Vision-Based Real-Time Camera Matchmoving with a Known Marker

Bum-Jong Lee, Jong-Seung Park, and Mee Young Sung

Dept. of Computer Science & Engineering, University of Incheon,
177 Dohwa-dong, Nam-gu, Incheon, 402-749, Republic of Korea
{leeyanga, jong, mysung}@incheon.ac.kr

Abstract. A primary requirement for practical augmented reality systems is a method of accurate and reliable camera tracking. In this paper, we propose a fast and stable camera matchmoving method aimed for real-time augmented reality application. A known marker is used for the fast detection and tracking of feature points. From the feature tracking of one of three different types of markers on a single frame, we estimate the camera position and translation parameters. The entire pose estimation process is linear and initial estimates are not required. As an application of the proposed method, we implemented a video augmentation system that replaces the marker in the image frames with a virtual 3D graphical object during the marker tracking. Experimental results showed that the proposed camera tracking method is robust and fast enough to interactive video-based applications.

1 Introduction

Tracking the camera pose using images it acquires while the camera is moving in unknown environments is considered as an important problem in image analysis. During the recent two decades, many research works have focused on vision-based camera tracking. Most representative approaches can be classified into two categories: fiducial marker-based approaches and feature-based approaches. The feature-based approaches exploit geometric constraints arising from the automatic selection and tracking of appropriate point features [1]. An example is the planar-surface tracking method which tracks a manually specified planar region and computes the planar homography to align the real and virtual coordinate systems [2]. The fiducial marker-based approaches rely upon the presence of known fiducial markers or a certain calibration object in the given environment [3].

Though feature-based approaches do not assume any known geometric objects, they require sets of corner matches over an image sequence for the pose estimation, which means the methods are designed to operate in a batch off-line mode. However, most augmented reality applications require the camera pose in online mode to project computer generated 3D graphics models into the real world view in real-time. Hence, utilization of a fiducial marker is a natural choice for the fast feature tracking as well as for the computation of initial camera pose.

The projective camera model or the affine camera model is frequently used in computer vision algorithms. The projective model has eleven degree of freedoms and it is unnecessarily complex for many applications. There are non-linear relationships for the camera model parameters and iterative least-squares minimization must be used to improve the solution for the cost of a great amount of time. Since the camera tracking for an augmented reality application must be fast enough to handle real-time interactions, appropriate restrictions on the camera model should be introduced as long as the approximation is not far from the optimal solution.

This paper describes a stable real-time marker-based camera tracking method for augmented reality systems working in unknown environments. We propose a fast linear camera matchmoving algorithm that the initial estimates are not required.

2 Simplified Camera Model for Linear Approximation

Assume an arbitrary point $\mathbf{x}$ in the 3D scene space is projected to an image point $\mathbf{m}$ where all points are represented by the homogeneous form. The projection of a point $\mathbf{x}=[x\ y\ z\ 1]^T$ into the image point $\mathbf{m}=[u\ v\ 1]^T$ is presented by a projection equation of the form $\lambda\mathbf{m}=\mathbf{Px}$ where λ is an arbitrary scalar constant. The 3×4 matrix $\mathbf{P}$, called the projection matrix, specifies how each 3D point is mapped to an image point.

When the transformation between the world space and the camera space is Euclidean, the projection matrix of the perspective camera model can be decomposed into two matrices and a vector of the form $\mathbf{P}=\mathbf{K}[\mathbf{R}|\mathbf{t}]$. The 3×3 matrix $\mathbf{K}$, called the camera calibration matrix, encodes all the intrinsic parameters. The matrix $\mathbf{R}$ and the vector $\mathbf{t}$ encode the pose of the object with respect to the camera. The calibration matrix has four intrinsic parameters: the focal length in two image directions (f_x and f_y) and the coordinates of principal point (p_x and p_y). The matrix $\mathbf{R}=[\mathbf{r}_x\ \mathbf{r}_y\ \mathbf{r}_z]^T$ is a 3×3 orthonormal matrix and $\mathbf{t}=[t_x\ t_y\ t_z]^T$ is a 3×1 vector. $\mathbf{R}$ and $\mathbf{t}$ represent the relative rotation translation between the model and the camera. The intrinsic parameters are unknown but, in most cases, they are constant or smoothly varying. The intrinsic parameters are determined through on the fly calibration performed prior to the pose recovery. Assume that the four intrinsic parameters encoded in $\mathbf{K}$ are known a priori. In a moment, we let $\mathbf{K}$ be an identity matrix only for simplicity purposes.

In the perspective model, the relations between image coordinates (u and v) and model coordinates (x, y and z) are expressed by non-linear equations. By imposing some restrictions on the projection matrix $\mathbf{P}$, linearized approximations of the perspective model are possible. A well-known linearized approximation of the perspective camera model is the weak-perspective model [4]. The weak-perspective model can be used instead of the perspective model when the dimensions of the object are relatively small compared to the distance between the object and the camera. In the weak-perspective model, all object points lie on a plane parallel to the image plane and passing through $\mathbf{x}_c$, the centroid of the object points. Hence, all object points have the same depth: $z_c=(\mathbf{x}_c-\mathbf{t})\cdot\mathbf{r}_z$.

The projection equations for an object point $\mathbf{x}$ to an image point $\mathbf{m}$ are:

$$u=(1/z_c)(\mathbf{r}_x\cdot(\mathbf{x}-\mathbf{t})),\ v=(1/z_c)(\mathbf{r}_y\cdot(\mathbf{x}-\mathbf{t})).$$

Assume $\mathbf{x}_c=0$, then $z_c=-\mathbf{t}\cdot\mathbf{r}_z$ and it leads the projection equations:

$$u=\check{\mathbf{r}}_x\cdot\mathbf{x}+x_c/z_c,\ v=\check{\mathbf{r}}_y\cdot\mathbf{x}+y_c/z_c \tag{1}$$

where $\check{\mathbf{r}}_x=\mathbf{r}_x/z_c$, $\check{\mathbf{r}}_y=\mathbf{r}_y/z_c$, $x_c=-\mathbf{t}\cdot\mathbf{r}_x$, and $y_c=-\mathbf{t}\cdot\mathbf{r}_y$.

The equation (1) corresponds to an orthogonal projection of each model point into a plane passing the origin of the object space and parallel to the image plane, followed by a uniform scaling by the factor ($1/z_c$). The equation (1) can be solved by the orthographic factorization method [5] with constraints $|\check{\mathbf{r}}_x|=|\check{\mathbf{r}}_y|=1$ and $\check{\mathbf{r}}_x\cdot\check{\mathbf{r}}_y=0$. Once $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$ are obtained, the motion parameters $\mathbf{r}_x$ and $\mathbf{r}_y$ can be computed by normalizing $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$, i.e., $\mathbf{r}_x=\check{\mathbf{r}}_x/|\check{\mathbf{r}}_x|$ and $\mathbf{r}_y=\check{\mathbf{r}}_y/|\check{\mathbf{r}}_y|$, and the translation along the optical axis is computed by $z_c=1/|\check{\mathbf{r}}_x|$.

Though camera pose estimation for coplanar points is also available [6], known methods using a closed form pose solution for coplanar points are not robust. We use a known marker in camera tracking mainly for the fast detection and tracking of feature points.

3 Camera Tracking Under Scaled-Orthographic Projection

The weak-perspective model approximates the perspective projection by assuming that all the object points are roughly at the same distance from the camera. The situation becomes true if the distance between the object and the camera is much greater than the size of the object.

We assume that the depths of all points of the object are roughly at the same depth. All depths of the object points can be set to the depth of a specific object point, called a reference point. Let $\mathbf{x}_0$ be such a reference point in the object and all other points have roughly same depth denoted by z_0. We consider the reference point $\mathbf{x}_0$ is the origin of the object space and all other coordinates of the object points are defined relative to the reference point. Consider the point $\mathbf{x}_i$ and its image $\mathbf{m}_i=[u_i\ v_i\ 1]^T$, which is the scaled orthographic projection of $\mathbf{x}_i$. From the equation (1), the relation can be written:

$$\check{\mathbf{r}}_x\cdot\mathbf{x}_i=u_i-u_0,\ \check{\mathbf{r}}_y\cdot\mathbf{x}_i = v_i-v_0 \tag{2}$$

where $\check{\mathbf{r}}_x=\mathbf{r}_x/z_0$, $\check{\mathbf{r}}_y=\mathbf{r}_y/z_0$, $u_0=-\mathbf{t}\cdot\mathbf{r}_x/z_0$, $v_0=-\mathbf{t}\cdot\mathbf{r}_y/z_0$, and $\mathbf{m}_0=[u_0\ v_0\ 1]^T$ is the image of the reference point $\mathbf{x}_0$. Since the object points are already known and their image coordinates $\mathbf{m}_i\,(0\leq i<N)$ are available, the equations (2) are linear with respect to the unknowns $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$.

3.1 Linear approximation

For the $N-1$ object points $(\mathbf{x}_1,\ldots,\mathbf{x}_{N-1})$ and their image coordinates $(\mathbf{m}_1,\ldots,\mathbf{m}_{N-1})$, we construct a linear system using equation (2) by introducing the $(N-1)\times 3$ argument matrix $\mathbf{A}$, the $(N-1)$-vector $\mathbf{u}$, the $(N-1)$-vector $\mathbf{v}$:

$$\mathbf{A}=[\mathbf{x}'_1\ \mathbf{x}'_2\ \ldots\ \mathbf{x}'_{N-1}]^T,\ \mathbf{u}=[u'_1\ u'_2\ \ldots\ u'_{N-1}]^T,\ \mathbf{v}=[\ v'_1\ v'_2\ \ldots\ v'_{N-1}]^T$$

where $\mathbf{x}'_i=\mathbf{x}_i-\mathbf{x}_0$, $u'_i=u_i-u_0$, and $v'_i=v_i-v_0$. All the coordinates are given by column vectors in non-homogeneous form. The unknowns $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$ can be obtained by solving the two linear least squares problems:

$$\mathbf{A}\check{\mathbf{r}}_x=\mathbf{u} \text{ and } \mathbf{A}\check{\mathbf{r}}_y=\mathbf{v}. \tag{3}$$

The solution is easily obtained using the singular value decomposition (SVD). The parameters $\mathbf{r}_x$ and $\mathbf{r}_y$ are computed by normalizing $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$.

3.2 Iterative refinement

Once the unknowns $\mathbf{r}_x$ and $\mathbf{r}_y$ have been computed, more exact values can be obtained by an iterative algorithm. Dementhon [7] showed that the relation of the perspective image coordinates (u_i and v_i) and the scaled orthographic image coordinates (u_i'' and v_i'') can be expressed by:

$$u_i''=u_i+\alpha_i u_i,\ v_i''=v_i+\alpha_i v_i$$

in which α_i is defined as $\alpha_i=\mathbf{r}_z\cdot\mathbf{x}_i/z_0$ where $\mathbf{r}_z=\mathbf{r}_x\times\mathbf{r}_y$. Hence, in the equations (2), we can replace u_i and v_i with u_i'' and v_i'' and we obtain:

$$\check{\mathbf{r}}_x\cdot\mathbf{x}_i=(1+\alpha_i)u_i-u_0,\ \check{\mathbf{r}}_y\cdot\mathbf{x}_i=(1+\alpha_i)v_i-v_0\ . \tag{4}$$

Once we have obtained initial estimates of $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$, we can compute α_i for each $\mathbf{x}_i$. Hence, the equations (4) are linear with the unknowns, $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$. The term α_i is the z-coordinate of $\mathbf{x}_i$ in the object space, divided by the distance of the reference point from the camera. Since the ratio of object size to z_0 is small, α_i is also small, which means only several iterations may be enough for the approximation.

4 Fast Pose Estimation Using a Known Geometric Marker

Assume the N object points $\mathbf{x}_0,\mathbf{x}_1,\ldots\mathbf{x}_{N-1}$ are observed in a frame and their image coordinates are given by $\mathbf{m}_0,\mathbf{m}_1,\ldots\mathbf{m}_{N-1}$ in a single frame. All the points are given by column vectors in non-homogeneous form. We automatically choose the most preferable reference point which minimizes the depth variation. The camera intrinsic parameters p_x, p_y, f_x, and f_y are also used for the better pose estimation. The overall steps of the algorithm are as follows:

- Step 1: Choose a reference point $\mathbf{x}_k$ satisfying $\arg\min_k \sum_i (z_i-z_k)^2$ where z_i is the z-coordinate of $\mathbf{x}_i$.
- Step 2: Translate all the input object points by $-\mathbf{x}_k$ so that the reference point $\mathbf{x}_k$ becomes the origin of object space. Also, translate all the input image points by $[-p_x\ -p_y]^T$ so that the location of principal point becomes the origin of image space.
- Step 3: Using object points $\mathbf{x}_i$ and their corresponding image points $\mathbf{m}_i\,(i=1,\ldots,N-1)$, build the $(N-1)\times3$ argument matrix $\mathbf{A}$, the $(N-1)$-vector $\mathbf{u}$, and the $(N-1)$-vector $\mathbf{v}$ shown in equation (3). Set $\mathbf{u}_t$ and $\mathbf{v}_t$ by $\mathbf{u}_t=\mathbf{u}$ and $\mathbf{v}_t=\mathbf{v}$.
- Step 4: Solve the two linear least squares problems, $\mathbf{A}\check{\mathbf{r}}_x=\mathbf{u}_t$ and $\mathbf{A}\check{\mathbf{r}}_y=\mathbf{v}_t$, for the unknowns $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$. The solution is easily obtained using the singular value decomposition (SVD).
- Step 5: Compute z_k by $z_k=2f_xf_y/(f_y|\check{\mathbf{r}}_x|+f_x|\check{\mathbf{r}}_y|)$ where f_x and f_y are the camera focal lengths by the x- and y-axis. Compute $\mathbf{r}_x$ and $\mathbf{r}_y$ by $\mathbf{r}_x=\check{\mathbf{r}}_x/|\check{\mathbf{r}}_x|$ and $\mathbf{r}_y=\check{\mathbf{r}}_y/|\check{\mathbf{r}}_y|$.
- Step 6: Compute α_i by $\alpha_i=\mathbf{r}_z\cdot\mathbf{x}_i/z_k$ where $\mathbf{r}_z=\mathbf{r}_x\times\mathbf{r}_y$. If α_i is nearly same to the previous one, stop the iteration.
- Step 7: Update $\mathbf{u}_t$ and $\mathbf{v}_t$ by $\mathbf{u}_t=(1+\alpha_i)\mathbf{u}$ and $\mathbf{v}_t=(1+\alpha_i)\mathbf{v}$. Go to Step 4.

The rotation matrix $\mathbf{R}$ is the arrangement of the three orthonormal vectors: $\mathbf{R}=[\mathbf{r}_x\ \mathbf{r}_y\ \mathbf{r}_z]^T$. The translation vector $\mathbf{t}$ is the vector from the origin of the camera space to the reference point. Hence, once we found $\check{\mathbf{r}}_x$ and $\check{\mathbf{r}}_y$, the depth of the reference point z_k is computed by $z_k=2f_xf_y/(f_y|\check{\mathbf{r}}_x|+f_x|\check{\mathbf{r}}_y|)$. Then, the translation $\mathbf{t}$ is obtained by $\mathbf{t}=[z_ku_k/f_x,\ z_kv_k/f_y,\ 2/(|\check{\mathbf{r}}_x|+|\check{\mathbf{r}}_y|)]^T$.

If at least three points are available other than $\mathbf{x}_k$ and not every point is on the same plane, the matrix $\mathbf{A}$ in equation (3) has rank 3 and the over-determined linear system in Step 4 can be solved. On the other hand, if all the object points are on a single plane, the matrix **A** has rank 2 and the linear system in equation (3) is ill-conditioned. For such rank deficient cases, the SVD-based method is a better choice among several least squares method then the Cholesky factorization method or the QR decomposition method. For the rank-deficient cases, the Cholesky method will fail and the QR decomposition method is unstable.

5 Experimental Results

To demonstrate the effectiveness of the proposed method we implemented the camera pose tracking system that relies on known marker tracking from a real video stream. The 3D coordinates of the marker are assumed to be known and are fixed in the program codes. For each frame, the feature points for the marker have been identified after converting the image frame to a binary image.

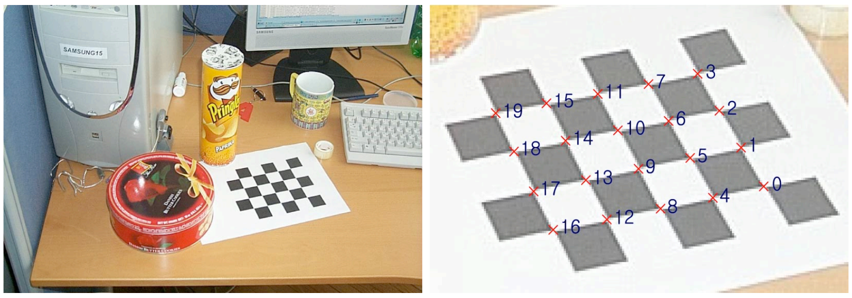


Fig. 1. A calibration pattern and its detected corners.

To recover the accurate shape geometry, it is required to know the camera intrinsic parameters, the focal lengths for two axes and the coordinates of the principal point. In our implementation, we equipped the system to deal with on-line camera calibration for convenience purpose only. The calibration algorithm is well described in [8]. When the system enters the calibration mode, the system monitors a grid pattern and it starts to accumulate detected grid pattern corners. When the grid pattern is found more than four frames the camera parameters are calculated.

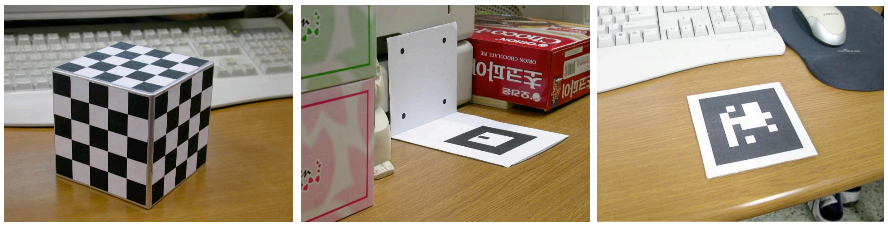


Fig. 2. Three different types of known markers: *Cube*, *TwoSidedMarker*, and *ARMarker*.

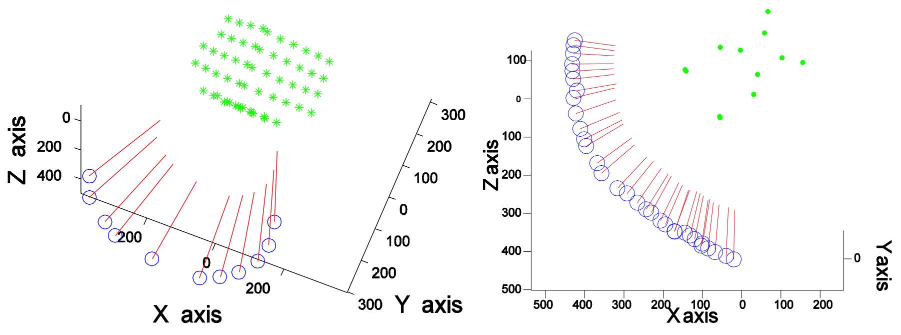


Fig. 3. Camera tracking results from two video frames each containing a known marker (*Cube* and *TwoSidedMarker* in Fig. 2).

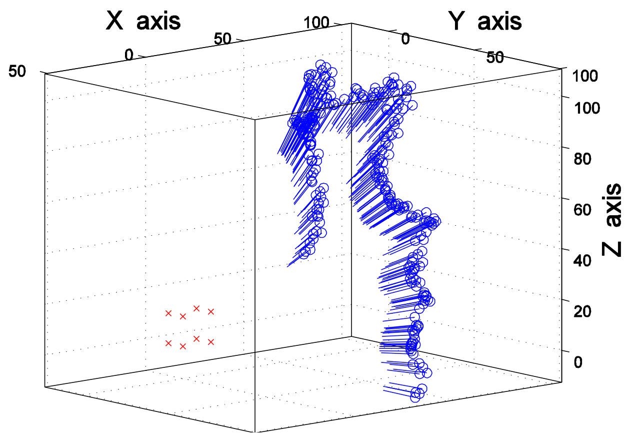


Fig. 4. Camera matchmoving for 507 frames using marker *Cube*.

As an example, we tested on a sequence of 768×512 image frames containing a 4×5 grid pattern as shown in Fig. 1. The computed focal lengths and principal point coordinates are as follows: f_x=1034.96, f_y=1024.62, p_x=360.79, p_y=204.29.

For each frame, the camera pose for the current frame is calculated using the tracked feature points on a marker from a single frame. The implemented system can recognize three types of markers (*Cube*, *TwoSidedMarker*, and *ARMarker*) as shown in Fig. 2. The continuous marker tracking and re-initialization are robust and also not sensitive to illumination changes. Fig. 3 shows the camera tracking results (recovered marker points and the camera pose trajectory). Fig. 4 shows a camera trajectory for a long sequence. The camera positions and orientations for the frames are shown relative to the marker points.

We compared the estimation accuracy of the proposed method with the linear scaled orthographic method (SOP) [9] and the iterative scaled orthographic method (POSIT) [7]. Using the estimated camera pose parameters, we projected 3D points onto the frames and compared the distances between the image features, which are shown in Fig. 5. The projection error is under 2 pixels in most cases and it is less than that of SOP or POSIT. The comparison of accuracy is shown in Fig. 6. We measured the error variation according to the relative distance of the marker from the camera divided by the marker radius, which is shown in Fig. 6(a). The average reprojection error is also measured when the scene point depth variation relative to the reference point increases, which is shown in Fig. 6(b). We found the proposed method is fast, stable, and versatile for various point distributions and various camera distances from a marker.

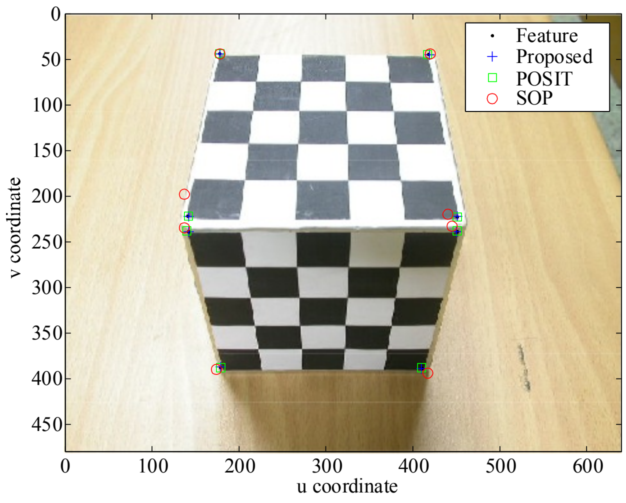


Fig. 5. Comparison of reprojected points.

Table 1. Accuracy and computing time with respect to marker types.

Marker type	#frame	#feature points	avg accuracy(pixel)	Time(ms)
CubeSparse	1000	5	1.64	1.73
TwoSidedMarker	500	16	0.815	1.986
ARMarker	300	4	0.435	1.672
CubeDense	700	72	0.327	2.113

Fig. 7 shows the stability of the camera pose estimation. The camera was moved very slowly and smoothly during 60 frames. The camera rotation parameters and translation parameters are stable along the frame sequence without serious oscillation. Fig. 8 shows the comparison of stability of the proposed method with other methods. The proposed method was stable when the number of feature points varies (See Fig. 8(a)). Since the proposed method is based on a single frame, any error from previous frames does not affect future frames (See Fig. 8(b)).

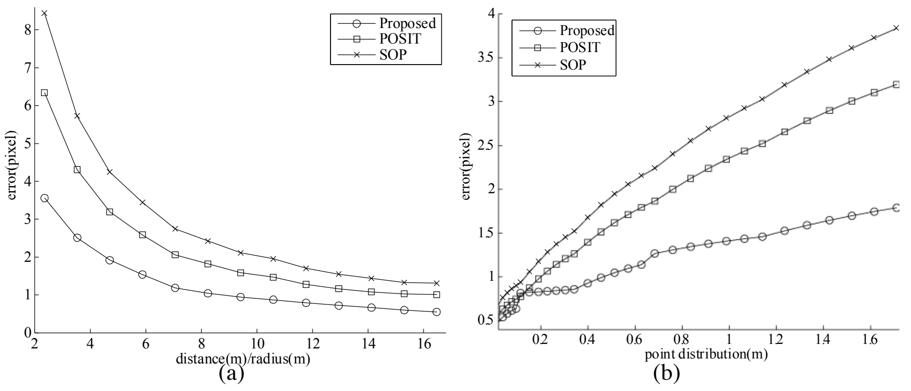


Fig. 6. Comparison of accuracy: (a) error variation with respect to the relative distance of the marker, (b) error variation with respect to the point distribution.

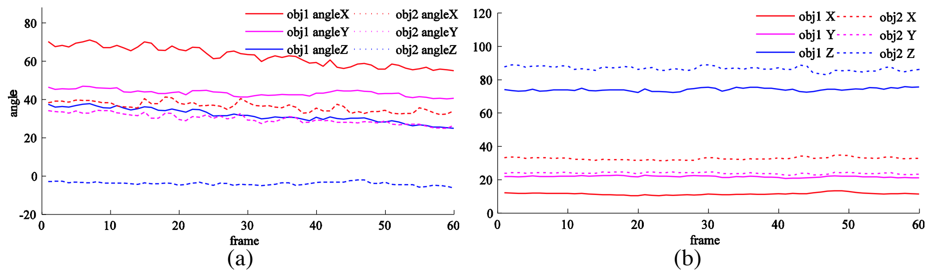


Fig. 7. The stability of camera pose parameters: (a) rotation parameters and (b) translation parameters.

The computation time for the proposed camera tracking method was measured. On a Pentium 4 (2.6GHz) computer, the processing speed is about 27 frames per second including all steps of the system such as frame acquisition, marker detection, feature extraction, pose estimation, and 3D rendering. The camera pose estimation process roughly takes 4 ms and 8 ms, respectively, and its speed is fast enough to be ignored for real-time augmented reality applications. Fig. 9 shows the camera pose computing time of the proposed method together with other methods. The camera pose estimation stage of the proposed method consumes about 1.8 ms CPU time per frame and, moreover, the number of points hardly increases the processing time.

Our method has also been applied to coplanar cases. We compared the results with ARToolKit method [10] which can estimate relative camera pose from a single rectangular marker. Although ARToolKit is fast and interactive, we found several critical problems exist in ARToolKit. First, the error of image corners of the marker rectangle critically increases errors. Second, when the camera moves away too much from the marker the marker detection step fails frequently. The proposed method is relatively more stable in most cases. The comparison of accuracy is shown in Fig. 10.

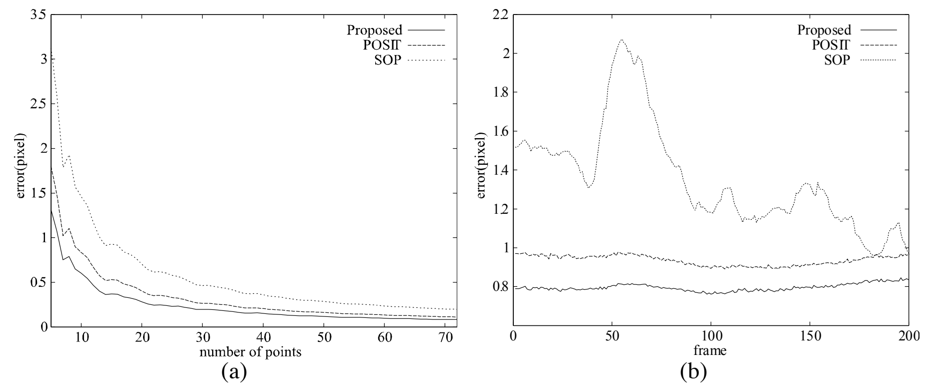


Fig. 8. Comparison of stability: (a) according to the number of points, (b) according to the frame moving.

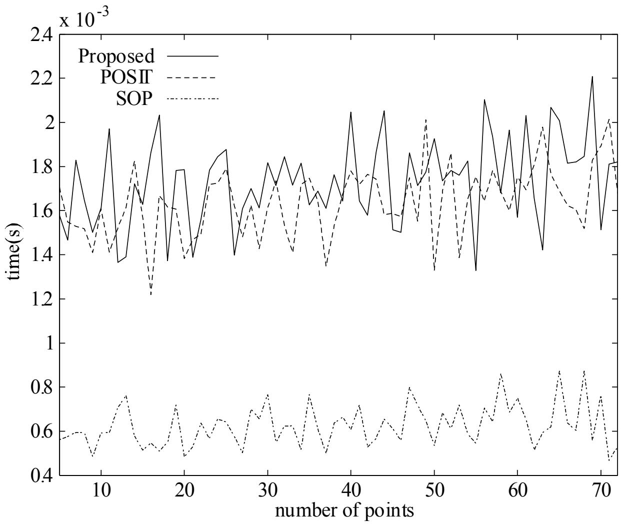


Fig. 9. Comparison of the computing time.

We have measured the accuracy and computing time with respect to different type of markers. We compared four different types of markers: *CubeSparse*, *TwoSidedMarker*, *ARMarker*, and *CubeDense* (See Table 1). Note that *CubeSparse* and *CubeDense* correspond to the same marker (*Cube* in Fig. 2) but they utilize different number of feature points on the marker.

The time is proportional to the number of feature points and the accuracy is inversely proportional to the number of feature points. Overall numerical values indicate that the type of markers does not affect the pose accuracy critically.

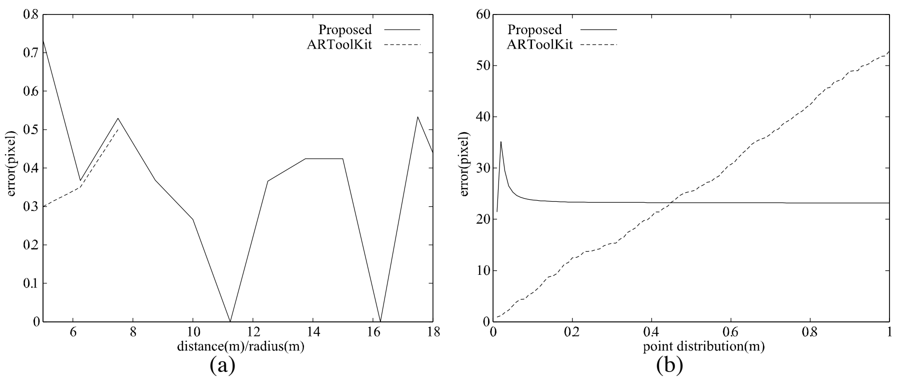


Fig. 10. Comparison of accuracy for coplanar cases: (a) error variation with respect to the relative distance of the marker, (b) error variation with respect to the point distribution.

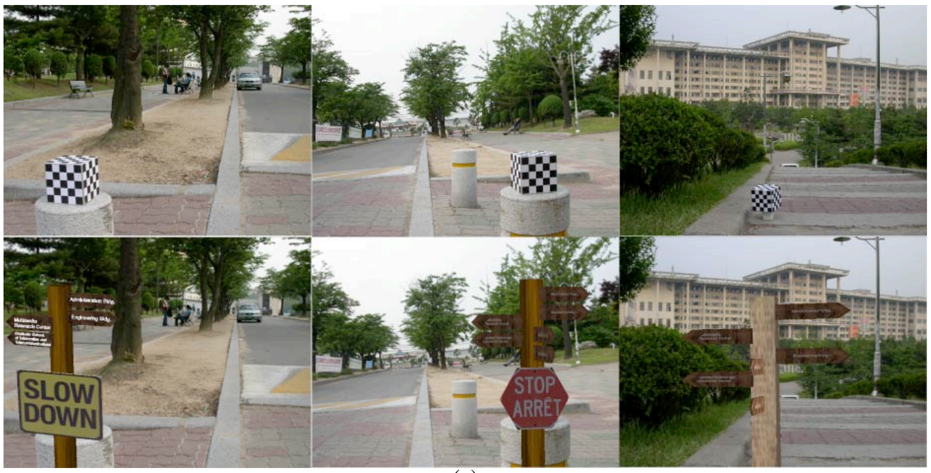
We implemented an interactive augmented reality system and tested camera tracking with real video frames. From a video stream, an image frame is captured and the known marker is detected in the frame. From the marker features, we estimate the camera pose. Then, we project some 3D graphical object onto the frame and render the projected virtual object together with the original input frame.

Fig. 11 shows the AR application which inserts a virtual object into a live video stream. The upper figure shows the insertion of a virtual flowerpot at the cube marker position. The lower figure shows the tracked planar marker is replaced by a wall clock.

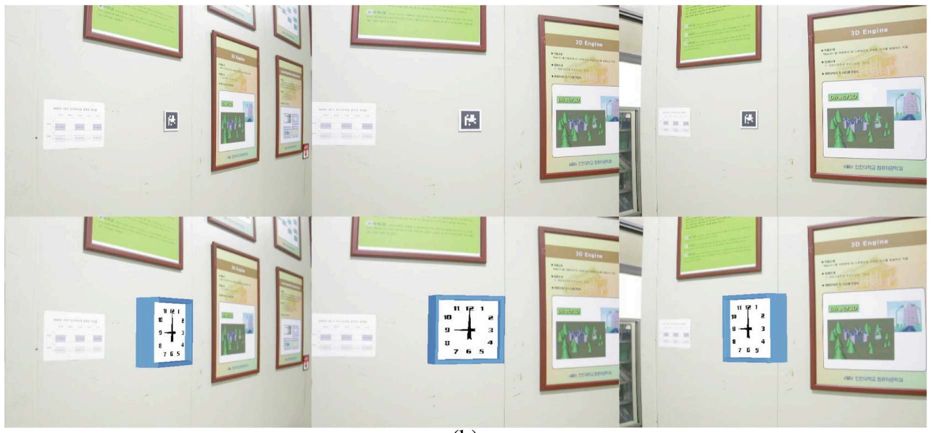
This article has presented a real-time camera pose estimation method assuming a known marker is visible. A fast detection and tracking of marker points is possible by assuming a known marker. From the marker tracking from a single frame, the camera position and translation parameters are estimated using a linear approximation. The pose estimation process is fast enough to real-time applications since entire pose estimation process is linear and initial estimates are not required. Compared with previous fast camera pose estimation methods, the camera pose accuracy is greatly improved without paying extra computing time. Another advantage is that it can cope with a planar marker since the SVD method can handle rank deficient cases.

As an application of the proposed method, we implemented an augmented reality application which inserts computer-generated 3D graphical objects into a live-action video stream of unmodeled real scenes. Using the recovered camera pose parameters, the marker in the image frames is replaced by a virtual 3D graphical object during the marker tracking from a video stream. Experimental results showed that the proposed camera tracking method is robust and fast enough to interactive video-based applications.

As future work, it would be preferable to extend the type of markers so that it accepts general primitives. Another possible direction concerns the use of features on a planar patch without using any known markers.



(a)



(b)

Fig. 11. An augmented reality application to insert virtual objects into a video stream: (a) using *Cube* maker, (b) using planar *ARMarker*.

Acknowledgments. This work was supported in part by grant No. RTI05-03-01 from the Regional Technology Innovation Program of the Ministry of Commerce, Industry and Energy(MOCIE) and in part by the Brain Korea 21 Project in 2006.

References

1. Gibson, S., Cook, J., Howard, T., Hubbold, R., Oram, D.: Accurate camera calibration for off-line, video-based augmented reality. In: Proceedings of International Symposium on Mixed and Augmented Reality (ISMAR 2002). (2002) 37-46
2. Simon, G., Fitzgibbon, A., Zisserman, A.: Markerless tracking using planar structures in the scene. In: Proceedings of IEEE and ACM International Symposium on Augmented Reality (ISAR 2000). (2000) 120-128

3. Kutulakos, K.N., Vallino, J.R.: Calibration-free augmented reality. IEEE Transactions on Visualization and Computer Graphics 4 (1998) 1-20
4. Carceroni, R., Brown, C.: Numerical method for model-based pose recovery (1997)
5. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: A factorization approach. International Journal of Computer Vision 9 (1992) 137-154
6. Oberkampf, D., DeMenthon, D.F., Davis, L.S.: Iterative pose estimation using coplanar feature points. Comput. Vis. Image Underst. 63 (1996) 495-511
7. Dementhon, D., Davis, L.: Model-based object pose in 25 lines of code. International Journal of Computer Vision 15 (1995) 123-141
8. Zhang, Z.: A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 1330-1334
9. Poelman, C., Kanade, T.: A paraperspective factorization method for shape and motion recovery. IEEE T-PAMI 19 (1997) 206-218
10. Kato, H., Billinghurst, M., Poupyrev, I., K., Tachibana, K.I.: Virtual object manipulation on a table-top AR environment. In: Proceedings of the International Symposium on Augmented Reality (ISAR 2000). (2000) 111-119