EUROGRAPHICS 2002 / I. Navazo Alvaro and Ph. Slusallek
(Guest Editors)

View Interpolation for Dynamic Scenes

Jiangjian Xiao Cen Rao Mubarak Shah

Computer Vision Lab
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, Florida 32816, USA
{jxiao, rcen, shah}@cs.ucf.edu

Abstract

This paper describes a novel technique for synthesizing a dynamic scene from two images without the use of a 3D model. A scene containing rigid or non-rigid objects, in which each object can move in any orientation or direction, is considered. It is shown that such a scene can be converted into several equivalent static scenes, where each scene only includes one rigid object. Our method can generate a series of continuous and realistic intermediate views from only two reference images without 3D knowledge. The procedure consists of three main steps: segmentation, morphing and postwarping. The key frames are first segmented into several layers. Each layer can be realistically morphed after determining its fundamental matrix. Based on the decomposition of 3D rotation matrix, an optimized and unique postwarping path is automatically determined by the least distortion method and boundary connection constraint. Finally, four experiments, which include morphing of a rotating rigid object in presence of occlusion and morphing of non-rigid objects (human), are demonstrated.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Viewing algorithms

1. Introduction

The technique for transforming a source view into a target view is called shape blending or view morphing. This creates a continuous and smooth transition between the two original views by 2D interpolation of shape and color. Several methods have been proposed in this area, and most of them are based on image-based rendering techniques[9, 15, 11]. A primary question in this context is how to retain the original appearance and 3D shape information after morphing.

In the shape blending area, there are several approaches to blend the shape based on boundaries or internal vertices. Sederberg et al.[7] presented shape blending to interpolate shape with boundary vertices. Tal and Elber[16] used Bspline curve matching to determine the boundary correspondences, and then constructed compatible triangulation to blending. Alexa[6] proposed another approach to construct an isomorphic triangle mesh using boundary vertex correspondences, and then to interpolate the two images by both boundary and interior vertices. All of these methods try to prevent shape bleeding and make the in-between shapes retain their original appearance as much as possible, but cannot preserve 3D shape information.

In feature-based morphing[15, 14], corresponding features are extracted from the images to generate the in-between views. Beier and Neely[15] presented a field morphing technique to improve 2D mesh warping. In their method, a uniform 2D mesh was mapped on the reference images. They utilized the continuity of the mesh to control image warping based on fields of influence. Lee et al.[11] used snakes (active contour) to determine the landmarks semi-automatically. Johan et al.[17] proposed a method for computing the correspondences between two feature curves by minimizing a cost function. Since the feature correspondences carry more information than boundaries and can partially imply 3D information, the appearance of images generated by this approach is better. This method is broadly used by the current commercial morphing software, such as Morpher, Morph man, Elastic Reality, and Morpheus, etc. All of the software can generate an in-between image sequence from two pictures based on corresponding features (points and line segments), which are manually specified. However, the existing com-

mercial software cannot fully preserve 3D physical characteristics.

Another interesting area is view morphing[1, 3, 4, 12], which uses basic principles of projective geometry to preserve 3D information implied in images. Seitz and Dyer[1] introduced this approach to synthesize a new view from two static views taken by two cameras. Manning and Dyer[4] extended this approach to rigid objects with translation, which is called dynamic view morphing. More recently, Wexler and Shashua[13] proposed another technique to morph a dynamic view with a moving object (along a straight line path) from three reference viewpoints. The advantage of view morphing is that it can reflect the physical change of scene accurately without 3D knowledge of the scene. The recovery of internal and external parameters of the camera is not necessary.

Previous work in dynamic view morphing has only dealt with translation objects. However, in the real world, most active objects can not only rotate and translate, but can also deform. Given two pictures of the rotating and translating object taken at different times and locations, constructing the in-between view is a very challenging problem. Another problem in this context is how to recover occluded facets of a rigid object when it rotates and when some facets are only visible in one reference view.

In this paper, we present a novel approach to overcome these problems. In particular, we extend the view morphing to a rotation case, and progressively apply it to non-rigid objects with complicated motion. Our scenario consists of rigid or non-rigid moving objects. Each object can rotate in any orientation and translate in any direction.

In our approach, we assume that non-rigid objects can be separated into several rigid parts, even though there is some apparent deformation at the joints between these parts. We segment the image into several layers, each layer containing one rigid part. For each part, we use its fundamental matrix to rectify the pair of layers to parallel views for morphing. Next, we use the least distortion method[6] for each layer to find an optimized position for postwarping. Finally, the boundary connection constraint with centroid adjustment is used to retain the spatial relationship of the layers. From only two key frames, we synthesize a continuous, smooth, realistic intermediate view.

The proposed technique can be applied in several applications, such as filling a gap in a movie, creating virtual views from static images, compressing movie sequences for fast transmission, and switching camera views from one to another.

The paper is organized as follows. Section 2 introduces the related work on view morphing, and identifies some drawbacks of this work. Section 3 presents an overview of the dynamic view interpolation algorithm for a non-rigid object. Section 4 shows how a dynamic scene can be separated into several layers. Section 5 discusses how to find the postwarping path using the least distortion method. Section 6 introduces the concept of the boundary connection constraint, and illustrates how to implement it for a non-rigid dynamic scene. Finally, in section 7, we demonstrate four experiments, which include morphing of a rotating rigid object in presence of occlusion, and morphing of non-rigid objects (human).

2. Related Work

In Seitz's method[1], the view morphing algorithm was implemented using three steps: prewarping, morphing, and postwarping.

In prewarping, a fundamental matrix, F, between the pair of layers such that $p_1^T F p_2 = 0$, where p_1 and p_2 are the corresponding points of key frames I_1 and I_2, is determined by Hartley's 8-point algorithm[5]. Prewarping projective transforms H_1, H_2 can determined by Equation 1[2].

$$(H_1^{-1})^T F H_2 = \hat{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad (1)$$

where $\hat{F}$ is the basic fundamental matrix for parallel views. I_1, I_2 are rectified to parallel views $\hat{I}_1, \hat{I}_2$ by applying transforms H_1^{-1} to I_1 and H_2^{-1} to I_2.

Next, dynamic programming[8] was used to find dense correspondences for each scan-line, or the corresponding lines were manually marked to morph the rectified images.

In postwarping step, the morphed result was reprojected to a proper final position. If the camera motion is only translation, the ideal final position is a physically correct position which can be obtained by linear interpolation of a quadrilateral in the key frames[2]. However, if there is some rotation, the linear interpolation will result in shrinkage or deformation. Seitz and Dyer manually adjusted the postwarping position to approximate the ideal position, which is hard to implement and cannot uniquely determine the solution, and no automatic method was discussed to get the correct position. Figure 4 (Top) shows that the area of a triangle (postwarping quadrilateral is the same) cannot be maintained by linear interpolation and the distortion will be intolerable if the object has a big rotational motion.

Manning and Dyer[4] extended Seitz and Dyer's method to dynamic view morphing. They considered several moving objects in the scene, where each object can only move along a straight line without any rotation. In order to simplify the motion problem, they first introduced a fixed camera model to compensate for the translation component of the camera, and then converted the dynamic scene to a static scene by adding the object's translation to the camera model.

3. Algorithm Overview

Dynamic view interpolation deals with scenes containing moving objects in presence of camera motion. In this case,

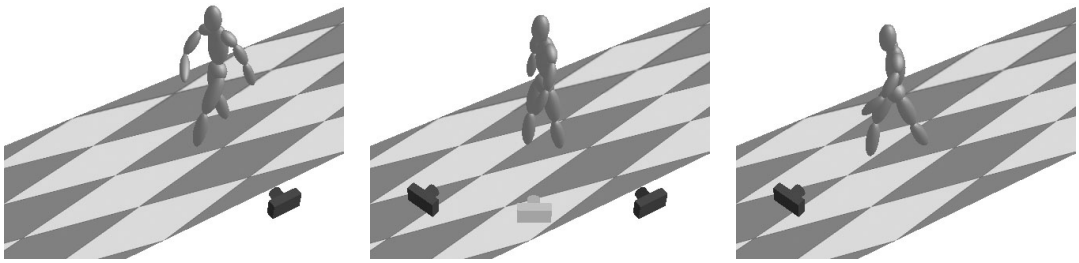

Figure 1: *A typical dynamic scenario. In this scene, a person is walking along a straight line. (a) and (c) are two reference views taken at different times from different viewpoints. (b) is the in-between view synthesized from the reference views. The camera shown in gray is the virtual camera, which is on the line connecting the two original cameras.*

the moving object can be a rigid or non-rigid object. Figure 1 shows a typical scenario for dynamic view morphing, where a person (non-rigid object) is walking along the straight line. Two reference views are taken at different times from different viewpoints, and a virtual view is generated from the two key frames. Our algorithm can effectively generate the realistic virtual view based on following three key steps: segmentation, morphing and postwarping.

First, given two reference images of a dynamic scene, we segment the scene into the background and moving object using edge detection. After extracting the contour of the whole object, we smooth the contour by cubic curve and detect the parts by using negative curvature and symmetry property[10]. During this procedure, some interactive adjustment is required. Based on the part detection, the two images are naturally segmented into several layers, where each pair of image layers, I_1 and I_2, only contain one rigid object.

The second step is a morphing procedure similar to Seitz and Dyer's view morphing algorithm. In this step, each pair of layers is rectified and morphed. First, eight or more corresponding point sets P_1 and p_2 for image layers I_1 and I_2 are determined manually. The fundamental matrix F and homographies H_1 and H_2 are computed. Next, transform H_1^{-1} is applied to I_1 and H_2^{-1} to I_2, and images I_1 and I_2 are rectified to parallel views $\hat{I}_1$ and $\hat{I}_2$. Using dynamic programming for each scan-line[8], all dense correspondences are automatically determined to blend the two images into new image $\hat{I}_s$.

Third, we use the least distortion method and boundary connection constraint to determine an optimized postwarping path. Four or more pairs of control points are selected from the boundary pixels of each layer. Based on the least distortion method and boundary connection constraint, each layer can be adjusted to a proper postwarping position. A homography H_s between control points P_s and $\hat{P}_s$ can be determined, where $P_s = H_s\hat{P}_s$. Finally, we warp the morphed image $\hat{I}_s$ to a new image I_s using H_s, and composite all image layers together to blend the new intermediate view.

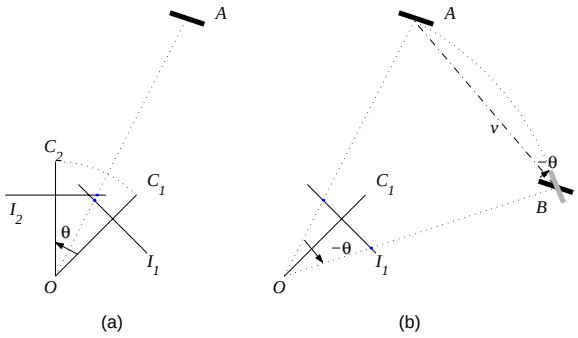

Figure 2: *(a) The object A is fixed, and the moving camera's optical axis rotates around the optical center O from OC_1 to OC_2 by θ. (b) The camera is fixed, this rotation can be compensated by adding rotation $-\theta$ to the object with a translation v. The gray object at B is the final position after compensation. I_1 and I_2 are the projection planes.*

4. Dynamic View Interpolation with Rigid Objects

In this section, we first show how to convert a moving camera model to a fixed camera model in the presence of rotation. Then we discuss a simple case involving two basic movements with a rigid object. Finally, we extend this to a general case which deals with the object motion in the presence of camera motion. We discuss how this can be converted into an equivalent case where the object is fixed and the camera moves. Moreover, we show that such a dynamic scene can be separated into several layers, where each can be considered as a static scene.

4.1. Fixed Camera Model

In a perspective camera model, the camera parameters can be divided into intrinsic and extrinsic parameters. For a fixed camera model, the intrinsic parameters such as focal length

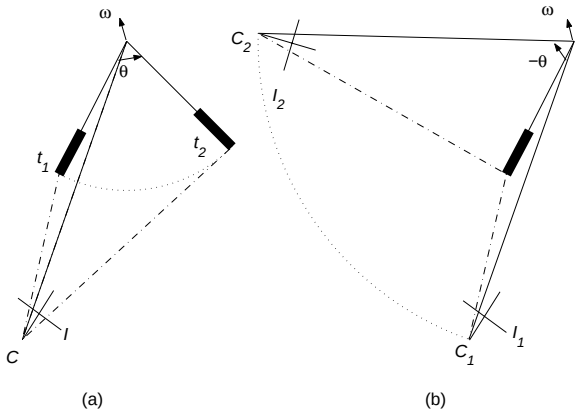

Figure 3: *Rotation case. (a) Original scenario, the object rotates by θ around axis ω and camera C is fixed. (b) Equivalent scenario, the camera rotates by −θ around axis ω and the object is fixed. The camera centers C_1 is at t_1, and C_2 is at t_2. I, I_1 and I_2 are the projection planes.*

may change with time, but the extrinsic parameters are fixed, and can be represented as $M_{ext} = RT$, where R and T are the rotation and translation of the camera motion respectively. If the camera motion is only translation, the moving camera model can be easily converted to a fixed camera model by adding an inverse translation vector, $-v$, to the object in the scene. For a pure rotation case (Figure 2),we can compensate for the rotation θ by adding $-\theta$ to the camera, and rotating every object by $-\theta$ around its own axis with some amount of translation v. For different objects, the translation may be different. If the camera has both translation and rotation, we also can convert it to a fixed camera model by adding a proper translation to the object with $-\theta$ rotation.

4.2. Object Translation and Rotation

First, we analyze the pure rotation case with a fixed camera model. In this case, the object rotates by an angle θ around axis ω with a constant speed, the camera C is fixed, and two pictures I_1 and I_2 are taken at time t_1 and t_2 as shown in Figure 3.a. This original rotation scenario can be converted into an equivalent case (Figure 3.b), such that the object is fixed and the camera rotates by $-\theta$ around axis ω with a constant speed, and two pictures I_1' and I_2' are taken at time t_1 and t_2. Therefore, in this case, the object has the same location and shape in images I_1 and I_2'. Moreover, the camera also can be considered rotating around its axis and moving along C_1C_2. The object's translation can be converted by the same method.

If the camera is not fixed, we first convert the moving camera model to a fixed camera model by adding proper translation and rotation to each object in the scene. As a result, each moving object may have a different fixed camera model with a different orientation and position. If we consider such a moving object with its fixed model, it also can be converted into an equivalent case as in Figure 3. Based on the analysis, we can separate the dynamic scene into layers, where each layer includes only one rigid object. The fundamental matrix of each layer may be different and only can be determined by using the points on the rigid object in that layer.

Even though we can use static view morphing algorithm to generate the morphing image for each layer, we cannot determine the correct final position for postwarping due to the rotation. In order to reduce the distortion and recover the correct postwarping position as much as possible, we use the least distortion method (section 5) and boundary connection constraint (section 6).

5. An Optimized Postwarping Using the Least Distortion Method

The 3D rotation of an object can be decomposed into three components α, β, γ around the axes X, Y, and Z respectively. X and Y are parallel to x and y on the image plane, and Z is perpendicular to that plane. The rotation transformation can be represented as $R = R_z \cdot R_y \cdot R_x$ due to Euler. Without the depth information, the rotations R_y and R_x are difficult to accurately recover using only two images. But the rotation R_z can be obtained by decomposing affine transformation into two matrices by Singular Value Decomposition (SVD).

For any 3 pairs of corresponding non-collinear points $P_1\{p_{11}, p_{12}, p_{13}\}$ and $P_2\{p_{21}, p_{22}, p_{23}\}$, we can form a triangle for each set of points. There is an affine transformation represented by matrix A and vector T between two original shapes such that $P_2 = AP_1 + T$. Since the translation doesn't change the shape of the triangle, the shape will be determined only by A. If we change the rotation and scaling parts of A, we can get a different intermediate shape $P_s = A(s)P_1$, where $A(s)$ is the corresponding intermediate transformation at $s(s \in [0,1])$. One possible representation for $A(s)$ based on linear interpolation can be expressed in Equation 2.

$$A(s) = (1-s)I + sA \tag{2}$$

However, using this linear transformation, the shape and area of the triangle cannot be preserved (Figure 4). In order to maintain the shape and area of the triangle during postwarping, we use another representation of $A(s)$, which can be implemented by the least distortion method[6].

The least distortion method is based on the decomposition of an affine transformation. The basic idea is that an affine transformation can be decomposed into a rotation matrix $R(\alpha)$ and a scaling-shearing matrix C by SVD:

$$A = SVD = S \begin{bmatrix} v_1 & 0 \\ 0 & v_2 \end{bmatrix} D,$$

where S and D are rotation matrices, and V is scaling matrix.

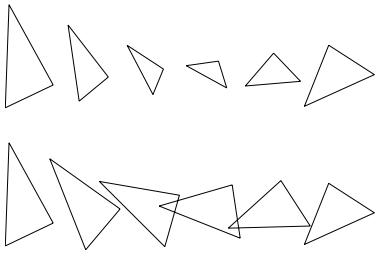

Figure 4: *Two representation of the affine transformation. Bottom: using the least distortion method (Equation 4), the triangle's area is kept constant. Top: using linear interpolation (Equation 2), the triangle shrinks at the middle position. The original triangles are shown at the left and right sides.*

Then, we can denote rotation matrix $R(\alpha) = SD$, and the scaling-shearing matrix $C = D^{-1}VD$.

$$\begin{aligned} A &= SVD = S(DD^{-1})VD = (SD)(D^{-1}VD) \\ &= R(\alpha)C = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} v_1 & v_h \\ v_h & v_2 \end{bmatrix} \end{aligned} \quad (3)$$

From Equation 3, we can construct a new intermediate transformation A(s) as below.

$$\begin{aligned} A(s) &= R(s)C(s) = R(s)((1-s)I + sC), \\ &= \begin{bmatrix} \cos(s\alpha) & -\sin(s\alpha) \\ \sin(s\alpha) & \cos(s\alpha) \end{bmatrix} ((1-s)I + sC), \end{aligned} \quad (4)$$

where the rotation matrix $R(s)$ is linearly changed by the rotation angle $s\alpha$, and the scaling-shearing matrix $C(s)$ is also linearly controlled by scale s.

Comparing Equations 2 and 4, the linear interpolation is the special case of the least distortion transformation when $R(s) = I$. The least distortion transformation can linearly change the affine transformation not only by the scale-shear matrix, but also by the rotation angle. Therefore, the triangle can control its orientation by the rotation angle and the shape by the scale-shear matrix separately. Figure 4 compares the intermediate shapes obtained using the two transformations. Using the least distortion method, the shape and area of the triangle can be kept steady, and the distortion can be minimized during the postwarping.

6. Boundary Connection Constraint

Even though an optimized solution for a single object can be determined by using the least distortion method, the rotation components *Rx* and *Ry* cannot be fully recovered, which may be converted to translation, scaling and shear. The scaling and shear can be partially recovered by scaling-shearing matrix *C*, but the translation cannot be recovered. Therefore, if a connected non-rigid or articulated object is morphed separately, after postwarping, the separate parts may get unconnected or squeezed together. In order to deal with this problem, we introduce the boundary connection constraint to force the connections among the parts.

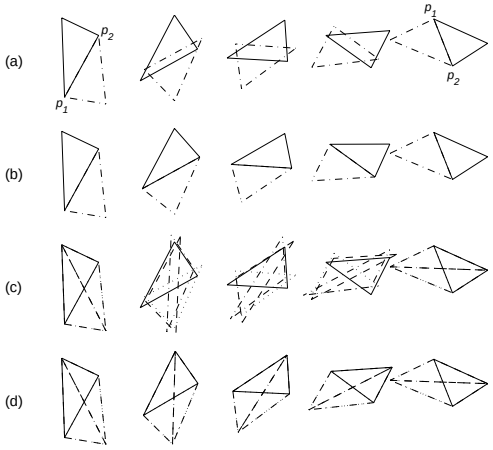

Figure 5: *Boundary connection constraint for a quadrilateral. (a) The least distortion method along one diagonal without using the boundary connection constraint. (b) (a) with the boundary connection constraint. (c) The least distortion method along two diagonals without using the boundary connection constraint. (d) (c) with the boundary connection constraint.*

Boundary Connection Constraint: the points shared by rigid parts should keep the same position after postwarping.

In addition, we also adjust every part's centroid to recover the lost displacement due to rotation.

In segmentation step, we select four or more control points to outline each rigid part as shown in Figure 6. Each set of control points can form a polygon, and any pair of neighboring parts can share several points at the boundary. First, we consider a simple case: using four points to outline a rigid part, which can form a quadrilateral (Figure 5). The quadrilateral can be decomposed into two triangles along one diagonal, which share the points p_1 and p_2 as shown in Figure 5.a. If we apply the least distortion transformation to the two triangles independently, the two triangles don't share the boundary points any more. In order to maintain boundary connection property, we compute new points by averaging the points from the two triangles as shown in Figure 5.b. However, this results in small shrinking along the other diagonal. Therefore, we decompose the quadrilateral along the both diagonals, and get four triangles instead. Then, we apply the least distortion transformation to each pair of triangles independently (Figure 5.c), and compute new points by averaging the points from the four triangles (Figure 5.d).

For more complicated cases, the points are not only shared in one rigid part, but are also shared between two neighboring parts. In this case, we assume that the whole object's centroid $C = \frac{1}{n}\sum_{i=1}^{n} p_i$ is moving along a straight line, even

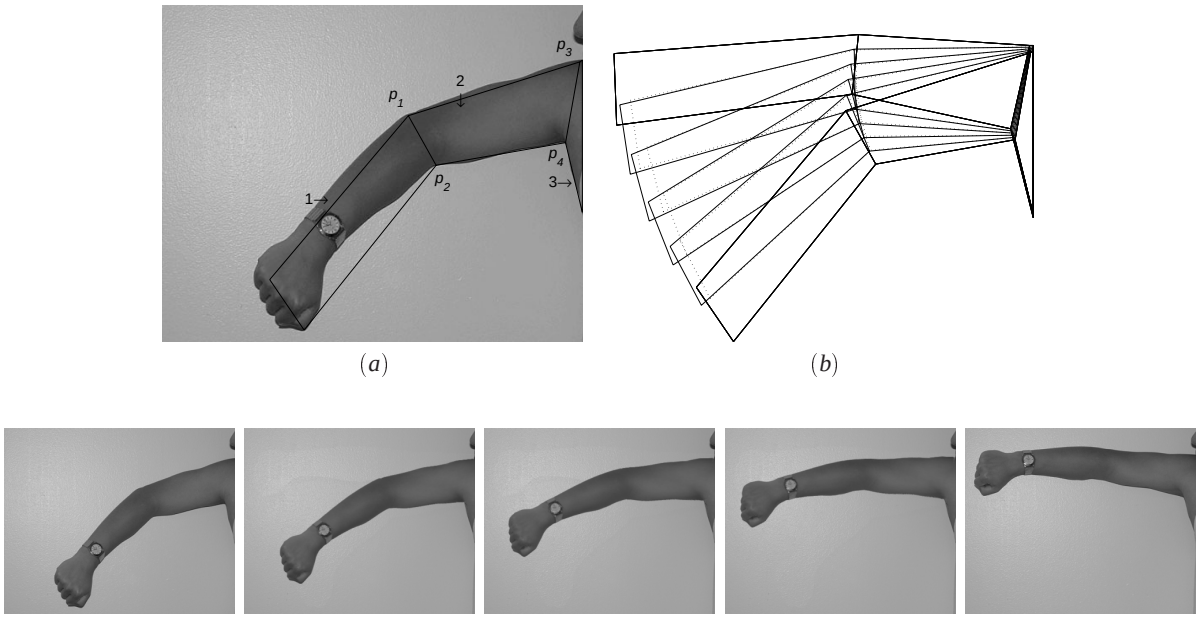

Figure 6: *Arm morphing. The lower arm rotates around the elbow, the upper arm rotates round the shoulder, and some apparent deformations can be seen at the joints. (a) Boundary connection constraint. (b) Postwarping path obtained by using the least distortion method (solid lines) and the linear interpolation method (dotted lines). In the bottom row, the first and last images are reference views; the other images are in-between views.*

though some parts may rotate or translate in different orientations or directions. First, we compute the centroid C_i for every part. The part's centroid C_A, which is close to object's centroid C, is fixed. Next, its neighboring part B's centroid C_B is adjusted by adding a vector v to every vertex of part B. v can be computed using their shared vertices $p_1, \cdots, p_k$ by Equation 5.

$$v = \frac{1}{k}\sum_{i=1}^{k}(p_{Ai} - p_{Bi}), \tag{5}$$

After all the centroids are adjusted, the final position of all original points are computed by averaging the corresponding adjusted vertices.

Figure 6.a shows a complicated case of a non-rigid object: a human arm moving from the lower to the upper part of the image. In this case, there are two rotations, one at the elbow and the other at the shoulder, and also some apparent deformations at the joints. We separate the images into lower arm (part 1), upper arm (part 2) and body (part 3). The points p_1 and p_2 are shared by parts 1 and 2; p_3 and p_4 are shared by parts 2 and 3. Figure 6.b shows the interpolation path obtained by two algorithms. The dashed lines show the paths generated by the linear algorithm; the solid lines show the paths generated by the least distortion algorithm using the connection constraint and the centroid adjustment. It is obvious that our method can overcome the shrinking during the interpolation in the presence of object rotation.

7. Experiments

Figure 7 shows the images of a box object, which is rotating around its symmetric axis and translating from right to left. Since the object has a big rotational motion, some parts are visible in both views, other parts are only visible in the left view, and the remaining parts are only visible in the right view. Therefore, we segment the object into different views, even though they have the same motion. In order to restore the relationship of these parts, we use their common boundary points to merge part 7.d with 7.c, and part 7.h with 7.g. In order to determine the parts' visibility, we use two conditions: depth order (foreground vs. background) and boundary information. Since the control points at the boundary can form a polygon or triangle, we use its normal to determine the visibility of the occluded parts. The final interpolation results are shown in Figure 8.

Figure 9 illustrates a dynamic view synthesis for human movement. In this experiment, a person is moving left to right with different postures, and the reference pictures are taken from two view points. For every part of the person, there are different rotations and translations. In order to obtain better results, we segment the person into eight layers: head, body, two upper arms, two lower arms, and two legs (left and right). For each part, we use the view morphing algorithm to obtain a new rectified image, and then use the least distortion method and boundary connection constraint

to determine the postwarping path to get the intermediate views as shown in Figure 9.b-e.

Figure 10 shows the dynamic view interpolation of human walking. Two reference images were taken from two view points when the person walked from right to left. During her walking, some part of her right leg was occluded by the right leg. Even more difficult problem was that her left and right legs switched order in the two key frames. Using our approach, we were able to overcome these problems and generate a reasonable image sequence without using human kinematic knowledge or a 3D model. Due to motion, disoccluded area is created in the background. The disoccluded area of one image can be filled by the pixels of the other image from the same area. However, if illumination conditions for two reference views are different, the shadows (ghost) will be observed in the intermediate views.

The video sequences of the above experiments can be found on our web site: http://www.cs.ucf.edu/ vision/projects/viewMorphing/.

8. Conclusion and Future Work

In this paper, we successfully generalized dynamic scene synthesis for scenes containing non-rigid objects. Our method can handle these cases mixed with dynamic rigid or non-rigid objects, including complicated objects such as humans. In our approach, if the non-rigid object can be segmented into several rigid parts, we can use the boundary information to connect adjacent parts during morphing. From only two reference views, we generate a series of continuous and realistic intermediate views without 3D knowledge.

Our approach provides a unique and reasonable solution for the dynamic scene. First, the least distortion method fully recovers one rotational component, R_z, and partially compensates the remaining two rotational components, R_x and R_y, by scaling-shearing matrix. Next, using boundary connection constraints with centroid adjustment, we can effectively compensate for the displacement due to rotation. Therefore the amount of shape shrinking is reduced. Moreover, due to the strength of the view morphing algorithm, our approach can preserve the realistic 3D effect for each layer.

In the future, we will combine kinematics to improve our approach and apply this algorithm to complicated movements such as running. Another issue for our future research is how to automatically recover the fundamental matrix from two image layers. Also, we would like to develop an algorithm to reduce the effect of shadow (ghost) on the disoccluded background region by removing the illumination component.

9. Acknowledgement

The research reported in this paper was partially supported by STRICOM under Contract No. N61339-02-C-0082. The content of the information herein does not necessarily reflect the position or the policy of the government, no official endorsement should be inferred.

References

1. S. Seitz, C. Dyer. View Morphing. *Proc. SIGGRAPH'96*, pp. 21–30, 1996. 2
2. S. Seitz. *Image-Based Transformation of viewpoint and scene appearance*. Dissertation, computer science, university of Wisconsin, 1997. 2
3. S. Chen, L. William. View Interpolation for Image Synthesis. *Proc. SIGGRAPH'93*, pp. 279–288, 1993. 2
4. R. Manning, C. Dyer. Interpolating View and Scene Motion by Dynamic View Morphing. *Proc. CVPR*, pp. 388–394, 1999. 2
5. R. Hartley. In defence of the 8-point algorithm. *Proc. Fifth Intl. Conference on Computer Vision*, pp. 1064–1070, 1995. 2
6. M. Alexa, D. Cohen-Or, D. Levin. As-Rigid-As-Possible Shape Interpolation. *Proc. SIGGRAPH'00*, pp.157–164, 2000. 1, 2, 4
7. T. Sederberg, P. Gao, G. Wang, H. Mu. 2-D Shape Blending: An Intrinsic Solution to the Vertex Path Problem. *Proc. SIGGRAPH'93*, pp.15–18, 1993. 1
8. Y. Otha, T. Kanade. Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming. *IEEE PAMI*, **7**(2), pp.139–154, March 1985. 2, 3
9. G. Wolberg. Image Morphing: a Survey. *The visual computer*, pp.360–372, 1998. 1
10. L. Zhao. *Dressed Human Modeling, Detection, and Parts Localization*. Dissertation, CMU, 2001. 3
11. S. Lee, K.Y. Chwa, J. Hahn, S. Shin. Image metamorphosis using snakes and free-form deformations. *Proc. SIGGRAPH'95*, pp. 439–448,1995. 1
12. O. Faugeras, L. Robert. What can two images tell us about a third one? *Proc. ECCV*, pp. 485–492, 1994. 2
13. Y. Wexler, A. Shashua. On the synthesis of dynamic scenes from reference views. *Proc. CVPR*, 2000. 2
14. P. Litwinowicz, L. Willians. Animating Images with Drawings. *Proc. SIGGRAPH'94*, 1994. 1
15. T. Beier, S. Neely. Feature-Based Image Matamorphosis. *Proc.SIGGRAPH'92*, pp35–42, 1992. 1
16. A. Tal, G. Elber. Image Morphing with Feature Preserving Texture. *Proc. EUROGRAPHICS'99*,**18**(3),pp. 339–348, 1999. 1
17. H. Johan, Y. Koiso, T. Nishita. Morphing Using Curves and Shape Interpolation Techniques. *Proc. of the Pacific Graphics 2000 (PG'00)*, 2000. 1

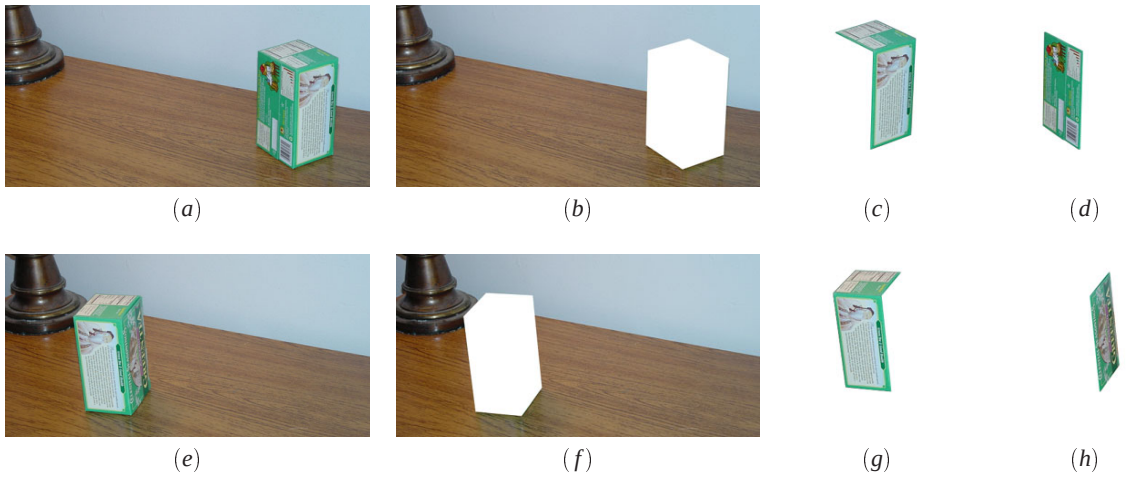

Figure 7: *Segmentation of two views of a box into three layers. (a)(e) two original views, (b)(f) background, (c)(g) the first part of the rotating object, (d) the second part of the rotating object, (h) the third part of the rotating object.*

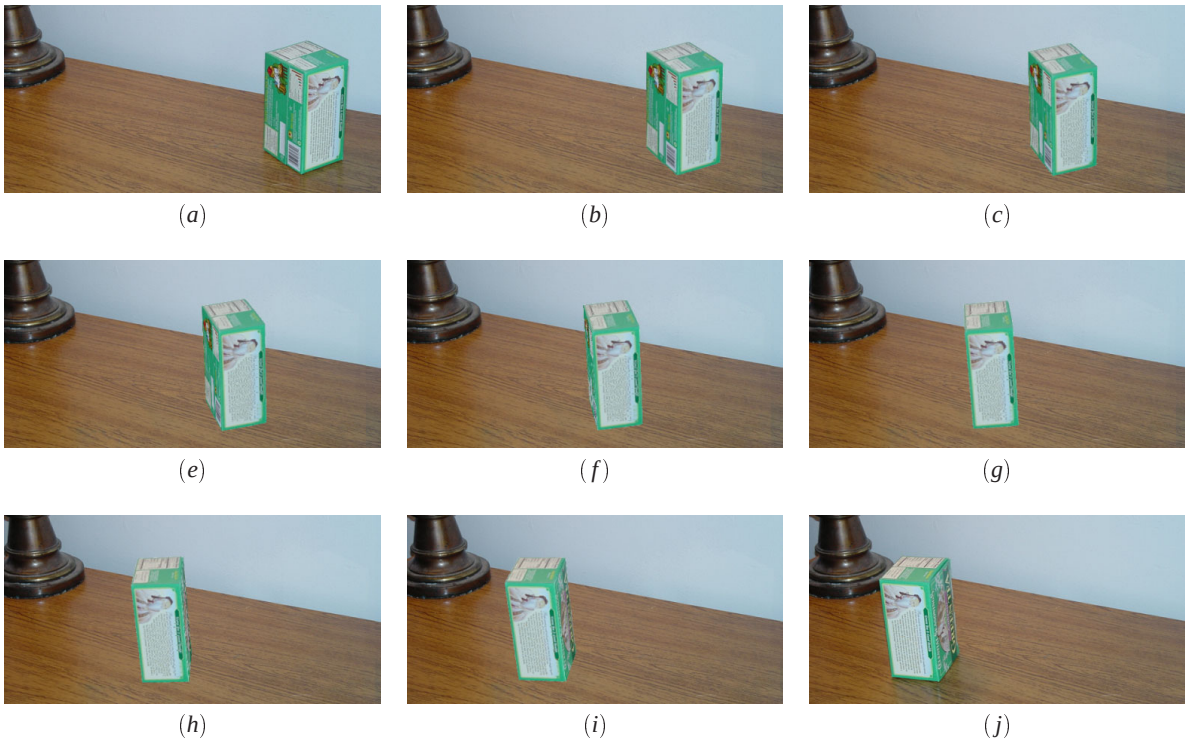

Figure 8: *Dynamic view interpolation of a rigid rotating object. (a) and (j) are reference images. Since the object has rotational motion, one side of the box is occluded by itself in (a) and (f), another side of the box is occluded by itself in (g) and (i). Our approach can correctly restore the geometric properties in the intermediate views (b)-(e).*

Figure 9: *Dynamic view interpolation of human motion. The first and last images are reference views. The person was moving from right to left with different postures, and the reference images were taken from different view points. Since the motion of the person is combined with multiple rotations and translations, we segmented the person into several parts based on the boundary connection constraint, and used our method to restore a reasonable postwarping path.*

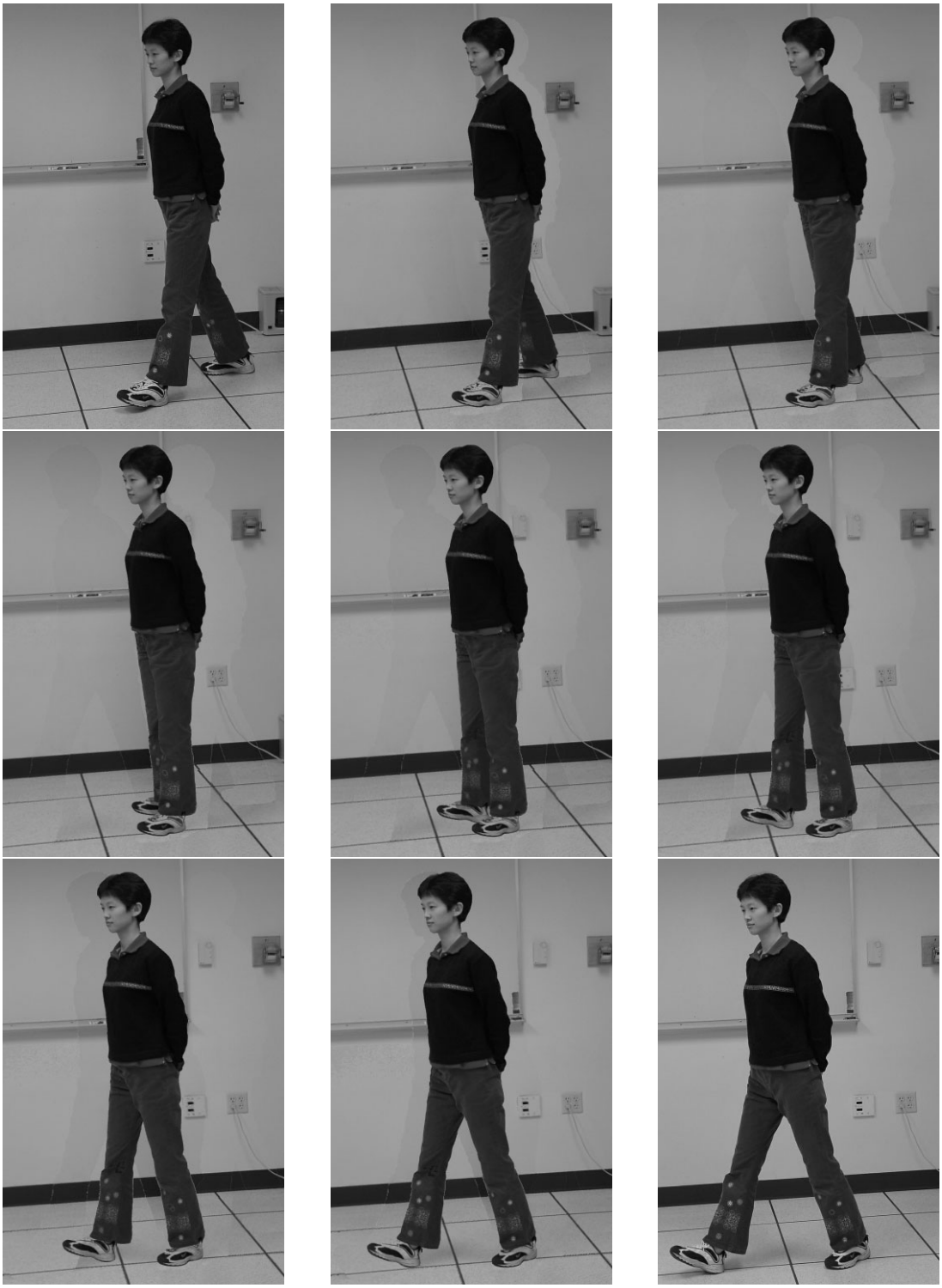

Figure 10: *Dynamic view interpolation of human walking. The first and last images are reference views which are taken from different view points. The person was walking from right to left. During her walking, some part of her right leg was occluded by the right leg, and the two legs switch order. We generated the intermediate images without using human kinematic knowledge or a 3D model.*