motion databases. In: Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, pp. 19–28 (2013)
19. Kovar, L., Gleicher, M.: Flexible automatic motion blending with registration curves. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '03, pp. 214–224 (2003)
20. Kovar, L., Gleicher, M.: Automated extraction and parameterization of motions in large data sets. ACM Trans. Graph. **23**(3), 559–568 (2004)
21. Kwon, T., Cho, Y.S., Park, S.I., Shin, S.Y.: Two-character motion analysis and synthesis. IEEE Trans. Vis. Comput. Graph. **14**(3), 707–720 (2008)
22. Kwon, T., Hodgins, J.: Control systems for human running using an inverted pendulum model and a reference motion capture sequence. In: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 129–138 (2010)
23. Kwon, T., Shin, S.Y.: Motion modeling for on-line locomotion synthesis. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 29–38 (2005)
24. de Lasa, M., Mordatch, I., Hertzmann, A.: Feature-based locomotion controllers. ACM Trans. Graph. **29**, 131:1–131:10 (2010)
25. Lee, T., Park, J., Kwon, T.: Adaptive locomotion on slopes and stairs using pelvic rotation. Vis. Comput. **31**(6), 873–881 (2015)
26. Liu, C.K., Popovic, Z.: Synthesis of complex dynamic character motion from simple animations. In: SIGGRAPH 2002 Conference Proceedings, pp. 408–416 (2002)
27. Lpez-mndez, A., Gall, J., Casas, J., Gool, L.V.: Metric learning from poses for temporal clustering of human motion. In: Proceedings of the British Machine Vision Conference, pp. 49.1–49.12 (2012)
28. Min, J., Chai, J.: Motion graphs++: a compact generative model for semantic motion analysis and synthesis. ACM Trans. Graph. **31**(6), 153 (2012)
29. Min, J., Chen, Y.L., Chai, J.: Interactive generation of human animation with deformable motion models. ACM Trans. Graph. **29**(1), 9:1–9:12 (2009)
30. Mukai, T., Kuriyama, S.: Geostatistical motion interpolation. ACM Trans. Graph. **24**(3), 1062–1070 (2005)
31. Müller, M., Baak, A., Seidel, H.P.: Efficient and robust annotation of motion capture data. In: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 17–26 (2009)
32. Müller, M., Röder, T.: Motion templates for automatic classification and retrieval of motion capture data. In: Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 137–146 (2006)
33. Müller, M., Röder, T., Clausen, M.: Efficient content-based retrieval of motion capture data. ACM Trans. Graph. **24**(3), 677–685 (2005)
34. van de Panne, M.: From footprints to animation. Comput. Graph. Forum **16**(4), 211–223 (1997)
35. Park, S.I., Shin, H.J., Shin, S.Y.: On-line locomotion generation based on motion blending. In: Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 105–111 (2002)
36. Porres, A.B., Pelechano, N., Kapadia, M., Badler, N.I.: Footstep parameterized motion blending using barycentric coordinates. Comput. Graph. **47**(C), 105–112 (2015)
37. Rose, C., Cohen, M.F., Bodenheimer, B.: Verbs and adverbs: multidimensional motion interpolation. IEEE Comput. Graph. Appl. **18**(5), 32–40 (1998)
38. Shin, H.J., Kovar, L., Gleicher, M.: Physical touch-up of human motions. In: Proceedings of the 11th Pacific Conference on Computer Graphics and Applications, PG '03, p. 194 (2003)
39. Shin, H.J., Oh, H.S.: Fat graphs: constructing an interactive character with continuous controls. In: Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 291–298 (2006)
40. Singh, S., Kapadia, M., Reinman, G., Faloutsos, P.: Footstep navigation for dynamic crowds. Comput. Anim. Virtual Worlds **22**(2–3), 151–158 (2011)
41. Sugihara, T.: Standing stabilizability and stepping maneuver in planar bipedalism based on the best com-zmp regulator. In: Proceedings of the 2009 IEEE International Conference on Robotics and Automation, pp. 1966–1971 (2009)
42. Wang, Y., Huang, K., Tan, T.: Human activity recognition based on r transform. Comput. Vis. Pattern Recognit. **2007**, 1–8 (2007)
43. Wang, Z., Feng, Y., Qi, T., Yang, X., Zhang, J.J.: Adaptive multi-view feature selection for human motion retrieval. Sig. Process **120**(C), 691–701 (2016)
44. Wu, C.C., Zordan, V.: Goal-directed stepping with momentum control. In: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '10, pp. 113–118 (2010)
45. Zhou, F., De la Torre, F., Hodgins, J.K.: Hierarchical aligned cluster analysis for temporal clustering of human motion. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) **35**(3), 582–596 (2013)

Taekhee Lee is currently a Researcher of the FXGear company. He was a research scientist of the computer graphics labs at the Hanyang University and the Ewha Womans University. He received his Ph.D. degree in computer science from the Seoul National University in 2009. His research interest includes Computer Graphics and Machine learning and Motion Planning.

Daeun Kang received a B.S. degree in computer science from Sookmyung Women's University, Korea, in 2013. She is currently working toward a Ph.D. degree at the Hanyang University in Korea. Her research intersts are computer graphics and data-driven character animation.

Taesoo Kwon is a professor of computer science at the Hanyang University since 2012. He also leads a computer animation research group. He received his Ph.D. degree in coputer science from the Korea Advanced Institute of Science and Technology (KAIST) in 2007. From 2007 to 2012, he worked as a post-doctral researcher at the Seoul National University and at the Carnegie Mellon University. His research interests center on improving the design and performance of animation techniques, mainly through the application of physics-based models and machine learning techniques.