Indexing and Retrieving Motions of Characters in Close Contact

Edmond S.L. Ho and Taku Komura

Abstract—Human motion indexing and retrieval is important for animators due to the need to search the databases for motions which can suitably be blended and concatenated. Most of the previous researches of human motion indexing and retrieval compute the Euclidean distance of joint angles or joint positions. Such approaches are difficult to apply for cases in which multiple characters are closely interacting with each other, as the relationships of the characters are not encoded in the representation. In this research, we propose a topology-based approach to index the motions of two human characters in close contact. We compute and encode how the two bodies are tangled based on the concept of rational tangles. The encoded relationships, which we define as *TangleList*, are used to determine the similarity of the pairs of postures. Using our method, we can index and retrieve motions such as one person piggy backing another, one person assisting another in walking, and two persons dancing/wrestling. Our method is useful to manage a motion database of multiple characters. We can also produce motion graph structures of two characters closely interacting with each other by interpolating and concatenating topologically similar postures and motion clips, which are applicable to 3D computer games and computer animation.

Index Terms—Content-based retrieval, character animation, human motion.

1 Introduction

Nowadays, the use of motion capture data is common in 3D computer games and animation. Therefore, there is a high demand for indexing and retrieving motion data efficiently so that animators can easily search for the motion they want in the database. Several methods have been proposed to search human motion data in the database by giving an example query. Such methods are targeted for motions of single characters, and most of them evaluate the similarities of the motions by comparing low-level attributes such as the joint angles or joint positions.

On the other hand, in computer animations and games, there are many scenes where multiple characters are densely interacting with each other. For example, in wrestling, the arms and legs of each character are tangled with those of the others in a complex way. To index such motions, we cannot simply apply the same methods as for single characters as the correlations between the characters will be ignored. Suppose one character is holding the neck of another character, as shown in Fig. 1. If we want to search and blend motions to these postures, we need to take into account the fact that the right arm of the black character is tangled with the neck of the gray character. Only motions that keep such relationships can be blended to the characters' motions. As previous indexing methods of human motions do not take into account the topological relationships of body segments, they will not work well for dense interactions of multiple characters.

• *The authors are with the School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, UK. E-mail: {s.l.ho, tkomura}@ed.ac.uk.*

Manuscript received 2 Mar. 2008; revised 18 Aug. 2008; accepted 8 Dec. 2008; published online 19 Dec. 2008.
Recommended for acceptance by R. Parent.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2008-03-0031.
Digital Object Identifier no. 10.1109/TVCG.2008.199.

In this paper, we use tangles [1] made between the segments to index the pair of postures of two characters. As there can be several tangles made by different segments of the body, we compose a data structure called a *TangleList* to represent the topological relationship of the two bodies. Given two *TangleLists*, we can compute their distance to evaluate the similarity of the set of postures. Then, it is possible to categorize the relationships of two characters and give an example relationship as a query to search for similar pairs of postures, as shown in Fig. 2. It is also possible to avoid interpolating/blending topologically dissimilar postures/motions which can cause body interpenetrations. As a result, our method is also useful for applications such as motion synthesis.

The paper is organized as follows: In Section 2, we discuss the related work. Section 3 explains how to extend the concept of tangles in knot theory to index, encode, and compare the relationships between two characters. In Section 4, experiments are conducted to show the performance of using the topological relationship for content-based retrieval and human animation. In Section 5, we discuss the possibilities of applying the topological relationship and conclude the work in this paper.

2 Related Work

Searching for similar postures or motions of humans is important for computer animation, as animators need to produce new motions by concatenating/interpolating different motions. In Motion Graphs [2], [3], [4], similar postures are searched and connected by edges to compose a graph structure. By traversing along the Motion Graph, it is possible to create a continuous animation of characters. In Motion Graphs, the postures are compared by calculating the difference of the joint angles/angular velocities [3] or the 3D locations/velocities of the joints [2], [4].

1077-2626/09/$25.00 © 2009 IEEE Published by the IEEE Computer Society

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

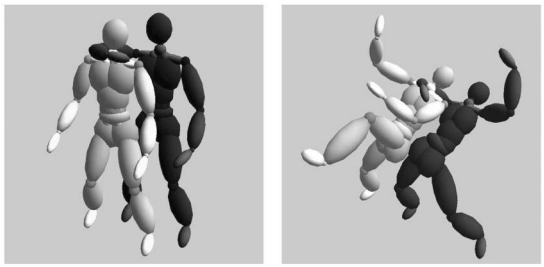


Fig. 1. The topological relationship where the arm is tangled with the neck is the same for the above two postures, although the kinematic joint angles or 3D location of the joints are different.

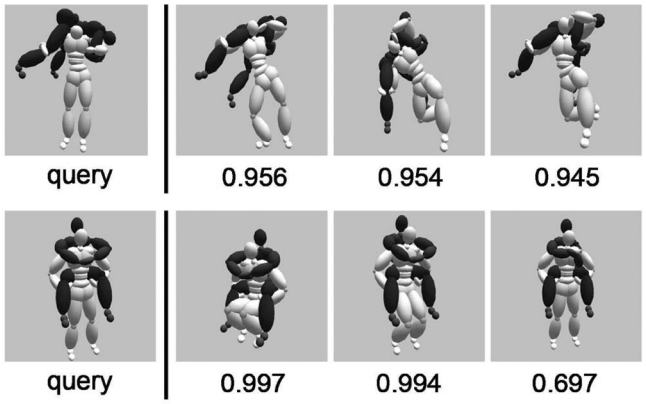


Fig. 2. The human motion retrieval based on topological relationships: three pairs of postures similar to the query postures are returned. The values on the bottom are the normalized similarity of the output posture with the query posture.

Content-based retrieval of human motions is used to search for similar motions that can be blended to synthesize new motions. Arikan et al. [5] propose using an SVM to classify the movements of characters. Feng et al. [6] propose building a hierarchical tree structure to quickly retrieve similar motions from the database. In order to consider temporal variations, dynamic time warping (DTW) is used to synchronize the timings of actions [7], [8]. Keogh et al. [9] present scaling which is more efficient than DTW and is more suitable for indexing large human motion databases. We categorize these methods as numerical methods, as the motions are classified based on distance of low-level attributes, such as the joint angles or 3D positions.

In many cases, when retrieving the motion data, users are more interested in the semantic similarities rather than the numerical closeness of low-level attributes. Müller et al. [10], [11] propose a semantic approach based on the correlation of four joint positions: a virtual plane is defined by the first three joint positions, and whether the last joint is in front or back of the plane is used as an operator to index the posture. They select a number of combinations of joints which are effective to distinguish human motions and use them to index and retrieve human motions. Such an approach is more robust in retrieving semantically similar motions, as the results are not affected by deviation of low-level attributes. However, they do not consider geometric features that are suitable for multiple characters.

In order to introduce a new systematic way to evaluate the relationships of different characters, topology is the key tool. Topology has been an important tool for reconstruction of 3D surfaces from cross-sectional images [12], [13], [14], shape modeling [15], image analysis [16], volume data analysis [17], and content-based retrieval of 3D objects [18]. Shinagawa et al. [12], [13], [14] extract the topological structure called a Reeb graph [19] from the cross-sectional images of biological tissues and use it to reproduce its 3D surface. Takahashi et al. [15] extend that idea to a 3D modeling system which lets the user specify the topological structure of the model. The users need to model only a portion of the surface and then the system estimates the rest of the shape based on the topological structure. In topology-matching [18], the Reeb graph based on the geodesic distances over the surface is composed, and the graph structures are matched to examine the similarity of the objects. The advantage of using topology for content-based retrieval is that it is invariant against orientation and local deformation. Topological information is also used for morphing 3D objects of different topological structures [20], [21]. In such cases, both the topological and geometric structures are used to interpolate the shape of the objects. However, none of these consider the topological relationships of multiple objects.

Ideas to handle the topological relationships can be found in knot theory; many invariants, such as crossing numbers and polynomials [22], are proposed to distinguish knots. In knot theory, there is a concept called tangles [1], which represents the relationship of two separate strings. Tangles are applied to describe the role of enzymes that change the topological structure of DNAs [23]. Ho and Komura [24], [25] use the concept of tangles for character animation: they detect tangles made between the two bodies by using Gauss Linking Integrals and apply them for character animation [24] and path planning [25]. Some basic ideas of their work, such as detecting the minimal tangles, are used in this paper.

In this research, we use the concept of rational tangles [1] to distinguish relationships of two characters. We extend the concept of tangles so that we can apply it not only to two-end strings, but also to tree structures such as humans, that are composed of multiple nodes and edges. Using this extended concept, we use the invariant of rational tangles for indexing and retrieving human motion data from the database.

3 Representation and Comparison of Topological Relationships

In this section, the methodology to compute and encode the way two human bodies are tangled with each other is explained.

The overview of the methodology is shown in Fig. 3. We first compute all the tangles made between the paths connecting the end effectors. The tangle information is then encoded into a data structure called a *TangleList*. We calculate the similarities between two pairs of postures by matching the tangles in the two *TangleLists*.

The rest of this section proceeds as follows: We first explain the concept of 2-tangles, which is the minimal unit for representing the topological relationship, and then

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

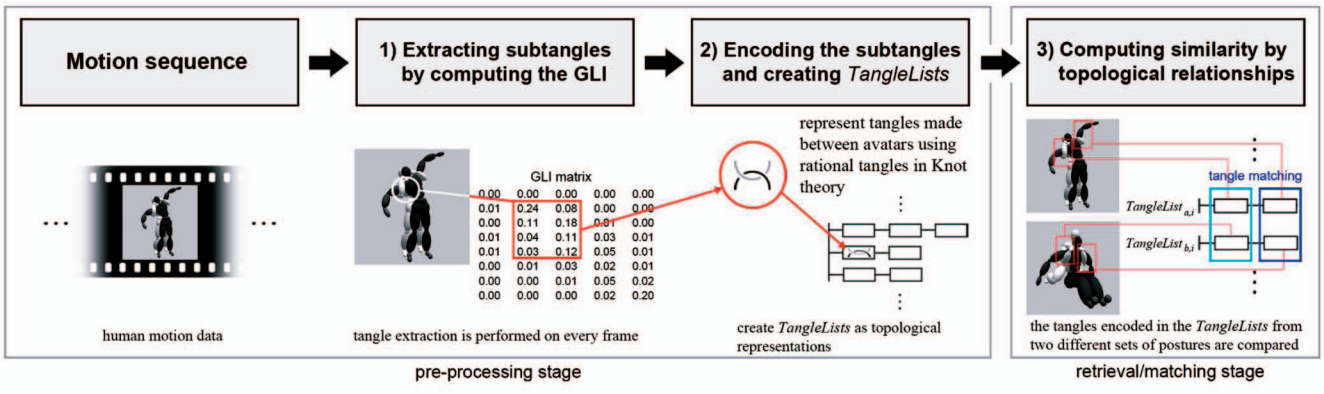


Fig. 3. The process of encoding the tangled postures. For every path connecting the end effectors, we 1) compute and 2) encode the tangle information, and 3) compare the results with those from other postures.

explain how to represent the relationship of body structures by a set of 2-tangles. Next, we explain how to compute and encode the tangle information from the 3D postures of the characters. Finally, we explain how to compare two sets of postures based on the encoded data.

3.1 2-Tangles

We use 2-tangles [1] as the minimal unit to represent the topological relationship of two characters. A 2-tangle is defined as a pair of strings whose end points are fixed in Euclidean 3D space. Examples of 2-tangles are shown in Figs. 4a, 4b, and 4c. Most of the tangles of the bodies can be represented by 2-tangles, or a set of 2-tangles.

2-Tangles can be categorized into rational tangles (Fig. 4a), self-knotted tangles (Fig. 4b), and prime tangles (Fig. 4c). The rational tangles are a group of tangles which can be composed of successive twists of two parallel strings around the vertical and horizontal axes. In this research, we limit the tangles made between the paths to rational tangles, because 1) they are the most basic tangles which can represent most of the postures of humans tangled with each other and 2) there is an invariant that can be used to distinguish every tangle from the others.

3.2 Tangles of Tree Structures

As we need to handle human characters, we have to compute the tangles made between tree structures. Trees are composed of edges and nodes, and therefore, tangles made between them will be more complex than those between single strings.

The tangles between trees can be examined by checking all the tangles made between the paths connecting the end effectors of the trees. The graph structure that is used to represent the human body in this research is shown in Fig. 5. There are 10 paths connecting the end effectors of this graph.

The topological relationships of two tree structures are considered equivalent when the 2-tangles of all the paths are equivalent. An example of a set of postures where the two characters are tangled is shown in Fig. 6a. In this case, path 5 of the gray character is tangled with paths 3, 4, 6, 7, 8, 9 of the black character. The advantage of representing the tangles of tree structures by the combination of all the 2-tangles is that although the tangle crosses a node while the characters are moving, as shown in Fig. 6b, the relationship is acknowledged unchanged. This kind of translation of tangles is called Reidemeister moves in knot theory. It is important that the state of the tangle does not change under Reidemeister moves.

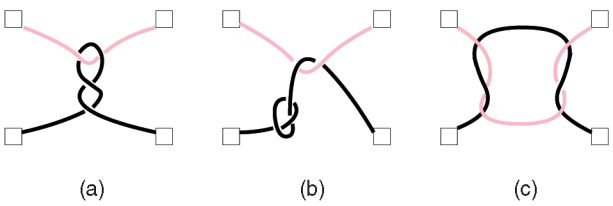


Fig. 4. Examples of 2-tangles: (a) rational, (b) self-knotted, and (c) prime tangles. The rational tangles can be composed by successive twists of two ends.

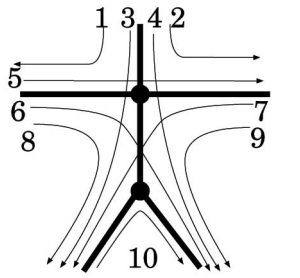


Fig. 5. The tree structure of the graph that is used to represent the body structure. There are 10 paths that connect the end effectors.

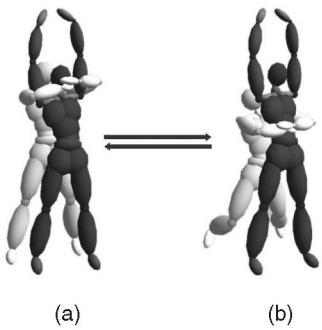


Fig. 6. The homeomorphic Reidemeister moves after which the relationship must be considered equivalent.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

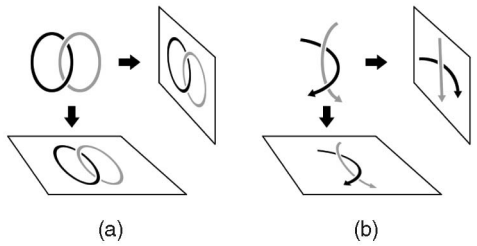


Fig. 7. The projection plane does not affect the minimum crossing numbers for links (a) but does for tangles (b).

3.3 Detecting the Tangles

In this subsection, we explain how to find out the tangles directly from the 3D position/orientation of the body segments by computing the Gauss Linking Integrals.

In knot theory, it is always assumed that the tangles or knots are projected onto a 2D plane. As knots or links are closed curves, the projection plane does not affect invariants such as the minimum crossing numbers (Fig. 7a). However, in case of tangles, the projection plane affects the number of crossings (Fig. 7b), and as a result, we cannot use the crossing numbers as the features. It is also difficult to define a plane to project the tangles onto, as the bodies are always moving around and their orientations are changing from time to time. Here we explain a method to compute the tangles directly from the 3D location/orientation of the bodies.

3.3.1 Gauss Linking Integrals

We use Gauss Linking Integral (GLI), which is the average number of crossings when viewing the tangle from all directions divided by two. In the case when the absolute value of the GLI is over 1 (Fig. 8a), the two curves are twisting around each other once. If it is over 0.5 (Fig. 8b), the two curves are tangled, which means that they cannot be separated into two by a plane between them without cutting either of them. And if it is less than 0.5 (Fig. 8c), the two curves are untangled. The GLI of two directed curves γ_1 and γ_2 can be computed by

$$GLI(\gamma_1, \gamma_2) = \frac{1}{4\pi} \int_{\gamma_1} \int_{\gamma_2} \frac{d\gamma_1 \times d\gamma_2 \cdot (\gamma_1 - \gamma_2)}{\|\gamma_1 - \gamma_2\|^3}, \quad (1)$$

where $\times$ and $\cdot$ are cross product and dot product operators, respectively. Since we are only interested in knowing whether the bodies are tangled or not, we compute the GLIs for the body segments, and if their absolute values are over 0.5, we tag them as tangled and use these results when

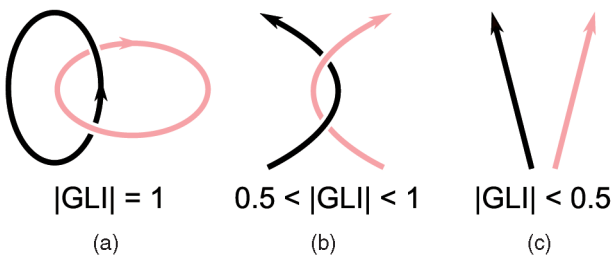


Fig. 8. The GLI of two directed curves when (a) one strand is surrounding the other, (b) singly tangled, and (c) untangled (c).

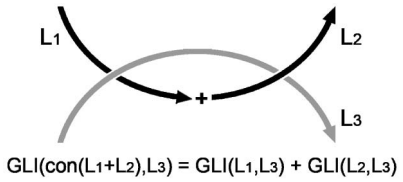


Fig. 9. GLI satisfies the commutative rule.

comparing the postures. The details are explained in the following subsections.

The GLI satisfies commutativity. It also has a distributive property over the operation of concatenating two strands. Mathematically, these can be written as

$$GLI(L_1, L_3) = GLI(L_3, L_1), \quad (2)$$

$$GLI(con(L_1, L_2), L_3) = GLI(L_1, L_3) + GLI(L_2, L_3), \quad (3)$$

where L_1, L_2, and L_3 represent strands, and $con()$ is an operator connecting two strands (see Fig. 9). These are convenient features when computing all the tangles made between the segments.

3.3.2 Efficient Detection of Tangles by the GLI Matrix

This process is similar to what has been done in the work of Ho and Komura [24], [25]; however, instead of conducting double integrations of Gauss Integrals for every path as stated in (1), we do this more efficiently by representing the segments connecting the joints by line segments and calculating the GLI between the line segments using the analytical solution [26]. The readers are referred to the Appendix for the details. As a result, the GLI between arbitrary local paths can be simply calculated by summing the GLI of segment pairs.

Now we explain how to find all the tangles based on the GLI. In the rest of the paper, we will call the entire tangle composed of the whole strands as tangles and their subsets composed of part of the strands as subtangles. We will detect the rational tangles made between two serial links of rigid segments. This starts by computing all the subtangles made between the two links. Suppose the two links A and B are composed of m and n segments, respectively. An $m \times n$ matrix that contains the GLI between every pair of body segments is composed. Let us define this matrix as the GLI matrix. First, we find out all the minimal subtangles, for which the absolute sum is larger than 0.5 by scanning all the submatrices in the GLI matrix.

An example of a 2-tangle and its GLI matrix is shown in Fig. 10. Five subtangles are found and the corresponding submatrices are surrounded by the rectangles.

3.4 Encoding the Tangles

Here we explain a method for encoding the tangles into a structure called a *TangleList*, which is equivalent to the continued fraction of rational tangles. The continued fraction is a complete invariant that can distinguish each kind of rational tangles from others [1]. It is the rational number that can be calculated by

$$a_n + \cfrac{1}{a_{n-1} + \cfrac{1}{a_{n-2} + \cdots + \cfrac{1}{a_1}}}, \quad (4)$$

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

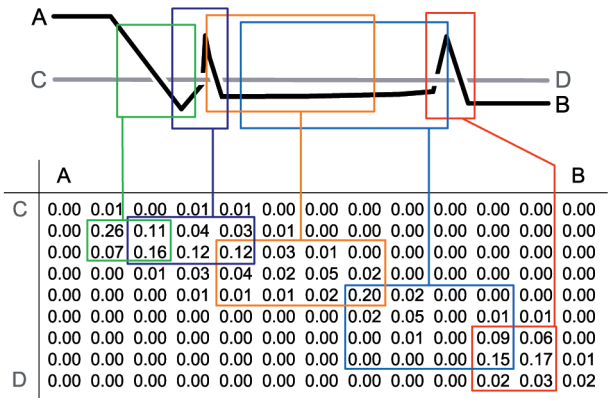


Fig. 10. A 2-tangle and its GLI matrix. The five subtangles and their corresponding submatrices are visualized.

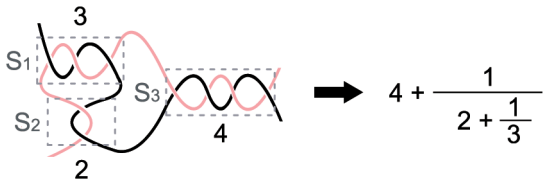


Fig. 11. An example of a rational tangle and its corresponding continued fraction.

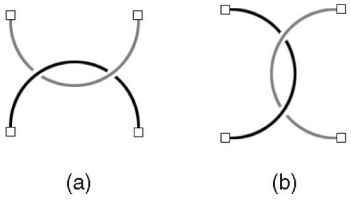


Fig. 12. Examples of (a) horizontal tangles and (b) vertical tangles.

where $(a_1, \ldots, a_n)$ are integers that represent the numbers of successive horizontal and vertical twists of the two strands. An example of a rational tangle and its corresponding continued fraction is shown in Fig. 11. As continued fraction requires the tangle to be projected onto a plane, we propose to use a data structure called TangleList, which is equivalent to the continued fraction but does not require the tangle to be projected onto a plane.

Let us define the rational tangle to be encoded by T. Rational tangles are composed of successive, integer numbers of horizontal twists (Fig. 12a), and vertical twists (Fig. 12b). The composition of the *TangleList* proceeds by untangling T, which is done by successive twists of the two ends, as shown in Fig. 13. In our case, we do not necessarily twist for an integer number of times, as the tangles are defined in the 3D space based on the location of the joint positions. Instead, we keep the GLI value of each twist. On the other hand, it is also difficult to define any absolute standards for vertical/horizontal twists as the bodies can be oriented in arbitrary ways. Therefore, we assume that the initial twist is a horizontal twist.

In order to correctly untwist tangle T, we need to define the type of the subtangles. Let us assume that a tangle T is composed of two strands a and b, and the directions are defined in both strands. The subtangles S_i composing T can be categorized into four types: those composed by

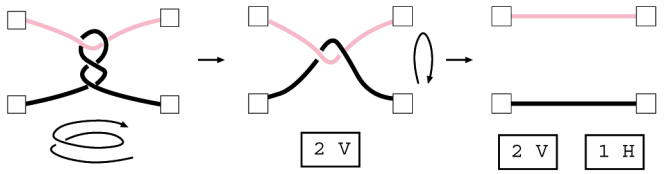


Fig. 13. An example of encoding a rational tangle while untangling it. The tangle is encoded as "two vertical twists (2V) and one horizontal twist (1H)."

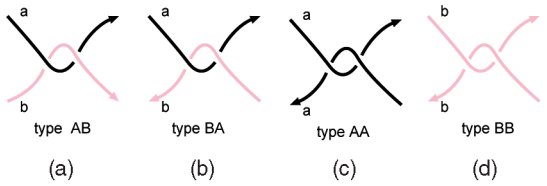


Fig. 14. The four types of subtangles composing the tangle made by two strands *a* and *b*. We assume that the directions of the strands are defined. These are composed of *a* and *b* ((a) and (b)), only by *a* (c), and only by *b* (d).

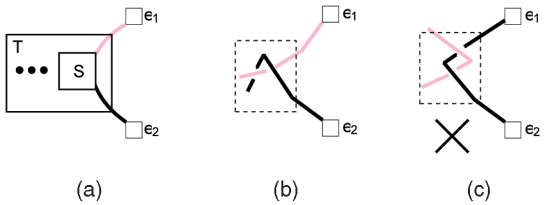


Fig. 15. The conditions for selecting the subtangle S to be untangled. (a) The two ends of S need to be the end points of T. (b) The closest minimal tangle from e_1 and e_2 must be the same. (c) e_1 and e_2 are not connected in this minimal tangle.

strand a and a (type AA), strand b and b (type BB), and by strand a and b in the forward direction (type AB), and in the opposite direction (type BA). The four types are shown in Fig. 14.

Now we can define three attributes for each subtangle: the $type$, GLI, and $twist$. The $type$ is either of BB, AB, BA, AA, the GLI keeps the GLI value of the subtangle, and $twist$ tells whether the subtangle is either made by a "vertical" (V) twist or an "horizontal" (H) twist.

The untwisting can be done systematically by the following process. First, we put all the subtangles into a group defined by G. We start by finding the subtangle to be untwisted in G. The subtangle S that satisfies the following conditions is selected:

1. Two end points of S, defined here by e_1 and e_2, are also the end points of T (Fig. 15a), which means there is no other subtangle between the end points of S and those of T.
2. S can be untangled by twisting e_1 and e_2.

The second condition can be judged by checking whether 1) the closest minimal tangle from e_1 and that from e_2 are the same (Fig. 15b) (remember the minimal tangles are those whose GLI values are above 0.5, which are found using the method explained in Section 3.3.2) and 2) e_1 and e_2 are not connected in this minimal tangle (Fig. 15c).

Once the subtangle S that satisfies these conditions is found, we virtually untangle this subtangle by removing it

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

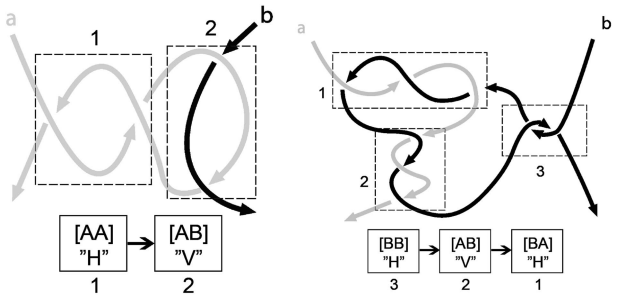


Fig. 16. Examples of encoding rational tangles. The encoded *TangleList* is shown on the bottom.

from G and putting it into the top of *TangleList*, and repeat the process iteratively until the whole tangle is untangled. When untangling a subtangle, we record its type (either of BB, AB, BA, AA). If it is different from that of the previous subtangle, we can say that the twist direction has switched too. Otherwise, the subtangles will be merged. This merging operation is the same as concatenating the rational tangles.

If no subtangle that satisfies conditions 1 and 2 is found, T is not a rational tangle. In that case, we do not use this tangle to evaluate the similarity. The process of encoding the tangle information is summarized in Algorithm 1.

Algorithm 1. Encoding the tangle information
Put all subtangles in G
while G is not empty **do**
Pick out the subtangle S in G that satisfies condition 1 and 2
if $S = NIL$ **then**
T is not a rational tangle. Exit()
end if
Check the two ends of S and set $S.type$.
if $TangleList$ is empty **then**
$S.twist = H$
else
if $S.type! = S_p.type$ **then**
$S.twist = !S_p.twist$
else
merge S and S_p and define it as S
$S.twist = S_p.twist$
$S.type = S_p.type$
end if
end if
$S_p = S$
Add S into $TangleList$
end while

Two examples of tangles encoded by this method are shown in Fig. 16. Whether the two tangles are the same or not can be checked by comparing their $TangleLists$. The details are explained in the next subsection.

3.5 Computing Similarities by Topological Relationships

Given two different sets of postures where the two characters are tangled with another, we compute the similarities of the postures/movements by using the encoded tangle information.

As shown in Fig. 5, there are 10 paths connecting every pair of end effectors in the articulated structure we use in the system. Therefore, there are $10 \times 10 = 100$ pairs of paths formed between two characters. For every pair of paths, the system builds the $TangleList$ structure, in which all the subtangle information is saved.

In order to compare the similarity between the pairs of postures, we have prepared two distance functions according to the application. Suppose we have two pairs of postures: the first pair is defined by u and the other by v. Each pair of postures has 100 $TangleList$s, each of which represents how the paths between the end effectors are tangled.

In the first distance function, we evaluate the overall topological similarity between two pose pairs, which will be useful for managing a database of human motions. This is done by accumulating the distance between the corresponding TangleLists in u and v

$$d = \sum_{i=1}^{100} dist(TangleList_i^u, TangleList_i^v), \tag{5}$$

where i is the index of path pairs, $TangleList_i^u$ and $TangleList_i^v$ are the ith TangleList in u and v, respectively, and $dist()$ is a function that computes the distance between two TangleLists. The distance between TangleLists is computed by finding the matching subtangles in the two TangleLists and comparing their GLI values. When matching a subtangle, we use a method similar to dynamic time warping: if the lth subtangle of TangleList 1 is matched with the kth subtangle of TangleList 2, subtangles of TangleList 1 whose index are larger than the l cannot be matched with subtangles of TangleList 2 whose index is smaller than k

$$\begin{aligned} &dist(TangleList_i^u, TangleList_i^v) \\ &\quad = \min_{j_l^u, j_l^v} \sum_{k=1} (TangleList_i^u[j_k^u]GLI - TangleList_i^v[j_k^v]GLI)^2 \\ &\qquad + P, \end{aligned} \tag{6}$$

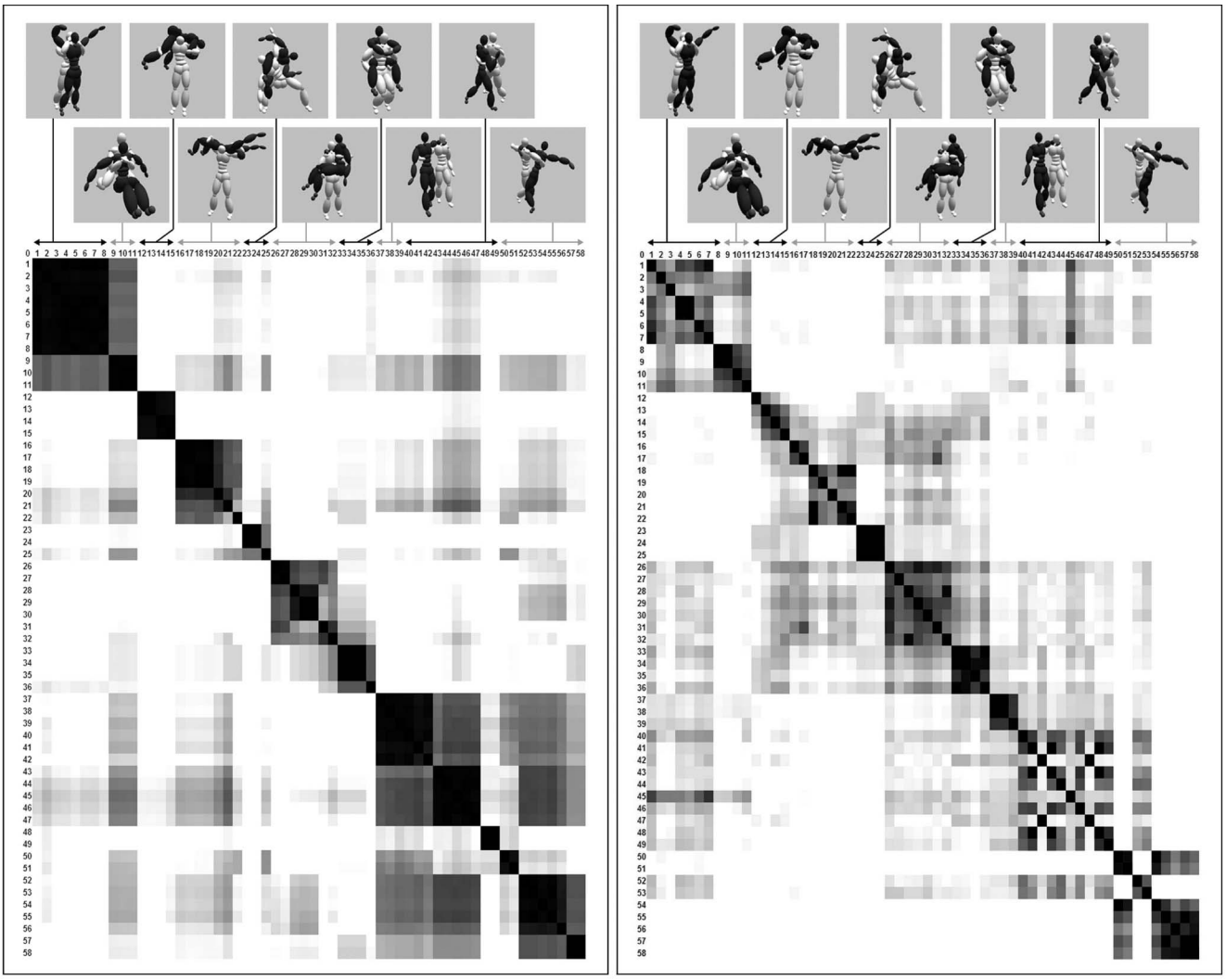
where $0 < j_1^u < \cdots < n_{u,i}$, $0 < j_1^v < \cdots < n_{v,i}$, $n_{u,i}$ and $n_{v,i}$ are the number of subtangles in $TangleList_i^u$ and $TangleList_i^v$, respectively, and P is the sum of squares of the GLI of the subtangles in $TangleList^u$ and $TangleList^v$ which could not find a matching subtangle.

The second distance function is used to judge whether postures and motions can be interpolated or blended without interpenetrations of the segments. For such an application, it is more meaningful to evaluate the maximum GLI difference rather than accumulating the difference of GLI for all the combination of paths

$$\begin{aligned} &blendable(TangleList_i^u, TangleList_i^v) \\ &\quad = \max_k (\min_{j_l^u, j_l^v} |TangleList_i^u[j_k^u]GLI - TangleList_i^v[j_k^v]GLI|). \end{aligned} \tag{7}$$

We can estimate that when $blendable()$ returns a value larger than 0.5, it is unlikely that the postures can be linearly

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.



(a) (b)

Fig. 17. A similarity matrix of different postures based on (a) topological relationships and (b) Euclidean distance computed by Kovar et al.'s point cloud metric [4].

interpolated without penetration, as some parts of the body need to be tangled/untangled.

4 Experimental Results

The results of computing the similarities of posture pairs based on the topological relationships are presented in this section. We also show examples of interpolating/concatenating different motion clips by using topological relationships as a measure.

4.1 Comparison between Topological and Euclidean Distance

We first compared the performance of the distance metrics of (6) which is based on topological relationship, and that based on Euclidean distance, to distinguish posture pairs which are semantically similar/dissimilar. Fifty eight posture pairs which can be divided into the following 10 categories were prepared:

1. Full Nelson holds (motions 1-8),
2. back holds (9-11),
3. firefighter carries (12-15),
4. backbreakers (16-22),
5. octopus holds (23-25),
6. one person carries (26-32),
7. piggy back carries (33-36),
8. walk support (37-39),
9. dancing (40-49),
10. Latin dance (50-58).

We computed the normalized similarities between the postures, whose values are between 0 (less similar) and 1 (highly similar) by using both the topological and Euclidean distance metrics. They are computed by $1 - d/d_{max}$, where d is the distance between the posture pairs and d_{max} is the maximum distance found among all the pair postures used in this research. For the Euclidean distance, we used the point cloud metric proposed by Kovar et al. [4].

The results are visualized in Fig. 17. The darker areas represent higher similarity and the lighter areas represent lower similarity. The postures of the same kind are grouped together along the row/columns.

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

Fig. 18. The representative postures of the three groups of dancing motions (42, 43, 48).

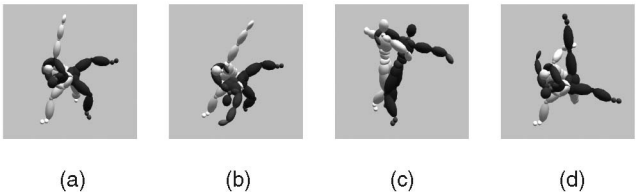


Fig. 19. The representative postures of the Latin dance motions (50, 51, 52, 57).

It can be observed that in the similarity matrix based on the topology, the distances between postures in the same group of actions are small. On the other hand, that does not necessarily apply to the results based on the Euclidean distance. For example, when using the topological distances, it can be observed that all the motions in the category of Full Nelson holds, back holds, and firefighter carries are evaluated as similar, while there are significant variations when using Euclidean distances.

In some categories, the topological relationships between the characters are different. These postures are 20, 21, and 22 in backbreakers, 25 in octopus holds, 28-32 in one person carries, and 36 in piggy back carries. Therefore, their topological distances from the other motions in the same category are relatively large. However, in some cases, the Euclidean distance between such postures with other postures in the same category are small because the positions of the joints are similar. This happens at posture 26 in octopus hold; although the Euclidean distance with the other postures are small, its topological relationship is actually different.

The dancing postures (40-49) are composed of three groups. In the first group (40-42, Fig. 18a), the white character is holding the torso of the black character and the black character is holding the neck of the white character. These postures are topologically equivalent to those of the walk assisting motion (37-39). In the second group (43-47, Fig. 18b), the white character is holding the torso of the other and the third group of postures is same as that of the first group, but the role of the two characters are switched (43-47, Fig. 18c). The proposed method can differentiate the postures correctly, as shown in Fig. 17a.

In all the Latin dance postures (50-58), the left arm of the white character is tangled with the torso of the black character. In addition to that, the black character's two arms (50, Fig. 19a), left arm (51, Fig. 19b) or right arm (52-58, Fig. 19c), are tangled with the neck of the white character. In posture 57-58, there is an additional tangle between the white character's left arm and the black character's right leg (Fig. 19d). Although the postures in the same group are topologically equivalent, they are kinematically dissimilar. It is difficult to classify them in the kinematic framework, as shown in Fig. 17b. Since the white character's right arm is tangled with the torso of the black character and the left arm of the black character is tangled with the neck of the white character in most of the dancing and Latin dance postures, the topological similarity of these postures are large, and as a result, a large gray area exists in the similarity matrix in Fig. 17a. Such topological similarity cannot be detected by Euclidean distance, and there is no consistency in the corresponding region of the similarity matrix shown in Fig. 17b.

In the similarity matrix computed by Euclidean distances, it can be observed that the variance of the similarity is large whether the postures are in the same or different groups. The large variance within the same group implies that the similarity is largely dependent on the kinematics and can result in inconsistencies. There is a lot of risk that it treats semantically different postures as similar and semantically similar postures as different. On the other hand, in the similarity matrix computed by topology distance, the variance is low. In some cases, the topology distance returns false positive results. For example, a gray area exists in 50-57 $\times$ 38-47 of the topology distance-based matrix. This is simply because these groups are topologically similar. The low variance suggests the consistency inside the same group and is not affected by the kinematics of the postures.

4.2 Content-Based Retrieval

Next, an experiment of content-based retrieval was done. The example postures were given as queries and the results together with the numerical similarity of each pair were computed and returned to the user.

Some of the results are shown in Figs. 2 and 20. The similarities of the postures are computed by (6). It can be observed that postures with similar topological relationships are at the top of the lists.

4.3 Creating Animations by Motion Graphs

Third, Motion Graphs [2], [3], [4] were created based on two sets of posture/motion data—one set on wrestling (*Group A*, Fig. 21) and the other on dancing (*Group B*, Fig. 22)—and we evaluated the animations created based on these Motion Graphs.

In Motion Graphs, the nodes represent postures and edges represent transition motions between the postures. They can be produced by comparing the distance between every posture of the captured motion data and connecting nodes by edges whose distances are below a given threshold. Here we created small-scale Motion Graphs using the postures in Section 4.1 and some additional short motion clips. We calculated the distance between all the postures and also the initial/final postures of the short motion clips, and connected the postures by edges if the distance was smaller than a threshold.

Three different methods were used to create the graphs and the results were compared. Firstly, we used the point-cloud distance metric in [4] to compose the graph. Secondly, we used the topological distance of (7) explained in Section 3.5 to compose the graph. This is because we want to find out whether the two postures can be blended or not

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

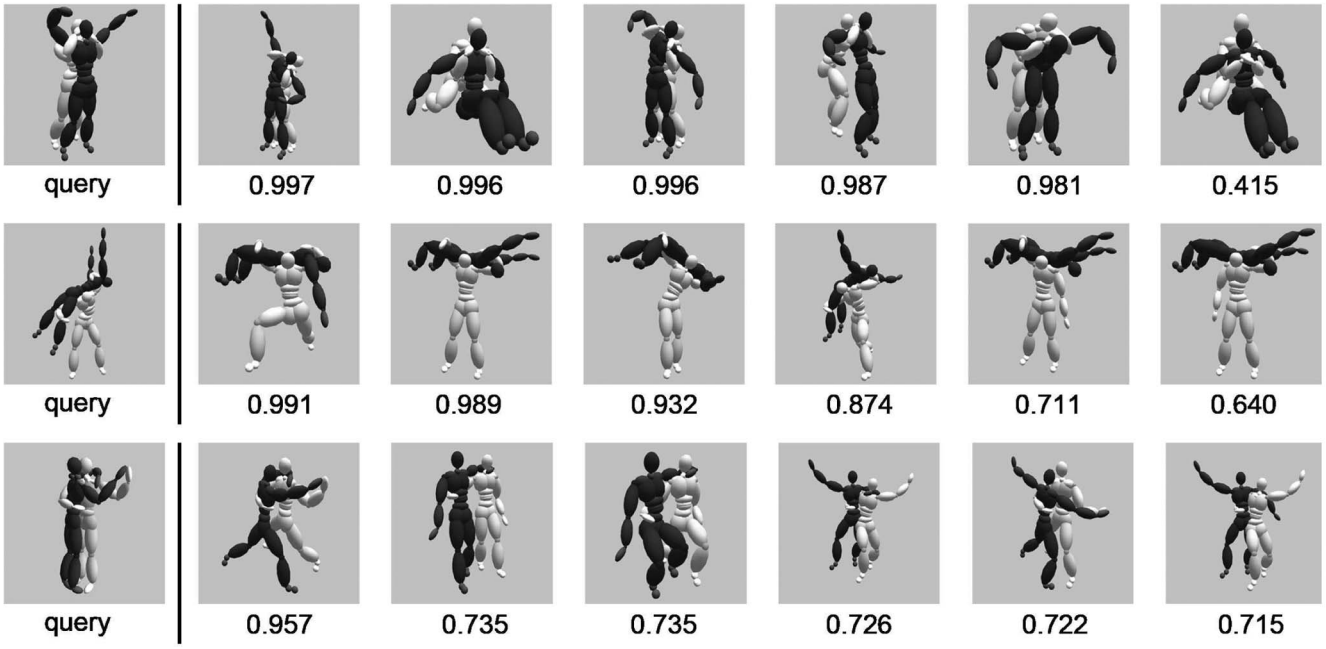


Fig. 20. Results of content-based retrieval based on the topological relationships. Despite the large variation of the postures, the pairs of postures with similar topological relationships return high scores.

rather than finding out the overall topological similarity of the two posture pairs. We had tightened the threshold to 0.45, which is slightly below 0.5 to reduce the risk of interpenetrations of the segments. Finally, we used a 2-pass method which first filters out postures whose topological relationships are too different and then compares the closeness based on the point-cloud metric.

The statistics of the Motion Graphs created based on the postures/motions of Groups A and B are shown in Tables 1 and 2, respectively. In the tables, *invalid edges* refer to the edges (transition motions) causing the bodies to intersect/penetrate each other. By using the point-cloud metric alone, a lot of invalid edges were found as many postures within each group are numerically similar, although their topological relationships are different. These apply to postures shown in Figs. 21b and 21d, for example.

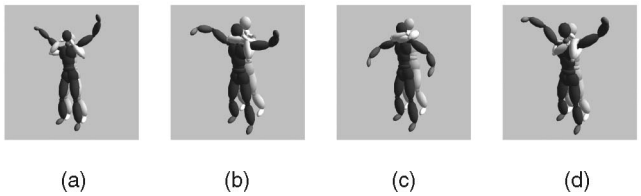


Fig. 21. The Full Nelson hold and Rear Chokehold postures/motions in Group A.

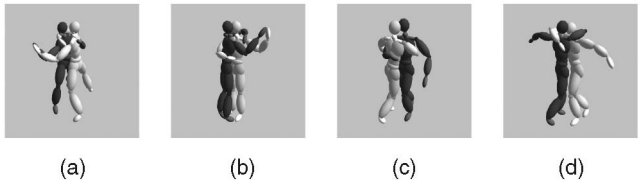


Fig. 22. The dancing postures/motions in Group B.

By using the topological distance alone, the number of invalid edges is reduced significantly. Since only topological distance is compared, some of the edges are connecting two numerically dissimilar postures. We found that most of these edges are valid edges. However, blending numerically dissimilar postures is not preferred in Motion Graphs since discontinued motions will be created. The risk of intersecting/penetrating will also rise as the bodies need to move a lot when interpolating the postures.

By using the 2-pass method, we only connected the postures whose point-cloud and topological distances are both small; as a result, no invalid edges are found. This implies that taking into account the topological distance between the postures is very useful in motion blending and concatenation, especially when the motion database contains a lot of motions whose topology is different.

The resulting motions created from the Motion Graphs composed using the point cloud metric and the 2-pass method are included in the attachment video, which can be

TABLE 1
Statistics of the Motion Graph Based on Wrestling Postures/Motions

	Euclidean	Topological	2-pass
# of edges	98	110	28
# of invalid edges	50	8	0

TABLE 2
Statistics of the Motion Graph Based on the Dancing Postures/Motions

	Euclidean	Topological	2-pass
# of edges	120	110	48
# of invalid edges	47	18	0

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

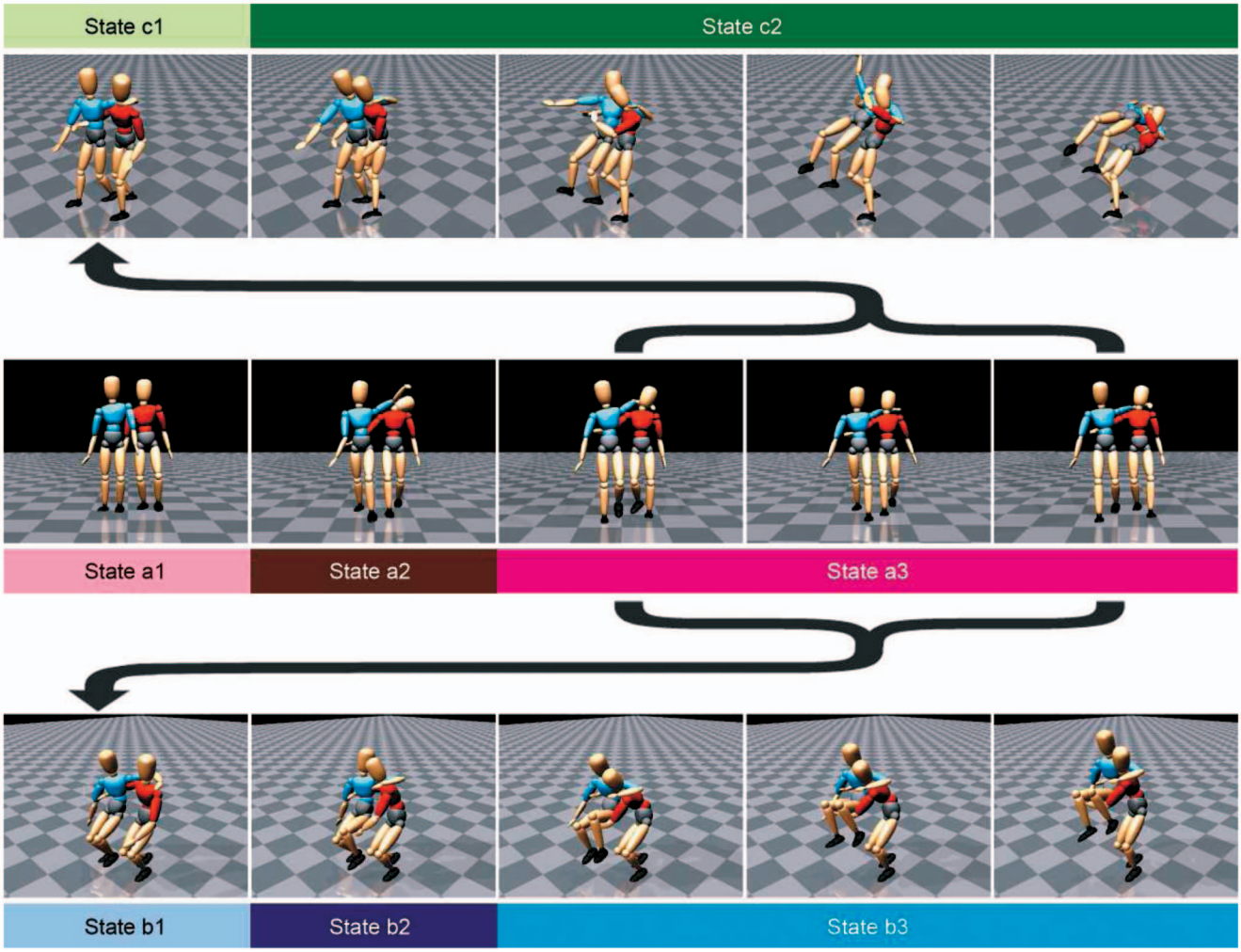


Fig. 23. The matching stages of the assist-walking motion, the backdrop motion, and the lifting motion.

found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2008.199.

4.4 Creating Animations by Concatenating Motion Clips

Finally, we used topological distance as a measure to evaluate whether motion clips can be concatenated or not. Three motion clips were prepared:

1. a person giving a shoulder to assist the walk of another (Fig. 23a);
2. a person doing a one-person arm carry to another (Fig. 23b);
3. a person conducting a back drop to another (Fig. 23c).

First, we computed the $TangleLists$ of the two bodies at every frame of the motion clips. Based on the topological relationship, the walking motion and the carrying motion are divided into three stages, and the backdrop motion is divided into two stages. By comparing the $TangleLists$ of every frame between the motions, it was found that the topological relationship at the third stage of the walking motion and at the first stage of the carry and backdrop motion are the same. We have further found the best frame to concatenate the walking motion and the carry motion, and the walking motion and the backdrop motion. By comparing the Euclidean distance of the joints, the smooth transitions from the walking motion to the carry motion and the walking motion to the backdrop motion were achieved. The readers are referred to the attachment video to see the resulting motions.

5 Discussions and Conclusions

In this paper, we have proposed a new method to index postures of two characters closely interacting with each other. The method is based on the theory of rational tangles, and it is shown that we can categorize various postures of two characters tangled with each other. We have also shown that a base line method using low-level attributes such as the position of the joints can suffer from categorizing such postures.

We have limited the tangles made between the segments to rational tangles. There are also different categories of tangles called self-knotted tangles and prime tangles. Such tangles have more complex structures and there is still no invariant known for them.

Regarding prime tangles, usually there are other rational tangles sharing the subtangles with them. Our system then encodes such rational tangles and uses them to distinguish the postures. As a result, by simply excluding the prime

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

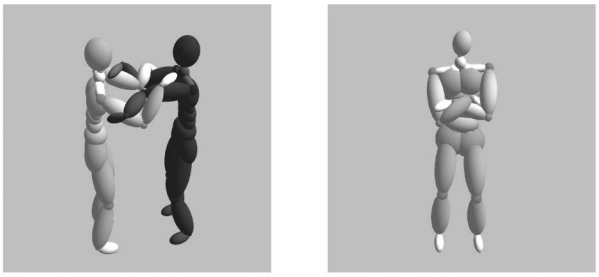


Fig. 24. Postures with: (a) prime tangle (b) self tangle.

tangles from considerations, we can compute the similarities of the pairs of postures by using the *TangleLists* of the rational tangles. Such an example can be found in Fig. 24a. The paths connecting the hands are forming a prime tangle equivalent to that in Fig. 4c. Although our method cannot index this tangle, other paths such as hand-foot get indexed as rational tangles. By comparing such paths, the system can successfully index this posture pair.

Regarding self-knotted tangles, because the human body is composed of a limited number of rigid segments, there are not so many self-knotted tangles that can be composed. One example of such posture is to hold the arms, as shown in Fig. 24b. As the main idea of this paper is to index the relationship of multiple characters, we did not provide solutions for such postures. One way to handle them is to compute the tangles made within the body by checking the GLI matrix of paths within the body. Actually, if we generate a GLI matrix of the left and right arms in the posture shown in Fig. 24b, a rational tangle of a single twist will be detected.

In the experimental results, we have shown several examples concatenating different motion clips by using Motion Graphs [2], [3], [4]. By comparing different distance metrics, the results clearly showed that using topological distance as a measure can reduce the number of collisions or penetrations in the blended motions. However, there are problems when interpolating topologically similar but kinematically dissimilar postures. For example, the location of the supporting feet can be quite different, which makes the resulting motions discontinuous and unnatural. By combining the topological and Euclidean distance metrics, the blended motions are free from collisions or penetrations, while the visual quality (in terms of continuity of the motions) is comparable to those created by conventional Motion Graphs. Taking into account the topological distance helps to generate collision-free motions automatically, especially when the motion database contains motions in which multiple characters closely interact with each other.

We mainly conducted experiments at the posture level instead of the motion level. It is straightforward to extend this concept to the motion level where the topological relationships change over time. As explained in Section 4.4, we can segment the motion at the postures where the topological relationship changes, and index or retrieve the motions using a sequence of topological relationships.

We proposed to encode the tangles made between the global paths connecting the end effectors; another approach to encode the tangles is to compute the local GLI between shorter paths such as those made by the limbs. Such an approach might be more efficient as we will only need to encode the local area where the tangles are composed. However, a drawback is that another approach to estimate the similarities of postures which are composed of different segments will be required. For example, the two postures in Fig. 6 are composed of different segments, although they should be considered similar. As our method is based on the topological relationship of global paths, such postures are treated as equivalent. We can further compare the details by comparing the kinematical difference.

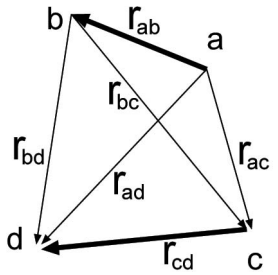


Fig. 25. The tetrahedron composed by two line segments a-b and c-d.

As a future work, one interesting approach is to use a hierarchical method in which the global similarities are first evaluated by global tangles and then the further details are compared by local tangles. Another interesting topic is to explore methods to interpolate postures based on the topology rather than interpolating the generalized coordinates.

APPENDIX: GLI OF LINE SEGMENTS

In this research, we assume that the two strands are represented by a chain of line segments. Suppose there are two serial chains C_1, C_2 composed of n_1 and n_2 line segments. The GLI of the two chains C_1, C_2 can be computed by

$$GLI(C_1, C_2) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} T_{i,j}, \tag{8}$$

where $T_{i,j}$ is the Gauss Linking Integrals of two line segments i and j. $T_{i,j}$ can be computed as follows [27]: Suppose points a, b and c, d are the end points of segments i and j, respectively. Let us define the vectors connecting a-b, a-c, a-d, b-c, b-d, and c-d by $r_{ab}, r_{ac}, r_{ad}, r_{bc}, r_{bd}, r_{cd}$, respectively (Fig. 25). Using these vectors, the normal vectors of the tetrahedron made by these four points can be calculated by

$$n_a = \frac{r_{ac} \times r_{ad}}{\|r_{ac} \times r_{ad}\|}, \quad n_b = \frac{r_{ad} \times r_{bd}}{\|r_{ad} \times r_{bd}\|},$$
$$n_c = \frac{r_{bd} \times r_{bc}}{\|r_{bd} \times r_{bc}\|}, \quad n_d = \frac{r_{bc} \times r_{ac}}{\|r_{bc} \times r_{ac}\|}.$$

Finally, $T_{i,j}$ is calculated by

$$T_{i,j} = \arcsin(n_a n_b) + \arcsin(n_b n_c) + \arcsin(n_c n_d) + \arcsin(n_d n_a).$$

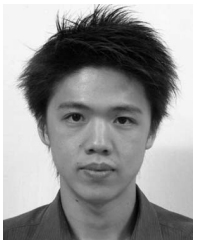
Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Paul Turner of Heriot-Watt Univeristy for the advice of Gauss Linking Integral. They also thank the reviewers for the helpful comments. This project was partly supported by a CERG grant from the Research Grant Council of Hong Kong (Ref:CityU1149/05) and the Wolfson Microelectronics Scholarship.

REFERENCES

[1] J. Conway, *An Enumeration of Knots and Links.* Pergamon Press, 1969.
[2] O. Arikan and D. Forsyth, "Motion Generation from Examples," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 483-490, 2002.
[3] 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 Trans. Graphics,* vol. 21, no. 3, pp. 491-500, 2002.
[4] L. Kovar, M. Gleicher, and F. Pighin, "Motion Graphs," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 473-482, 2002.
[5] O. Arikan, D.A. Forsyth, and J. O'Brien, "Motion Synthesis from Annotations," *ACM Trans. Graphics (Proc. ACM SIGGRAPH '03),* vol. 33, no. 3, pp. 402-408, 2003.
[6] F. Liu, Y. Zhuang, F. Wu, and Y. Pan, "3D Motion Retrieval with Motion Index Tree," *Computer Vision and Image Understanding,* vol. 92, no. 2-3, pp. 265-284, 2003.
[7] L. Kovar and M. Gleicher, "Flexible Automatic Motion Blending with Registration Curves," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation,* pp. 214-224, 2003.
[8] L. Kovar and M. Gleicher, "Automated Extraction and Parameterization of Motions in Large Data Sets," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 559-568, 2004.
[9] E.J. Keogh, T. Palpanas, V.B. Zordan, D. Gunopulos, and M. Cardle, "Indexing Large Human-Motion Databases," *Proc. 30th VLDB Conf.,* pp. 780-791, 2004.
[10] M. Muller, T. Roder, and M. Clausen, "Efficient Content-Based Retrieval of Motion Capture Data," *ACM Trans. Graphics,* vol. 24, no. 3, pp. 677-685, 2005.
[11] M. Muller and T. Roder, "Motion Templates for Automatic Classification and Retrieval of Motion Capture Data," *Proc. 2006 ACM SIGGRAPH/Eurographics Symp. Computer Animation,* pp. 137-146, 2006.
[12] Y. Shinagawa and T.L. Kunii, "The Homotopy Model: A Generalized Model for Smooth Surface Generation from Cross Sectional Data," *Visual Computer,* vol. 7, nos. 2-3, pp. 72-86, 1991.
[13] Y. Shinagawa, Y.L. Kergosien, and T.L. Kunii, "Surface Coding Based on Morse Theory," *IEEE Computer Graphics and Applications,* vol. 11, no. 5, pp. 66-78, 1991.
[14] Y. Shinagawa and T.L. Kunii, "Constructing a Reeb Graph Automatically from Cross Sections," *IEEE Computer Graphics and Applications,* vol. 11, no. 6, pp. 44-51, 1991.
[15] S. Takahashi, Y. Shinagawa, and T. Kunii, "A Feature-Based Approach for Smooth Surfaces," *Proc. ACM Symp. Solid Modeling,* pp. 97-110, 1997.
[16] Y. Shinagawa and T.L. Kunii, "Unconstrained Automatic Image Matching Using Multiresolutional Critical-Point Filters," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 9, pp. 994-1010, Sept. 1998.
[17] I. Fujishiro, Y. Takeshima, T. Azuma, and S. Takahashi, "Volume Data Mining Using 3D Field Topology Analysis," *IEEE Computer Graphics and Applications,* vol. 20, no. 5, pp. 46-51, Sept./Oct. 2000.
[18] M. Hilaga, Y. Shinagawa, T. Komura, and T.L. Kunii, "Topology Matching for Full Automatic Similarity Estimation of 3D Shapes," *Proc. ACM SIGGRAPH '01,* pp. 203-212, 2001.
[19] G. Reeb, "Sur Les Points Singuliers d'fune Forme de Pfaff Completement Integrable ou d'fune Fonction Numerique [on the Singular Points of a Completely Integrable Pfaff form or of a Numerical Function]," *Comptes Randus Acad. Sciences Paris,* vol. 222, pp. 847-849, 1946.
[20] D. DeCarlo and J. Gallier, "Topological Evoloution of Surfaces," *Graphics Interface,* pp. 194-203, 1996.
[21] S. Takahashi, Y. Kokojima, and R. Ohbuchi, "Explicit Control of Topological Transitions in Morphing Shapes of 3D Meshes," *Proc. Ninth Pacific Conf. Computer Graphics and Applications (PG '01),* p. 0070, 2001.
[22] C.C. Adams, *The Knot Book.* W.H. Freeman and Co., 1994.
[23] M. Vazquez and D.W. Sumners, "Tangle Analysis of Gin Site-Specific Recombination," *Math. Proc. Cambridge Philosophical Soc.,* vol. 136, pp. 565-582, 2004.
[24] E.S.L. Ho and T. Komura, "Wrestle Alone: Creating Tangled Motions of Multiple Avatars from Individually Captured Motions," *Proc. Pacific Graphics 2007,* pp. 427-430, 2007.
[25] E.S.L. Ho and T. Komura, "Planning Tangling Motions for Humanoids," *Proc. Humanoids 2007,* 2007.
[26] M. Levitt, "Protein Folding by Restrained Energy Minimization and Molecular Dynamics," *J. Molecular Biology,* vol. 170, pp. 723-764, 1983.
[27] K. Klenin and J. Langowski, "Computation of Writhe in Modeling of Supercoiled DNA," *Biopolymers,* vol. 54, pp. 307-317, 2000.



Edmond S.L. Ho received the BSc degree (first-class honors) in computer science from the Hong Kong Baptist University in 2003 and the MPhil degree from the City University of Hong Kong in 2006. He is currently working toward the PhD degree in the School of Informatics, University of Edinburgh. His research interests include physically based animation, and human motions analysis and synthesis.

Taku Komura received the BSc, MSc, and PhD degrees in information science from the University of Tokyo in 1995, 1997, and 2000, respectively. He is currently a lecturer in the School of Informatics at the University of Edinburgh. His research interests include human motion analysis and synthesis, topology-based retrieval and editing, physically based animation, and real-time computer graphics.

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

Authorized licensed use limited to: IEEE Xplore. Downloaded on April 9, 2009 at 05:00 from IEEE Xplore. Restrictions apply.