

Analysis and Synthesis of Interactive Two-Character Motions

Sang Il Park

Taeso Kwon

Hyun Joon Shin

Sung Yong Shin

CS/TR-2004-194

January 26, 2004

K A I S T

Department of Computer Science

Analysis and Synthesis of Interactive Two-Character Motions

Sang Il Park Taesoo Kwon Hyun Joon Shin
Sung Yong Shin

Abstract

In this paper, we deal with the problem of synthesizing novel motions performed by a pair of human-like characters while reflecting their interactions. Adopting an example-based paradigm, we address three non-trivial issues embedded in this problem: motion labeling, interaction modeling, and motion coupling. For the first issue, we provide a genetic search scheme for motion segmentation and classification, exploiting the structural and geometric features of motion segments. For the second issue, we propose a coupled movement transition graph to capture the interactions between the two characters and their individual behaviors. Based on a coupled hidden Markov chain, the transition graph is trained with example motions. For the last issue, we propose a scheme for synthesizing a novel sequence of coupled movements guided by the transition graph. We believe that our method can be used for two-character motion synthesis in game production.

1 Introduction

Interactions with other people are the most essential part of human life since a human being, as articulated by Aristotle, is a social animal. In particular, interactions between pairs of people are so ubiquitous that they arise naturally in various forms, e.g., martial arts such as Taekwondo and Judo, sports games such as boxing and tennis, coupled dances, and so on, daily person-to-person activities notwithstanding. Scenes of such forms can be observed very often in computer games and character animations. However, this issue has not been addressed well in the computer animation community.

Convincing interactive motions between two characters cannot be achieved by simply juxtaposing their individual motions. The motion of a character should be chosen in accordance with that of the other, and vice versa. That is, each action by

a character should be accompanied by a proper reaction by the other to alternate in combination. Moreover, the motions need to be adapted both in space and time so that actions and reactions are exchanged at the right place and time.

In this paper, we present an example-based method to synthesize novel motions performed by a pair of characters while properly reflecting their interactions. The example motions consist of a sequence of frames, each of which has the postures of both characters. Our problem of two-character motion synthesis involves three major issues in character animation: motion labelling, interaction modelling, and motion coupling.

Given a (two-character) motion sequence, we segment each (single-character) motion sequence into basic movements and categorize them into a collection of groups according to their similarities in structural and geometric features such that the movements in the same group can be blended together [33, 30]. To model each individual motion sequence we employ the movement transition graph in [18], in which nodes and edges represent movement groups and their transitions, respectively. Thus, the first issue is reduced to motion labelling. We address this issue by providing an effective solution for motion segmentation and classification.

Given a pair of motion transition graphs, each representing an individual motion sequence, our second issue is how to assemble them to capture the interactions between two characters. A pair of nodes in different graphs are connected by a “cross” edge if the corresponding group of movements affect those of the other. The resulting graph is called a “coupled movement transition graph,” which models the interactions between the characters. We adopt the notion of a coupled Markov chain as proposed in [6, 34] to train the transition model consisting of the two individual transition graphs.

Provided with the coupled graph, the last issue is how to make timely transitions to produce a sequence of interactive movements. Two characters interact with each other via cross edges guided by the coupled transition model while traversing their individual transition graphs from node to node to produce a sequence of movements. One of the characters can be designated as an avatar, if desired, which is controlled by a user.

The remainder of this paper is organized as follows: We first review related work in Section 2. In Section 3, we mainly deal with the motion labelling issue. In Section 4, a coupled movement transition graph is presented to capture the interactions between two characters embedded in the example motions. In Section 5, we show how to synthesize two-character motions based on the coupled transition graph. Experimental results are provided in Section 6. In Sections 7 and 8, we discuss limitations and further issues and finally conclude this paper.

2 Related Work

Example-based motion synthesis has been a popular research area in computer animation. We classify related results in this area into two groups, motion rearrangement [1, 2, 11, 19, 20, 22, 31, 37] and motion blending [14, 30, 33, 38]. Using a set of long motion sequences (possibly with some annotations), the former group of methods rearrange motion segments while trying to preserve the details of the original motions. Thus, the resulting motions, in general, are realistic although searching for a desired motion is time-consuming for a large motion database. On the other hand, the latter group of methods synthesize a desired motion by blending a set of labelled motions after establishing their temporal correspondence. The methods in this group have exhibited on-line controllability with real-time performance regardless of the database size. However, the details of the resulting motions may not be maintained when the labelled motion set is coarse.

Recently, Kim et al. [18] presented a hybrid method to generate rhythmic motions by combining the advantages of the two groups. Given an unlabelled example motion sequence, this method decomposes the sequence into a set of motion segments called “basic movements” and clusters them to obtain a collection of labelled sets of basic movements by exploiting their rhythmic structure. They modelled the example motion sequence as a “movement transition graph,” where a node and an edge represent a set of labelled movements and the transition from one labelled movement set to the other. Given a piece of input music, this graph is traversed from node to node guided by transition probabilities while synthesizing a movement at each visited node by blending the labelled movements assigned to the node. Unfortunately, this method cannot be applicable to non-rhythmic motions. We generalize the method for motion synthesis in general, specifically, two-character motion synthesis.

Zordan and Hodgins [39] presented motion capture-driven simulations that respond to a variety of unexpected impacts in the upper body. They demonstrated the capability of their method by simulating players in table tennis and boxing. In particular, the boxers exchanged “action and reaction”, although the interaction paradigm itself was not their main concern. Kim et al. [18] used the motion transition graph to synthesize animations for a dancing couple. Although the animation was interesting, the couple were treated in unity, that is, a single character with at least twice as many degrees of freedom (DOFs) as a human-like character.

To represent an unlabelled example motion sequence as a motion transition graph, the example motion needs to be segmented into labelled movements, which are basic motion segments. Although there have been rich research results on motion segmentation in computer vision, few results have dealt with full 3D human

motions. Recently, Fod et al. [10] presented a scheme to find a set of motion primitives from a sequence of motion data, based on principal component analysis (PCA) and K-means clustering. They segmented the motion sequence at the moments of zero-crossing of the angular acceleration. In computer graphics, Bindiganavale and Badler [3] employed the notion of zero-crossing in order to detect moments of interaction with environments. Kim et al. [18] used a similar idea for rhythmic motion segmentation. Arikan et al. [2] employed a vector supporting machine classifier to interactively annotate motion data. Inspired by the scheme of Fod et al. [10], we adopt K-means clustering for motion classification while exploiting the moments of zero-crossing for motion decomposition. Unlike their scheme, our method computes the number K of clusters automatically and is intended to handle full-body 3D motion data.

Yet another important related work concerns how to handle the interactions between two characters. In general, most important interactions occur at moments of collision. The problem of collision detection and response has been well addressed to provide exact solutions [9, 24, 23, 26, 25, 28]. However, the method of our choice is that of [17, 16] due to its on-line, real-time performance. To model the interactions, we employ the coupled Markov chain in [6, 5, 34] with slight simplifications.

3 Motion Labelling

3.1 Preliminaries

In this section, we describe how to partition a set of (individual) example motions into a collection of groups of motion segments called basic movements. Our objective for motion labelling is to model the behavior of a character (sampled by the example motions) as a movement transition graph [18], in which a node and an edge represent a group of basic movements and a transition from one group to another, respectively.

Motion transition graphs have shown their effectiveness in motion synthesis. Based on on-line motion blending, these graphs have been instrumental in combining the advantages of motion rearrangement and those of motion blending. The major premise of this approach is the availability of a motion labelling scheme to facilitate blending motions of a similar structure.

Kim et al. [18] labelled rhythmic motions, exploiting “motion-beat” patterns embedded in those motions. However, such patterns are not available in non-rhythmic motions. For non-rhythmic motions, motion labelling has been done manually, which limits the scope of motion blending approaches to be applica-

ble [30, 33]. Therefore, motion labelling itself is important in its own right, aside from two-character motion modelling and synthesis.

The issues in motion labelling are two-fold: segmentation and classification. Within the framework of genetic search, we attack these issues simultaneously. That is, given a sequence of candidate time instances for motion segmentation, a genetic search scheme is devised to produce promising sequences of time instances called segmentation sequences. For a given segmentation sequence, we cluster the motion segments into a collection of groups satisfying clustering criteria by employing a K -mean clustering algorithm [13] while determining the number k of clusters automatically. Our objective function selects the best among the solutions enumerated by the generic search scheme.

A two-character motion $\hat{\mathcal{M}}$ consists of a sequence of frames, each containing an ordered pair of postures denoted by $(\mathbf{m}^1(i), \mathbf{m}^2(i))$, where $\mathbf{m}^c(i)$ is the posture of an individual character c for $c = 1, 2$ at frame i , $1 \leq i \leq F$, and F is the number of frames in $\hat{\mathcal{M}}$. That is,

$$\hat{\mathcal{M}} = ((\mathbf{m}^1(1), \mathbf{m}^2(1)), \dots, (\mathbf{m}^1(F), \mathbf{m}^2(F))). \quad (1)$$

In the remainder of this paper, $\hat{\mathcal{M}}$ is often referred to as a coupled motion. The coupled motion $\hat{\mathcal{M}}$ can be considered to be composed of two individual motions of different characters tightly coupled at the frame level. We denote each individual motion by \mathcal{M}^c :

$$\mathcal{M}^c = (\mathbf{m}^c(1), \mathbf{m}^c(2), \dots, \mathbf{m}^c(F)), c = 1, 2. \quad (2)$$

A posture \mathbf{m} of a character is represented as

$$\mathbf{m} = (\mathbf{p}, \mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_J), \quad (3)$$

where $\mathbf{p} \in \mathcal{R}^3$ and $\mathbf{q}_1 \in \mathcal{S}^3$ describe the position and orientation of the root segment, $\mathbf{q}_j \in \mathcal{S}^3$ gives the orientation of joint j , $1 \leq j \leq J$, and J is the number of joints.

Let $\mathcal{E} = \{\hat{\mathcal{M}}_1, \hat{\mathcal{M}}_2, \dots, \hat{\mathcal{M}}_H\}$ the set of all coupled example motions performed by the same pair of characters. In the following sections, each individual motion of $\hat{\mathcal{M}}_k$, $1 \leq k \leq H$ in \mathcal{E} is divided into a sequence $\bar{\mathcal{M}}_k^c$ of basic movements, that is,

$$\bar{\mathcal{M}}_k^c = (\bar{\mathbf{M}}_1^c, \bar{\mathbf{M}}_2^c, \dots, \bar{\mathbf{M}}_{N_k^c}^c), \quad 1 \leq k \leq H \text{ and } c = 1, 2, \quad (4)$$

where $\bar{\mathbf{M}}_t^c$, $1 \leq t \leq N_k^c$ is a basic movement, and N_k^c is the number of basic movements in $\bar{\mathcal{M}}_k^c$. The basic movements of the same character from all example motions are classified into a collection \mathcal{S}^c of groups as follows:

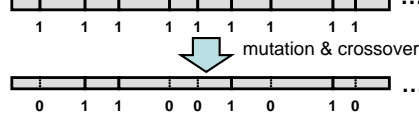


Figure 1: The initial gene (top), and a mutated gene (bottom).

$$\mathcal{S}^c = \{\mathbf{S}_1^c, \mathbf{S}_2^c, \dots, \mathbf{S}_{K^c}^c\}, \quad c = 1, 2, \quad (5)$$

where \mathbf{S}_j^c is a group of basic movements $1 \leq j \leq K^c$ and K^c is the number of groups in \mathcal{S}^c . The movements in the same group have a similar structure, such that they can be regarded as variations of the same movement.

3.2 Initial Segmentation

Given the example motions of a character, we first find a set of candidate time instances for segmentation. Those time instances should satisfy the following requirements:

- The time instances are sufficiently dense to segment every basic movement.
- The moments at which every basic movement starts and ends are included in the set.
- The number of time instances is not too large for efficient candidate sequence enumeration.

For two-character motions, most basic movements start and end with footsteps, as observed in martial arts and coupled dancing. For example, kicks and footwork in martial arts end with a footstep, and the ensuing movement also commences with a step. This observation suggests choosing the moments at which a foot touches the ground as the candidate time instances. As a foot may stay on the ground for a sequence of consecutive frames, the problem now is reduced to how to choose the right frame. We select the frame coinciding with the zero-crossing moment of toe acceleration as the candidate time instance in the sequence. The resulting time instances might yield finer segmentation than an “ideal” solution. However, they are adequate for the input to the next step (Section 3.3).

3.3 Segmentation Sequence Enumeration

Suppose that T candidate time instances are marked for motion segmentation. We can then represent the set \mathcal{T} of all segmentation sequence as follows:

$$\mathcal{T} = \{(b_1, b_2, \dots, b_T) : b_i, 1 \leq i \leq T \text{ is a binary digit}\} \quad (6)$$

such that

$$b_i = \begin{cases} 1 & \text{if the } i\text{-th candidate time instance is chosen,} \\ 0 & \text{otherwise,} \end{cases} \quad (7)$$

for $1 \leq i \leq T$. Each bit pattern in \mathcal{T} is defined as a gene encoding a segmentation sequence. The number of possible genes is 2^T , which is prohibitive to enumerate them all for large T .

Our genetic search scheme produces a sequence of promising genes, guided by a fitness function \mathcal{G} . Let $\mathcal{S} = \{\mathbf{S}_1, \mathbf{S}_2, \dots, \mathbf{S}_K\}$, be a collection of groups of motion segments obtained from a gene after cluster analysis (See Section 3.4.) Then, \mathcal{G} is defined as follows:

$$\mathcal{G} = \frac{\sum_{i=1}^K n_i}{K}, \quad (8)$$

where K is the number of groups (clusters), and n_i is the number of motion segments in group i , $1 \leq i \leq K$.

Our objective is to maximize \mathcal{G} while satisfying the clustering conditions, that is, the constraints on clustering such as the radius of a group and the inter-center distance between a pair of groups, as described in Section 3.4. By maximizing \mathcal{G} , the average number of motion segments is maximized, which eventually minimizes the number of groups in \mathcal{S} while satisfying the constraints.

The genes iteratively evolve by random mutation and cross-over. At each evolution step, the fitness function \mathcal{G} is used to evaluate the genes at the current generation based on their outcomes, that is, collections of groups computed in Section 3.4. The evolution terminates when the maximum number of iterations is reached. In practice, we found that 10,000 iterations are sufficient to find a high quality result when the population at each generation is 40.

Before evaluating the outcome of a gene with the fitness function \mathcal{G} , we adopt two heuristics to further refine the gene (See Figure 2.) Empirically, these two heuristics greatly accelerate our genetic search while yielding rapid convergency.

Our first heuristic prevents excessively long motion segments. In particular, we impose a restriction on the bit pattern of a gene so that no more than L consecutive bits are turned off. Because of mutation, a gene may result in more than L consecutive zero bits. In this case, we explicitly turn on the bit in the middle of the zero bit sequence to avoid such a gene.

Our second heuristic is to increase the average number of motion segments. The main objective of clustering motion segments is to facilitate motion blending. Thus, a group \mathbf{S}_i , $1 \leq i \leq K$ should contain as many ‘‘blendable’’ motion segments as possible. Thus, we examine a singleton group for possible elimination.

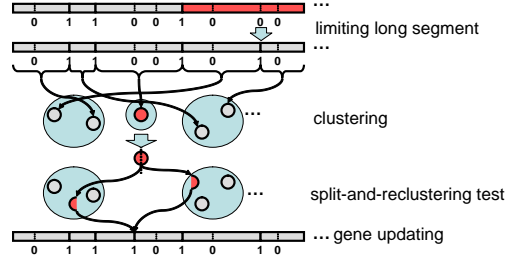


Figure 2: Explicit segmentation refinement.

Let M be a motion segment in a singleton group S_i for some i . Suppose that M contains one or more candidate time instances for motion segmentation. Then, we enumerate every such time instance to split M into two new motion segments M_a and M_b at that time instance. M_a and M_b are subject to reclassification into other groups in the collection S . Failure in reclassification of any of these segments leads to merging of the unsuccessful one with its neighbor for a retry. A successful reclassification results in a new collection of groups with a better fitness function value, since the group S_i containing M is eliminated. Accordingly, the bit pattern of the gene is updated.

Finally, the outcome of every gene at the current generation is evaluated with the fitness function \mathcal{G} to choose the superior genes. These genes go through mutation and cross-over to populate the genes in the next generation.

3.4 Segment Grouping

In this section, we deal with the motion classification issue: Given a segmentation sequence (represented by a gene) and the clustering criteria, we cluster the motion segments into a collection of groups that satisfy the criteria. Our classification scheme consists of two parts: We first classify the motion segments according to their structural feature similarities and then reclassify each class of motion segments according to their geometric feature similarities.

3.4.1 Classification by structures

To facilitate motion blending, the example motions should have similar structures [36, 33, 30]. We mainly use the information on footsteps since such information characterizes well various two-character motions in coupled dancing, boxing, Taekwondo, tennis, etc. In particular, we collected the structural information of

Table 1: Structural information $\mathbf{F}(\mathbf{M})$.

Symbols	Description	foot	time	value
$f_1(\mathbf{M})$	ground touching state	left	starting frame	0 or 1
$f_2(\mathbf{M})$	ground touching state	right	starting frame	0 or 1
$f_3(\mathbf{M})$	ground touching state	left	ending frame	0 or 1
$f_4(\mathbf{M})$	ground touching state	right	ending frame	0 or 1
$f_5(\mathbf{M})$	number of steps	left	entire duration	integer
$f_6(\mathbf{M})$	number of steps	right	entire duration	integer
$f_7(\mathbf{M})$	interaction state	-	entire duration	0 or 1

a motion segment \mathbf{M} into a seven-tuple $\mathbf{F}(\mathbf{M}) = (f_1(\mathbf{M}), f_2(\mathbf{M}), \dots, f_7(\mathbf{M}))$, where $f_i(\mathbf{M}), 1 \leq i \leq 7$ is defined in Table 1.

For example, $f_1(\mathbf{M})$ gives the left foot contacting state at the starting frame of the motion segment \mathbf{M} , which takes on the value 0 or 1. $f_5(\mathbf{M})$ denotes the number of left footsteps in \mathbf{M} . $f_7(\mathbf{M})$ takes on the value 0 or 1 depending on whether or not an end-effector is involved in an interaction with the other character. $f_i(\mathbf{M}), 1 \leq i \leq 6$ can be extracted automatically with methods in [27]. To identify the interacting end-effector, if any, we used an on-line puppetry technique [35] after modelling characters as sphere-trees.

After extracting the all structure tuples of motion segments, we lexicographically sort them to obtain a collection of movement classes such that the motion segments in the same class share the same structure (tuple value).

3.4.2 Classification by features

Given a collection of classes of motion segments, each with similar structure, our problem now is to reclassify those segments class by class according to their geometric features to form a collection of groups of motion segments, such that the movement groups satisfy the cluster conditions. To solve this problem, we need to specify the features of a motion segment and the clustering conditions.

Since the structural similarity among the basic movement in a class established in the linear space, the features of our choice are the end-effector positions together with the height of the hip center. These features are not only intuitive but also easily observable. Moreover, they match well with our clustering criteria to be provided soon. Let $\mathbf{e}(\mathbf{m}(i))$ be the feature vector of a posture $\mathbf{m}(i)$ at frame i . Then, $\mathbf{e}(\mathbf{m}(i)) = (\mathbf{e}_1(\mathbf{m}(i)), \mathbf{e}_2(\mathbf{m}(i)), \dots, \mathbf{e}_6(\mathbf{m}(i)))$, where $\mathbf{e}_j(\mathbf{m}(i)), 1 \leq j \leq 6$ is defined in Table 2. The geometric features $\mathbf{e}_j(\mathbf{m}(i)), 1 \leq j \leq 5$ are specified in the local coordinate frame of the hips. By collecting $\mathbf{e}(\mathbf{m}(i))$, the feature vector $\mathbf{E}(\mathbf{M})$ of a motion segment \mathbf{m} is given as follows.

$$\mathbf{E}(\mathbf{M}) = (\mathbf{e}(\mathbf{m}(1)), \mathbf{e}(\mathbf{m}(2)), \dots, \mathbf{e}(\mathbf{m}(F_{\mathbf{M}}))), \quad (9)$$

where $F_{\mathbf{M}}$ is the number of frames in \mathbf{M} .

The motion segments in a class have different numbers of frames in general. For effective classification, we adopt a dynamic time-warping technique [15, 7] to align their keytimes and resample the frames so that the motion segments in a class have the same number of frames. The motion segment with the longest frame sequence is used as the reference for dynamic time warping.

To provide the clustering conditions, let $\mathcal{S} = \{\mathbf{S}_1, \mathbf{S}_2, \dots, \mathbf{S}_K\}$ be the collection of groups of motion segments obtained from a segmentation sequence encoded in a gene. Then, the motion segments in the same group should be similar enough to be blended, and a pair of distinct groups should have sufficiently different types of motion segments. Given two motion segments M_i and M_j , their geometric feature similarity is measured in a Euclidean metric: $d(\mathbf{M}_i, \mathbf{M}_j) = \|\mathbf{E}(\mathbf{M}_i) - \mathbf{E}(\mathbf{M}_j)\|$. Using $d(\cdot)$, we define the radius of a group and the distance between a pair of groups as follows:

$$D(\mathbf{S}_j, \mathbf{c}_j) = \max_{\mathbf{M} \in \mathbf{S}_j} d(\mathbf{M}, \mathbf{c}_j), \text{ and } D(\mathbf{S}_i, \mathbf{S}_j) = d(\mathbf{c}_i, \mathbf{c}_j), \quad (10)$$

where $\mathbf{c}_j, 1 \leq j \leq K$ denotes the center of group \mathbf{S}_j . We specify the clustering condition:

$$\max_j \{D(\mathbf{S}_j, \mathbf{c}_j)\} \leq \gamma, \text{ and } \min_{i \neq j} \{D(\mathbf{S}_i, \mathbf{S}_j)\} \geq \delta. \quad (11)$$

That is, the radius of each group is smaller than γ and the inter-center distance between every pair of groups is larger than δ .

Now, we are ready to explain our clustering scheme. For each class of motion segments obtained in Section 3.3, we cluster the motion segments into groups. Here, our underlying assumption for the class-wise clustering is that the Euclidean distance between two motion segments in different classes is so large that they cannot be in the same group. Initially, a pair of motion segments \mathbf{M}_i and \mathbf{M}_j with the largest distance is chosen and set $K = 2$ to employ a K -means clustering technique [13] with the initial group centers $\mathbf{E}(\mathbf{M}_i)$ and $\mathbf{E}(\mathbf{M}_j)$. In general, we check

Table 2: Elements in a feature vector.

Symbols	Description	space
e_1	The position of the left foot	R^3
e_2	The position of the right foot	R^3
e_3	The position of the left hand	R^3
e_4	The position of the right hand	R^3
e_5	The position of the head top	R^3
e_6	The height of the hip center	R^1

the clustering criteria to provide new K and group centers for the next iteration. The iteration is repeated until the clustering criteria are satisfied.

Specifically, at each iteration, we find the group \mathbf{S}^* with the maximum radius and the pair of groups \mathbf{S}_i^* and \mathbf{S}_j^* that give the minimum inter-center distance. If $D(\mathbf{S}^*, \mathbf{c}^*) > \gamma$, then the group \mathbf{S}^* is split into two groups: Let \mathbf{M} be the movement in \mathbf{S}^* such that $d(\mathbf{M}, \mathbf{c}^*)$ is largest. Then, the new group $\{\mathbf{M}\}$ with its center $\mathbf{E}(\mathbf{M})$ is added to the collection and K is incremented by one. If $D(\mathbf{S}_i^*, \mathbf{S}_j^*) < \delta$, then \mathbf{S}_i^* and \mathbf{S}_j^* are deleted, a new group $\{\mathbf{S}_i^*, \mathbf{S}_j^*\}$ with its center $(\mathbf{S}_i^* + \mathbf{S}_j^*)/2$ is added, and K is decremented by one.

4 Interaction Modelling

In this section, we model the individual behaviors of characters and their interactions using movement transition graphs [18], based on hidden Markov models [6].

4.1 Individual Movement Transition Graphs

Provided with the collection of movement groups \mathcal{S}^c for each character c and the segmented example motions $\bar{\mathcal{M}}_k^c$, for $c = 1, 2, 1 \leq k \leq H$ (See Section 3), we employ the method in [18] to build individual movement transition graphs. We only describe how to construct the movement transition graph for character 1 since that of the other can be done in the same manner. Let $G^1 = (V^1, E^1)$ be the movement transition graph for character 1, where V^1 and E^1 denote the node and edge sets, respectively. A node in V^1 represents a group of basic movements in the collection \mathcal{S}^1 . We derive the edges in E^1 from the segmented example motions $\bar{\mathcal{M}}_k^1, 1 \leq k \leq H$.

A segmented motion $\bar{\mathcal{M}}_k^1$ consists of a sequence of basic movements, each of which is contained in $\mathbf{S}_j^1 \in \mathcal{S}^1$ for some $1 \leq j \leq K^1$. For each consecutive pair of basic movements, $(\bar{\mathbf{M}}_j^1, \bar{\mathbf{M}}_{j+1}^1)$ in $\bar{\mathcal{M}}_k^1, 1 \leq k \leq H$, suppose that $\bar{\mathbf{M}}_j^1 \in \mathbf{S}_a^1$ and $\bar{\mathbf{M}}_{j+1}^1 \in \mathbf{S}_b^1$. Then, we connect the pair of nodes corresponding to \mathbf{S}_a^1 and \mathbf{S}_b^1 by an edge. The direction of the edge is from the node for \mathbf{S}_a^1 to that for \mathbf{S}_b^1 . If $a = b$, the edge represents a self-transition. Such an edge reflects the ‘‘behavior continuity,’’ captured from the example motions. As suggested in [18], we also connect a pair of nodes by an edge if the last posture of the movement of one node is similar to the first posture of the other to reflect ‘‘kinematic continuity.’’ We explain how to assign the transition probability after fusing the individual graphs into a coupled transition graph.

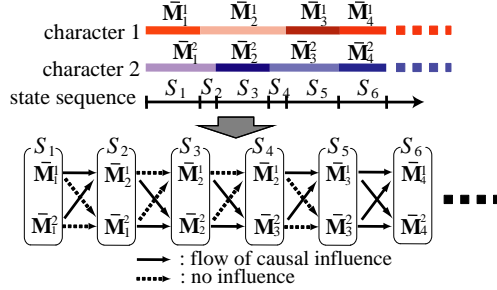


Figure 3: **Segmented motion data and their corresponding training state sequence.**

4.2 Coupled Movement Transition Graph Construction

In this section, we describe how to capture the interactions between the two characters, whose behaviors are modelled separately by their movement transition graphs. Coupled Markov chains have been used to model two or more processes interacting each other, each of which is modelled by a conventional hidden Markov chain [4, 6]. Interpreting a coupled example motion as the signals that are produced by two processes, we adopt a coupled hidden Markov chain to model them.

Consider a coupled motion sequence as shown in Figure 3. By performing the cluster analysis on an individual motion sequence as described in Section 3, the motion sequence is converted into a sequence of states, which correspond to the groups of basic movements. Thus, the cluster analysis enables us to extract a semi-optimal, if not optimal, sequence of states deterministically from lengthy raw motion data in which individual motion segments are not even identified. Combining the two sequences of individual states, we obtain the sequence of coupled states as illustrated in the bottom part of Figure 3.

However, we can observe that two individual sequences do not agree in transition times. For example, movement \bar{M}_2^1 partially or completely overlaps movements \bar{M}_1^2 , \bar{M}_2^2 , and \bar{M}_3^2 in time. That is, when a Markov chain makes a transition from one state to another, the other remains in the same state. In this case, we synchronize the two processes by introducing a virtual transition point at which the self-transition probability is one. The transition point can easily be detected in practice since the staying time in a state can be estimated as a process makes a transition to the state.

Exploiting the notion of a virtual transition, the coupled movement sequence in Figure 3 can be represented as a sequence of coupled states: $S = (S_1, S_2, S_3, S_4, S_5, S_6, \dots)$, where

$$\begin{aligned}
S_1 &= \begin{pmatrix} S^1(\bar{\mathbf{M}}_1^1) \\ S^2(\bar{\mathbf{M}}_1^2) \end{pmatrix}^T, S_2 = \begin{pmatrix} S^1(\bar{\mathbf{M}}_2^1) \\ S^2(\bar{\mathbf{M}}_2^2) \end{pmatrix}^T, S_3 = \begin{pmatrix} S^1(\bar{\mathbf{M}}_2^1) \\ S^2(\bar{\mathbf{M}}_2^2) \end{pmatrix}^T, \\
S_4 &= \begin{pmatrix} S^1(\bar{\mathbf{M}}_2^1) \\ S^2(\bar{\mathbf{M}}_3^2) \end{pmatrix}^T, S_5 = \begin{pmatrix} S^1(\bar{\mathbf{M}}_3^1) \\ S^2(\bar{\mathbf{M}}_3^2) \end{pmatrix}^T, S_6 = \begin{pmatrix} S^1(\bar{\mathbf{M}}_4^1) \\ S^2(\bar{\mathbf{M}}_4^2) \end{pmatrix}^T, \dots,
\end{aligned} \tag{12}$$

and $S^c(\bar{\mathbf{M}}_t^c)$, $c = 1, 2$ gives the state corresponding to the movement group containing $\bar{\mathbf{M}}_t^c$. From now we use states and groups interchangeably depending on contexts.

Brand [6] described how to choose a small number of coupled state sequences for training a coupled hidden Markov model without enumerating all possible sequences of coupled states. We instead used the coupled state sequences extracted from the example motions, since those sequences give all pairs of (individual) states in interaction that are embedded in the example motions.

4.3 Training with Example Motions

As proposed in [6], we first train each of the individual model parameters employing Baum-Welch’s EM (expectation-maximization) method [32]. Then, the transition probabilities across different processes are trained with the coupled state sequences extracted from the examples. A cross edge between two nodes in different movement transition graphs is built when the transition probability from one node to the other is greater than some threshold. Unlike a conventional hidden Markov model, the state sequences are explicitly given for our case, which greatly simplifies the procedure to estimate the model parameters.

The interactions between two characters largely depend on their spatial relationship. In particular, we choose two parameters to characterize this relationship: the distance between the two characters and their relative orientation. For an individual character, let $\mathbf{O} = (r, \theta)$ be the parameter vector for the spatial relationship, where r is the distance and θ is the orientation difference. Both of the parameters are extracted from the projected centers of root segments onto the ground plane on which the characters stand. In each (individual) state of a character, \mathbf{O} is used as the observation at the first frame of a basic movement performed in this state. The probability $P_i^c(\mathbf{O})$ of observing \mathbf{O} in state \mathbf{S}_i^c for character c is modeled as a Gaussian, that is,

$$P_i^c(\mathbf{O}) = N(\mathbf{O}; \mu_i^c, \mathbf{K}_i^c / \beta) \tag{13}$$

where μ_i^c and \mathbf{K}_i^c are the mean and covariance for character c , respectively, and β is a control parameter.

We now explain how to obtain the individual movement transition model for each character. Consider a known state sequence $\hat{S}^c = \hat{S}_1^c, \hat{S}_2^c, \dots, \hat{S}_{T_c}^c$ for char-

acter c , where T^c is the number of states in the sequence and $\hat{S}_t^c \in \mathcal{S}^c$. Let $\gamma_t^c(i)$ be the probability of being at state \mathbf{S}_i^c at time t , and $\xi_t^c(i, j)$ the probability of being in state \mathbf{S}_i^c at time t and \mathbf{S}_j^c at time $t + 1$. According to Baum-Welch's EM method, the transition probability \mathbf{a}_{ij}^c from state \mathbf{S}_i^c to state \mathbf{S}_j^c is estimated as follows [32]:

$$\mathbf{a}_{ij}^c = \frac{\text{expected \# of transitions from } \mathbf{S}_i^c \text{ to } \mathbf{S}_j^c}{\text{expected \# of transitions from } \mathbf{S}_i^c} = \frac{\sum_{t=1}^{T^c-1} \xi_t^c(i, j)}{\sum_{t=1}^{T^c-1} \gamma_t^c(i)}. \quad (14)$$

Based on the training state sequence \hat{S}^c , $\gamma_t^c(i)$ and $\xi_t^c(i, j)$ are computed deterministically, that is,

$$\gamma_t^c(i) = \begin{cases} 1 & \text{if } \hat{S}_t^c = \mathbf{S}_i^c; \\ 0 & \text{otherwise.} \end{cases}, \text{ and} \quad (15)$$

$$\xi_t^c(i, j) = \begin{cases} 1 & \text{if } \hat{S}_t^c = \mathbf{S}_i^c \text{ and } \hat{S}_{t+1}^c = \mathbf{S}_j^c; \\ 0 & \text{otherwise.} \end{cases}. \quad (16)$$

Plugging Equations (15) and (16) into Equation (14), we obtain the transition probability as follows:

$$\bar{\mathbf{a}}_{ij}^c = \frac{\text{\# of transitions from } \mathbf{S}_i^c \text{ to } \mathbf{S}_j^c \text{ in } \hat{S}^c}{\text{\# of transitions from } \mathbf{S}_i^c \text{ in } \hat{S}^c}. \quad (17)$$

To reflect kinematic continuity, we further adjust $\bar{\mathbf{a}}_{ij}^c$ as suggested in [18]. The initial state probability π_i^c is also defined based on the given state sequence. That is,

$$\bar{\pi}_i^c = \frac{\text{\# of times in } \mathbf{S}_i^c \text{ at } t = 1}{\sum_{k=1}^{K^c} \text{\# of times in } \mathbf{S}_k^c \text{ at } t = 1}. \quad (18)$$

Finally, given the coupled sequence \hat{S} , we construct cross edges between the two individual graphs. For simplicity, we only provide the cross edge from character 1 to character 2 since a cross edge from character 2 to character 1 can be derived symmetrically. Let $\xi_t^{12}(i, j)$ be the probability of being in state \mathbf{S}_i^1 at time t for character 1 and \mathbf{S}_j^2 at time $t + 1$ for character 2. From the results in [6], the transition probability \mathbf{a}_{ij}^{12} of a cross edge from state \mathbf{S}_i^1 to state \mathbf{S}_j^2 is obtained as follows:

$$\mathbf{a}_{ij}^{12} = \frac{\text{expected \# of transitions from } \mathbf{S}_i^1 \text{ to } \mathbf{S}_j^2}{\text{expected \# of transitions from } \mathbf{S}_i^1} = \frac{\sum_{t=1}^{T-1} \xi_t^{12}(i, j)}{\sum_{t=1}^{T-1} \gamma_t^1(i)}, \quad (19)$$

where T is the number of coupled states in sequence \hat{S} . By a similar argument in deriving $\bar{\mathbf{a}}_{ij}^c$, Equation (19) is reduced to

$$\bar{a}_{ij}^{12} = \frac{\# \text{ of transitions from } \mathbf{S}_i^1 \text{ to } \mathbf{S}_j^2 \text{ in } \hat{S}}{\# \text{ of transitions from } \mathbf{S}_i^c \text{ in } \hat{S}}. \quad (20)$$

Finally, from the transition model of the coupled Markov chain [6], the transition probability from one coupled state $(\mathbf{S}_i^1, \mathbf{S}_j^2)$ at time t to another $(\mathbf{S}_k^1, \mathbf{S}_l^2)$ at time $t + 1$ with the observations $(\mathbf{O}^1, \mathbf{O}^2)$ at time $t + 1$ is determined as follows:

$$\begin{aligned} P [\mathbf{s}_{t+1} = (\mathbf{S}_k^1, \mathbf{S}_l^2), \mathbf{o}_{t+1} = (\mathbf{O}^1, \mathbf{O}^2) | \mathbf{s}_t = (\mathbf{S}_i^1, \mathbf{S}_j^2)] \\ = \bar{a}_{ik}^1 \cdot \bar{a}_{jl}^2 \cdot \bar{a}_{il}^{12} \cdot \bar{a}_{jk}^{21} \cdot P_k^1(\mathbf{O}^1) \cdot P_l^2(\mathbf{O}^2). \end{aligned} \quad (21)$$

5 Motion Coupling

In this section, we describe how to synthesize a two-character motion while traversing the coupled movement transition graph. Interpreting each character as a motion-generating process, the two processes corresponding to the characters exchange actions and counter-actions via cross edges while traversing their own individual movement transition graphs. An action (or counter-action) can be regarded as the information on the next basic movement to be performed by a character. One of the character may be designated as an avatar. In this case, the character is under the control of a user while still communicating with the other via cross edges. We first describe our basic model for two-character motion synthesis, assuming that neither of them is under the user control. Then, we extend the basic model to incorporate the user control.

5.1 Basic Model

5.1.1 State Transition

The coupled transition graph represents a hidden Markov chain employed to model the process producing a coupled motion as its signal. An individual motion is given as a sequence of states (basic movements), each of which produces an observation. Thus, the two-character motion can be represented as a sequence of coupled states, $S = (S_1, S_2, S_3, \dots)$, as described in Equations (5) and (12).

First, we choose the coupled state $\mathbf{s}_1 = (\mathbf{S}_i^1, \mathbf{S}_j^2)$ randomly according to the initial state probabilities

$$P [\mathbf{s}_1 = (\mathbf{S}_i^1, \mathbf{S}_j^2), \mathbf{o}_1 = (\mathbf{O}^1, \mathbf{O}^2)] = \pi_i^1 P_i(\mathbf{O}^1) \pi_j^2 P_j(\mathbf{O}^2),$$

where π_i^1 and π_j^2 are the initial state probabilities of \mathbf{S}_i^1 and \mathbf{S}_j^2 , respectively (See Equation (18)). The initial individual observations \mathbf{O}^1 and \mathbf{O}^2 are computed after choosing at random the positions and orientations of the roots for both characters.

Then, we move from state to state guided by the coupled movement transition graph while synthesizing a basic movement at every visited node in the individual transition graphs. A basic movement synthesized at a node can be stitched with that of the following node seamlessly in most cases (For exception, see Section 5.1.4) since the node-to-node transitions reflect the kinematic consistency between the basic movements and their natural flow embedded in the example motions [30, 18].

Suppose that the (coupled) process is in state $(\mathbf{S}_i^1, \mathbf{S}_j^2)$ at time t , that is, $\mathbf{s}_t = (\mathbf{S}_i^1, \mathbf{S}_j^2)$, where \mathbf{S}_i^1 and \mathbf{S}_j^2 are the individual states for characters 1 and 2, respectively. Then, the process moves to the next state $\mathbf{s}_{t+1} = (\mathbf{S}_k^1, \mathbf{S}_l^2)$ and produce an observation $\mathbf{o}_{t+1} = (\mathbf{O}^1, \mathbf{O}^2)$ according to the probability

$$P [\mathbf{s}_{t+1} = (\mathbf{S}_k^1, \mathbf{S}_l^2), \mathbf{o}_{t+1} = (\mathbf{O}^1, \mathbf{O}^2) | \mathbf{s}_t = (\mathbf{S}_i^1, \mathbf{S}_j^2)]$$

given in Equation (21). As explained in Section 4.1, it may be possible that either $\mathbf{S}_i^1 = \mathbf{S}_k^1$ or $\mathbf{S}_j^2 = \mathbf{S}_l^2$ (but not both) because of a virtual transition. To obtain this probability, we need to not only specify $\mathbf{o}_{t+1} = (\mathbf{O}^1, \mathbf{O}^2)$ but also detect a virtual transition that any individual process makes.

Whenever moving into a new state, an individual process synthesizes a basic movement according to a movement specification, which initiates an actual transition to the next state of an individual process. The next transition time of the coupled process is the minimum of the two individual transition times. One of the processes (characters), which has not got the current basic movement done by the transition time, experiences a virtual transition.

Suppose that there is no virtual transition, that is, neither $\mathbf{S}_i^1 = \mathbf{S}_k^1$ nor $\mathbf{S}_j^2 = \mathbf{S}_l^2$. From the signal generation (motion synthesis) point of view, observation $\mathbf{o}_{t+1} = (\mathbf{O}^1, \mathbf{O}^2)$ can be regarded as characteristics of a desirable signal (coupled motion segment) to be synthesized at time t . As defined in Section 4.3, $\mathbf{O}^c = (\mathbf{r}^c, \theta^c)$, $c = 1, 2$, where \mathbf{r}^c and θ^c are the (projected) position and orientation differences with respect to the root of character c , respectively. Thus,

$$\mathbf{r}^1 = \mathbf{r}^2 \quad \text{and} \quad \theta^1 = 2\pi - \theta^2. \quad (22)$$

Assuming that the observations are similar at consecutive frames, we take the observation \mathbf{O}^c at the last frame of the current basic movement as the observation to be produced at the first frame in the next individual state of character c . Let $\Gamma(\mathbf{S}_i^1)$ be the set of incident nodes from \mathbf{S}_i^1 in the individual movement transition graph for character 1 and $\Gamma(\mathbf{S}_j^2)$ that of incident nodes from \mathbf{S}_j^2 for character 2. Then, the coupled state \mathbf{s}_{t+1} is chosen such that

$$\mathbf{s}_{t+1} = \arg \max_{\mathbf{S} \in \Gamma(\mathbf{S}_i^1) \times \Gamma(\mathbf{S}_j^2)} \{ P [\mathbf{S}, (\mathbf{O}^1, \mathbf{O}^2) | (\mathbf{S}_i^1, \mathbf{S}_j^2)] \}.$$

Now, suppose that a virtual transition is made by a character. Without loss of generality, we assume that $\mathbf{S}_i^1 = \mathbf{S}_k^1$, that is, the individual process corresponding to character 1 makes a self-transition at the frame of the virtual transition point. This frame is coincident with the first frame of the next basic movement performed by character 2. As character 1 stays in the same individual state, the observation $\mathbf{O}^1 = (\mathbf{r}^1, \theta^1)$ at the virtual transition point can be obtained trivially. From Equation (22), $\mathbf{O}^2 = (\mathbf{r}^1, 2\pi - \theta^1)$ at that frame which is taken as the observation in the next state of character 2. In this case, we also need to adjust transition probabilities \bar{a}_{ik}^1 and \bar{a}_{jk}^{21} , since character 1 stays in state \mathbf{S}_k^1 deterministically while character 2 moves to the next state \mathbf{S}_l^2 . Therefore, setting $\bar{a}_{ik}^1 = 1$ and $\bar{a}_{jk}^{21} = 1$, we choose \mathbf{s}_{t+1} as follows:

$$\mathbf{s}_{t+1} = \arg \max_{\mathbf{S} \in \{\mathbf{S}_i^1\} \times \Gamma(\mathbf{S}_j^2)} \{P[\mathbf{S}, (\mathbf{O}^1, \mathbf{O}^2) | (\mathbf{S}_i^1, \mathbf{S}_j^2)]\}.$$

5.1.2 Movement Synthesis

In this section, we provide a scheme for synthesizing a basic movement in an individual state. We employ an on-line motion blending scheme [30] to support interactive applications. Based on multi-dimensional data interpolation [33, 36], this scheme is composed of two stages : preprocessing and motion blending.

The preprocessing is a single-shot task for building a movement transition graph, in which a node represents a group of basic movements (See Section 3). First, every node is labeled with a group identification and a movement type (normal, defensive, or offensive). Then, the basic movements of each node is parameterized node by node. With emphasis placed on two-character motions such as boxing and Taekwondo, we choose three parameters: the height of target position, speed, and turning angle. The last two parameters of a basic movement can be extracted from the movement itself. However, to capture the first parameter, we need to compute the actual contact position on the body of the opponent. The target position is captured approximately by adopting the on-line puppetry technique in [35] after modelling the characters as sphere trees. The first parameter is mainly for interaction, and the last two parameters are to capture the locomotive aspects of a basic movement.

The basic movements in the node have the same number of key-times, that is, the moments of interaction of a character with the environment and the counterpart, such that the movements can be blended together. Typical examples are moments of hill-strike, toe-off, punching-on-the-body, etc. The keytimes are used for time-warping, that is, temporal alignment of the structural features of the example motions for effective blending. An interaction time is a special-type keytime at which the characters contact physically with each other, for example, moments

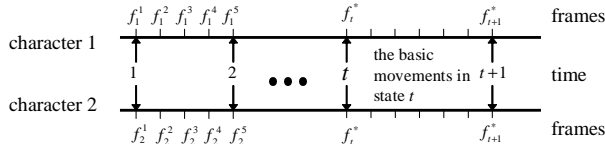


Figure 4: Frames and time.

of high-five, hitting-the-face, etc. We assume that there is at most one interaction time in a basic movement, since a basic movement is too short in general to accommodate two or more interaction moments. As mentioned previously, the keytimes can be marked automatically [35, 27].

The motion blending stage is performed in run-time to synthesize a basic movement at a node of an individual movement transition graph. This stage consists of four tasks: motion specification, weight computation, time-warping, and posture-blending. The last three tasks are typical components for motion blending, which are elaborated in [36, 33, 30]. We concentrate on the motion specification task, while minimizing our efforts on the others.

For on-line character control, the parameter vector $\alpha^c(f)$ of a basic movement at frame f for character c is specified frame by frame as follows:

$$\alpha^c(f) = (h^c(f), v^c(f), d^c(f)), i = 1, 2, \quad (23)$$

where h, v , and d denote target height, speed, and turning angle, respectively. So far “frame” and “time” are sometimes used interchangeably. From now, we use time exclusively for state transitions.

As illustrated in Figure 4, every state transition time is coincident with a frame. Suppose that the coupled process has made a transition from state $\mathbf{s}_t = (\mathbf{S}_i^1, \mathbf{S}_j^2)$ to $\mathbf{s}_{t+1} = (\mathbf{S}_k^1, \mathbf{S}_l^2)$. We derive the parameter vector $\alpha_{t+1}^c, c = 1, 2$ at time t guided by the example motions. The parameter vector α_{t+1}^1 should be constructed so that the corresponding basic movement looks like a natural successor of the movement with parameter α_t^1 while matching well that of the counterpart corresponding to α_{t+1}^2 , and vice versa. To achieve this, we exploit the root configuration $(\bar{\mathbf{p}}(\mathbf{M}(f)), \bar{\mathbf{q}}(\mathbf{M}(f)))$ of a character where $\mathbf{M}(f)$ denotes the posture of movement \mathbf{M} at frame f , and $\bar{\mathbf{p}}(\mathbf{M}(f)) \in \mathcal{R}^2$ and $\bar{\mathbf{q}}(\mathbf{M}(f)) \in \mathcal{S}^1$ represent the root position and orientation of the posture $\mathbf{M}(f)$ projected onto the plane where the character stands.

There are two cases depending on whether or not there occurs a virtual transition at time $t + 1$. First, we consider the case in which no virtual transitions occur at time $t + 1$. Let f_{t+1}^* be the first frame of the basic movements in state \mathbf{s}_{t+1} (See figure 4). Let

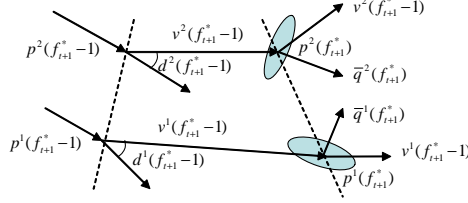


Figure 5: Movement parameters at frame f_{t+1}^* .

$$\alpha_{t+1}^c(f_{t+1}^*) = (h^c(f_{t+1}^*), v^c(f_{t+1}^*), d^c(f_{t+1}^*)), c = 1, 2, \quad (24)$$

be the parameter vector of character c at frame f_{t+1}^* . To give more variety to the synthesized movement, the target height $h^c(f_{t+1}^*)$ is chosen randomly at frame f_{t+1}^* within its range trained by the example motions and fixed with this value for the entire frames in the state.

To compute the remaining two parameters, we fit our motion specification strategy to the framework of scattered data interpolation. Our idea is to exploit of the group of basic movements and their blending weight values that have been used to produce the posture at the previous frame. Remember the collection of movement groups, $\mathcal{S}^c = \{\mathbf{S}_1^c, \mathbf{S}_2^c, \dots, \mathbf{S}_k^c\}$, $c = 1, 2$ defined in Equation (5), where \mathbf{S}_b^c , $1 \leq b \leq K_c$ is a group. Suppose that the posture $\hat{\mathbf{M}}(f-1)$ has been produced at frame $f-1$ in state \mathbf{S} by blending the corresponding posture $\mathbf{M}(f-1)$ of each basic movement \mathbf{M} contained in the group corresponding to \mathbf{S} . Then,

$$\hat{\mathbf{M}}(f-1) = \sum_{\mathbf{M} \in \mathbf{S}} w \cdot \mathbf{M}(f-1). \quad (25)$$

For later use, we maintain a set of ordered tuples

$$\mathbf{W}(f-1) = \left\{ (w, \mathbf{M}) : \hat{\mathbf{M}}(f-1) = \sum_{\mathbf{M} \in \mathbf{S}} w \cdot \mathbf{M}(f-1) \right\}, \quad (26)$$

where $w \in R$ is a weight value.

Initially (at frame f_{t+1}^*), the speed parameter at frame f_{t+1}^* is set to that at f_{t+1}^*-1 as shown in Figure 5, that is, $v^c(f_{t+1}^*) = v^c(f_{t+1}^*-1)$. For each remaining frame f in the next state \mathbf{S} (\mathbf{S}_k^1 or \mathbf{S}_l^2), the speed parameter $v^c(f)$ is computed from example motion data. Let

$$\vec{v}_e^c(\mathbf{M}(f)) = \bar{p}^c(\mathbf{M}(f)) - \bar{p}^c(\hat{\mathbf{M}}(f-1)) \quad (27)$$

be the displacement vector of character c from the (blended) root position $\bar{p}^c(\hat{\mathbf{M}}(f-1))$ at frame $f-1$ to the (pure) root position at frame f in a basic movement

$\mathbf{M} \in \mathbf{S}$. The displacement vector $\vec{v}^c(f)$ at frame f in the next state \mathbf{S} is obtained by blending $\vec{v}_e^c(\mathbf{M}(f))$:

$$\vec{v}^c(f) = \sum_{\mathbf{M} \in \mathbf{S}} w \cdot \vec{v}_e^c(f(\mathbf{M})), \quad (28)$$

where $(w, \mathbf{M}) \in \mathbf{W}(f - 1)$. From $\vec{v}^c(f)$, we obtain the speed parameter $v^c(f)$ by setting $v^c(f) = \|\vec{v}^c(f)\|$.

The turning angle $d^c(f)$ for the movement in the next frame can also be derived in a similar fashion. At the first frame f_{t+1}^* , we set $d^c(f_{t+1}^*) = d^c(f_{t+1}^* - 1)$. For each remaining frame f , $d^c(f)$ is obtained as follows:

$$d^c(f) = \sum_{\mathbf{M} \in \mathbf{S}} w \cdot \theta_e^c(\mathbf{M}(f)), \quad (29)$$

where $(w, \mathbf{M}) \in \mathbf{W}(f - 1)$, and

$$\theta_e^c(\mathbf{M}(f)) = \bar{\mathbf{q}}^c(\mathbf{M}(f)) - \bar{\mathbf{q}}^c(\hat{\mathbf{M}}(f - 1)) \quad (30)$$

is the angular displacement from the (blended) root orientation at frame $f - 1$ to the (pure) root orientation at frame f in a movement $\mathbf{M} \in \mathbf{S}$.

Now we consider the other cases, in which one of the individual transition is virtual. Without loss of generality, the individual process corresponding to character 1 makes the virtual transition. In this case, we make a trivial modification so that character 1 continues the current movement: By setting f_{t+1}^* to the next frame of the current basic movement of character 1, the same movement specification scheme can be applied.

5.1.3 Interaction Enhancement

A character may initiate a tightly-coupled movement with the other, for example, hitting-the-face, punching-on-the-body, and high-five, etc. In this case, the other character should choose the right basic movement to respond properly. An interaction moment is marked as a special keytime and annotated with a movement type such as offensive or defensive movement. There is at most one interaction time in a basic movement as mentioned in Section 5.1.2.

To facilitate correct interactions, we enhance the coupled Markov model with two heuristics. Our first heuristic is to restrict the state transition. Suppose that the coupled process makes a transition from $(\mathbf{S}_i^1, \mathbf{S}_j^2)$ to $(\mathbf{S}_k^1, \mathbf{S}_l^2)$ at time $t + 1$, that is, character 1 has just got a basic movement done in state \mathbf{S}_i^1 to perform a new movement in \mathbf{S}_k^1 . Moreover, suppose that \mathbf{S}_k^1 contains a group of basic movements each having an interaction time.

To make the other character perform the correct response, each basic movement in \mathbf{S}_l^2 should also have an interaction time. In particular, if \mathbf{S}_k^1 represents offensive movements, then \mathbf{S}_l^2 has to represent defensive ones, and vice versa. Therefore, we choose \mathbf{S}_l^2 among the incident nodes from \mathbf{S}_j^2 that are connected to \mathbf{S}_i^1 via cross edges and also have a different movement type.

Our second heuristic is to make a timely interaction, that is, temporal alignment of a pair of interaction moments. We first predict the interaction times using the initial parameter vectors. When the (predicted) interaction times are sufficiently close (in three frames for our experiments), then they are left intact. Otherwise, the defensive (less dynamic) movement is time-warped to align the interaction times.

5.1.4 Motion Stitching

To synthesize a movement at each node of a motion transition graph, we employ the on-line motion blending method proposed by Park et al. [30]. The difference between the final posture at the last frame in the previous state and the first frame in the current state is very small by the way in which the movement transition graph is constructed. Although this method in general produces high-quality movement at a node, there may be some discontinuity introduced during movement transition because of the time-warping for interaction enhancement. To fix this problem in an on-line manner, we adopt the notion of a motion displacement map in [12]. Unlike their original version, we leave the last frame of the movement in the previous state as it was and compute the displacement map for the current movement only, using the Hermit interpolation.

5.2 Interactive Model

In this section, we extend our basic model to allow user interactions. One of the characters is designated as an avatar controlled by a user to support on-line applications. To control the avatar, the user supplies movement specifications which are stored in a queue. On finishing the current movement, the avatar dequeues a motion specification one by one to drive the next movement. According to the motion specification, our model chooses a node of the movement transition graph for the avatar and extracts the movement parameters to resume state transition under our coupled Markov model. Whenever the queue is empty, the avatar is back to being under the control of the basic model. In the current model, our method supports a simple user interface of a "keyboard-and-mouse" type.

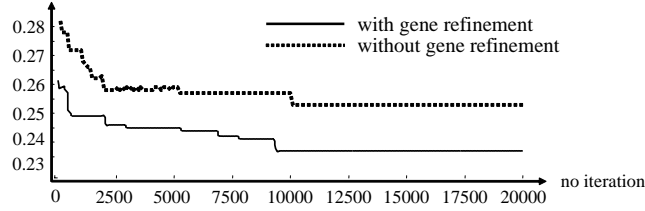


Figure 6: Convergence of $1/\mathcal{G}$.

Table 3: Test Data.

motions	sampling rate	# of frames	# of characters
<i>The Waltz</i>	30 fps	1800	2
boxing	120 fps	14810	1
Taekwondo	30 fps	3670	2

6 Experimental Results

We performed our experiment on an Intel[®] Pentium[®] PC (P4 2.2GHz Processor and 1GB RAM). Our character models had 43 DOFs (Degree Of Freedom) that consist of 6 DOFs for the pelvis, 3 DOFs for the spine, 7 DOFs for each limb, 3 DOFs for the neck, and 3 DOFs for the head. We used three motion sequences as test data (Table 3). We first present experimental results for motion labelling and then provide those for motion synthesis.

6.1 Motion Labelling

We apply our motion labelling scheme to all test motions given in Table 3, to show its effectiveness. As summarized in Table 4, our scheme worked well for those data. *The Waltz* is a typical coupled rhythmic motion, which was modelled as an

Table 4: Motion labelling results.

motions	# of frames	# of candidate time instances	Final results		# of iterations	fitness function values	processing time (min)
			# of basic movements	# of groups			
<i>The Waltz</i>	1800	58	36 (30)	11 (5)	10000	3.27	18
boxing	14810	274	199	57	10000	3.49	412
Taekwondo	3670	239	161	39	10000	4.13	156

individual movement transition graph in [18]. For this motion, the actual number of basic movements and that of groups were given in parentheses of columns 4 and 5, respectively. Our motion labelling scheme produces a few more basic movements and about two times as many groups. We believe that the results could be improved further by exploiting domain-specific knowledge such as dancing rules. The boxing motion was performed by a single actor. We used this motion as test data for segmentation only. The Taekwondo motion was performed by a pair of actors. Every group consists of short, similar basic movements with the same structure and thus can be blended together as demonstrated in the later experiments for motion synthesis.

Figure 6 exhibits a typical behavior of our genetic search scheme, in particular, the plotting of experimental results for the Taekwondo motion. x - and y -axes are the number of iterations and the reciprocal of the fitness function values, respectively. The solid curve illustrates the behavior of our scheme when equipped with the gene refinement heuristics. The dotted curve shows the behavior without these heuristics. The former converges much faster than the latter to yield a motion segmentation of similar quality. With the latter scheme, we found that 10,000 iterations with 40 genes at each generation yield good results. For classification of motion segments, we set $\gamma=40$ and $\delta=20$ for our experiments. Timing data for motion labelling are summarized in Table 4.

6.2 Motion Synthesis

To demonstrate the effectiveness of our scheme, we performed four experiments on two coupled motion data in Table 3: one for *The Waltz* and three for Taekwondo. Based on on-line motion blending [30], we can synthesize coupled motions at a rate of more than 1500 Hz.

Our first experiment was to synthesize *The Waltz* motion (See Figure 7). A couple performed a pair of individual motion in unison. Thus, we used the same parameter for each pair of basic movements performed simultaneously. In particular, the leading character’s movement parameter was used. The features of handling interaction moments were turned off since a dancing couple always was in contact. We did not exploit any knowledge on the rhythmic structure of the motion.

The remaining three experiments were performed on the Taekwondo motion to show the full capability of our synthesis scheme in three different modes: automatic, constrained, and interactive modes.

In the automatic mode (See Figure 8), the pair of characters moved autonomously guided by our coupled transition model, that is, the coupled movement transition graph trained with the example motion. As observed in the accompanying video,

the characters exhibited “statistically-similar” behaviors captured from the test motion, that is, a time-varying sequence of motion patterns including searching for chances and exchanging attacks and reactions.

In the constrained mode (See Figure 9), a character was constrained to move along a given trajectory while interacting with the opponent. In this experiment, the dark character following a circular trajectory was been chasing by the opponent. They exchanged kicks and blows with each other guided by the coupled Markov model. As in the previous mode, the distance and orientation difference between two characters changed the observation probabilities from time to time for the characters to choose right movements.

In the interactive mode (See Figure 10), one of the characters (dark) was designated as an avatar. The trajectory for this character was not known in advance. We directed the motion of the avatar in an on-line manner by pressing keys to choose movements while moving the mouse pointer to supply the trajectory. With these constraints imposed, the avatar interacted with the opponent according to the coupled Markov model.

7 Discussion

Although focused on locomotive motions, our motion labelling scheme is in fact quite general. In the current version, motions are classified mainly according to information on foot movements and end-effector positions. We believe that our scheme can be extended by incorporating similar information on hand movements. Being equipped with the powerful tool of motion labelling, the blending-based paradigm for motion synthesis will benefit most greatly from this scheme.

In an experiment, a coupled motion of Taekwondo was used as an example motion. However, it must be noted that this motion would be quite different from an actual (coupled) motion. Specifically, many combinations of actions and counter-actions were simulated by a pair of performers to avoid injury to any of the participants. This raises a serious problem in motion capture which is inherent in martial arts and fighting sports. One solution may be to incorporate the motion capture-driven simulation technique in [39] into our method to add more realism to actions involving actual contacts between characters.

The coupled transition graph effectively captures the general behaviors of the characters. To control their behaviors more directly, the transition graph should be used together with behavioral rules, which will also be an interesting research topic.



Figure 7: *The Waltz* motion.

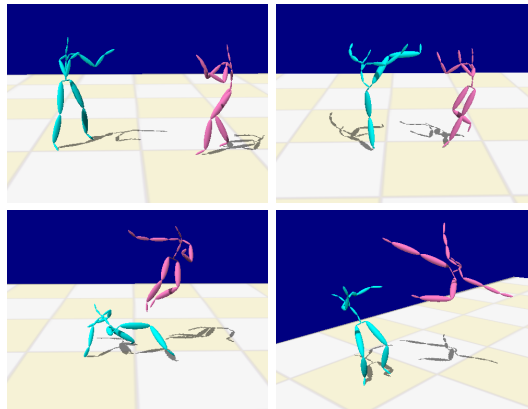


Figure 8: Taekwondo: automatic mode.

8 Conclusion

In this paper, we present an example-based method for synthesizing novel motions performed by a pair of characters. Based on a coupled hidden Markov chain, we model two-character motions as a coupled motion transition graph. A key component of the method is our motion labelling scheme, which greatly simplified the modelling task. This scheme can also be used as a general tool for motion labelling. We are aiming at elaborating our method for use in game production, in which it may find immediate applications.

References

- [1] O. Arikan and D. A. Forsyth. Interactive Motion Generation from Examples. *ACM Transactions on Graphics (Proc. SIGGRAPH 2002)*, 21(3):483–490, July 2002.



Figure 9: Taekwondo: constrained mode.

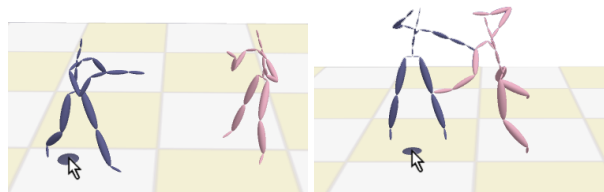


Figure 10: Taekwondo: interactive mode.

- [2] O. Arikan, D. A. Forsyth, and J. F. O'Brien. Motion synthesis from annotations. *ACM Transactions on Graphics (Proc. SIGGRAPH 2003)*, 22(3):402–408, July 2003.
- [3] R. Bindiganavale and N. I. Badler. Motion abstraction and mapping with spatial constraints. In *Proceedings of International Workshop CAPTECH'98*, pages 70–82, 1998.
- [4] M. Brand and A. Hertzmann. Style machines. *Computer Graphics(Proc. SIGGRAPH 2000)*, 34:183–192, 2000.
- [5] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 994 – 999, 1997.
- [6] Matthew Brand. Coupled hidden markov models for modeling interacting processes. Technical Report 405, MIT Media Lab, <http://xenia.media.mit.edu/brand/Publications.html>, 1996.

- [7] Armin Bruderlin and Lance Williams. Motion signal processing. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, pages 97–104, August 1995.
- [8] S. M. Chu and T. S. Huang. Audio-visual speech modeling using coupled hidden markov models. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 2009 – 2012, 2002.
- [9] J. Cohen, M. Lin, D. Manocha, and M. Ponamgi. I-collide: An interactive and exact collision detection system for large-scale environments. In *Proceedings of ACM Symposium on Interactive 3D Graphics*, pages 189–196, 1995.
- [10] A. Fod, M. J. Mataric, and O.C. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12(1):39–54, 2002.
- [11] A. Galata, N. Johnson, and D. Hogg. Learning variable-length markov models of behavior. *Computer Vision and Image Understanding*, 81:398–413, 2000.
- [12] Michael Gleicher, Hyun Joon Shin, Lucas Kovar, and Andrew Jepsen. Snap-together motion: Assembling run-time animation. *ACM Transactions on Graphics*, 22(3):702–702, July 2003.
- [13] Earl Gose, Richard Johnsonbaugh, and Steve Jost. *Pattern Recognition and Image Analysis*. Prentice Hall, 1996.
- [14] S. Guo and J. Robergé. A High-Level Control Mechanism for Human Locomotion Based on Parametric Frame Space Interpolation. In *Proceedings of Eurographics Workshop on Computer Animation and Simulation 96*, pages 95–107, August 1996.
- [15] Jelinek and Frederick. *Statistical methods for speech recognition*. MIT Press, 1997.
- [16] Dohan Kim, Ho Kyung Kim, and Sung Yong Shin. An event-driven approach to crowd simulation with example motions. Technical Report CS-TR-2003-186, Computer Science Division, Korea Advanced Institute of Science and Technology, <http://cs.kaist.ac.kr/library/tr/Archive/CS-TR-2003-186.pdf>, 2003.
- [17] Dong-Jin Kim, Leonidas J. Guibas, and Sung Yong Shin. Fast collision detection among multiple moving spheres. *IEEE Transactions on Visualization and Computer Graphics*, 4:230–242, 1998.

- [18] Tae Hoon Kim, Sang Il Park, and Sung Yong Shin. Rhythmic-motion synthesis based on motion-beat analysis. *ACM Transactions on Graphics (Proc. SIGGRAPH 2003)*, 22(3):392–401, July 2003.
- [19] L. Kovar, M. Gleicher, and Frédéric Pighin. Motion Graphs. *ACM Transactions on Graphics (Proc. SIGGRAPH 2002)*, 21(3):473–482, July 2002.
- [20] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard. Interactive Control of Avatars Animated with Human Motion Data. *ACM Transactions on Graphics (Proc. SIGGRAPH 2002)*, 21(3):491–500, July 2002.
- [21] X. Li and M. Parizeau. Training hidden markov models with multiple observations - a combinatorial method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4):371–377, 2000.
- [22] Y. Li, T. Wang, and H. Shum. Motion Texture: A Two-Level Statistical Model for Character Motion Synthesis. *ACM Transactions on Graphics (Proc. SIGGRAPH 2002)*, 21(3):465–472, July 2002.
- [23] M. Lin. *Efficient Collision Detection for Animation and Robotics*. PhD thesis, Dept. of Electrical Eng. and Computer Science, Univ. Of California, Berkeley, December 1993.
- [24] M. Lin and J. Canny. A fast algorithm for incremental distance calculation. In *Proceedings of IEEE Int'l Conf. Robotics and Automation*, pages 1008–1024, 1991.
- [25] M. Lin and D. Manocha. Fast interference detection between geometric models. *The Visual Computer*, 11(10):542–561, 1995.
- [26] M. Lin, D. Manocha, and J. Canny. Fast contact determination in dynamic environments. In *Proceedings of IEEE Int'l Conf. Robotics and Automation*, pages 602–608, 1994.
- [27] C. Karen Liu and Zoran Popovic. Synthesis of complex dynamic character motion from simple animations. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002.
- [28] B. Mirtich and J. Canny. Impulse-based simulation of rigid bodies. In *Proceedings of ACM Symposium on Interactive 3D Graphics*, pages 181–188, 1995.

- [29] A. V. Nefian, Luhong Liang, Xiaobo Pi, Liu Xiaoxiang, C. Mao, and K. Murphy. A coupled hmm for audio-visual speech recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 2013 – 2016, 2002.
- [30] S. I. Park, H. J. Shin, and S. Y. Shin. On-line locomotion generation based on motion blending. In *Proceedings of ACM SIGGRAPH Symposium on Computer Animation*, pages 105–111, July 2002.
- [31] K. Pullen and C. Bregler. Motion Capture Assisted Animation: Texturing and Synthesis. *ACM Transactions on Graphics (Proc. SIGGRAPH 2002)*, 21(3):501–508, July 2002.
- [32] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [33] C. Rose, M. F. Cohen, and B. Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications*, 18(5):32–40, Sept. 1998.
- [34] J.M. Sanchez, X. Binefa, and J.R. Kender. Coupled markov chains for video contents characterization. In *Proceedings of International Conference on Pattern Recognition (ICPR'2002)*, pages II: 461–464, 2002.
- [35] Hyun Joon Shin, Jehhee Lee, Sung Yong Shin, and Michael Gleicher. Computer puppetry: An importance-based approach. *ACM Trans. Graph.*, 20(2):67–94, 2001.
- [36] P. Sloan, C. F. Rose, and M. F. Cohen. Shape by example. In *Proceedings of 2001 ACM Symposium on Interactive 3D Graphics*, pages 135–144, 2001.
- [37] L. M. Tanco and A. Hilton. Realistic synthesis of novel human movements from a database of motion capture examples. In *Proceedings of the IEEE Workshop on Human Motion*, pages 137–142, 2000.
- [38] D. J. Wiley and J. K. Hahn. Interpolation synthesis for articulated figure motion. *IEEE Computer Graphics and Applications*, 17(6):39–45, 1997.
- [39] V. B. Zordan and J. K. Hodgins. Motion capture-driven simulations that hit and react. In *Proceedings of ACM SIGGRAPH Symposium on Computer Animation*, pages 89–96, July 2002.