

Example-Based Human Motion Extrapolation and Motion Repairing Using Contour Manifold

Nick C. Tang, Chiou-Ting Hsu, *Member, IEEE*, Ming-Fang Weng, Tsung-Yi Lin, and Hong-Yuan Mark Liao, *Fellow, IEEE*

Abstract—We propose a human motion extrapolation algorithm that synthesizes new motions of a human object in a still image from a given reference motion sequence. The algorithm is implemented in two major steps: *contour manifold construction* and *object motion synthesis*. Contour manifold construction searches for low-dimensional manifolds that represent the temporal-domain deformation of the reference motion sequence. Since the derived manifolds capture the motion information of the reference sequence, the representation is more robust to variations in shape and size. With this compact representation, we can easily modify and manipulate human motions through interpolation or extrapolation in the contour manifold space. In the object motion synthesis step, the proposed algorithm generates a sequence of new shapes of the input human object in the contour manifold space and then renders the textures of those shapes to synthesize a new motion sequence. We demonstrate the efficacy of the algorithm on different types of practical applications, namely, motion extrapolation and motion repair.

Index Terms—Contour manifold, motion synthesis, motion transfer.

I. INTRODUCTION

IN recent years, human motion synthesis has become one of the most popular research fields because of its potential application in motion creation, motion editing, and recovery operations [1]–[3]. As a result, many algorithms have been developed, e.g., [4]–[9] to create or edit human motion sequences. The objective is to synthesize visually pleasing human motion sequences with realistic motions and appearance. Conventional human motion synthesis algorithms can be divided into two categories: methods based on a model-driven architecture (model-

driven methods) [5], [8], [9] and methods based on a data-driven architecture (data-driven methods) [4], [6], [7]. Model-driven methods provide an interactive system to help animators specify the pose of a character. Then, constrained optimization techniques can be used to compute the details of physically valid motions for the character. In [5], Fang and Pollard used a validity constraint optimization approach that iteratively adjusts a synthesized motion to satisfy the animator’s requirement for realistic motions. Liu and Popovic [8] proposed a system that synthesizes realistic character motions by regulating a small set of linear and angular momentum constraints; and Safonova *et al.* [9] utilized a motion capture database to find a low-dimensional space that captures the characteristics of a desired motion and then solved the optimization problem in that low-dimensional space. Although model-driven algorithms guarantee that the synthesized motions will be as realistic as possible, they are not practical for real-world applications due to the need for a complex motion control interface and a massive amount of user intervention.

Data-driven algorithms reduce the amount of user intervention needed to synthesize visually pleasing human motions. Specifically, they use reference motions as a guide to synthesize the desirable motions of a human object. In [7], Kovar *et al.* proposed an example-based algorithm called a motion graph that synthesizes new motions by using examples of motion data with 3D models. Cheung *et al.* [4] developed a computer vision-based system to synthesize new motions of a human object. First, they built detailed kinematic models of a human object to capture the motion data of each part of the human body in the input motion sequence. Then, they used an image-based rendering algorithm to render the captured motion of another human object. In [6], Gleigher presented a technique that retargets motions from one character to another by adapting the motion of one articulated figure to another figure that has an identical structure. Most data-driven algorithms try to use the motion data obtained from reference motions to synthesize desirable motions directly with less user intervention. However, the inaccessibility of motion capture devices and the complex settings of the experimental environment make these algorithms impractical for ordinary users.

To help users synthesize new human motions from input video sequences, Kobayashi *et al.* [10] developed an interactive framework that uses shape matching techniques to extract video objects and build a 2D motion graph. The user can then synthesize a new motion sequence of the extracted video object by dragging the detected feature points to search for available object images in the 2D motion graph. However,

Manuscript received December 19, 2012; revised May 14, 2013, July 09, 2013; accepted July 10, 2013. Date of publication September 27, 2013; date of current version December 12, 2013. This work was supported in part by the Taiwan E-learning and Digital Archives Program (TELDAP) and the National Science Council of Taiwan under NSC Grant NSC99-2631-H-001-020. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ebroul Izquierdo.

N. C. Tang and M.-F. Weng are with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan (e-mail: nickctang@iis.sinica.edu.tw; mfueng@iis.sinica.edu.tw).

C.-T. Hsu is with the Department of Computer Science, National Tsing Hua University, Hsinchu 300, Taiwan (e-mail: cthsu@cs.nthu.edu.tw).

T.-Y. Lin is with the Electrical and Computer Engineering Department, University of California at San Diego, La Jolla, CA 92093 USA. (e-mail: tsl008@ucsd.edu)

H.-Y. M. Liao is with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan, and also with the National Chiao Tung University, Hsinchu 300, Taiwan. (e-mail: liao@iis.sinica.edu.tw).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TMM.2013.2283844

the algorithm can only generate new motion sequences for the original human object, and an ordinary user cannot apply the constructed motion graph to other human objects. Xu *et al.* [11] proposed a method to rearrange available motion templates with few user assistance to form a new motion sequence by minimizing the shape context distance in the proposed motion cycle reconstruction algorithm. We believe that data-driven algorithms are more effective than model-driven algorithms in synthesizing human motions with the minimum cost and the least amount of user intervention, but some issues need to be addressed to tackle the above mentioned constraints.

Understanding human motions is one of the most important issues in transferring motions successfully from a reference motion sequence to another human object. Human motion is difficult to synthesize because of its non-rigid nature and high dimensionality. Several algorithms have been proposed to compute the underlying structure of motion data for synthesizing human motions, e.g., [9], [12]–[19]. In [12], Hsieh *et al.* proposed a skeleton-based posture recognition method that converts each sequence of human movements into a posture sequence to analyze the human object's behavior. Dimensionality reduction techniques are often used to simplify a large collection of motion data. It is then possible to compute the data's intrinsic structure and construct a low-dimensional representation. Safonova *et al.* [9] used Principal Component Analysis (PCA) to reduce the dimensionality of human motions and showed that the articulated motion could be optimized effectively in the lower-dimensional space by satisfying user-specified motion constraints. Shin and Lee [13] used a multi-dimensional scaling algorithm to project human motions on to a two-dimensional space and designed an interface that allows the user to sketch a curve on the constructed two-dimensional space and choose the desired motions. Chai and Hodgins [14] proposed an animation system that uses low-dimensional control signals obtained from a few markers on a human body to control the motion of animated characters. In [15], Elgammal *et al.* introduced a technique that learns low-dimensional activity manifolds from silhouettes and then conducts analysis and tracking in the low-dimensional manifold. Subsequently, they proposed an efficient human gait tracking algorithm [16] with an explicit manifold modeling for body configuration to model shape variations in human motion sequences. Liang *et al.* [17] proposed an approach that learns low-dimensional posture manifolds and performs human motion analysis in the low-dimensional manifold space. In [18], Ding *et al.* proposed a method which uses a set of low-dimensional points computed by Locally Linear Embedding to represent the high dimensional motion frames in original images and then repaired damaged motion sequences by interpolating low dimensional points corresponding to missing or occluded motion frames.

In a previous work [19], we used the shape distance between any two motions as input and then exploited isometric feature mapping to construct a low-dimensional structure of a human motion database. When a user inputs specific starting and ending poses, the system synthesizes the desirable interior motions of the user-specific 2D human object. However, in our experiments, we found that constructing low-dimensional embedding for the whole human motion database is time con-

suming. In fact, the major limitations of our previous method and those proposed in [14] are the time complexity and the lack of flexibility. In other words, the methods can only produce human motion sequences based on the postures collected in their respective motion databases.

The above limitations motivate us to develop a low-cost, easy-to-use system that will help ordinary users create, edit and recover the motions of a human object. In this paper, we propose a human motion extrapolation algorithm that can synthesize new motions of a human object in a single image from a given reference motion sequence. Our goal is to capture and transfer the motion from the reference sequence to the input human object to help ordinary user extrapolate motion sequences they would like to have, even though their shape and size may be very different. Basically, the method we proposed is a data-driven based approach. We use a reference motion sequence as a guide to synthesize the desirable motions of a human object. Fig. 1 shows the proposed framework, which consists of two major phases: *contour manifold construction* and *object motion synthesis*. The first phase, contour manifold construction, uses local features to describe the articulated motion of a human body, and then constructs contour manifolds to represent the human motion data in the manifold embedding space. Using the proposed contour manifold to represent human motion ensures that our method is robust to variations in the size, shape and kinematics of human bodies. The second phase, object motion synthesis, first generates a sequence of new shapes of the input human object in the contour manifold space. It then renders the textures of the derived shapes to synthesize the new motion sequence.

From a technical perspective, the contribution of this work can be summarized in the following. Basically, our work contributes to the society with two things. First, the proposed contour manifold provides a compact representation of high dimensional human motions, and it makes human motions analyzable and maneuverable. Since a human motion is composed of a number of articulated motions, it is hard to represent in the spatial domain. Besides, a same motion done by different individuals may be very different. For example, different people have different sizes and the speed they perform the motion may be quite different. By transforming the spatial domain into the contour manifold domain, many problems that are originally existing in the spatial domain can now be solved. Second, the constructed contour manifolds provide a systematic way to manipulate human motion efficiently in a low dimensional space. The proposed algorithm can easily produce visually pleasing motion sequences of an input object, with any number of motions and for different purposes, because it exploits the constructed contour manifold. In addition, the proposed method is developed in a general manner. One can apply it to repair damaged human subject sequences in digitized vintage films (as shown in Fig. 2) for maintaining the original values. Another potential application is to generate new video sequence by editing or creating new video contents of video objects for movie industry.

The remainder of this paper is organized as follows. Section II presents the proposed human motion extrapolation algorithm which includes contour manifold construction for human motion and object motion synthesis. Section III details the experiment results; and concluding remarks are drawn in Section IV.

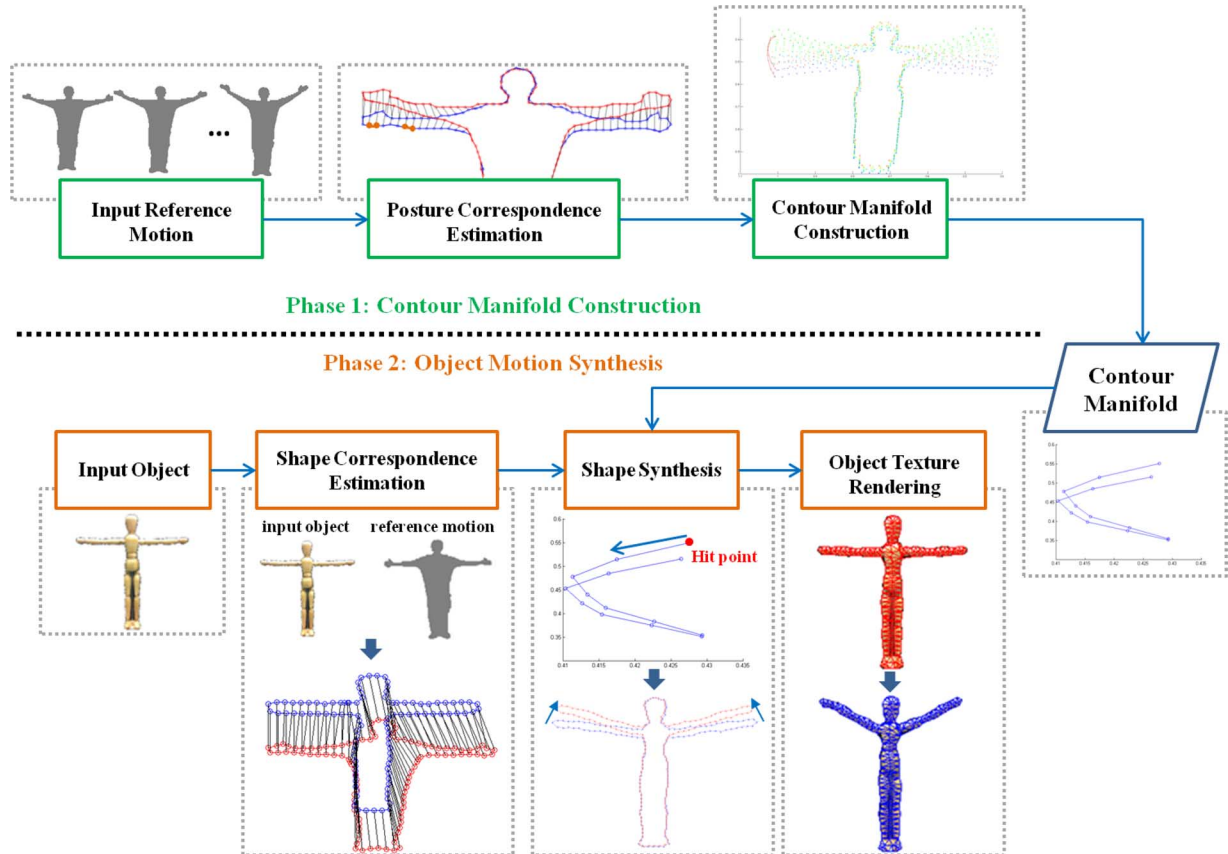


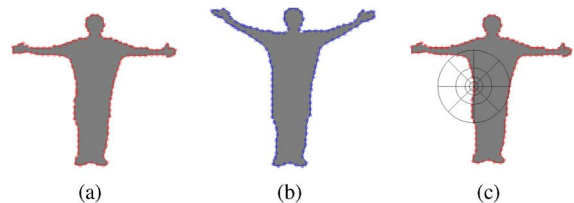
Fig. 1. Proposed framework.



Fig. 2. An example of damaged human subject repairing in a digitized vintage film.

II. HUMAN MOTION EXTRAPOLATION

The synthesis and manipulation of human motion is a very challenging task that usually involves high-dimensional data. It also requires consistency in the spatiotemporal domain to ensure the smoothness of a realistic human motion. In this paper, we propose a novel motion extrapolation algorithm that automatically synthesizes a sequence of human motions from a single still image \mathcal{O} of one person and a given reference motion sequence $M_t (t = 1, 2, \dots, m)$, which can be performed by another person. The representation of human motions usually relies on sets of densely sampled contour points. As shown in Fig. 3(a), a set of contour points is uniformly sampled on the shape of M_t . However, this representation would require a massive amount of high-dimensional data to describe a sequence

Fig. 3. Example of a shape context descriptor: (a) and (b) show the uniformly sampled contour points $\{\mathbf{p}_i^t\}$ and $\{\mathbf{p}_{i+1}^t\}$ on the shape of motion M_t and M_{t+1} ; and (c) a diagram of the log-polar space.

of human motions, whose intrinsic structure tends to lie on a lower-dimensional manifold. To manipulate human motions effectively, we propose a representation scheme called *contour manifold*, which characterizes the time-evolving trajectories of each contour point along their corresponding manifolds. Then, we use the constructed contour manifolds as guides to synthesize new motions $S_t (t = 1, 2, \dots, m)$ of the input shape \mathcal{O} . We describe the proposed algorithm in detail in the following subsections.

A. Contour Manifold Construction

Given a motion sequence $M_t (t = 1, 2, \dots, m)$, we first uniformly sample a set of contour points on the silhouette of each single motion (or posture). Then, we estimate the correspondence of the contour points between two consecutive postures M_t and M_{t+1} to obtain the motion trajectory for each contour point. To construct the contour manifolds, we apply a linear

dimensionality reduction technique called Orthogonal Locality Preserving Projection (OLPP) [20], which projects the motion trajectories into lower-dimensional manifolds by preserving the local neighborhood information in the spatial and temporal domain. The advantage of using OLPP is that the continuity of each contour point on spatial and temporal domain will be preserved during the manifold construction process. The computational efficiency of the proposed method combined with the data reconstruction ability of OLPP ensures that it is suitable for practical applications.

1) *Posture Correspondence Estimation*: Establishing correspondences between different postures is not a trivial task, especially for non-rigid human motions. To resolve the problem, we propose using a cost-minimization method to initiate the correspondences and then apply an error-match correction mechanism to refine them further.

Two consecutive postures, M_t and M_{t+1} , are first uniformly sampled into sets of contour points \mathbf{p}_t^i and \mathbf{p}_{t+1}^i ($i = 1, \dots, n$), as shown in Fig. 3(a) and (b) respectively. Let $M_t = \{\mathbf{p}_t^i\}$, $i = 1, \dots, n$, where $\mathbf{p}_t^i = [p_{t,x}^i, p_{t,y}^i]^T$. The goal of posture correspondence is to determine a one-to-one matching ($\mathbf{p}_t^i, \mathbf{p}_{t+1}^{\pi(i)}$) by minimizing the total cost as follows:

$$H(\pi) = \sum_{i=1}^n \text{Cost} \left(\mathbf{p}_t^i, \mathbf{p}_{t+1}^{\pi(i)} \right), \quad (1)$$

where $\pi : \{1, 2, \dots, n\} \rightarrow \{1, 2, \dots, n\}$ is a permutation function and $\text{Cost}(\mathbf{p}_t^i, \mathbf{p}_{t+1}^{\pi(i)})$ indicates the cost of matching the contour point \mathbf{p}_t^i in M_t with $\mathbf{p}_{t+1}^{\pi(i)}$ in M_{t+1} .

In (1), an effective cost measurement $\text{Cost}(\mathbf{p}_t^i, \mathbf{p}_{t+1}^{\pi(i)})$ between a pair of matching points \mathbf{p}_t^i and $\mathbf{p}_{t+1}^{\pi(i)}$ need to be developed. To this end, we utilize the *shape context* [21] as the descriptor, which measures a coarse diagram in a log-polar space for each contour point. A shape-context based descriptor has good discriminative power and good robustness against deformations. Therefore, it can help obtain a good initial estimate for determining posture correspondences. The cost of a matching pair can be determined by the shape context as follows

$$\text{Cost} \left(\mathbf{p}_t^i, \mathbf{p}_{t+1}^{\pi(i)} \right) = \frac{1}{2} \sum_{k=1}^K \frac{\left[h_t^i(k) - h_{t+1}^{\pi(i)}(k) \right]^2}{h_t^i(k) + h_{t+1}^{\pi(i)}(k)}, \quad (2)$$

where $h_t^i(k)$ and $h_{t+1}^{\pi(i)}(k)$ denote the k -th bin of the normalized histogram in the log-polar space (as shown in Fig. 3(c)) around \mathbf{p}_t^i and $\mathbf{p}_{t+1}^{\pi(i)}$ respectively. Hence, the cost minimization defined in (1) is regarded as an assignment problem and solved by using a bipartite graph matching method [22]. However, as shown in Fig. 4(b), the estimated correspondences contain several error-matches because the cost function only considers the cost of matching two isolated points and ignores the global consistency.

To correct the error-matches in the initial estimate of π , we employ an error-match correction mechanism, which is implemented in two steps: *error-match detection* and *correspondence re-estimation*. In the first step, we assume that the correct correspondences will preserve a linear order of the contour points sampled from a posture to maintain the global

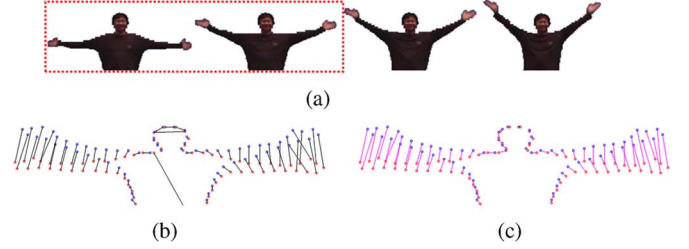


Fig. 4. Example of estimated correspondence between M_t and M_{t+1} . (a) The example of M_t and M_{t+1} . (b) The estimation done by bipartite correspondence matching π ; and (c) the corrected correspondences π' done by our method. Purple line segments shown in (c) represent the perfectly matched parts between the ground truth correspondences and the corrected correspondences.

consistency. Hence, we refer error-matches to those pairs that do not satisfy the following rule:

$$\text{if } i_1 < i_2 < \dots < i_n, \text{ then } \pi(i_1) < \pi(i_2) < \dots < \pi(i_n). \quad (3)$$

After determining all the error-matches in π , we re-estimate the correspondence between those contour points by minimizing the coordinate transformation error:

$$\pi' = \arg \min_{\pi} \left\{ \frac{1}{n} \sum_i^n \left\| \mathbf{p}_{t+1}^{\pi(i)} - F(\mathbf{p}_t^i) \right\|_2 \right\}, \quad (4)$$

where $F(\cdot)$ denotes the TPS (Thin Plate Spline) transform estimated by using the approach in [23]. Finally, we re-arrange the contour points in M_{t+1} according to the permutation π' . Fig. 4(a) shows an example of consecutive postures, M_t and M_{t+1} , which are bounded by red rectangle. To precisely evaluate the performance of the proposed correspondence estimation method, we manually connect the ground truth contour points and the estimated corresponding contour points. Fig. 4(b) shows the connections linked by the original bipartite correspondence matchings. Fig. 4(c), on the other hand, shows the correspondence matchings done by our proposed method. The average success rate of the original correspondence matching method was about 73% and the average success rate of our proposed method was about 90%. Therefore, we use the proposed method to perform correspondence estimation. Note that, to simplify the notation, we use $M_{t+1} = \{\mathbf{p}_{t+1}^i\}$, $i = 1, \dots, n$ to denote the permuted contour points $\{\mathbf{p}_{t+1}^{\pi'(i)}\}$, $t = 1, \dots, m - 1$ in the following sections.

2) *Contour Manifold Construction*: After determining the correspondences between consecutive postures, we can represent the human motion in terms of n motion trajectories $(\mathbf{p}_1^i, \mathbf{p}_2^i, \dots, \mathbf{p}_m^i)$, $i = 1, \dots, n$, as shown in Fig. 5(a). Although the trajectories provide a good representation of the reference motion sequence M_t ($t = 1, 2, \dots, m$), if the input shape \mathcal{O} is very different from that of the reference sequence, it may be difficult to transfer and synthesize a new motion sequence for \mathcal{O} . However, as the motion trajectory of each contour point lies on a lower-dimensional manifold embedded in the 3-dimensional space (x, y, t) , we should be able to construct a lower-dimensional manifold to characterize the non-rigid deformation of each contour point. In summary, to

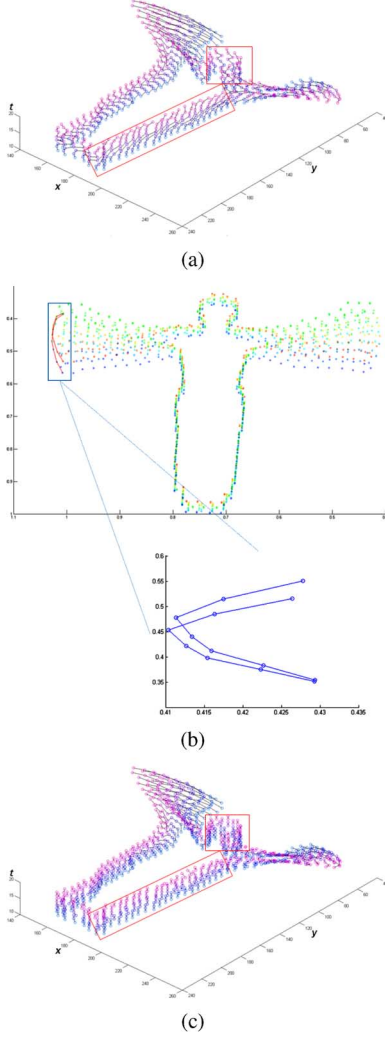


Fig. 5. Example of contour manifold construction: (a) the motion trajectory of each contour point in the original space; (b) the constructed OLPP embedding \mathbf{y}_t^i and the contour manifold of a contour point; (c) the back projected results of \mathbf{y}_t^i . One thing to be noted is that the areas bounded by red rectangles in (a) and (c) look different. It is obvious that the noises caused by the segmentation problem are handled during the manifold construction process.

better analyze and manage motion sequences, we utilize contour manifolds to find a compact representation that can capture the continuous motion of each contour point independent of the shape of the reference motion sequence.

Let $\tilde{\mathbf{p}}_t^i = [\mathbf{p}_t^i, t]^T = [p_{t,x}^i, p_{t,y}^i, t]^T \in \mathbb{R}^3$ be the 3-dimensional representation of the contour point \mathbf{p}_t^i and let $P = [\tilde{\mathbf{p}}_1^1, \dots, \tilde{\mathbf{p}}_1^n, \tilde{\mathbf{p}}_2^1, \dots, \tilde{\mathbf{p}}_2^n, \dots, \tilde{\mathbf{p}}_m^1, \dots, \tilde{\mathbf{p}}_m^n]^T \in \mathbb{R}^{3 \times mn}$ be the matrix of the input data, where n is the number of sampled contour points in each posture and m is the length of the input motion sequence. We apply OLPP [20] and use $\tilde{\mathbf{p}}_t^i$ as inputs to construct contour manifolds for each contour point. The objective is to preserve the continuity of each contour point along the temporal dimension. In addition, because spatially neighboring contour points usually have very similar motion trajectories, we include the spatial continuity in the contour manifold construction process. Next, we describe the steps of the construction process.

- 1) **PCA Projection:** First, we project the input motion data P of the $m \cdot n$ contour points into the PCA subspace

by removing the components with a zero eigenvalue. In this step, the transformation matrix of PCA is denoted by A_{PCA} .

- 2) **Constructing the Adjacency Graph:** Let G denote the graph with the $m \cdot n$ nodes, as represented in the columns of the matrix P . Two nodes are connected by an edge when they are neighbors in the spatial domain (i.e., $\tilde{\mathbf{p}}_t^i$ and $\tilde{\mathbf{p}}_t^{i+1}$) or in the temporal domain (i.e., $\tilde{\mathbf{p}}_t^i$ and $\tilde{\mathbf{p}}_{t\pm 1}^i$).
- 3) **Constructing the Weight Matrix:** After deriving the adjacency relationship in graph G , we compute the edge weight to construct a sparse symmetric weight matrix $W \in \mathbb{R}^{mn \times mn}$. The weight of the edge between two nodes is calculated as follows:

$$W_{ij} = \begin{cases} \exp\left(\frac{-\|\tilde{\mathbf{p}}_{t_i}^i - \tilde{\mathbf{p}}_{t_j}^j\|^2}{\kappa}\right), & \text{if } \tilde{\mathbf{p}}_{t_i}^i \text{ and } \tilde{\mathbf{p}}_{t_j}^j \text{ are connected} \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

The setting of the parameter κ can be referred to [24]. The weight matrix W of graph G characterizes the local structure of the contour points in both the spatial domain and the temporal domain.

- 4) **Computing the Orthogonal Basis Functions:** In this step, we define D as a diagonal matrix whose entries are the column sums of W , i.e., $D_{ii} = \sum_j W_{ji}$; and we let L be the Laplacian matrix defined by $L = D - W$. We select k orthogonal basis vectors $\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_k\}$ which correspond to the k largest eigenvalues and compute as follows:

- Compute \mathbf{a}_1 as the eigenvector of $(PDP^T)^{-1}PLP^T$ associated with the smallest eigenvalue.
- Compute \mathbf{a}_k as the eigenvector of

$$M^{(k)} = \left\{ I - (PDP^T)^{-1}A^{(k-1)}[B^{(k-1)}]^{-1}[A^{(k-1)}]^T \right\} \times (PDP^T)^{-1}PLP^T, \quad (6)$$

where

$$B^{(k-1)} = [A^{(k-1)}]^T (PDP^T)^T A^{(k-1)}, \quad (7)$$

$$A^{k-1} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{k-1}] \quad (8)$$

associated with the smallest eigenvalue of $M^{(k)}$.

- 5) **Computing OLPP Embedding:** Let $A_{OLPP} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_k] \in \mathbb{R}^{3 \times k}$. The k -dimensional OLPP embedding Y is computed as follows:

$$Y = A^T P, \quad (9)$$

where $A = A_{PCA}A_{OLPP}$ is the transformation matrix. Fig. 5(b) shows an example of the constructed contour manifolds in the OLPP embedding Y with $k = 2$.

Using the proposed manifold representation to represent human motion in a lower-dimensional space has two major benefits. First, the proposed contour manifolds provide a compact representation that describes how the body shape varies in a motion sequence. Because the computed contour manifolds characterize the non-rigid deformation of each local part of a human body, the representation is invariant to the scales of

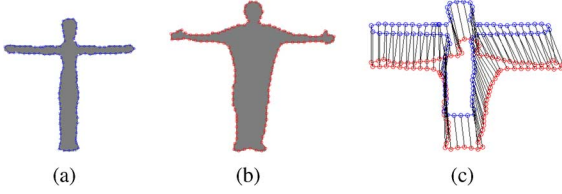


Fig. 6. Estimating the correspondence between the shape of the input object and the reference posture: (a) the shape of input object, (b) the reference posture, and (c) the estimated correspondence.

shapes and poses. Second, the contour manifolds provide a systematic way to manipulate human motion in a low dimensional space. By using the continuous bijective mapping between the lower-dimensional embedding and the original human motion representation, we can easily modify and manipulate human motion through interpolation or extrapolation on the contour manifolds.

B. Object Motion Synthesis

After constructing the contour manifolds of the reference motion sequences $M_t, t = 1, 2, \dots, m$, we can synthesize new motions of the input object \mathcal{O} . The proposed motion synthesis process involves two steps. The first step, *shape synthesis*, generates a sequence of new shapes of \mathcal{O} by deforming the latter's shape in the embedding space of contour manifolds. The second step, *object texture rendering*, implements two processes, *triangular mesh deformation* and *texture transfer*, which work together to render the corresponding textures of the synthesized shapes.

1) *Shape Synthesis*: To synthesize new shapes of \mathcal{O} , we must establish the correspondences between the shape of \mathcal{O} and the first reference posture M_1 so as to deform each contour segment of \mathcal{O} individually. Once again, we use the correspondence estimation method described in Subsection II.A to determine the correspondence π between \mathcal{O} and M_1 . Let $S_1 = \{\mathbf{q}_1^i\}, i = 1, 2, \dots, n$ denote the set of contour points of \mathcal{O} in the order determined by π , as shown in Fig. 6. The sets of uniformly sampled contour points of the shapes S_1 and M_1 are shown in Fig. 6(a) and (b) respectively; and Fig. 6(c) shows the estimated correspondence.

After obtaining the set of contour points $S_t = \{\mathbf{q}_t^i\}, i = 1, 2, \dots, n$ in the t -th frame, we synthesize a new shape S_{t+1} by deforming each contour segment of S_t in the constructed contour manifolds. Let $C_t^i = (\mathbf{q}_t^i, \mathbf{q}_t^{i+1})$ denote the contour segment between two contour points \mathbf{q}_t^i and \mathbf{q}_t^{i+1} in S_t . We calculate the coordinate of \mathbf{q}_t^i on S_{t+1} by mapping the contour embedding \mathbf{y}_t^i back to the input motion space as follows:

$$\tilde{\mathbf{q}}_{t+1}^i = A \cdot \mathbf{y}_t^i, \quad (10)$$

where A is the computed transformation matrix in (9), and $\mathbf{y}_t^i = A^T \tilde{\mathbf{q}}_t^i$ is the embedding of $\tilde{\mathbf{q}}_t^i = [q_{t,x}^i, q_{t,y}^i, t]^T$ in the i -th contour manifold. Fig. 5(c) shows an example of back projection result of \mathbf{y}_t^i . The coordinates \mathbf{q}_{t+1}^i of all the contour points in S_{t+1} are then extracted from $\tilde{\mathbf{q}}_{t+1}^i$ by $\mathbf{q}_{t+1}^i = [\tilde{q}_{t+1,x}^i, \tilde{q}_{t+1,y}^i]^T$. We also need to determine the coordinates of all the pixels $\mathbf{c}_{t+1}^i \in C_{t+1}^i$ located in the contour segment $C_{t+1}^i = (\mathbf{q}_{t+1}^i, \mathbf{q}_{t+1}^{i+1})$. Since we have the

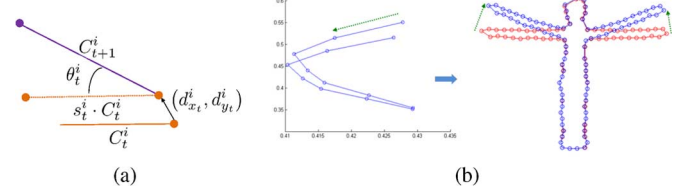


Fig. 7. An example of shape synthesis: (a) the rotation θ_t^i , scale s_t^i and translation $(d_{x_t}^i, d_{y_t}^i)$ of a contour segment C_t^i ; and (b) the shape S_t (in red) and the synthesized shape S_{t+1} (in blue).

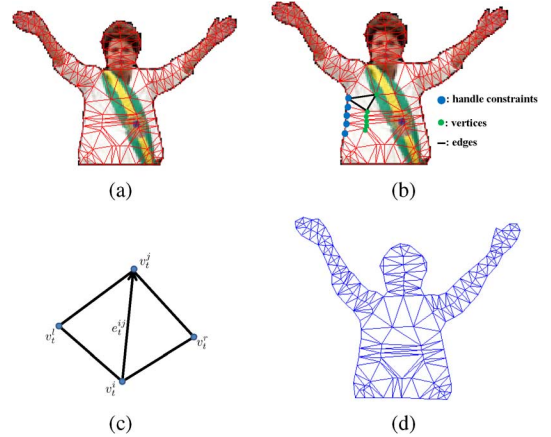


Fig. 8. Triangular mesh deformation: (a) an example of mesh triangulation; (b) the handle constraints, vertices and edges of the meshes; (c) the surrounding vertices of the edge e_t^{ij} ; (d) the deformed meshes.

corresponding contour segment $C_t^i = (\mathbf{q}_t^i, \mathbf{q}_t^{i+1})$ in S_t for each contour segment C_{t+1}^i in S_{t+1} , we perform affine transformation directly on \mathbf{c}_t^i to obtain \mathbf{c}_{t+1}^i . As illustrated in Fig. 7(a), the contour segment C_{t+1}^i is affine transformed from C_t^i with rotation angle θ_t^i , scaling s_t^i , and translation displacement $(d_{x_t}^i, d_{y_t}^i)$. That is,

$$\begin{bmatrix} \mathbf{c}_{t+1}^i \\ 1 \end{bmatrix} = D_t^i \begin{bmatrix} \mathbf{c}_t^i \\ 1 \end{bmatrix} = s_t^i \begin{bmatrix} \cos \theta_t^i & -\sin \theta_t^i & d_{x_t}^i \\ \sin \theta_t^i & \cos \theta_t^i & d_{y_t}^i \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{c}_t^i \\ 1 \end{bmatrix}. \quad (11)$$

Fig. 7(b) shows an example of S_t and the synthesized S_{t+1} . After the new shapes S_{t+1} of the input object \mathcal{O} corresponding to the $(t+1)$ -th reference posture M_{t+1} has been synthesized, we start to render the object's texture.

2) *Object Texture Rendering*: To render the texture of the synthesized shapes S_{t+1} from S_t for $t = 1, 2, \dots, m-1$, we first decompose S_t into a set of triangular meshes. Then, we exploit a triangular mesh deformation process to deform every triangular mesh from S_t to S_{t+1} . We describe the steps in detail below.

First, we take the contour points $\{\mathbf{q}_t^i\}$ on S_t as inputs and apply the constrained conforming Delaunay triangulation algorithm [25] to decompose S_t into triangular meshes. Let E_t and V_t denote, respectively, the set of directed edges and the set of vertices in S_t . Fig. 8(a) shows an example of mesh triangulation, and Fig. 7(b) shows the contour points $\{\mathbf{q}_t^i\}$, the derived vertices V_t and edges E_t of S_t .

Next, we use a modified version of an edge-based mesh deformation algorithm [26] to deform all the triangular meshes

in S_t by minimizing the distortion of the deformed edges. In this algorithm, we use the edges E_t as inputs and the contour points $\{\mathbf{q}_t^i\}$ and $\{\mathbf{q}_{t+1}^i\}$ on S_t and S_{t+1} as the given constraints (called ‘‘handles’’ in the deformation algorithm) to guide the deformation process. The goal is to find the vertex set V_{t+1} by minimizing the distortion between the directed edges under the handle constraint:

$$V_{t+1} = \arg \min_{\mathbf{v}_{t+1} \in V_{t+1}} \left\{ \sum_{(\mathbf{v}_t^i, \mathbf{v}_t^j) \in E_t} \left\| \left(\mathbf{v}_{t+1}^j - \mathbf{v}_{t+1}^i \right) - R_{ij} \left(\mathbf{v}_t^j - \mathbf{v}_t^i \right) \right\|^2 + \omega \sum_{\mathbf{q}_t^i \in S_t} \left\| \left(\mathbf{v}_{t+1}^i - \mathbf{q}_t^i \right) \right\|^2 \right\}, \quad (12)$$

where \mathbf{v}_t^i is the vertex coordinate of the triangular meshes on S_t , \mathbf{v}_{t+1}^i is the vertex coordinate of the triangular meshes on the synthesized shape S_{t+1} , and ω is a weight factor. The rotation matrix R_{ij} is used to convert the vertices around the edge (as shown in Fig. 8(c)) into new positions that are as close as possible in a least-square sense. It can be obtained by

$$R_{ij} = \arg \min_R \left\{ \sum_{\mathbf{v}_t \in N(\mathbf{v}_t^i, \mathbf{v}_t^j)} \left\| R\mathbf{v}_t - \mathbf{v}_{t+1} \right\|^2 \right\}, \quad (13)$$

where $N(\mathbf{v}_t^i, \mathbf{v}_t^j) = \{\mathbf{v}_t^i, \mathbf{v}_t^j, \mathbf{v}_t^l, \mathbf{v}_t^r\}$ is the set of vertices around the edge $(\mathbf{v}_t^i, \mathbf{v}_t^j)$, as shown in Fig. 8(c). Details of the derivation of the closed-form solution can be found in [26]. Since the contour points $\{\mathbf{q}_t^i\}$ and $\{\mathbf{q}_{t+1}^i\}$ were obtained by (10) in the *shape synthesis* step, we can now obtain all the vertices of the deformed triangular meshes in S_{t+1} automatically. Fig. 8(d) shows the deformed triangular meshes.

After obtaining the vertex set V_{t+1} of the deformed triangular meshes, we transfer the texture from the triangles at the t -th posture to the deformed triangles at the $(t+1)$ -th posture. Let $T_t = (\mathbf{v}_t^1, \mathbf{v}_t^2, \mathbf{v}_t^3)$ be a triangle at the t -th posture, and let $T_{t+1} = (\mathbf{v}_{t+1}^1, \mathbf{v}_{t+1}^2, \mathbf{v}_{t+1}^3)$ be its corresponding triangle at the $(t+1)$ -th posture. We convert each point \mathbf{r}_t inside the triangle T_t into the barycentric coordinate system by

$$\mathbf{r}_t = \lambda_1 \mathbf{v}_t^1 + \lambda_2 \mathbf{v}_t^2 + \lambda_3 \mathbf{v}_t^3, \quad (14)$$

where λ_1 , λ_2 , and λ_3 are the barycentric coordinates and are subject to the constraint $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

During the texture transfer operation, we need to determine the corresponding point \mathbf{r}_t for each \mathbf{r}_{t+1} , as shown in Fig. 9. Since all the vertices of the two triangles are given, we use linear interpolation to determine the point \mathbf{r}_t by solving the following linear system:

$$\begin{cases} \mathbf{r}_t = \lambda_1 \mathbf{v}_t^1 + \lambda_2 \mathbf{v}_t^2 + (1 - \lambda_1 - \lambda_2) \mathbf{v}_t^3 \\ \mathbf{r}_{t+1} = \lambda_1 \mathbf{v}_{t+1}^1 + \lambda_2 \mathbf{v}_{t+1}^2 + (1 - \lambda_1 - \lambda_2) \mathbf{v}_{t+1}^3 \end{cases}. \quad (15)$$

Fig. 10 shows an example of the texture transfer operation, and Fig. 11 shows two rendered objects.

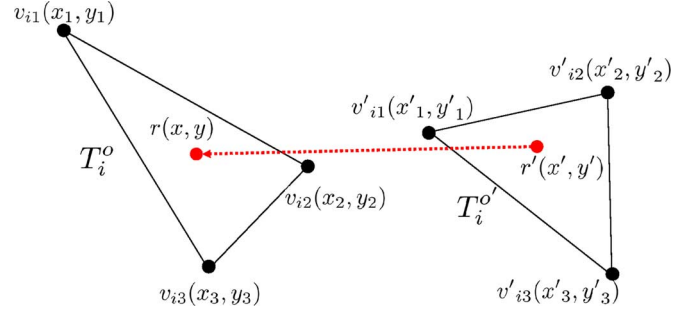


Fig. 9. Illustration of the point correspondence between the original object and its deformed triangular meshes.

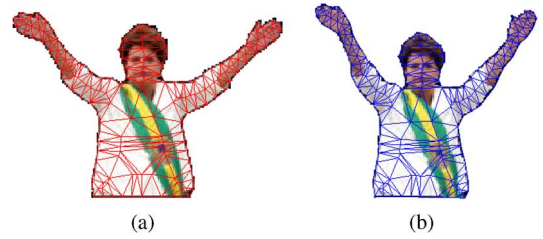


Fig. 10. An example of texture transfer: (a) the input object and its triangular meshes; (b) the result of texture transfer.

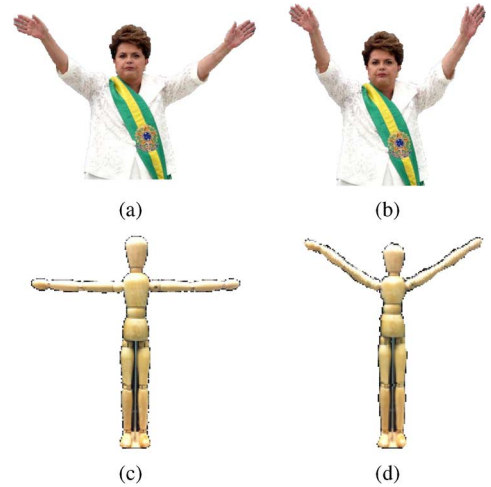


Fig. 11. Examples of object texture rendering: (a) and (c) are two original objects, and (b) and (d) are the corresponding rendered results.

III. EXPERIMENT RESULTS

To assess the effectiveness of the proposed human motion extrapolation method, we conducted three experiments for different applications, namely, motion extrapolation and motion repair. We implemented the proposed algorithm on an i7-3.4 GHz machine. All the experimental results related to this work can be found at <http://research.twnct.net/MExtrapolation/>. To extract the accurate object images from all motion sequences, we first apply our previous work [27] to obtain a background image and then use it as basis to extract the foreground object by using a well-known background subtraction algorithm proposed by Barnich and Van Droogenbroeck [28].

In the first experiment, we used a set of images downloaded from the Internet as input objects and adopted our own motion sequences as reference motions to synthesize new sequences of the



Fig. 12. Experiment 1—Motion extrapolation on sequences #1, #2, #3, #4 and #5: (a), (c), (e), (g) and (i) are the input objects downloaded from the Internet and (b), (d), (f), (h) and (j) are the respective motion extrapolation results derived by the proposed algorithm.

TABLE I
MOTION SEQUENCES USED IN MOTION EXTRAPOLATION EXPERIMENTS. CMC STANDS FOR CONTOUR MANIFOLD CONSTRUCTION. OMS STANDS FOR OBJECT MOTION SYNTHESIS

Fig. #	Size of input object	Average Computation Time per Frame (second)			
		Isomap [29]	CMC LLE [30]	Ours	OMS Ours
12(a)	549 * 554	14.22	2.36	0.60	1.658
12(c)	552 * 376	108.47	2.28	0.26	2.232
12(e)	300 * 387	66.92	2.40	0.45	1.587
12(g)	556 * 818	65.84	2.34	0.63	3.585
12(i)	500 * 333	68.13	2.52	0.57	2.103

input objects. Fig. 12(a), (c), (e), (g) and (i) are the input objects. The corresponding motion extrapolation results are shown in

Fig. 12(b), (d), (f), (h) and (j). Table I shows the test sequences used in the first set of experiments and their corresponding

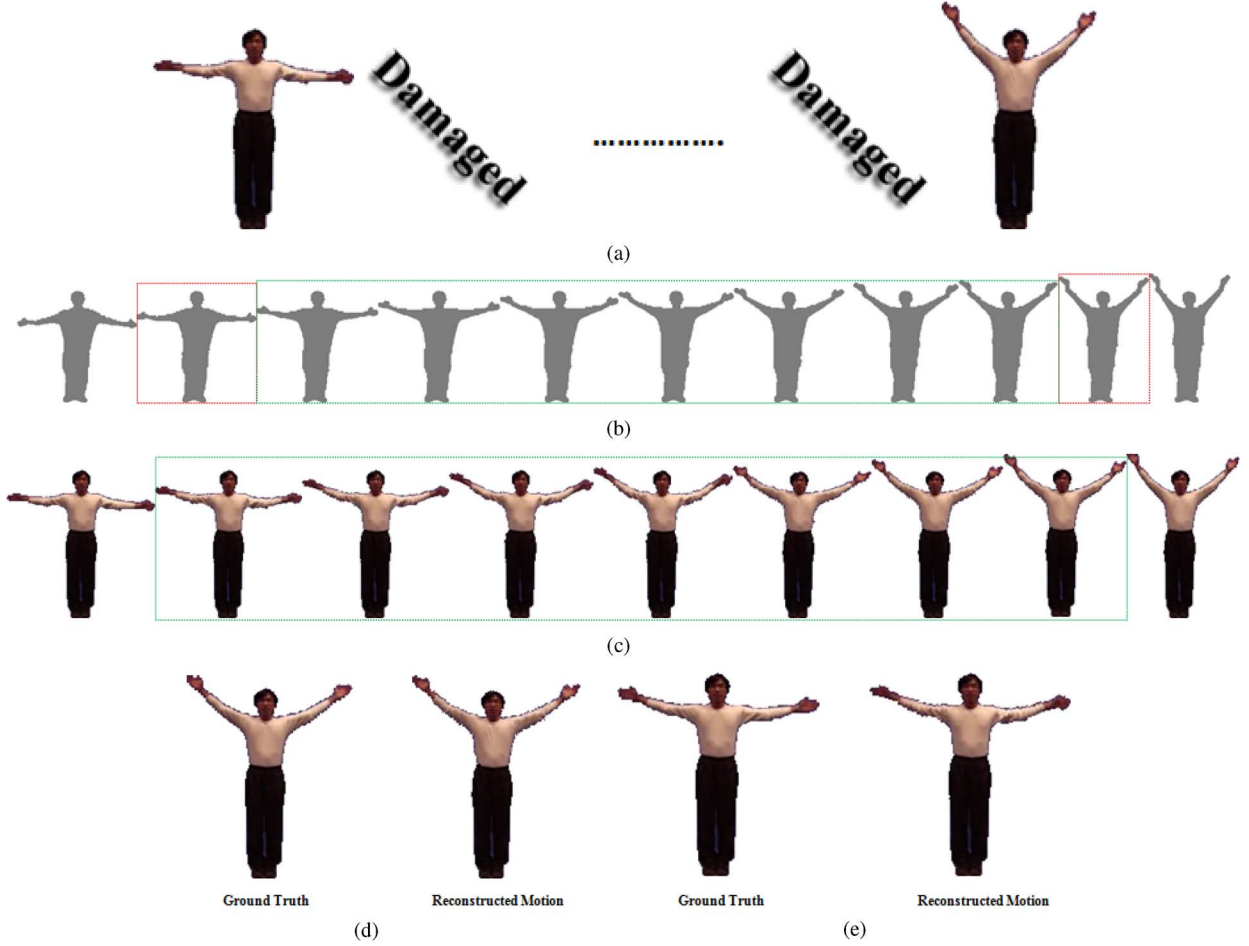


Fig. 13. Experiment 2—Motion repair in sequence #5: (a) part of the damaged motion sequence; (b) the reference motions performed by an actor; (c) part of the reconstructed motions derived by the proposed algorithm; (d) and (e) the reconstructed motions and the ground truth.

execution time. The computation time of Isomap [29], LLE [30] and our method respectively illustrated in Table I. The results of this experiment demonstrate that the proposed method can synthesize new motions of a human object successfully, even if the user only provides a single image of the object.

In the second experiment, we applied the proposed algorithm to a damaged motion sequence to repair the missing motions. The goal was to reconstruct the missing motions from undamaged motions while maintaining the spatiotemporal continuity of the repaired sequence. For example, in Fig. 12, assume that the consecutive motions between the two undamaged motions M_s and M_e are missing. Because the undamaged motions only contain limited information, applying linear interpolation in the temporal domain would result in a poor performance. Therefore, we hired an actor to perform a sequence of motions $\{M_1^r, \dots, M_m^r\}$ (shown in Fig. 12(b)) as a reference between the undamaged motions M_s and M_e (marked by red rectangles in Fig. 12(b)). Then, with the reference motion sequence $\{M_t^r\}$, we searched for the motions that were most similar to M_s and M_e in $\{M_t^r\}$ based on the shape similarity defined by

$$D(M_t^r, M_s) = \frac{1}{|\mathbf{p}'_i|} \sum_{\mathbf{p}_i \in M_t^r} Cost(\mathbf{p}'_i, \mathbf{p}_{\pi i}) + \frac{1}{|\mathbf{p}'_j|} \sum_{\mathbf{p}_j \in M_t^r} Cost(\mathbf{p}_j, \mathbf{p}'_{\pi j}). \quad (16)$$

In (16), \mathbf{p}'_i and \mathbf{p}_j are the uniformly sampled contour points on the shapes of the motion in M_t^r and M_s (or M_e) respectively; and the function $Cost(\cdot)$ measures the cost of matching two sets of contour points, as defined in (2). After deriving the most similar motions M_{ts}^r and M_{te}^r of M_s and M_e , we use the sequence of reference motions between M_{ts}^r and M_{te}^r to synthesize the missing motions with the proposed motion extrapolation algorithm. Fig. 13(a) shows part of the damaged motion sequences; Fig. 13(b) shows part of the filmed reference motion sequences $\{M_1^r, \dots, M_m^r\}$ and the discovered reference postures M_{ts}^r and M_{te}^r (marked by green rectangles in the figure); Fig. 13(c) shows the reconstructed motions derived by the proposed motion extrapolation algorithm; and Fig. 13(d) and (e) compare the reconstructed motions with the ground truth.

To check whether a repaired motion sequence is visually pleasant or not, we compare our method with the methods proposed in [11], [18]. We execute our algorithm and the two algorithms proposed in [11] and [18] to repair the missing motions of two damaged motion sequences prepared by ourselves (<http://research.twncn.net/MExtrapolation/vote/>). Figs. 14 and 16 show, respectively, the results of the three compared methods applied on the two damaged motion sequences. In the two figures, the black parts are the difference portions belonging to the ground truth postures and the red parts are the difference portions belonging to the reconstructed postures. As to the blue













Ours	Reconstruction Result				
	Accuracy	95.4%	96.7%	96.2%	98.7%
Xu et al. [11]	Reconstruction Result				
	Accuracy	88.2%	90.0%	93.2%	96.1%
Ding et al. [18]	Reconstruction Result				
	Accuracy	91.51%	92.68%	93.42%	95.53%

Fig. 14. Sequence #6: Comparison of the ground-truth motions and reconstructed missing motions (the parts in black, red, and blue represent the ground-truth motions, the reconstructed motions, and the perfectly matched portions respectively).

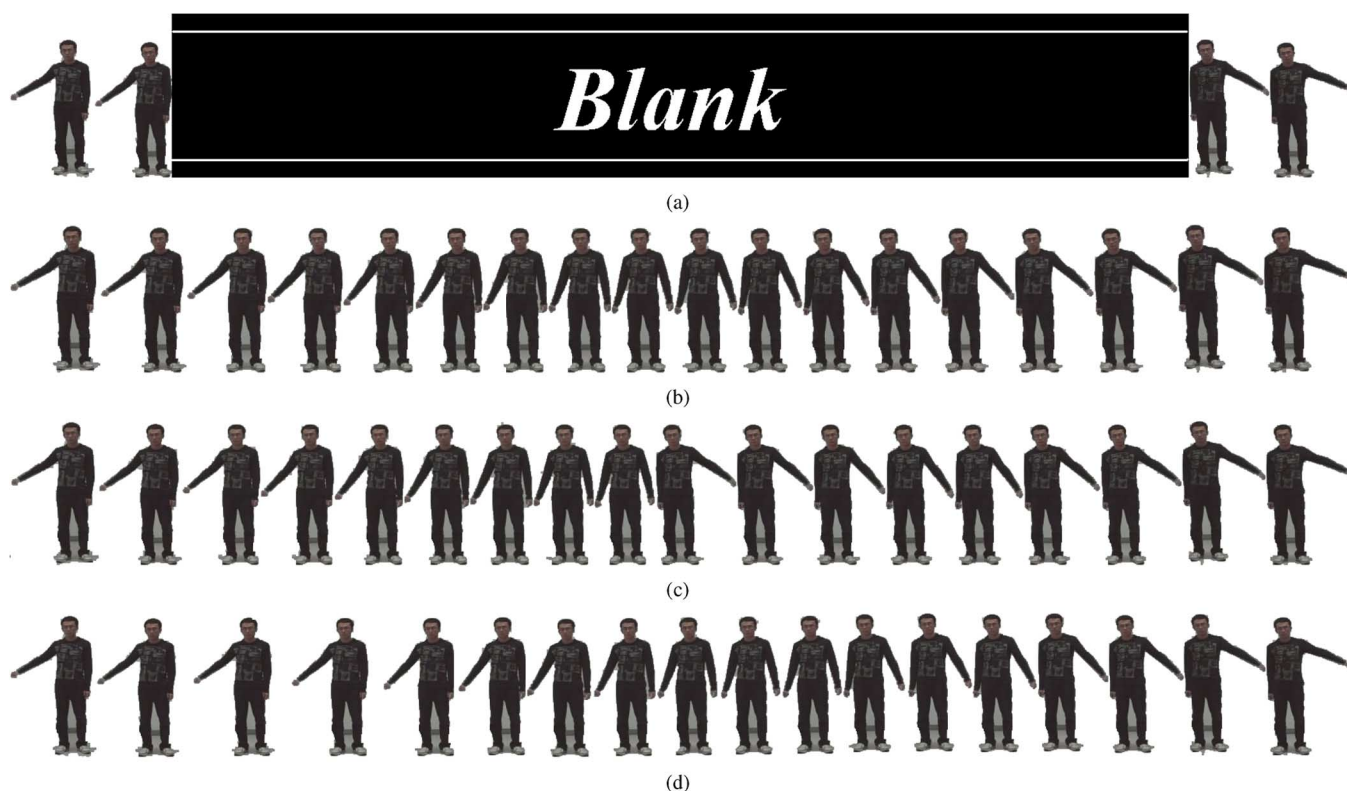


Fig. 15. Experiment 2—Motion repairing result of sequence #6. (a) “Blank” represents the damaged portions of the damaged motion sequence; (b), (d) and (d) are the reconstructed motions done by the proposed algorithm, Xu *et al.*'s algorithm [11] and Ding *et al.*'s algorithm [18] respectively.

parts, they represent the perfectly matched parts between the ground truth postures and the reconstructed postures. Figs. 15 and 17 show the motion repairing results of motion sequence #6 and #7, respectively. The “Blank” portion of Figs. 15(a) and 17(a) represent the damaged video frames of sequence #6

and #7, respectively. Figs. 15(b)–(d) and 17(b)–(d) are reconstructed motion sequences done by the proposed algorithm, Xu *et al.*'s algorithm [11], and Ding *et al.*'s algorithm [18], respectively. It is obvious that the work done by our algorithm is much better than the work done by the two comparison methods.













Ours	Reconstruction Result				
	Accuracy	93.53%	93.30%	93.37%	93.87%
Xu <i>et al.</i> [11]	Reconstruction Result				
	Accuracy	87.83%	87.01%	91.5%	90.95%
Ding <i>et al.</i> [18]	Reconstruction Result				
	Accuracy	87.3%	92.48%	91.59%	91.24%

Fig. 16. Sequence #7: Comparison of the ground-truth motions and reconstructed missing motions (the parts in black, red, and blue represent the ground-truth motions, the reconstructed motions, and the perfectly matched portions respectively).

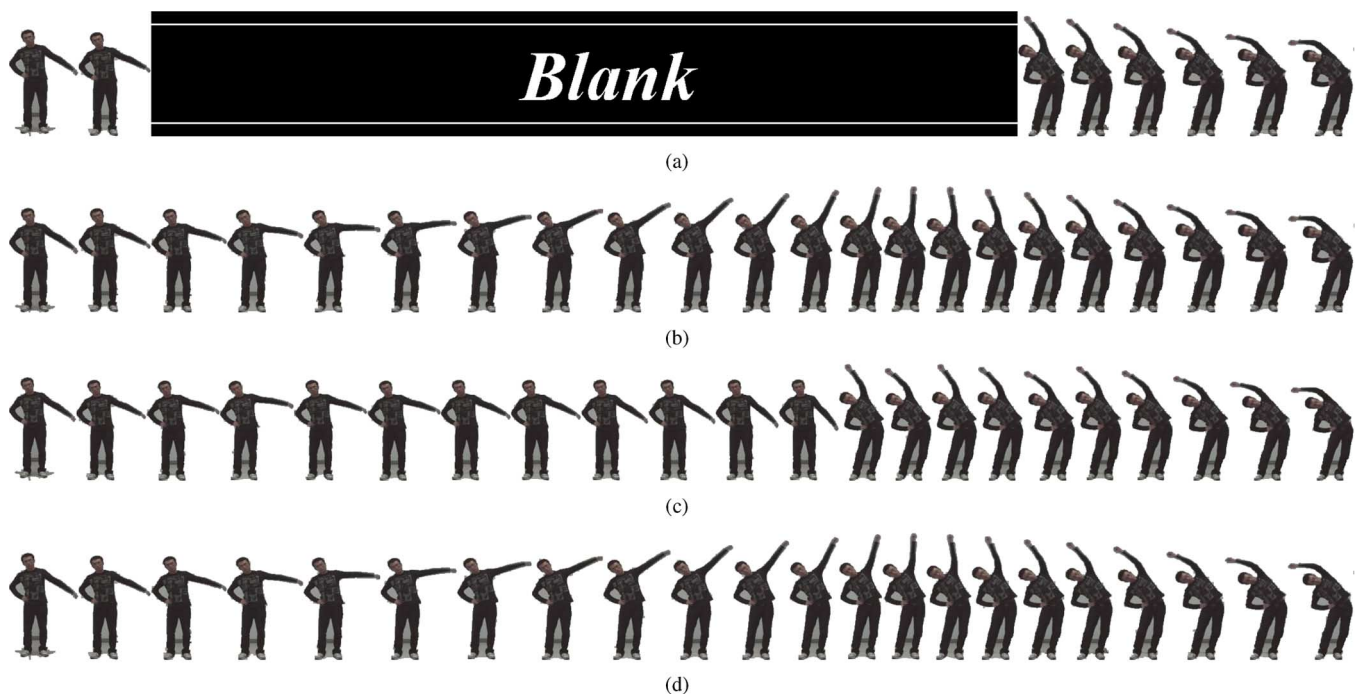


Fig. 17. Experiment 2—Motion repairing result of sequence #7. (a) “Blank” represents the damaged portions of the damaged motion sequence; (b), (c) and (d) are the reconstructed motions done by the proposed algorithm, Xu *et al.*'s algorithm [11] and Ding *et al.*'s algorithm [18] respectively.

To check the degree of user satisfaction, we create a website <http://research.twncf.net/MExtrapolation/vote/>, and upload the motion sequences #6 and #7 as well as the reconstructed results done by our algorithm, Xu *et al.*'s algorithm [11] and Ding *et al.*'s algorithm [18], respectively. Fig. 18 shows for sequence #6, our result received the highest score, Ding *et al.*'s algorithm the second, and Xu *et al.*'s algorithm the third. In the case of sequence #7, our algorithm, again, received the highest score.

Ding *et al.*'s algorithm and Xu *et al.*'s algorithm are the second and third again.

IV. DISCUSSION AND CONCLUSION

We have proposed a novel human motion extrapolation algorithm for synthesizing new motion sequences of an object with minimum user intervention. To demonstrate the efficacy of the algorithm, we conducted three sets of experiments on several

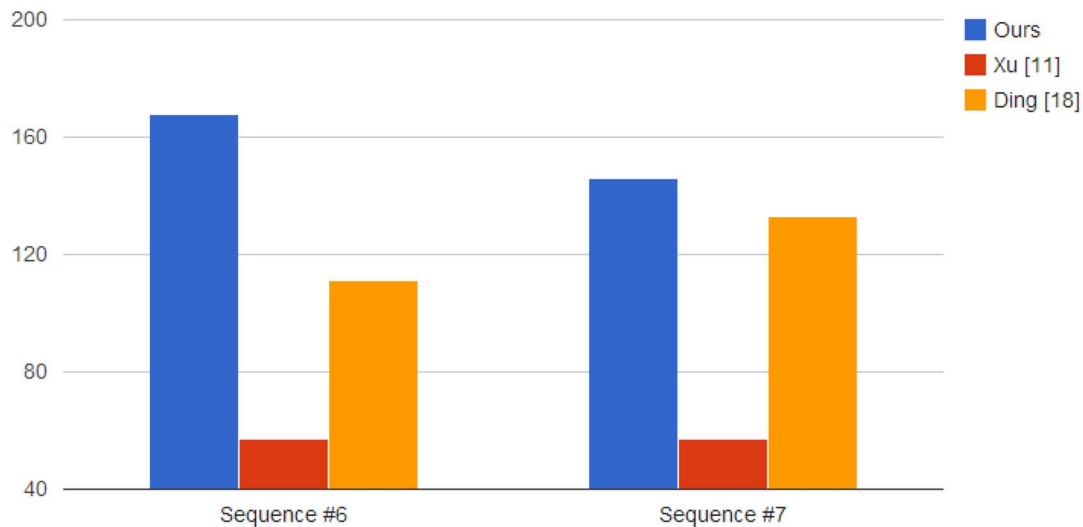


Fig. 18. User satisfaction questionnaire result of the motion repairing result derived by the proposed algorithm, Xu *et al.* [11] and Ding *et al.* [18].

test sequences. The results demonstrate that the algorithm can produce visually pleasing results in a wide range of applications, such as motion extrapolation, motion synthesis and motion repair operations. In addition, all of the results are produced automatically. The user only needs to input a human object and a reference motion sequence. The contribution of this work is twofold. First, the proposed contour manifold provides a compact representation of high dimensional human motions, and it makes human motions analyzable and maneuverable. Second, The constructed contour manifolds provide a systematic way to manipulate human motion efficiently in a low dimensional space.

The proposed contour manifold computation algorithm has two limitations, which we will address in our future work. First, the algorithm relies heavily on the results of shape correspondence estimation. If the reference motion sequence contains many self occlusions the proposed correspondence estimation method would fail to estimate the correspondence of the occluded parts. Second, because the object motion synthesis algorithm uses the textures of an input object to synthesize new images of the object, visual defects may appear in the resulting motion sequences if the new motion of the input object is very different from the original one. To solve the self occlusion problem, we will try to improve the feasibility of our shape correspondence estimation method by incorporating different features.

REFERENCES

- [1] M. Gleicher, "Motion path editing," in *Proc. Symp. Interactive 3D Graphics*, 2001, pp. 195–202.
- [2] R. Heck and M. Gleicher, "Parametric motion graphs," in *Proc. Symp. Interactive 3D Graphics and Games*, 2007, pp. 129–136.
- [3] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," in *ACM SIGGRAPH Papers*, 2005, pp. 1134–1141.
- [4] G. Cheung, S. Baker, J. Hodgins, and T. Kanade, "Markerless human motion transfer," in *ACM SIGGRAPH Sketches*, 2004, pp. 31–36.
- [5] A. C. Fang and N. S. Pollard, "Efficient synthesis of physically valid human motion," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 417–426, Jul. 2003.
- [6] M. Gleicher, "Retargetting motion to new characters," in *Proc. 25th Annu. Conf. Comput. Graph. Interactive Tech.*, 1998, pp. 33–42.
- [7] L. Kovar, M. Gleicher, and F. H. Pighin, "Motion graphs," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 473–482, 2002.
- [8] C. K. Liu and Z. Popović, "Synthesis of complex dynamic character motion from simple animations," in *Proc. 29th Annu. Conf. Comput. Graphics Interactive Tech.*, 2002, pp. 408–416.
- [9] A. Safonova, J. K. Hodgins, and N. S. Pollard, "Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces," in *ACM SIGGRAPH Papers*, 2004, pp. 514–521.
- [10] J. Kobayashi, C. Bi, and S. Takahashi, "Sophisticated construction and search of 2D motion graphs for synthesizing videos," in *Proc. 4th Pacific-Rim Symp. Image and Video Technol.*, 2010, pp. 487–494.
- [11] X. Xu, L. Wan, X. Liu, T.-T. Wong, L. Wang, and C.-S. Leung, "Animating animal motion from still," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 117:1–117:8, Dec. 2008.
- [12] J.-W. Hsieh, Y.-T. Hsu, H.-Y. M. Liao, and C.-C. Chen, "Video-based human movement analysis and its application to surveillance systems," *IEEE Trans. Multimedia*, vol. 10, no. 3, pp. 372–384, Apr. 2008.
- [13] H. J. Shin and J. Lee, "Motion synthesis and editing in low-dimensional spaces," *J. Visualization Comput. Animation*, vol. 17, no. 3–4, pp. 219–227, 2006.
- [14] J. Chai and J. K. Hodgins, "Performance animation from low-dimensional control signals," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 686–696, Jul. 2005.
- [15] A. Elgammal and C.-S. Lee, "Inferring 3D body pose from silhouettes using activity manifold learning," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition*, 2004, pp. 681–688.
- [16] C.-S. Lee and A. Elgammal, "Style adaptive contour tracking of human gait using explicit manifold models," *Mach. Vis. Applications*, vol. 23, no. 3, pp. 461–478, 2012.
- [17] Y.-M. Liang, S.-W. Shih, A. C.-C. Shih, H.-Y. M. Liao, and C.-C. Lin, "Unsupervised analysis of human behavior based on manifold learning," in *Proc. IEEE Int. Symp. Circuits Syst.*, Taipei, Taiwan, May 24–17, 2009, pp. 2605–2608.
- [18] T. Ding, M. Sznajder, and O. Camps, "A rank minimization approach to video inpainting," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.
- [19] N. C. Tang, C.-T. Hsu, T.-Y. Lin, and H.-Y. M. Liao, "Example-based human motion extrapolation based on manifold learning," in *Proc. 19th ACM Int. Conf. Multimedia*, 2011, pp. 1305–1308.
- [20] D. Cai, X. He, J. Han, and H. Zhang, "Orthogonal Laplacian faces for face recognition," *IEEE Trans. Image Process.*, vol. 15, no. 11, pp. 3608–3614, Nov. 2006.
- [21] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, Apr. 2002.
- [22] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, no. 4, pp. 325–340, Nov. 1987.

- [23] F. Bookstein, *Morphometric Tools for Landmark Data: Geometry and Biology*. Cambridge, U.K.: Cambridge Univ., 1991.
- [24] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," *Adv. Neural Inf. Process. Syst.*, vol. 14, pp. 585–591, 2001.
- [25] J. R. Shewchuk, "Triangle: Engineering a 2d quality mesh generator and Delaunay triangulator," in *Proc. WACG*, 1996, vol. 1148, pp. 203–222.
- [26] T. Igarashi and Y. Igarashi, "Implementing as-rigid-as-possible shape manipulation and surface flattening," *J. Graphics, GPU, & Game Tools*, vol. 14, no. 1, pp. 17–30, 2009.
- [27] N. C. Tang, C.-T. Hsu, C.-W. Su, T. K. Shih, and H.-Y. M. Liao, "Video inpainting on digitized vintage films via maintaining spatiotemporal continuity," *IEEE Trans. Multimedia*, vol. 13, no. 4, pp. 602–614, Aug. 2011.
- [28] O. Barnich and M. V. Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1709–1724, Jun. 2011.
- [29] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.
- [30] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.



Nick C. Tang received the B.S. and M.S. degrees from Tamkang University, Tamsui, Taiwan, in 2003 and 2005, respectively. He also received the Ph.D. degree from Tamkang University in 2008. Currently, he is a Postdoctoral Fellow with the Institute of Information Science, Academia Sinica, Taiwan. His research interests include image and video analysis, computer vision, computer graphics, and their applications.



Chiou-Ting Hsu (M'98) received the Ph.D. degree in computer science and information engineering from National Taiwan University (NTU), Taipei, Taiwan, in 1997. From 1998 to 1999, she was with Philips Innovation Center Taipei, Philips Research, as a senior research engineer. She joined the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, as an Assistant Professor in 1999 and is currently a Professor. Her research interests include multimedia signal processing, video analysis and content-based retrieval.

Prof. Hsu received the Citation Classic Award from Thomson ISI in 2001 for her paper Hidden digital watermarks in images. She was an Associate Editor of *Advances in Multimedia* during 2006–2012, and is current an Associate Editor of the *IEEE Transactions on Information Forensics and Security*.



and multimedia applications.

Ming-Fang Weng received the B.S. degree and M.S. degree from National Chiao Tung University, Hsinchu, Taiwan, in 1998 and 2000 respectively, Ph.D. degree from National Taiwan University in 2010, all in computer science and information engineering. He was a Postdoctoral Fellow in the Institute of Information Science, Academia Sinica, Taiwan, and is currently a Principal Engineer in the Institute for Information Industry, Taiwan. His research interests include digital content analysis, image/video information retrieval, computer vision,



Tusng-Yi Lin received the B.S. in electrical engineering from National Taiwan University, Taipei, Taiwan, at 2009 and is currently pursuing the Ph.D. degree at University of California, San Diego (UCSD). He is a member of Computer Vision Laboratory, UCSD. His research interests include computer vision and machine learning.



Mark Liao (F'13) received his Ph.D. degree in electrical engineering from Northwestern University in 1990. In July 1991, he joined the Institute of Information Science, Academia Sinica, Taiwan and currently, is a Distinguished Research Fellow. He has worked in the fields of multimedia signal processing, image processing, computer vision, pattern recognition, video forensics, and multimedia protection for more than 25 years. During 2009–2011, he was the Division Chair of the computer science and information engineering division II, National Science Council of Taiwan. He is jointly appointed as a Professor of the Computer Science and Information Engineering Department of National Chiao-Tung University and the Department of Electrical Engineering of National Cheng Kung University. During 2009–2012, he was jointly appointed as the Multimedia Information Chair Professor of National Chung Hsing University. Since August 2010, he has been appointed as an Adjunct Chair Professor of Chung Yuan Christian University. He received the Young Investigators' Award from Academia Sinica in 1998; the Distinguished Research Award from the National Science Council of Taiwan in 2003 and 2010; the National Invention Award of Taiwan in 2004; the Distinguished Scholar Research Project Award from National Science Council of Taiwan in 2008; and the Academia Sinica Investigator Award in 2010. His professional activities include: Co-Chair, 2004 International Conference on Multimedia and Exposition (ICME); Technical Co-chair, 2007 ICME; General Co-Chair, 17th International Conference on Multimedia Modeling; President, Image Processing and Pattern Recognition Society of Taiwan (2006–08); Editorial Board Member, *IEEE Signal Processing Magazine* (2010–present); Associate Editor, *IEEE Transactions on Image Processing* (2009–present), *IEEE Transactions on Information Forensics and Security* (2009–12) and *IEEE Transactions on Multimedia* (1998–2001). He has been a Fellow of the IEEE since 2013 for contributions to image and video forensics and security.