Real-time View Morphing for Web Applications

Mikio Terasawa*

Yasushi Yamaguchi†

Kinji Odaka‡

*College of Economics, Nihon University
1-3-2 Misaki-cho, Chiyoda-ku, Tokyo 101-8360 Japan
terasawa@eco.nihon-u.ac.jp
†Graduate School of Arts and Sciences, The University of Tokyo
3-8-1 Komaba, Meguro-ku, Tokyo 153-8902 Japan
yama@graco.c.u-tokyo.ac.jp
‡Nabla Inc.
EG Bldg., 3-3-2 Nakameguro, Meguro-ku, Tokyo 153-0061 Japan
kinji@nabla.co.jp

ABSTRACT

In this paper, real-time view morphing that is an extension of view morphing is proposed for web applications such as on-line shopping and remote instruction. Real-time view morphing is an image-based rendering method that generates an image of an intermediate view from two or more photographs in real-time without 3D models. The method is combined with conventional view morphing and 2D polygon rendering with texture mapping for real-time processing. Techniques to avoid discontinuity of texture and artifacts at contours without generating holes are proposed to keep the quality of original images. The advantages of the method are small data size, high image quality and real-time rendering that are important for web applications. Users can change object-centered viewpoints interactively in a web browser on a local machine that is connected to the narrow band Internet. The real-time view morphing program also runs in banner advertisements and desktop accessories. An editor is developed for preparing the data such as corresponding points, faces, and face order manually or semi-automatically using the epipolar constraint.

Keywords: image-based rendering, view morphing, epipolar geometry.

1 INTRODUCTION

It is important to look at an object from arbitrary viewpoints to see the detail of for web applications such as on-line shopping, remote instruction, and remote diagnosis. In addition to image quality and easy data acquisition that are required in general computer graphics applications, data size and fast interaction is crucial for the web applications because they have to run on standard PCs that are connected to the narrow band Internet in general. The data to be transmitted, therefore, should be as less as possible.

In order to realize an object-centered view, two approaches can be taken. One is a geometry-based approach and the other is an image-based approach. A major geometry-based approach is so called Web3D that uses 3D models and is typically described by VRML. Users can rotate, translate and deform 3D objects as well as change color and/or texture. However, it is not suitable for personal applications such as on-line auction because realistic 3D models require either much labor to create or special devices to directly obtain 3D data such as 3D scanners.

A hybrid method[Debevec96] that generates 3D models from images and basic models can reduce the labor though the application is limited to ar-

chitectures whose shapes can be approximated by simple primitives, e.g., blocks and wedges. A relief texture method[Oliveira00] can reduce the number of 3D faces and improve the image quality by mapping pre-warped texture images.

The image-based approach is a more convenient way for data generation because photographs of real objects are directly used. Light field[Levoy96] and Lumigraph[Gortler96] can reconstruct images at various view points for both real and virtual scenes by storing the intensity of light beams in 3D space rather than 3D geometric information. However, the data size is too large for web applications. If the cameras are calibrated, the geometry of objects is automatically reconstructed from two or more images[Maybank92][Luong97]. The 3D objects can be obtained in Virtualized Reality[Rander97] using a set of calibrated cameras. A faster image-based reconstruction method is proposed using a set of contour images[Matusik00], although it cannot deal with concave objects. An image-based system is developed[Oh01] for generating new images from a single photograph by manual interaction.

Most of other approaches are based on the stereo vision techniques. View interpolation[Chen93] generates an image of an intermediate view as a set of blocks representing the pixel of the source images without 3D models. The blocks are computed using a quadtree subdivision while Bao et al. modified it by a binary decomposition[Bao98] to fill the holes. The framework of view interpolation and other image-based approaches are uniformly represented as plenoptic modeling[McMillan95a]. The method requires many images to compute the dense correspondence of the pixels. The cameras must be calibrated for automatic pixel correspondence. The assumption of calibration, however, is not practical in general web applications since photographs are generally taken by a hand-held camera whose focal length is not constant.

As an uncalibrated method, the most popular commercial product is QuickTime VR[Chen95] that uses panoramic images to obtain new images of different views. The speed of rendering and downloading is fast because it uses a single stitched image. A robust stitching algorithm with a few parameters is proposed to generate mosaics[Szeliski97]. The object-centered concentric mosaic[Shum99] is useful for web applications. However, the quality of stitched images is not satisfactory compared to the original images.

View transfer[Laveau94] generates an intermediate image from widely separated object-centered views under the epipolar constraint[Faugeras93]. In the case of two images, the distortion may occur because of the degrees of freedom for the projective reconstruction.

View morphing[Seitz96] also generates an image of an intermediate view under the epipolar constraint. It is useful for web applications as for the data size because only two images of widely separated views are necessary. However, the image morphing[Beier 92] that is used in the process of view morphing is time consuming because it is proportional to the number of the feature lines and the pixels. Moreover, it is not easy to specify the proper values of weight factors because of the distortion in the pre-warped images.

Therefore, we modified the image morphing process to 2D polygon rendering with texture mapping for real-time processing while keeping the image quality of original photographs. It is suitable for web applications due to easy preparation, small data to be transferred, and rendering. Two-dimensional faces are defined for texture mapping. Since the occlusion is inevitable in view morphing because of the widely separated views, the faces that are seen only in one image should be defined to avoid the holes seen in the pixel-based methods such as view interpolation. The drawing order of the faces is also important because the faces have no depth information and the sorting methods for dense correspondence[McMillan95b][Fu98] are no longer available. We have developed an interactive system to define faces to deal with the problems. Our system enables users look at an object from more than two viewpoints by applying view morphing for each pair of two images.

Section2 explains the algorithm of real-time view morphing and rendering techniques to improve image quality. Section 3 describes how to generate the data manually or semi-automatically. Section4 shows examples of a real-time view morphing system that runs in a web browser.

2 REAL-TIME VIEW MORPHING

2.1 Overview

We aimed at extending the view morphing for real-time processing. We call our method real-time view morphing. In real-time view morphing, objects are represented by faces rather than by feature lines. Figure1 shows an illustrative example of input and output for real-time view morphing. Faces are defined on the two source images

as shown in the figure. One thing we must note is that the faces are defined in 2D space, namely in images, though the faces in Figure 1 look like forming a 3D shape of a car. The resulting image is generated by combining the faces with texture. Notice that some of the faces defined in the front and back part of the car are invisible. The invisible faces prevent holes caused by the occlusion that may appear in the pixel-based methods. The drawing order of faces is determined by using the epipolar geometry.

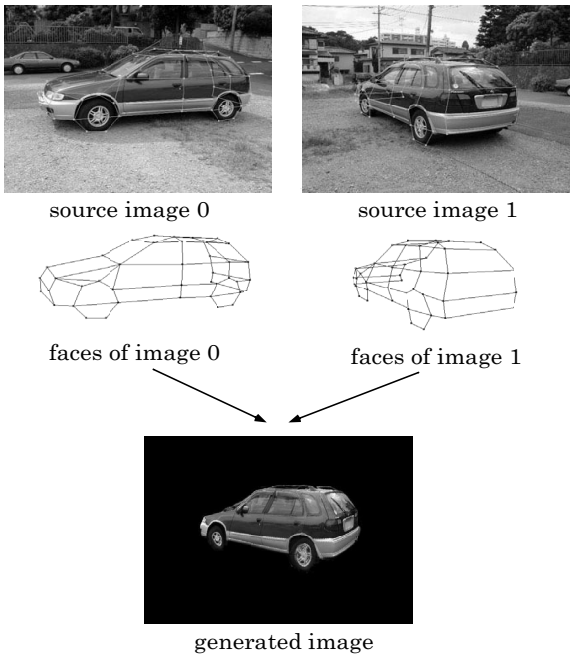


Figure 1: Image and face definition and an interpolated image

The texture images are blended in advance to avoid the discontinuity at the face boundaries. The view morphing transformation is applied to each vertex of the faces. Texture images on the faces are mapped after all vertices are transformed. Texture mapping, which is a 2D-to-2D mapping instead of an ordinary 2D-to-3D mapping, is performed in real-time because it is a standard function of graphic libraries with or without help of graphics hardware. If there are more than two images, the algorithm is applied for each pair of two neighboring images.

The data of the faces and other information is prepared in an off-line process and stored in a web server, while the real-time view morphing process runs in a web browser of a local machine.

2.2 Interpolation of vertices

The problem is to find an intermediate point in an image I_s from the points in two source images I_0 and I_1. Let the points in homogeneous coordinates in images I_0, I_1 and I_s that correspond to one point of an object be $\boldsymbol{p}_0$, $\boldsymbol{p}_1$ and $\boldsymbol{p}_s$ respectively(Figure 2). The points in the source images $\boldsymbol{p}_0$ and $\boldsymbol{p}_1$ are known while $\boldsymbol{p}_s$ is to be calculated.

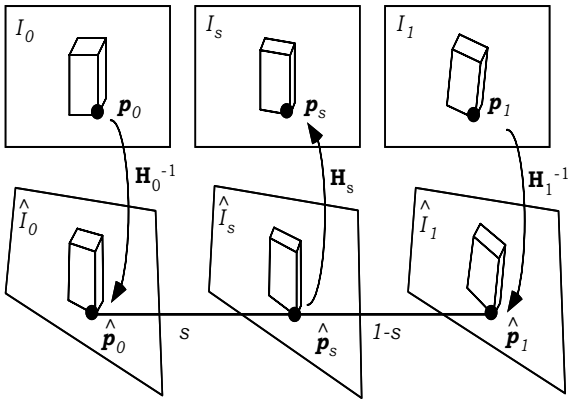


Figure 2: Interpolation of a vertex

The first step of the algorithm is to rectify the images. The rectification transforms the source images to the images whose corresponding points lie on the same scan line. The 3×3 projective transformations $\mathbf{H}_0$ and $\mathbf{H}_1$ for the rectification are represented as a multiplication of rotation matrices and a translation matrix for a stereo view. They are computed from a fundamental matrix and epipoles that are determined by eight or more sets of corresponding points[Hartley97]. Several linear and non-linear algorithms are proposed for robust solution of a fundamental matrix[Luong93]. The points $\boldsymbol{p}_0$ and $\boldsymbol{p}_1$ in the source images I_0 and I_1 are transformed to the points $\hat{\boldsymbol{p}}_0$ and $\hat{\boldsymbol{p}}_1$ in the rectified images $\hat{I}_0$ and $\hat{I}_1$ by Eq (1).

$$\begin{aligned} \hat{\boldsymbol{p}}_0 &= \mathbf{H}_0^{-1}\boldsymbol{p}_0, \\ \hat{\boldsymbol{p}}_1 &= \mathbf{H}_1^{-1}\boldsymbol{p}_1. \end{aligned} \tag{1}$$

The second step is to interpolate an intermediate point in the intermediate rectified image by the ratio of s $(0 \leq s \leq 1)$.

$$\hat{\boldsymbol{p}}_s = (1-s)\hat{\boldsymbol{p}}_0 + s\hat{\boldsymbol{p}}_1. \tag{2}$$

The third step computes an intermediate projective transformation $\mathbf{H}_s$. It is approximated by the linear interpolation of the boundary points as done by Seitz and Dyer. The fourth step computes the intermediate point $\boldsymbol{p}_s$ by $\boldsymbol{p}_s = \mathbf{H}_s\hat{\boldsymbol{p}}_s$.

Finally, the texture image is mapped to the intermediate faces after intermediate points are computed for all vertices of the faces.

2.3 Face information

A face is defined by a list of 2D vertices in counter-clockwise order with a texture image. A face is divided into convex sub-faces internally for texture blending and hardware texture mapping. Each face is associated with only one texture. A face, however, usually have two possible texture images derived from the source images. Thus, we have to carefully define a texture image in order to accomplish a high quality result.

We defined two criteria to determine which image should be used. The first criterion is the direction of the face. If one of the two corresponding faces is backward, i.e. the vertices of the face are traced in clockwise order, the image of the forward face is chosen as a texture image. The image I_1 is used for texture mapping in the case of face1 in Figure3.

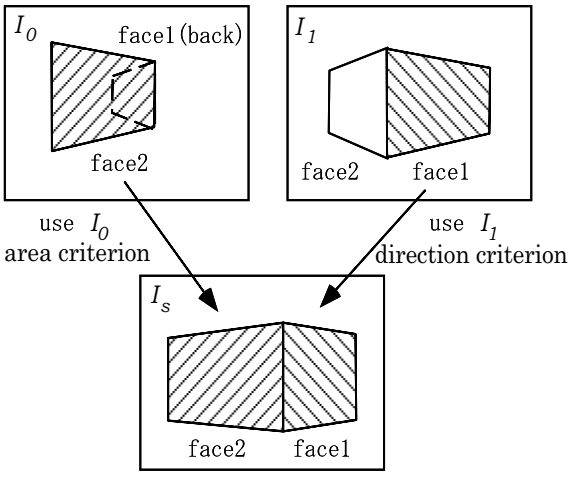


Figure 3: Criteria of texture reference

The second criterion is the area of the face. If both of the faces are oriented forward, a texture image of a larger face is chosen. The image I_0 is used for texture mapping in the case of face2 in Figure 3. The area criterion is useful to keep the quality of images because an image of better resolution is used.

Another important aspect of faces is drawing order. Since real-time view morphing deals with only two-dimensional information, we cannot expect depth sorting for hidden surface removal. We used the epipoles of the images to determine the drawing order. The depth information is approximated by the distance of the vertices of the faces from the epipole because the epipole indicates the position of a viewpoint that is projected to the other image. In Figure4, $\boldsymbol{v}_0$, $\boldsymbol{v}_1$ represent the virtual viewpoints and $\boldsymbol{e}_0$, $\boldsymbol{e}_1$ represent the epipoles for images I_0 and I_1 respectively. In this case, faces are drawn in the order of F_0, F_1 and F_2 as a result of comparison of the distance d_0, d_1 and d_2. If occlusion occurs, the ordering is decided locally in the image where the faces are visible. Since this method is approximation, the exact order may be modified manually.

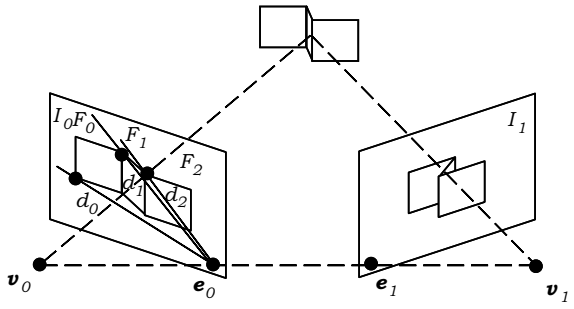


Figure 4: Drawing order of faces

2.4 Blending texture images

As a result of the texture selection described in Section2.3, adjacent faces may use texture images from the different source images. It will cause the discontinuity of images at the face boundaries as shown in Figure5(1). The color of the two texture images is smoothly blended to avoid the discontinuity. The texture modification is a pre-process that are performed only once. The modified texture is used if the adjacent face uses the texture of the other image.

Assume that an image I_0 is chosen as the texture for a face. The modified texture color $\boldsymbol{C}'_0(\boldsymbol{p}_0)$ of each pixel $\boldsymbol{p}_0$ inside the face is linearly interpolated by the color of the source images I_0 and I_1, which are written as $\boldsymbol{C}_0(\boldsymbol{p}_0)$ and $\boldsymbol{C}_1(\boldsymbol{p}_1)$ respectively.

$$\boldsymbol{C}'_0(\boldsymbol{p}_0) = (1-w)\boldsymbol{C}_0(\boldsymbol{p}_0) + w\boldsymbol{C}_1(\boldsymbol{p}_1). \qquad (3)$$

The total weight w that has influence on the pixel is obtained by $w = \min\left(\sum_{i \in A} w_i, 1\right)$ where A is a set of edges shared by two faces with different texture images and w_i is defined by Eq (4).

$$w_i = \begin{cases} \frac{1}{2}\left(1 - \frac{d_i}{r_i}\right) & 0 \le d_i < r_i \\ 0 & d_i \ge r_i \end{cases} \qquad (4)$$

where d_i is the distance between the pixel and an edge i and r_i is the distance between the center of the face and an edge i. The result of the blending procedure is shown in Figure 5(2).

(1) without blending (2) with blending

Figure 5: Effect of texture blending

2.5 Offset procedure at contours

Since real-time view morphing is polygon-based texture mapping, undesirable artifacts may occur at the contours as in 3D polygon rendering. Offset faces with alpha masks are attached to avoid the artifacts. An offset face is located along the contour edge that belongs to only one visible face(Figure 6). The direction to be pulled from the edge is determined by a vertex offset vector that is an average of edge offset vectors. An edge offset vector is perpendicular to the edge and oriented outside from the face center. The length of the vertex offset vector is specified by users to contain the object. The offset faces are drawn before the ordinary faces.

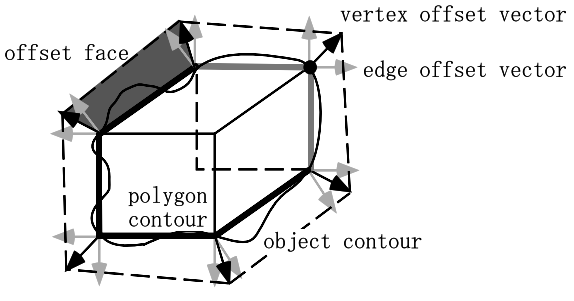


Figure 6: Offset faces at contours

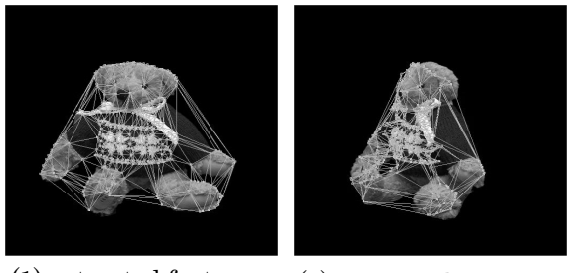
Although only the vertices are subject to the epipolar constraint, the distortion inside the offset faces is small because they are close to the edges.

3 DATA PREPARATION

3.1 Semi-automatic generation of faces

We have developed a real-time view morphing system that consists of an editor and a viewer. In the editor, the faces can be generated semi-automatically using the epipolar constraint and pattern matching when the occlusion is negligible. First of all, eight or more pairs of corresponding points are computed for the source images using a correlation method similar to Zhang's[Zhang94] for weak calibration, which is to calculate the fundamental matrix and the epipoles. Our system also allows users to specify those corresponding points by mouse operation that is usually more stable and faster than the automatic pattern matching process.

The feature points are extracted in one of the two images as shown in Figure 7(1) to define the vertices of the faces. In this implementation, the Susan operator[Steve97] is used for the feature extraction. It is fast and easy to use because the feature points are found by local masking operation. Then, the corresponding vertices in the other image are computed using the epipolar constraint and local pattern matching in a search window. The result is shown in Figure 7(2). If there are three images, the third corresponding vertex is analytically obtained by the epipolar constraints from the two corresponding vertices and the two fundamental matrices though some error correction is necessary. The face topologies are created



(1) extracted feature points and faces (2) computed corresponding points

Figure 7: Face generation

by spanning edges between the points using an incremental Delaunay triangulation[Lischinski94] in the visible image. The face is sorted by the distance from the epipoles for the hidden surface removal as stated in Section 2.3.

3.2 Manual Interaction

Since the algorithm uses pattern matching of images, it cannot deal with the case of occlusion

that occurs in general examples. In such a case, users use the semi-automatic process partly and modify the undesirable vertices manually.

Users can specify faces and additional information for the source images with the mouse operation in the editor. The data includes corresponding points, vertices, faces, drawing order, texture blending parameters, and contour offset.

The fundamental matrix and the epipoles points help users to specify the corresponding points of the faces because the points are constrained on the epipolar lines.

When two or more forward faces overlap, the image-based algorithm is not available any more. In that case, users specify layer groups of the vertices manually and apply an incremental Delaunay triangulation to generate faces and the method to determine the drawing order layer by layer. The editor shown in Figure 8 supports the manual interaction.

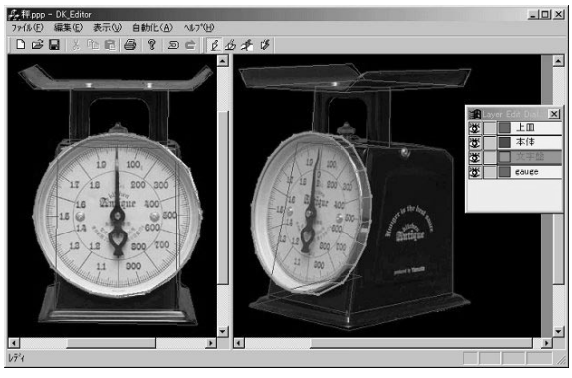


Figure 8: Specification of drawing order by a layer structure

4 EXAMPLES

The viewer of our system runs in a general region of a web browser such as Internet Explorer on a local machine. There are an ActiveX version and a Java version. The file size of the ActiveX version is currently 348KB.

An example of a torso is shown in Figure 9. A user can rotate and scale the torso continuously by the mouse operation. The source pictures are nine JPEG images of 256×512 that are taken from the surrounding nine viewpoints by an uncalibrated hand-held camera. About 170 faces are defined in each pair of the two images. The total file size of the nine images is 75.3KB while the additional data is 64.6KB. The additional information could be reduced because it is represented in the ASCII format in our current implementation. However, the total file size 140KB is practically acceptable for the use of the narrow band Internet. It takes 0.027 seconds per frame for the view morphing transformation and rendering by Pentium III 866MHz with a GeForce2 accelerator, while it takes more than 5 minutes by conventional view morphing.

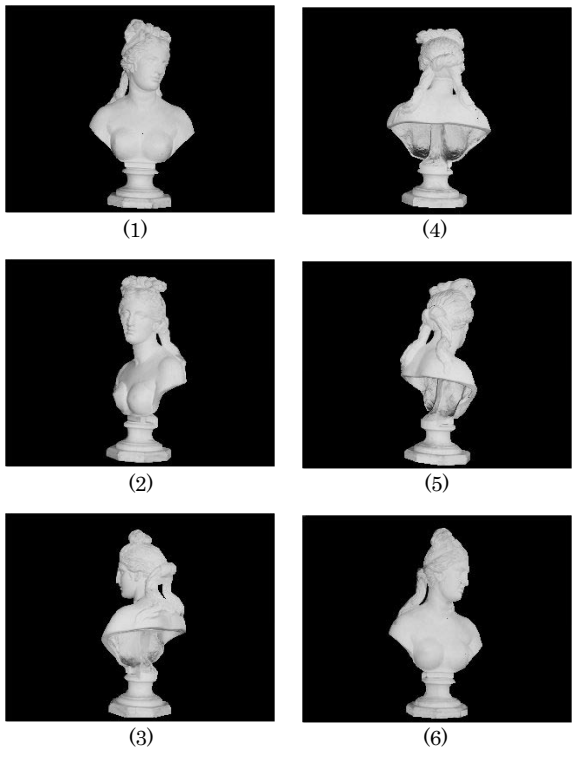


Figure 9: Example of a torso

Figure 10 shows an example of a human. The total file size of the eight 512×512 images is 75.6KB. The additional data is 41.6KB for about 290 faces. The offset procedure at the contours effectively shows the detail of hair and fingers that are difficult to represent by 3D models.

In addition to the general region of a web browser, the viewer runs in a region of a banner advertisement as shown in Figure 11. Users can rotate objects inside the banner advertisement interactively. The ActiveX version also runs on a desktop as shown in Figure 12.

The pre-process of texture blending is not effective for shiny or glossy objects. In that case, users can choose an overlapping mode that blends the original texture images by the interpolation ratio instead of blending the images in advance.

The number of pictures depends on the complex-

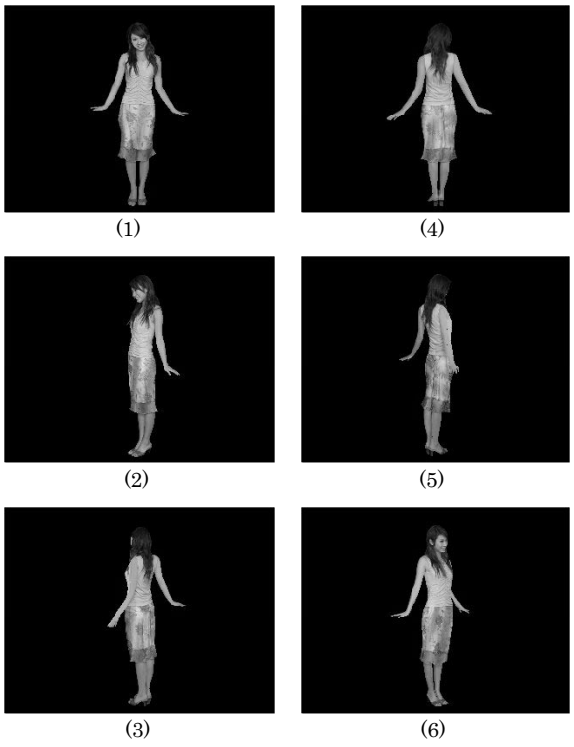


Figure 10: Example of a real human

Figure 11: Application to a banner advertisement

Figure 12: Application to a desktop accessory

ity of objects for reconstructing images looked from anywhere around them. If the objects are simple with little occlusion, small number, i.e., four to six, of images should be enough. Otherwise, larger number, i.e., ten to twelve, of images or much manual interaction will be necessary.

5 CONCLUSION

We proposed an image-based rendering method, which is called real-time view morphing, for interpolating object-centered views from two or more photographs. It is suitable for web applications because of small data size and fast rendering. Users can feel three-dimensional view motion around an object by applying pair-wise view morphing although the procedure is two-dimensional. Furthermore, it is useful because it utilizes photographs that are taken by uncalibrated hand-held cameras without any special devices or markers. It is applicable to many practical applications such as on-line shopping, remote instruction, and remote diagnosis. A tentative virtual shopping mall that is based on real-time view morphing is available at `http://www.nabla.co.jp/PhotoPopper.html`.

An intermediate viewpoint is limited on a curve that connects the two viewpoints by a straightforward implementation of real-time view morphing. The extension for interpolating three viewpoints at the same time is now under development.

The proposed method can generate faces semi-automatically if eight or more sets of corresponding points are specified. However, the method cannot deal with the case of occlusion completely because it is based on an image matching technique. A stable method to deal with the occlusion has to be developed for practical usage because the occlusion is inevitable in widely separated views.

ACKNOWLEDGEMENT

This research was supported by research and development fund for advanced technology of TAO. A part of the research was supported by basic research 21 for breakthroughs in infocommunications of ministry of public management, home affairs, posts and telecommunications.

REFERENCES

[Bao98] Bao, H., Chen, L., Ying, J. and Peng, Y.: Nonlinear View Interpolation, Proc. Pacific Graphics 98, pp. 61–68, 1998.

[Beier 92] Beier, T. and Neely, S.: Feature-based Image Metamorphosis, Proc. SIGGRAPH92, pp. 35–42, 1992.

[Chen93] Chen, S. E. and Williams, L.: View Interpolation for Image Synthesis, Computer Graphics Proceedings, Annual Conference Series, pp. 279–288, 1993.

[Chen95] Chen, S. E.: QuickTime V R — An Image-based Approach to Virtual Environment Navigation, Computer Graphics Proceedings, Annual Conference Series, pp. 29–38, 1995.

[Debevec96] Debevec, P. E., Tayler, C. J. and Malik J.: Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-based Approach, Computer Graphics Proceedings, Annual Conference Series, pp. 11–20, 1996.

[Faugeras93] Faugeras, O. D.: Three-dimensional Computer Vision: A Geometric Viewpoint, MIT Press, 1993.

[Fu98] Fu, C., Wong, T. and Heng, P.: Triangle-based View Interpolation without Depth-buffering, Journal of Graphics Tools, Vol. 3, No. 4, pp. 13–31, 1998.

[Gortler96] Gortler, S. J., Grzeszczuk, R., Szeliski R. and Cohen, M. F.: Lumigraph, Computer Graphics Proceedings, Annual Conference Series, pp. 43–54, 1996.

[Hartley97] Hartley, R. I.: In Defense of the Eight-point Algorithm, Pattern Analysis and Machine Intelligence, Vol. 19, No. 6, pp. 580–593, 1997.

[Laveau94] Laveau, S. and Faugeras, O. D.: 3-D Scene Representation as a Collection of Images and Fundamental Matrices, INRIA Report, No. 2205, pp. 1–25, 1994.

[Levoy96] Levoy, M. and Hanrahan, P.: Light Field Rendering, Computer Graphics Proceedings, Annual Conference Series, pp. 31–42, 1996.

[Lischinski94] Lischinski, D.: Incremental Delaunay Triangulation in Graphics Gems IV, AP Professional, 1994.

[Luong93] Luong, Q. T., Deriche, R., Faugeras, O. D. and Papadopoulo, T.: On Determining the Fundamental Matrix: Analysis of Different Methods and Experimental Results, INRIA Report, No. 1894, pp. 1–28, 1993.

[Luong97] Luong, Q. T. and Faugeras, O. D.: Self-calibration of a Moving Camera from Point Correspondences and Fundamental Matrix, International Journal of Computer Vision, Vol. 22, No. 3, pp. 261–289, 1997.

[Maybank92] Maybank, S. J. and Faugeras, O. D.: A Theory of Self-calibration of a Moving Camera, International Journal of Computer Vision, Vol. 8, No. 2, pp. 123–152, 1992.

[Matusik00] Matusik, W., Buehler, C., Raskar, R. Gortler, S. J. and McMillan L.: Image-Based Visual Hulls, Computer Graphics Proceedings, Annual Conference Series, pp. 369–374, 2000.

[McMillan95a] McMillan, L. and Bishop, G.: Plenoptic Modeling: An Image-based Rendering System, Computer Graphics Proceedings, Annual Conference Series, pp. 39–46, 1995.

[McMillan95b] McMillan, L.: Computing Visibility without Depth, UNC Technical Report TR95-047, Univ. North Carolina, 1995.

[Oh01] Oh, B.M., Chen, M., Dorsey, J. and Durand, F.: Image-Based Modeling and Photo Editing, Computer Graphics Proceedings, Annual Conference Series, pp. 433–442, 2001.

[Oliveira00] Oliveira, M. M., Bishop, G. and McAllister, D.: Relief Texture Mapping, Computer Graphics Proceedings, Annual Conference Series, pp. 359–368, 2000.

[Rander97] Rander, P., Narayanan, P.J. and Kanade, T.: Virtualized Reality: Constructing Time-Varying Virtual Worlds from Real Events, Proc. IEEE Visualization'97, pp. 277-283, 1997.

[Seitz96] Seitz, S. M. and Dyer, C. R.: View Morphing, Computer Graphics Proceedings, Annual Conference Series, pp. 21–30, 1996.

[Shum99] Shum, H.Y. and He, L.W.: Rendering with Concentric Mosaics, Computer Graphics Proceedings, Annual Conference Series, pp. 299–306, 1999.

[Steve97] Steve, S. M. and Brady, M.: SUSAN — A New Approach to Low Level Image Processing, International Journal of Computer Vision, Vol. 23, No. 1, pp. 45–78, 1997.

[Szeliski97] Szeliski, R. and Shum, H. Y.: Creating Full View Panoramic Image Mosaics and Environment Maps, Computer Graphics Proceedings, Annual Conference Series, pp. 251–258, 1997.

[Zhang94] Zhang, Z., Deriche, R., Faugeras, O. and Luong, Q. T.: A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry, INRIA Report, No. 2273, pp. 1–38, 1994.