Superimposing 3D Virtual Objects using Markerless Tracking

Sang-Cheol Park, Sang-Woong Lee and Seong-Whan Lee
Dept. of Computer Science and Engineering, Korea University
Anam-dong, Seongbuk-ku, Seoul 136-713, Korea
{*scpark, sangwlee, swlee*}*@image.korea.ac.kr*

Abstract

This paper presents a novel methods to estimate the coordinates of a 3D object using the four vertices of a quadrangle and to track the markerless feature points. These methods are basic and important problems in augmented reality. However, many tracking solutions are dependent on fiducial markers in video or known coordinate systems which are required to superimpose virtual objects on frames. In this paper, we begin with the fact that the rectangular objects in 3D real world are projected the perspective quadrangle onto image plane through camera. We can estimate 3D object coordinates from 4 vertices of quadrangle through transformations of image coordinates. Also, the markerless tracking method of 4 vertices is presented and the camera motion parameters between successive frames are calculated using epipolar geometry.

1. Introduction

Augmented reality (AR) combines a virtual environment with the real world, in order to help people to understand the real world more easily by providing additional information about interesting objects in the real environment. An AR system should be able to 1) combine real environments and computer-generated virtual objects, 2) operate virtual objects interactively with the change in the real world, and 3) align virtual graphic objects onto real environments[1].

Video-based AR makes use of specific information such as square markers of known size, fiducial marks of a specific color, and the depth information assessed by stereo-camera systems. It is possible to obtain such specific information in very limited environments where artificial makers exist for AR systems. And a stereo camera is limited to use due to its high cost. In an attempt to overcome such limitations, our method presented by this paper examined in general video environments for video-based AR applications, so we found that many videos included artificial objects, which are rectangular and can be projected onto an image plane as a perspective-transformed shape.

In our method, the user selects just four points for a projected perspective quadrangle, and the detailed locations of the vertices are obtained by adjusting user's point according to the results of feature points extraction. Using rectangular objects projected and transformed into perspective shapes in sequential frames, we can estimate the direction of the Z-axis continuously by rotating the image coordinates (X, Y) at each vertex of the rectangle, and the vertices of a quadrangle that is defined in a reference frame are tracked using the properties of these pixels. In consecutive frames, we can calculate the parameters to indicate camera motion, so we can estimate 3D virtual object coordinate system. This coordinate system is computed for the essentially stereoscopic matrix using epipolar geometry involving rotation and translation transformation, and the estimated coordinates of the 3D object are determined from the camera motion parameters [2]. Using the estimated coordinates of a 3D object, a 3D graphical object can be superimposed in a sequence using OpenGL and a bitmap mask.

2. Related Works

A video-based AR can overcome the registration problem through the interaction between the AR system and features in the images under the following assumption; When a virtual object is superimposed in a reference frame, the frame should contain one plane with which a 3×3 planar homography can be found [3][4][5][6]. The homography is the transformation modeling the 2D movement of coplanar points under perspective projection. To obtain another planar homography between two consecutive frames in a sequence, different methods calculating camera motion to use multiple planes have been considered [5].

Kutulakos and Vallino proposed a system that could represent 3D graphic objects using four pairs of prior affine basis points that corresponded to a sequence of images extracted from two uncalibrated affine cameras[7]. This method imposed minimal computational requirements, and generated accurate, real-time video overlays, even when the

The 18th International Conference on Pattern Recognition (ICPR'06)
0-7695-2521-0/06 $20.00

camera parameters varied dynamically. Nevertheless, it required some fiducial markers in the video and a stereo camera system.

Seo and Hong proposed a system that allowed high performance registration[8]. Unlike the previous system that used an affine camera model, the system involved a perspective camera model. Although it was more difficult to estimate the projective reconstruction from perspective views than using affine reconstruction from orthographic views, the camera model was very good for superimposing 3D graphic objects.

3. 3D Object Registration

3.1. Feature points extraction

In previous methods, the user supplied coordinate information directly, in order to overcome the problem of the ambiguous selection of points. However, this caused registration problems in AR [8]. Although the user could specify exact feature points, in many cases the designated feature points were ambiguous, unless the location of the image input by the user was near a feature point that was being tracked. The optical flow method, general method to track feature points, was calculated from the difference in pixel intensity using Equation 1.

$$\mathbf{G} = \sum_w \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix}, det(\mathbf{G} - \lambda \mathbf{I}) = 0 \qquad (1)$$

where I_u and I_v were the differences in intensity; u and v were the locations of the pixel in the image; λ was the eigenvalue; $\mathbf{I}$ was the unit matrix; and λ_t was a threshold defined by the user. The features selected by Equation 2 were interpreted as corners after calculating all the eigenvalues of the image.

$$\min(\lambda_1, \lambda_2) < \lambda_t \qquad (2)$$

3.2 Estimating the coordinates of a 3D object

Our method uses a rectangular object from the real world, so the user has to select a rectangular object from a reference frame. This rectangular object is projected onto the image plane through a camera. Then, this rectangle is deemed to be one side of a rectangular parallelepiped. Consequently, its X-, Y- and Z-axes are at right angles to each other. Based on this fact, the Z-axis of the rotation angle is determined via complex rotations of the X- and Y-axes in the real world. The estimated direction of the Z-axis can be used to calculate the angles among the X-, Y- and Z-axes, and these angles are used when overlaying a 3D graphic object on the frame image.

The image coordinate system is determined from the columns and rows of the image plane and the direction of view that is vertical to both the columns and rows. The coordinates of a 3D object can be established from the image coordinates by creating a perspective quadrangle. This quadrangle is considered one side of a rectangular parallelepiped.

Consequently, the X-, Y- and Z-axes are at 90 degrees to each other. To establish the direction of the Z-axis, the image coordinates are rotated by applying the Euler-angle to each axis. This transformation does't rotate the object about the basis axes, but rotates it about an arbitrary axis. The vertices and the center point of a quadrilateral which the user designates from a reference frame are applied to Equation 3.

$$\begin{bmatrix} 1 + D \cdot E & -C + A \cdot H & B + A \cdot J \\ C + D \cdot H & 1 + D \cdot F & -A + D \cdot I \\ -B + D \cdot J & A + D \cdot I & 1 + A \cdot G \end{bmatrix} \qquad (3)$$

where

$$A = a \cdot \sin\theta, B = b \cdot \sin\theta, C = c \cdot \sin\theta,$$
$$D = (1 - \cos\theta), E = a^2 - 1, F = b^2 - 1,$$
$$G = c^2 - 1, H = ab, I = bc, J = ac.$$

Then, the unit vector in the direction of the Z-axis can be calculated. Equation 3 is the transformation matrix used for rotating an object about an arbitrary axis.

Figure 1. labelEstimateCoord2Estimating the direction of Z-axis by rotating the image coordinates

3.3. Tracking feature points and calculating camera motion parameters

In order to apply the estimated coordinates of 3D object to frames, we use tracking method based on the optical flow constraint. Equation 4 computes the value of d, which is the degree of change between two corresponding pixels within two consecutive frames in which d is minimized in the search region, on when comparing the frame and the previous frame, are tracked in a sequence of frames.

$$\begin{cases} d_0 = 0 \\ d_{k+1} = d_k + \mathbf{G}^{-1} \sum_w [(I(\mathbf{x}, t) \\ \qquad -I(\mathbf{x} + d_k, t + 1))\nabla I(\mathbf{x}, t)] \end{cases} \qquad (4)$$

This paper develops another method for extracting camera motion parameters using mono-scopic system. The camera motion parameters are calculated between pairs of successive frames in a sequence using epipolar geometry. The proposed method treats the initial frame as the image from the left camera and the second frame as the image from the right camera.

In this paper, most of the video sequences are pictures in which the intrinsic parameters of the camera are unknown. Therefore, we assume that the intrinsic parameters of the camera are set to a fixed skew of 0, an aspect ratio of 1, and the principle point is in the center of the quadrangle. The extrinsic parameters are calculated as an essential matrix. The images of left and right camera in Equation 5 can be adapted to a pair of successive frames to a sequence of frames. A pair of successive frames has the similar property with images of left and right camera. The $\mathbf{R}$ and $\mathbf{T}$ are rotation and translation between frames, respectively. To get the cross product with vector and matrix, the $\mathbf{S}$ is a skew symmetric matrix. And, the $\mathbf{R} \cdot \mathbf{S}$ matrix, an essential matrix is computed by Equation 5.

$$
\begin{aligned}
q_r &= \mathbf{R}(q_l - \mathbf{T}) \\
q_r^T \cdot (\mathbf{R} \cdot \mathbf{S}) \cdot q_l &= 0 \\
\mathbf{S} &= \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix}
\end{aligned}
\tag{5}
$$

The essential matrix $\mathbf{R} \cdot \mathbf{S}$ is computed using Equation 5, using an 8-point algorithm. In addition, $\mathbf{E}$ in a 3×3 matrix is computed using Equation 6, 7 and the property of the epipolar constraint. In order to calculate the essential matrix, we expand the equation,

$$
\mathbf{x'^T E x} = \mathbf{0}
$$

$$
\begin{bmatrix} u' & v' & 1 \end{bmatrix} \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = 0. \tag{6}
$$

For 8 point correspondences, Equation 6 becomes

$$
A\mathbf{e} = \mathbf{0} \tag{7}
$$

where

$$
A = \begin{bmatrix} u'_1u_1 & u'_1v_1 & u'_1 & v'_1u_1 & v'_1v_1 & v'_1 & u_1 & v_1 & 1 \\ u'_2u_2 & u'_2v_2 & u'_2 & v'_2u_2 & v'_2v_2 & v'_2 & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u'_8u_8 & u'_8v_8 & u'_8 & v'_8u_8 & v'_8v_8 & v'_8 & u_8 & v_8 & 1 \end{bmatrix}
$$

and $\mathbf{e} = (e_{11}, e_{12}, e_{13}, e_{21}, e_{22}, e_{23}, e_{31}, e_{32}, e_{33})$.

In Equation 7, $(u_1 \ldots_8, v_1 \ldots_8)$ and $(u'_1 \ldots_8, v'_1 \ldots_8)$ are obtained points from the images of the left and right cameras, where $e_1 \ldots_8$ are the components of essential matrix $\mathbf{E}$. Since Equation 7 is too time-consuming to process, $e_1 \ldots_8$ are computed using singular values decomposition (SVD).

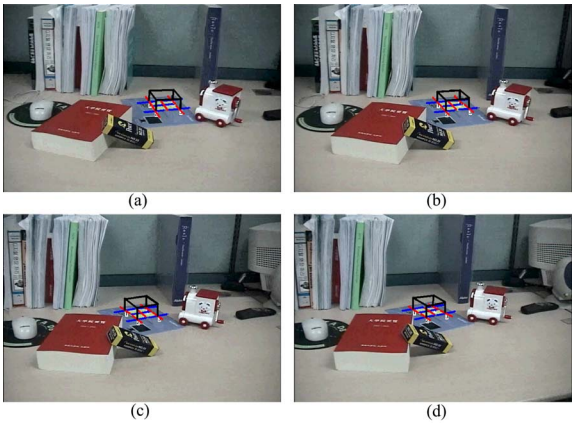


Figure 2. Applying camera motion parameters to frames : (a) Estimating the coordinates of a 3D object in the reference frame. (b) 10^{th} frame. (c) 20^{th} frame. (d) 30^{th} frame.

4. Experiment Results and Analysis

We tested the proposed method estimating the coordinates of a 3D object on a camera. The used camera system was a Sony Digital Cam (DCR-PC33) which was an ordinary digital camcorder, and the recording media was a Mini DV. The proposed method, which estimated the coordinates of a 3D object using a planar structure for video-based AR, tested in two steps. The first approach was to estimate the coordinates of a 3D object using input points specified by the user. The second approach was to apply the camera motion information to the coordinates of a 3D object created on an image in a sequence of frames. The estimated coordinates of a 3D object were applied to the camera motion parameters in the frames of a video. Therefore, the results could be compared with the real direction of the Z-axis on the image. Figure 3 summarized the direction error resulting from estimation of the coordinates of a 3D object. All of the errors were within 1 pixel. Our proposed method applied the camera motion parameters to the coordinates of a 3D object in the frames of a video, and compared the estimated direction of the Z-axis with the direction of the real Z-axis. Figure 4 showed the measured difference between the estimated and real directions of the Z-axis. The accumulated error increased towards the end of sequences.

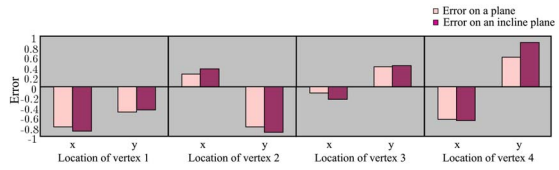


Figure 3. The difference in the direction of the real and estimated Z-axes.

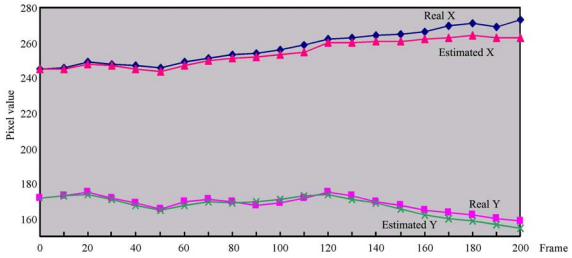
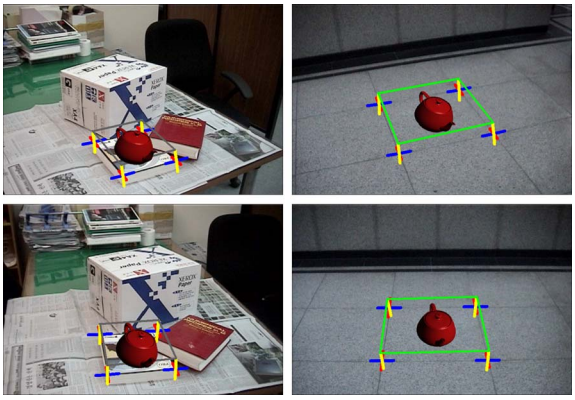


Figure 4. Comparison for the registration between the estimated and real directions of Z-axes.

5. Conclusions and Future Works

This paper proposed a method to estimate the coordinates of a 3D object contained in a sequence of frames using the four vertices of a rectangle and the camera motion parameters, which were computed from the essential stereoscopic matrix using epipolar geometry for rotation and translation, from a sequence of frames for video-based AR. Our method enabled computer-generated graphical objects to be registered to a real environment without using a calibration pattern or fiducial markers. Instead of using square fiducial markers of a known size, calibration patterns, or a stereo camera system, we was able to calculate the 3D coordinate parameters from the coordinates of a 3D object in a reference frame, using epipolar geometry in general videos. Since we used an ordinary digital camcorder, this extended the scope of video-based AR. This result was an accumulated error in the estimated direction of the Z-axis over all frames. To eliminate this registration error, we planned to increase the number of feature points tracked in a sequence and to make a buffer to store previously referred frames to help determine the locations of lost vertices.

As future works, our method will be extended to examine the superimposition of virtual images to any places on a real image, and to locate objects with rectangle shape automatically without human interaction.



(a) (b)

Figure 5. Superimposition a teapot in video sequence.

Acknowledgements

This research was supported by the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Ministry of Commerce, Industry and Energy of Korea. This research was also partially supported by the Brain Korea 21 Project in 2006.

References

[1] R.T. Azuma, "A Survey of Augmented Reality," *Teleoperators and Virtual Environments*, Vol. 6, No. 4, 1997, pp. 355-385.

[2] K. Rakesh, H. Sawhney and A.R. Hanson, "3D Model Acquisition From Monocular Image Sequences," *Proc. of the Conf. on Computer Vision and Pattern Recognition*, 1992, pp. 209-215.

[3] G. Simon and M.-O. Berger, "Estimation for Planar Structures," *IEEE Computer Graphics and Applications*, Vol. 22, 2002, pp.46-53.

[4] S.J.D. Prince, K. Xu, and A.D. Cheok, "Augmented Reality Camera Tracking with Homographies," *IEEE Computer Graphics and Applications*, Vol. 22, 2002, pp.39-45.

[5] G. Simon, A.W. Fitzgibbon and A. Zisserman, "Markerless Tracking using Planar Structures in the Scene," *Proc. International Symp. Augmented Reality*, 2000, pp. 137-146.

[6] P. Sturm, "Algorithms for Plane-Based Pose Estimation," *Proc. of the Conf. on Computer Vision and Pattern Recognition*, 2000, pp. 1010-1017.

[7] K.N. Kutulakos and J.R. Vallino, "Calibration-Free Augmented Reality," *IEEE Trans. on Visualization and Computer Graphics*, Vol. 4, 1998, pp. 1-20.

[8] Y.D. Seo and K.S. Hong, "Calibration-Free Augmented Reality in Perspective," *IEEE Trans. on Visualization and Computer Graphics*, Vol. 4, No. 6, 2000, pp. 346-359.