International Conference on Image Processing (ICIP 97), vol. 1, pp. 444-447, Santa Barbara, October 1997.

Model-based Synthetic View Generation from a Monocular Video Sequence

Chun-Jen Tsai*, Peter Eisert+, Bernd Girod+, Aggelos K. Katsaggelos

Electrical and Computer Engineering
Northwestern University, Evanston, IL, USA
tsai, aggk@ece.nwu.edu

+Telecommunications Institute
University of Erlangen-Nuremberg, Erlangen, Germany
eisert, girod@nt.e-technik.uni-erlangen.de

Abstract

In this paper a model-based multi-view image generation system for video conferencing is presented. The system assumes that a 3-D model of the person in front of the camera is available. It extracts texture from a speaking person sequence images and maps it to the static 3-D model during the video conference session. Since only the incrementally updated texture information is transmitted during the whole session, the bandwidth requirement is very small. Based on the experimental results one can conclude that the proposed system is very promising for practical applications.

1 Introduction

A number of researchers have been working on synthetic multi-view image generation in the past few years [1, 2]. There are several potential applications of this research. For example, the elimination of inappropriate eye-contact in video conference systems [3], the production of depth perception for stereoscopic displays [4], object-based video coding/decoding [5], etc. Most of the recent investigations use two cameras for image acquisition. A disparity estimation process is then applied to the acquired two-channel video sequence. Finally, virtual images are synthesized by interpolating between the two image channels. With such an approach, the permissible viewing angle of the virtual scene is constrained to be in between the two real cameras. The allowable viewing angle could be increased using extrapolation. However, the quality of an extrapolated virtual scene is usually unacceptable when the viewing position is very different from the positions of the real cameras. In addition to the viewing angle problem, another issue for a binocular synthetic multi-view image generation system is the efficient transmission of the virtual scenes across a communication channel with low bit rate [6].

In this paper, a different approach is investigated. It uses a 3-D model and a monocular video sequence. Assuming that 3-D models of the objects in the virtual scene are available, we can perform the task of multi-view image generation with the use of only one camera. This approach, in comparison with the previous schemes, has the advantages of larger permissible viewing angle of virtual scenes, lower transmission bandwidth requirement, and higher virtual image quality. As can be seen from the experiments, this method is very promising for applications where 3-D models of the underlying scenes can be obtained in advance.

The paper is outlined as follows: A general introduction to model-based synthetic view generation is given in section 2. The algorithm for aligning the model with a camera image is explained in section 3. Section 4 describes the techniques for texture composition, section 5 shows some experimental results, and conclusions are given in section 6.

2 Model-based Synthetic View Generation

As mentioned in the introduction, a major drawback of the current two-camera synthetic view generation approaches is that the permissible viewing angles of the synthesized images are very limited. In addition, the continuous transmission of the two-channel image sequence and/or the disparity fields over the communication channel during an application session is very bandwidth-consuming.

However, in some applications, we can obtain the 3-D model of the target scene. For example, in a video conferencing scenario, assuming that the 3-D model of the person in front of the camera is known and transmitted to the receiver before the session, it is possible to generate synthetic views based on only a monocular image sequence. With this approach, there is no need to modify the existing video conference systems, and the required bit-rate is much lower than with previous techniques. Our new approach can be divided into two parts. The first part is the alignment of the 3-D model with each of the 2-D camera images so that the texture on each 2-D image can be mapped to the 3-D model. Due to different shadowing and alignment between every two camera images, a straightforward mapping of two textures on the 3-D model produces

*Work carried out with support of Graduiertenkolleg 3D Image Analysis and Synthesis, University of Erlangen-Nuremberg, Germany.

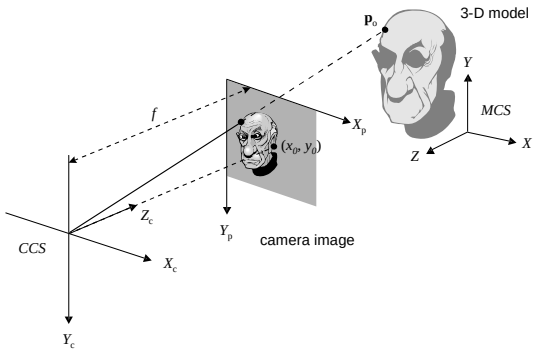

Figure 1: The alignment of the model and a camera image

some artifacts along the boundary. The second part in our approach is a combination of textures from different camera images so that the final texture map of the 3-D model produces convincing synthetic views.

3 Texture Mapping from a Camera Image to the 3-D Model

Before mapping the object texture in a camera image to the 3-D model, we must align the 3-D model with the camera image so that the texture can be back-projected from the camera lens center to the 3-D model pixel-by-pixel (see Figure 1). This can be done by selecting some feature points in the 3-D model and aligning these feature points with the corresponding 2-D feature points. It has been pointed out that the selection and tracking of a few points in a sequence of camera images is feasible and can be done efficiently in practice [7, 8]. There is also an increasing interest in head-tracking algorithms for multimedia applications. Since the main purpose of this paper is to investigate the mapping of a sequence of a talking person images to a static 3-D model, we assume that the locating and tracking of a few 2-D feature points (like eyes, nostrils, mouth tips, etc.) has been implemented in some practical way.

In this study, a 3-D model is described by a triangle mesh. The triangles used in the model are described with respect to a Model Coordinate System (MCS), while the camera images are defined in the Camera Coordinate System (CCS). The problem now is to find the appropriate rotation matrix (denoted by R) and translation vector (denoted by T) between MCS and CCS so that the (known) correspondence is established. This problem is similar to the camera calibration problem. The transformation of an object point $\mathbf{p}_o = (x, y, z)^T$ into camera coordinates is given by:

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = R \begin{pmatrix} x \\ y \\ z \end{pmatrix} + T. \qquad (1)$$

The imaging process then records the texture at point $\mathbf{p}_o$ on the image plane at point $(x_p, y_p)^T$ via a perspective transformation:

$$x_p = -f_x \frac{x_c}{z_c} + x_0 \quad y_p = f_y \frac{y_c}{z_c} + y_0. \qquad (2)$$

Here, f_x and f_y are the focal lengths measured by the size of the CCD array pixels in X and Y directions, respectively. $(x_0, y_0)^T$ is the piercing point of the optical axis on the CCD array chip. f_x, f_y, x_0, and y_0 are the intrinsic camera parameters. They can be estimated by camera calibration before we solve the alignment problem. By combining equation (1) and equation (2), we have

$$\begin{aligned} \frac{x_p - x_0}{f_x} &= -\frac{x_c}{z_c} = -\frac{R_1^T \cdot \mathbf{p}_o + t_x}{R_3^T \cdot \mathbf{p}_o + t_z} \\ \frac{y_p - y_0}{f_y} &= \frac{y_c}{z_c} = \frac{R_2^T \cdot \mathbf{p}_o + t_y}{R_3^T \cdot \mathbf{p}_o + t_z}, \end{aligned} \qquad (3)$$

where R_i is the ith row of the rotation matrix R, and t_x, t_y, t_z are the components in the translation vector T.

To compute the transformation from the MCS to CCS, we can rewrite equation (3) and obtain matrices A and x as follows:

$$\begin{aligned} A &= \begin{pmatrix} f_x x & f_x y & f_x z & 0 & 0 & 0 \\ 0 & 0 & 0 & -f_y x & -f_y y & -f_y z \\ (x_p - x_0)x & (x_p - x_0)y & (x_p - x_0)z & & & \\ (y_p - y_0)x & (y_p - y_0)y & (y_p - y_0)z & & & \\ f_x & 0 & x_p - x_0 & & & \\ 0 & -f_y & y_p - y_0 & & & \end{pmatrix} \\ x &= \begin{pmatrix} r_{11} & r_{12} & \dots & r_{33} & t_x & t_y & t_z \end{pmatrix}^T. \end{aligned} \qquad (4)$$

In equation (4), $r_{11}, ..., r_{33}$ are the entries in the rotation matrix R. Although we have 12 entries here, the degree of freedom for a rotation matrix is only 6. Each corresponding pair of 2-D and 3-D points, $(x_p, y_p)^T$ and $(x, y, z)^T$, give rise to a 2×12 system:

$$Ax = 0 \qquad (5)$$

By substituting the Euler angle rotation entries into (r_{11}, $\dots, r_{33}$), we have a nonlinear system with 6 unknowns, namely, $(\phi, \theta, \psi, t_x, t_y, t_z)$. Based on (5), we can define the following nonlinear cost function for alignment error:

$$F(\phi, \theta, \psi, t_x, t_y, t_z) = \|Ax\|^2. \qquad (6)$$

The minimization of equation (6) gives us the rotation and translation parameters needed to align the model with the camera image. We used the simplex routine in Matlab to minimize equation (6) because we usually have a good initial guess of the alignment parameters in a video conference scene. The simplex method almost always converges to the correct values for $(\phi, \theta, \psi, t_x, t_y, t_z)$. Figure 2 shows some examples of the model and the camera image after alignment.

4 Texture Composition with Multiple Camera Images

Because we do not assume ambient lighting conditions and because there are always some errors in the alignment parameters estimation, artifacts like inconsistent colors on the skin or mapping of the background texture on the face arise when one tries to put the textures in different camera images on the 3-D model. The techniques we use in the

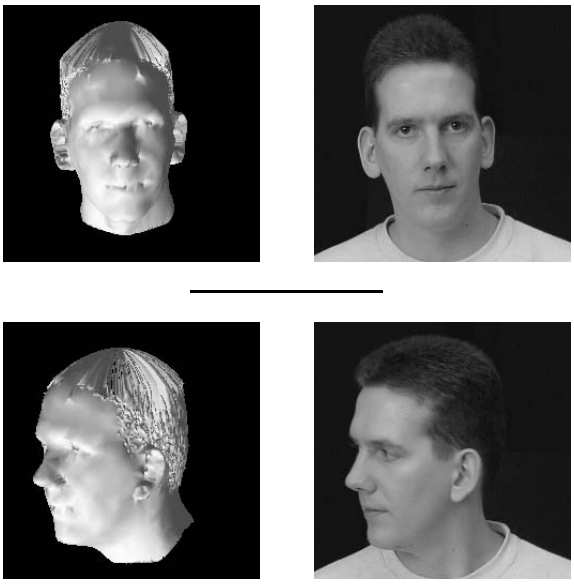

Figure 2: The camera images and the models (left) after alignment.

composition of the 3-D texture from several camera images are described below:

1. Alpha blending to combine textures from different camera images. A texture map (see Figure 3) of the model is stored in the system. In the beginning of a video conference session, this texture map is empty. As the camera starts acquiring images, the extracted texture from the camera images will be projected on this texture map. If there is no existing texture at the projected position, the extracted texture is stored at that position. Otherwise, the following equation is used to incorporate the new texture into the texture map:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_M = (1-\alpha) \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}_M + \alpha \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}_C \quad (7)$$

where the vector $(R\ G\ B)^T_M$ is the color in our texture map, and $(R\ G\ B)^T_C$ is the color of the extracted texture from the camera image, and α is the blending factor. It is set equal to 1.0 if there is no previous texture value in the map and equal to 0.5 (determined experimentally), otherwise.

2. Omission of texture around the object boundary in each camera image. Although the alignment process in section 3 is quite robust, its accuracy is limited by the number of reliable feature points which we can extract from a facial image. For camera calibration, 50 to 200 corresponding pairs of 2D-3D feature points are usually needed to get accurate results. Furthermore, these points must be distributed evenly in the working volume. Unfortunately, for a video conference scene, the number of reliable feature points on a person's face is usually less than

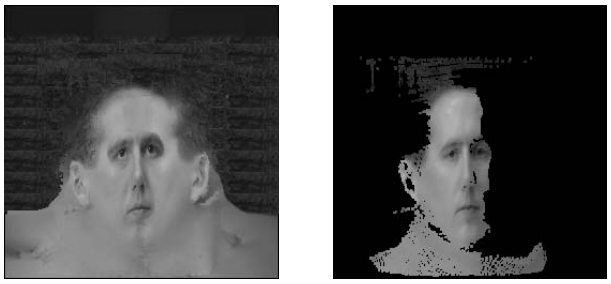

Figure 3: Examples of texture maps. Left: a fully constructed texture map. Right: a partial texture map extracted from a single image.

8. To account for the inaccuracy in the alignment parameters, we discard the texture around the object boundary in each camera image so that the background texture will not be mapped erroneously on the face of the model.

3. Extraction of only the part that covers the person's expression for a frontal view (the person in front of the camera is looking directly into the camera). The heuristics here are to reduce the effect of different shadowing between the frontal and other views. In addition, the alpha-blending method we use tends to over-smooth the important features like eyes, nose, mouth, etc. Therefore, for a frontal view image, the value $\alpha = 1.0$ is used to overwrite the current expression on the face of the model. This way, we are guaranteed to have the most recent expression of the person for image synthesis.

4. Use of model-based interpolation to fill in the holes in the final texture. Since the camera usually does not cover every aspect of the person even after the video conference session is on for a while, we must find some way to interpolate/extrapolate the missing parts of the texture map. Because we assume that the 3-D model is known in advance, we know which parts of the texture map correspond to hairs, face, etc. We can use different interpolation/extrapolation algorithms for different parts of the map. For example, we use the texture of the hair in the front to extrapolate the hair in the back and we use linear interpolation to fill the holes in the skin.

5 Experimental Results

Some results from our experiments are shown in Figures 4 and 5. These experiments are based on a nine-frame video sequence. The resolution of each image is 352×288 pixels. The model of the person was constructed using a Cyberware 3-D color scanner 3030 RGB/PS. The computation and rendering of the synthesized images was done on an SGI Indy workstation. Figure 4 shows the synthesized image after the 8th frame from 4 different viewing angles. At the 9th frame, the person spoke. After extracting the texture from this frame, the rendering of the synthesized images from 4 different viewing angles are shown in Figure 5. Note that the permissible viewing angles are very large in these synthesized images.

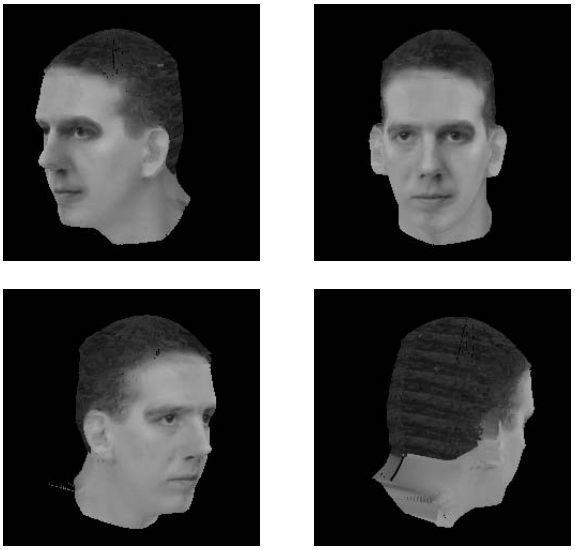

Figure 4: Some synthesized views of the 3-D model.

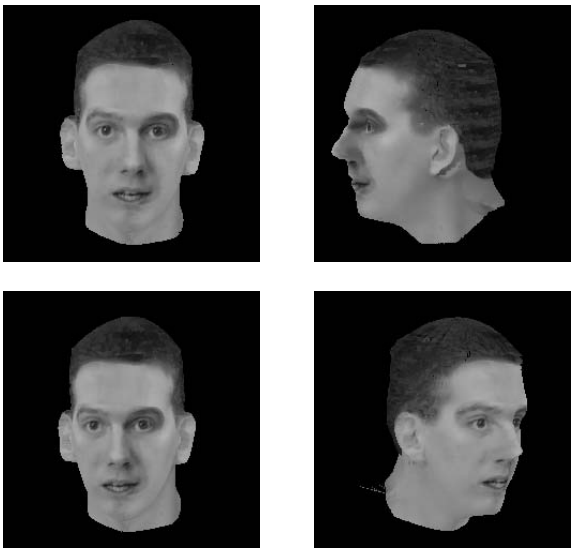

Figure 5: Some results of mapping a speaking person to the model. Note that the image in the upper-right corner is an extreme case where the difference between the model and the actual camera image is apparent around the mouth.

6 Conclusions

In this paper we have presented a method for generating synthetic multi-view images from monocular video sequences. A 3-D wireframe model is used to describe the shape and texture of the person. The model is aligned according to the position of the head in the video sequence and textural information is extracted from the camera images. This information is then used together with the texture from previous frames to update the texture map of the 3-D model. Here, we use a priori knowledge about the position of facial features to avoid artifacts due to illumination changes and local deformation caused by facial expressions. Experimental results show that this approach performs well generating synthetic images from arbitrary viewing points.

References

[1] J.S. McVeigh, M.W. Siegell, and A.G. Jordan, "Intermediate view synthesis considering occluded and ambiguously referenced image regions," *Signal Processing: Image Communication*, pp.21-28, 1996.

[2] P.A. Redert, E.A. Hendriks, and J. Biemond, "Synthesis of multi viewpoint images at non- intermediate positions," *Proceedings of ICASSP* München, Germany, pp. 2749-2752, 1997.

[3] J. Liu, I.P. Beldie and M. Wöpking, "A computational approach to establish eye-contact in videocommunication," *in Proceedings of the International Workshop on Stereoscopic and Three Dimensional Imaging (IWS3DI)*, Santorini, Greece, pp. 229-234, 1995.

[4] M.R. Jewell, G.R. Chamberlin, D.E. Sheat, P. Cochrane, and D.J. McCartney, "3-D imaging systems for video communication applications." *Proc. SPIE Vol 2409. Stereoscopic Displays and Virtual Reality Systems II*, San Jose, CA., Feb., 1995.

[5] V.M. Bove, Jr., "Multimedia Based on Object Models: Some Whys and Hows," *IBM System Journal*, **35**, pp.337-348, 1996.

[6] D. Tzovaras, N. Grammalidis, and M.G. Strintzis "Depth map coding for stereo and multiview image sequence transmission," *in Proceedings of the International Workshop on Stereoscopic and Three Dimensional Imaging (IWS3DI)*, Santorini, Greece, pp. 75-80, 1995.

[7] A. Azarbayejani and A. Pentland, "Camera self-calibration from one point correspondence," *MIT media laboratory, perceptual computing technical report #341*, 1995.

[8] L. Zhang, "Estimation of eye and mouth corner point positions in a knowledge-based coding system," *Proc. SPIE Vol 2952.*, pp. 21-18, 1996.