Image and Vision Computing 27 (2009) 1484–1496

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Multiple planes based registration using 3D Projective Space for Augmented Reality

Yuko Uematsu *, Hideo Saito

Keio University, School of Science and Technology, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-8522, Japan

ARTICLE INFO

Article history:
Received 6 November 2006
Received in revised form 4 November 2008
Accepted 28 January 2009

Keywords:
Augmented Reality
Mixed Reality
Vision-based registration
Multiple planes
Projective Space

ABSTRACT

We propose a vision-based registration method for Augmented Reality (AR) that uses real scenes containing multiple planar structures. Our method provides users with an enhanced view of a real scene by overlaying computer-generated virtual objects onto images captured with a movable camera. The virtual objects are associated with a 3D planar structure and appeared to move with that structure. To align the coordinates of the virtual objects with the coordinates of the images captured with the camera, the camera's motion should be estimated for every frame by tracking and integrating the multiple planes. In contrast with related work, the proposed method does not require a priori knowledge about the geometrical relationships among the multiple planes because the geometrical relationship is automatically estimated by "Projective Space". This space is 3D and is defined by projective reconstruction of two reference images. Automatic estimation using the Projective Space can eliminate the time-consuming task of measuring the planes and the constraints of plane arrangement, e.g. only vertical or co-planar. Therefore, the proposed method can be applied even to complicated multi-planar scenes.

1. Introduction

Augmented Reality (AR) enhances the visual information in a real scene with computer-generated virtual objects, such as computer graphics, texts, etc. The virtual objects are superimposed onto images of the real scene, which are captured with a movable video camera, as shown in Fig. 1. A user can see the real world with the superimposed virtual objects through displays, such as handheld monitors, head mounted displays and 3D displays. This means that AR can provide the user with more informative views in its mixed world than can Virtual Reality (VR), in which the real world around the user is completely replaced by a virtual world [1,2].

One of the most important issues for AR is geometrical registration between the coordinates of the real world and the virtual world. To align the virtual objects with the images of the real world as captured by a moving camera, the camera's position and pose (parameters of rotation and translation) with regard to the real world must be estimated frame-by-frame. After that, the virtual objects are superimposed onto the images frame-by-frame based on the estimated parameters.

We propose a new approach to estimate the camera motion that is used to register the virtual objects for AR. The camera's position and pose with regard to the real world are estimated by tracking multiple planes in the real world. Feature points on the planes are tracked throughout the image sequence and used to estimate camera motion with a planar constraint derived from the points. Although the camera's position and pose can be estimated with only a single plane, our method uses multiple planes to estimate more accurately and extend the user's movable space. The camera's position and pose are estimated from the feature points on every plane, and then the estimated parameters from all planes are integrated. The virtual objects are registered by overlaying them onto the captured images by computing 2D coordinates of the virtual objects on the images from the camera's estimated position and pose. The virtual objects are placed on one of the planes in the real world, which is selected in advance. Through a display the user sees the virtual objects as though they really exist in the real world plane.

2. Related work

Magnetic and gyro sensors may be used to track a camera in AR. Such sensors stabilize the registration of the virtual objects against a change in illuminations and when the camera moves rapidly between frames. However, the camera's rotations and translations as obtained by such sensors are not accurate enough to achieve complete geometrical registration. Furthermore, there are practical limitation on the use of sensors, such as a user's limited area of movement and perturbation caused by the environment.

Vision-based registration, by contrast, requires no special devices other than cameras, so many approaches have been proposed, in which artificial markers [3,4], prepared 3D models [5–7], and natural features are used to estimate camera motion. The natural feature-based approaches use various features in the real world such

* Corresponding author. Tel.: +81 45 563 1141x43230; fax: +81 45 566 1747.
E-mail addresses: yu-ko@ozawa.ics.keio.ac.jp (Y. Uematsu), saito@ozawa.ics.keio.ac.jp (H. Saito).

0262-8856/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2009.01.003

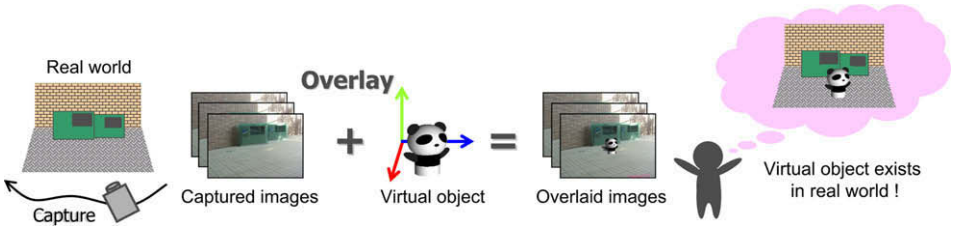


Fig. 1. Concept of vision-based Augmented Reality: a user can watch both real and virtual worlds by overlaying virtual objects onto images. Users see a mixed world.

as feature points [8], edges, and curves. However, detecting and tracking the natural features offers no stability in the real environment. Therefore, it is important to use the features effectively.

We focus on planar structures in the real world. That is, our method uses natural feature points on planes to track the camera. By using only feature points on planes, our method can easily and stably compute camera position and pose. Moreover, there are a lot of planar structures in the real world, such as indoor and outdoor walls and floors. Planar markers are also frequently used because it is easy to make them. Therefore, using feature points on the planar structures is a reasonable approach. When using planar structures to track the camera, using multiple planes is better than using a single plane because the user can move around a wide space by switching the planes based on the user's point of view. Moreover, using multiple planes simultaneously stabilizes tracking and improves its accuracy.

To use multiple planes, however, we have to know the geometrical relationships of the planes. Simon et al. have proposed related approaches for AR, which use multiple planes in the real world [9–11]. In one study [9], they tracked feature points on a plane in the real world and used them to estimate camera motion. Then, they overlaid virtual objects onto the plane based on the estimated camera motion. In another study [10], they extended this approach [9] in order to use multiple planes in the real world. A constraint they used was that the planes used to track the camera were perpendicular to a reference plane. In [11], they used multiple planes oriented in arbitrary positions and directions. The geometrical relationships between these planes and the camera motion were estimated by bundle adjustment, which is carried out over all frames.

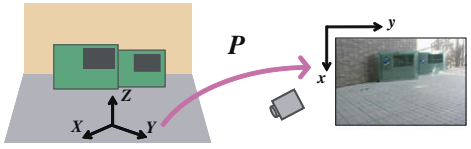
Planar-markers like AR-Toolkit [3] have been applied to various AR applications in marker-based approaches. When multiple markers are used for registration, the arrangement of the markers also has to be measured in advance. Therefore, to measure the arrangement, the markers are placed on the same plane or onto the surface of a 3D object, like a cube, whose shape is already known.

One advantage of the proposed method is that it can use multiple planes even if they are oriented in arbitrary positions and directions. This means that no prior knowledge of the geometrical relationships among the planes is required to track camera motion. In our method, the geometrical relationship among the planes is automatically estimated using "Projective Space". 3D Projective Space is defined by projective reconstruction of two reference images that are captured from two different viewpoints. In contrast with related work, our method avoids the time-consuming tasks of measuring the planes and the constraints on arranging the planes, e.g. as only vertical or co-planar planes. Therefore, the method can be applied even to a complicated multi-planar scene. Moreover, since it can estimate camera motion and overlay virtual objects onto images of the real world frame-by-frame, the proposed method can be extended to on-line applications.

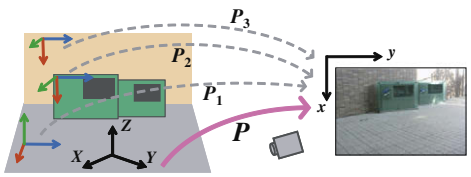
3. Overview of our method

As described above, the camera's position and pose with regard to the real world must be estimated frame-by-frame to register virtual objects. This corresponds to computing a projection matrix from the coordinate system of the real world $(X-Y-Z)$ to the coordinate system of the input image $(x-y)$, which is captured by a camera, as shown in Fig. 2a. Simon et al. have shown that the projection matrix can be computed from planar homography [9,10]. Although the projection matrix can be computed using only a single plane [10], the accuracy of the projection matrix can be improved by using multiple planes.

In the proposed method, a projection matrix is computed from each plane independently, and then the multiple projection matrices are integrated into one matrix, as shown in Fig. 2b. Use of the



(a) Projection matrix between real world and input image is computed from one plane.



(b) Projection matrices computed from the multiple planes are integrated into one matrix.

Fig. 2. Projection matrix from real world to input image.

geometrical relationship among the multiple planes is necessary to integrate those projection matrices into one matrix, as described in Section 2. Our method can automatically estimate the geometrical relationship using "3D Projective Space", in which the position and pose of the planes are estimated. This makes manual measurements of the planes and restrictions on their placement unnecessary.

3.1. Definition of coordinate systems

In this section, we introduce some coordinate systems and transformation matrices among them used in our method. To track the camera's position and pose with regard to the real world and align the coordinates of the virtual objects with the input image, our method defines three different coordinate systems to represent the real world ($X_i - Y_i - Z_i$), the input image ($x - y$), and a virtual space ($P - Q - R$), respectively. We define such coordinate systems as shown in Fig. 3.

First, a 3D coordinate system ($X_i - Y_i - Z_i$) is independently designated to each plane i and a 2D coordinate system ($x - y$) is designated to the input image. Here, we select one plane as a base plane whose 3D coordinate system is especially described as ($X_{base} - Y_{base} - Z_{base}$). Then the 3D coordinates of the virtual objects are described in ($X_{base} - Y_{base} - Z_{base}$).

Since the multiple planes occur in arbitrary positions and directions in the real world, the relationship among ($X_i - Y_i - Z_i$) is

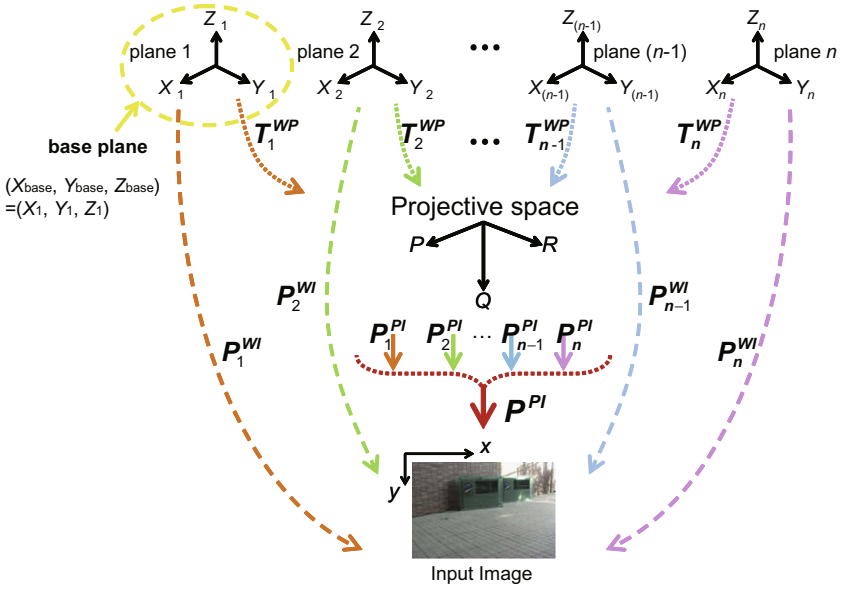


Fig. 3. Coordinate systems defined in our method.

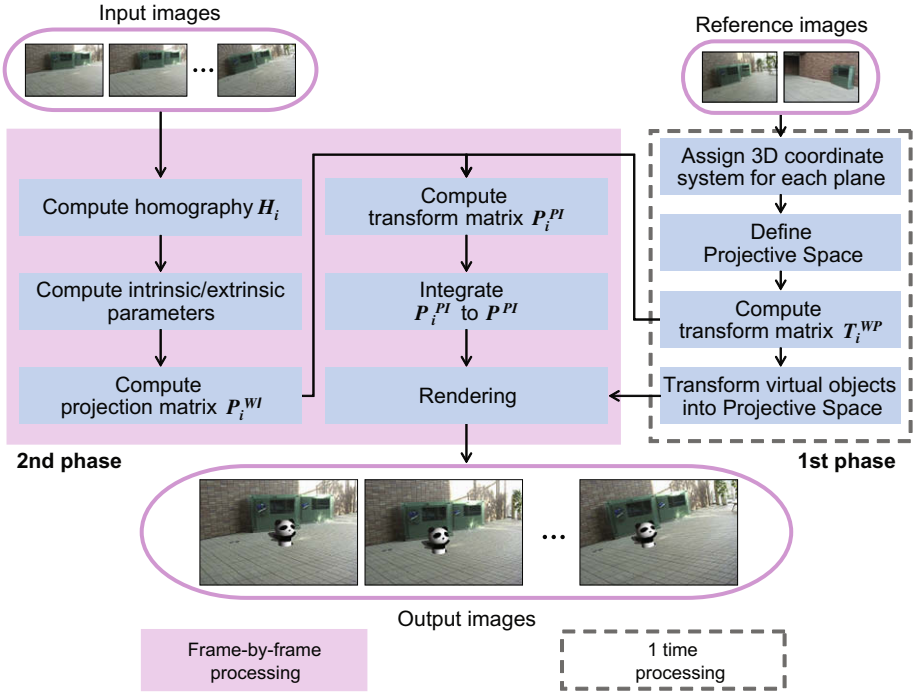


Fig. 4. Overview of our approach.

unknown. We introduce "Projective Space" $(P-Q-R)$ to estimate the relationship. This space is a 3D non-Euclidean coordinate system and is defined by projective reconstruction of two images that are called "reference images" and are captured from two different viewpoints. This will be described in Section 4.3.

3.2. Flow of our method

Our method can be divided into two phases as shown in Fig. 4. In the first phase, the object scene is captured from two different viewpoints. These images are reference images A and B, which are used to define a Projective Space. Here, we specify four points on each plane in each of the two reference images in order to define the $(X-Y)$ axes of a 3D coordinate system for each plane, as shown in Fig. 5. The Z axis is defined so that each plane becomes $Z=0$.

Next, 3D Projective Space is defined by projective reconstruction of the two reference images. Then, $\boldsymbol{T_i^{WP}}$ is computed based on each plane, which is a transformation matrix from each 3D coordinate system of the plane $(X_i-Y_i-Z_i)$ to the Projective Space $(P-Q-R)$. $\boldsymbol{T_i^{WP}}$ is computed from five or more corresponding points between $(X_i-Y_i-Z_i)$ and $(P-Q-R)$ obtained by the reference images.

The 3D coordinates of the virtual objects are described in $(X_{base}-Y_{base}-Z_{base})$, which is the 3D coordinate system designated to one of the multiple planes. After computing $\boldsymbol{T_i^{WP}}$ for all planes, then, the 3D coordinates of virtual objects are transformed into the coordinate system of the Projective Space by $\boldsymbol{T_{base}^{WP}}$.

In the second phase, a projection matrix $\boldsymbol{P_i^{WI}}$ based on each plane i that is visible in the current frame of the input image sequence is computed from a homography, which is computed from four or more corresponding points between 2D points on the current frame and 3D points on $(X-Y)$ plane in the real world $(Z=0)$.

After computing all $\boldsymbol{P_i^{WI}}$ for the plane in the current frame, $\boldsymbol{P_i^{PI}}$ is computed by Eq. (1) from $\boldsymbol{P_i^{WI}}$ and $\boldsymbol{T_i^{WP}}$, which have been computed in the first phase. Then all $\boldsymbol{P_i^{PI}}$ are integrated into one projection matrix $\boldsymbol{P^{PI}}$.

Finally, the virtual objects described in the Projective Space are projected onto the current frame by $\boldsymbol{P^{PI}}$ as the output image. These processes of the second phase are repeated in every frame until the end of the input image sequence. Computing $\boldsymbol{T_i^{WP}}$ corresponds to indirect estimation of the geometrical relationship among the planes via the Projective Space. Computing $\boldsymbol{P_i^{WI}}$ for every frame corresponds to the camera tracking in the current frame.

4. Registration method of virtual objects

4.1. Relationship of coordinate systems

The geometrical relationship among the planes is estimated by computing each transformation matrix. This relates $(X_i-Y_i-Z_i)$ to $(P-Q-R)$. The camera's position and pose in every frame are tracked by computing each projection matrix from $(X_i-Y_i-Z_i)$ to $(x-y)$. These two kinds of matrices, which are based on the plane i, are represented as $\boldsymbol{T_i^{WP}}$ and $\boldsymbol{P_i^{WI}}$, respectively, as shown in Fig. 3. The details of the computation will be described in Section 4.

When $\boldsymbol{T_i^{WP}}$ and $\boldsymbol{P_i^{WI}}$ are obtained based on each plane i, a 3D point (P,Q,R) in the Projective Space is independently projected to a 2D point (x,y) in the image coordinate system by the following equation,

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \simeq \boldsymbol{P_i^{WI}}(\boldsymbol{T_i^{WP}})^{-1} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix} \simeq \boldsymbol{P_i^{PI}} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix}, \tag{1}$$

where $\boldsymbol{P_i^{PI}}$ is a projection matrix computed based on plane i. Since each $\boldsymbol{P_i^{PI}}$ represents the projection from $(P-Q-R)$ to $(x-y)$, all $\boldsymbol{P_i^{PI}}$ should coincide with each other. Therefore, we can uniquely define the projection from the Projective Space to the input image by integrating these $\boldsymbol{P_i^{PI}}$ into one projection matrix. The integrated projection matrix is called $\boldsymbol{P^{PI}}$ as shown in Fig. 3. $\boldsymbol{P^{PI}}$ includes information on the geometrical relationship among the planes and camera's position and pose of the current frame.

As discussed in the previous section, the 3D coordinates of the virtual objects are described in $(X_{base}-Y_{base}-Z_{base})$, which is designated to the base plane. If the virtual objects are transformed into the coordinate system of the Projective Space by $\boldsymbol{T_{base}^{WP}}$, the virtual objects can be projected onto the input image by $\boldsymbol{P^{PI}}$. Therefore, to overlay the virtual objects onto the input image based on the camera's motion, our method computes $\boldsymbol{P^{PI}}$ from $\boldsymbol{T_i^{WP}}$ and $\boldsymbol{P_i^{WI}}$ at every frame.

4.2. Designation of 3d coordinate systems on multiple planes

In our method, a 3D coordinate system is independently designated to each plane that is used for registration. We specify the 3D coordinate system through the reference images used to define a Projective Space.

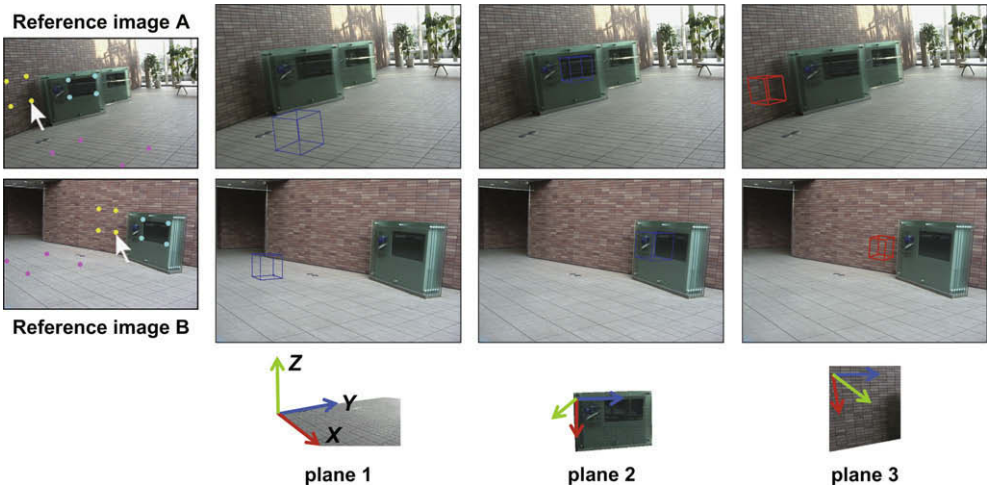


Fig. 5. 3D coordinate systems designated to planes.

First, four points on each plane are clicked on the reference images A and B, as shown in Fig. 5. Here, we specify the four points so that they become a square on the $X-Y$ axes. The Z axis is perpendicular to the plane. Next, one of the planes is selected as a base plane, and then the 3D coordinates of the four points on the base plane are arbitrarily defined as $(X, Y, 0)$. Using the four points on the two reference images, two projection matrices, which relate to the 3D coordinate system of the base plane and the reference images respectively, are computed by homographies between 3D points on the base plane and 2D points on the reference images, as described in Section 4.5. A similar approach is also taken in [10]. The 3D coordinates of the four points on the other planes are computed by 3D reconstruction using a stereo algorithm as described in Section A of the appendix. These four points are used for computing homography as described in Section 4.5.1.

4.3. Definition of Projective Space

3D Projective Space is used to estimate the geometrical relationship among the multiple planes that occur in arbitrary positions and poses. The Projective Space is defined by projective reconstruction of two images that are captured from two different viewpoints and are called reference images.

As shown in Fig. 6, the object scene is captured from two viewpoints and the captured images become reference images A and B. Then, a 3D coordinate system $(P-Q-R)$ is defined so that it has the following projective relationships with each of the reference images. This 3D coordinate system is called 3D Projective Space.

$$\begin{bmatrix} u_A \\ v_A \\ 1 \end{bmatrix} \simeq \boldsymbol{P_A} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix}, \quad \begin{bmatrix} u_B \\ v_B \\ 1 \end{bmatrix} \simeq \boldsymbol{P_B} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix} \tag{2}$$

$$\boldsymbol{P_A} = [\mathbf{I}|\mathbf{0}], \quad \boldsymbol{P_B} = \left[-\frac{[\boldsymbol{e_B}]_\times \boldsymbol{F_{AB}}}{\|\boldsymbol{e_B}\|^2} | \boldsymbol{e_B} \right] \tag{3}$$

Here, $[u_A, v_A, 1]^\top$ and $[u_B, v_B, 1]^\top$ are homogeneous coordinates of 2D points in the reference images, and $[P, Q, R, 1]^\top$ is a homogeneous coordinate of a 3D point in the Projective Space. $\boldsymbol{F_{AB}}$ is a fundamental matrix from the image A to image B. $\boldsymbol{e_B}$ is an epipole on the image B, and $[\boldsymbol{e_B}]_\times$ is the skew-symmetric matrix of $\boldsymbol{e_B}$ [12].

Our method computes $\boldsymbol{F_{AB}}$ and $\boldsymbol{e_B}$ by eight or more corresponding points between the reference images, which are manually specified, and then $\boldsymbol{P_A}$ and $\boldsymbol{P_B}$ are defined by Eq. (3). 3D coordinates in $(P-Q-R)$ space are computed by $\boldsymbol{P_A}, \boldsymbol{P_B}$ and 2D projected points onto the reference images. The detail is described in Section 4.4.1.

4.4. Computation of $\boldsymbol{T_i^{WP}}$ (between plane's coordinate system and Projective Space)

Since the coordinate system associated with a plane i $(X_i - Y_i - Z_i)$ and on the Projective Space $(P-Q-R)$ are both 3D coordinate system, the transformation matrix $\boldsymbol{T_i^{WP}}$ is a 4×4 matrix and is computed from five or more corresponding points between $(X_i - Y_i - Z_i)$ and $(P-Q-R)$.

4.4.1. How to get corresponding points

We assume that a 3D point in $(X_i - Y_i - Z_i)$ is $\boldsymbol{X_W} \simeq [X, Y, Z, 1]^\top$ and that the point is projected onto reference images A and B as 2D points (u_A, v_A) and (u_B, v_B), respectively. When these 2D points are projected into the Projective Space as a 3D point, $\boldsymbol{X_P} \simeq [P, Q, R, 1]^\top$, we can write

$$\begin{bmatrix} \boldsymbol{p_A^1} - u_A \boldsymbol{p_A^3} \\ \boldsymbol{p_A^2} - v_A \boldsymbol{p_A^3} \\ \boldsymbol{p_B^1} - u_B \boldsymbol{p_B^3} \\ \boldsymbol{p_B^2} - v_B \boldsymbol{p_B^3} \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix} = \mathbf{0}, \tag{4}$$

where, $\boldsymbol{p_A^j}$ and $\boldsymbol{p_B^j}$ are the jth row vectors of $\boldsymbol{P_A}$ and $\boldsymbol{P_B}$ in Eq. (3). Then $\boldsymbol{X_P}$ is computed by Singular Value Decomposition of the 4×4 matrix on the left-hand side of Eq. (4). Therefore, the corresponding points $\boldsymbol{X_W}$ and $\boldsymbol{X_P}$ are obtained through the 2D points (u_A, v_A) and (u_B, v_B) on the reference images. Since five or more corresponding points are required to compute $\boldsymbol{T_i^{WP}}$, we draw a cube on each plane as shown in Fig. 5. The size of the cube is already known when defining 3D coordinate systems. Then, using eight vertices from each cube, eight corresponding points between $(X_i - Y_i - Z_i)$ and $(P-Q-R)$ can be obtained by Eq. (4).

4.4.2. How to compute the matrix

As described in Section 4.1 and Fig. 3, the relationship between the coordinate system of the plane i and the Projective Space is expressed by the following equation.

$$\boldsymbol{X_P} \simeq \boldsymbol{T_i^{WP}} \boldsymbol{X_W} \tag{5}$$

$$\boldsymbol{T_i^{WP}} = \begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ t_{41} & t_{42} & t_{43} & 1 \end{bmatrix} \tag{6}$$

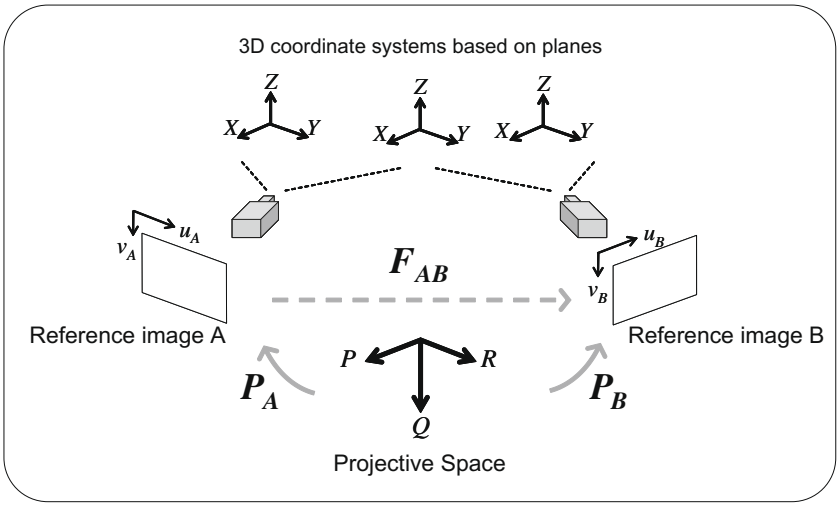


Fig. 6. 3D Projective Space.

To obtain $\boldsymbol{T_i^{WP}}$, this equation can be written as

$$\boldsymbol{Mt} = \boldsymbol{b}, \tag{7}$$

where

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{X_W}^\top & 0 & 0 & -XP & -YP & -ZP \\ 0 & \boldsymbol{X_W}^\top & 0 & -XQ & -YQ & -ZQ \\ 0 & 0 & \boldsymbol{X_W}^\top & -XR & -YR & -ZR \end{bmatrix} \tag{8}$$

$$\boldsymbol{t} = [\, t_{11} \quad t_{12} \quad t_{13} \quad \cdots \quad t_{41} \quad t_{42} \quad t_{43} \,]^\top \tag{9}$$

$$\boldsymbol{b} = [\, P \quad Q \quad R \,]^\top \tag{10}$$

If there are j corresponding points of $\boldsymbol{X_W}$ and $\boldsymbol{X_P}$ ($j \geqslant 5$), Eq. (7) is

$$\begin{bmatrix} \boldsymbol{M}_1 \\ \vdots \\ \boldsymbol{M}_j \end{bmatrix} \boldsymbol{t} = \begin{bmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b_j} \end{bmatrix} \tag{11}$$

Therefore, the parameters of $\boldsymbol{T_i^{WP}}$ are obtained by Least Square Method of Eq. (11) (See Fig. 7).

4.5. Computation of $\boldsymbol{P_i^{WI}}$ (between plane's coordinates and image coordinate system)

In this section, we explain how to compute $\boldsymbol{P_i^{WI}}$ which projects a 3D point in the coordinate system of the plane i ($X_i - Y_i - Z_i$) to a 2D point on the coordinate system of the current frame of the input images ($x - y$). In our method, a homography between each plane i and the input image is computed first, then the $\boldsymbol{P_i^{WI}}$ is computed from the homography. Since a projection matrix consists of intrinsic and extrinsic parameters, we separately compute those parameters from the homography. The details of computing the parameters will be described in Section B of the appendix.

4.5.1. Computation of $\boldsymbol{H_i}$ (between $X - Y$ plane and $x - y$ image plane)

A homography represents the projective transformation between planes and is computed from four or more corresponding points on both planes.

In the first frame of the input images, we specify four points of each plane in the image. These four points corresponds to the four points which are clicked on the reference images when X and Y axes are designated in the first phase, as shown in Fig. 5 and described in Section 4.2. $\boldsymbol{H_i}$ is computed based on each plane using these four corresponding points.

After the second frame, feature points on each plane in the input images are tracked by the KLT Feature Tracker [13] and used to compute the homography of each plane between the current frame and the previous frame. Then, a new $\boldsymbol{H_i}$ is obtained by multiplying the homography between the current frame and the previous frame by $\boldsymbol{H_i}$, which relates the previous frame to the $X - Y$ plane as shown in Fig. 7.

4.6. Computation and integration of $\boldsymbol{P_i^{PI}}$ (between Projective Space and image coordinate system)

In this section, we explain how to compute $\boldsymbol{P_i^{PI}}$, which is a projection matrix from the Projective Space to the coordinate system of the input image, and how to integrate all $\boldsymbol{P_i^{PI}}$ into $\boldsymbol{P^{PI}}$.

After computing $\boldsymbol{P_i^{WI}}$ based on each plane, $\boldsymbol{P_i^{PI}}$ is obtained by using $\boldsymbol{P_i^{WI}}$ and $\boldsymbol{T_i^{WP}}$, which is computed in the first phase.

$$\boldsymbol{P_i^{PI}} = \boldsymbol{P_i^{WI}} (\boldsymbol{T_i^{WP}})^{-1} \tag{12}$$

As described in Section 4.1, all $\boldsymbol{P_i^{PI}}$ should all coincide with each other because they represent a common geometrical projection between the Projective Space and the input image. Therefore, all $\boldsymbol{P_i^{PI}}$ are integrated into a single $\boldsymbol{P^{PI}}$. The details are as follows.

When $\boldsymbol{P_i^{PI}}$ is computed for each plane, a 2D point $\boldsymbol{x_j}$ in the input image, which corresponding to a 3D point $\boldsymbol{X_{P_j}} \simeq [P_j, Q_j, R_j, 1]^\top$ in the Projective Space can be obtained by the following equation.

$$\boldsymbol{x_j} \simeq \boldsymbol{P_i^{PI}} \boldsymbol{X_{P_j}} \tag{13}$$

If there are m corresponding points and n planes in the real world, the integrated projection matrix $\boldsymbol{P^{PI}}$ is computed as follows ($m \geqslant 6$).

$$\begin{bmatrix} \boldsymbol{M}_1 \\ \vdots \\ \boldsymbol{M_n} \end{bmatrix} \boldsymbol{p} = \begin{bmatrix} \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_n \end{bmatrix} \tag{14}$$

where $\boldsymbol{M_i}$ is

$$\boldsymbol{M_i} = \begin{bmatrix} \boldsymbol{X}_{P_1}^\top & 0 & -P_1x_1 & -Q_1x_1 & -R_1x_1 \\ 0 & \boldsymbol{X}_{P_1}^\top & -P_1y_1 & -Q_1y_1 & -R_1y_1 \\ & & & \vdots & \\ \boldsymbol{X}_{P_m}^\top & 0 & -P_mx_m & -Q_mx_m & -R_mx_m \\ 0 & \boldsymbol{X}_{P_m}^\top & -P_my_m & -Q_my_m & -R_my_m \end{bmatrix} \tag{15}$$

$$\boldsymbol{b_i} = \begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_m \\ y_m \end{bmatrix}, \quad \boldsymbol{p} = \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ \vdots \\ p_{31} \\ p_{32} \\ p_{33} \end{bmatrix}, \quad \boldsymbol{P^{PI}} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & 1 \end{bmatrix} \tag{16}$$

Therefore, the parameters of $\boldsymbol{P^{PI}}$ are obtained by Least Square Method of Eq. (14).

An alternative we have considered and rejected is determining the projection matrix by simply selecting the most accurate single projection matrix. The difficulty with this approach, and our reason for not adopting it, is that it introduces instability in moving from frame to frame, because the projection matrix may be computed from different planes on successive frames. Therefore we employ the integration process instead of switching the planes.

5. Experimental results

In this section, the experimental results are shown to demonstrate the effectiveness of the proposed method. We implemented

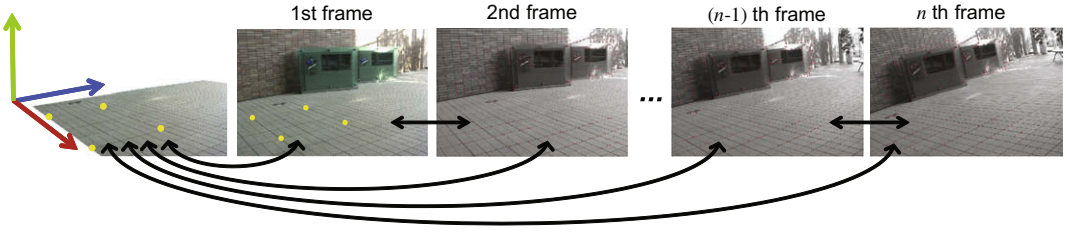


Fig. 7. Homography between 3D plane and current frame.

the AR system based on our method using only a PC (OS: Windows XP, CPU: Intel Pentium IV 3.20 GHz) and a CCD camera (SONY DCR-TRV900). The resolution of the input image is 720 × 480 pixels, and the graphical views of virtual objects are rendered using OpenGL.

The images produced by our augmentation method are shown in Figs. 8 and 9. In the sequence in Fig. 8, three planes (a floor, a front display and a back wall) are used for registration, and then a virtual object (a figure of a panda) is overlaid onto the floor plane. In the sequence in Fig. 9, three planes (a laptop display, a book and a mouse pad) are used to overlay the virtual object onto the laptop, which is the same as the mouse pad plane. As shown by these results, our registration method can successfully overlay the virtual object onto the input images based on the camera motion even though the geometrical relationship among the planes is unknown.

Next, we evaluate the registration accuracy of our method by implementing the same process for the synthesized images generated with OpenGL, as shown in Fig. 10. Then we compare the image coordinates of the overlaid virtual object with the ground truth. In Fig. 11a, we compare the ground truth with the registration results for 120 frames using a single plane and the eight planes shown in Fig. 10a. It can be seen that using eight planes causes fewer registration errors and jitters than using only a single plane. This means that registration accuracy can be improved by increasing the number of the planes. Moreover, the geometrical relationship among

(a) Frame 0
Rot. (± 0, ± 0, ± 0)
Trans. (± 0, ± 0, ± 0)

(b) Frame 60
Rot. (-1.33, +0.84, -2.51)
Trans. (+5.29, -0.95, +1.58)

(c) Frame 120
Rot. (-0.18, -0.36, +4.56)
Trans. (-6.51, +1.54, +1.33)

(d) Frame 180
Rot. (-1.82, +8.58, +8.00)
Trans. (-8.11, +4.14, -0.79)

(e) Frame 240
Rot. (-4.07, +11.53, +8.25)
Trans. (-8.68, +4.36, -0.80)

(f) Frame 300
Rot. (-1.90, +4.20, +5.07)
Trans. (-7.19, +4.78, +2.14)

Fig. 8. Overlaid image sequence of a virtual object. Camera's rotation and translation with respect to Frame 0 are represented as Rot. (X, Y, Z) and Trans. (X, Y, Z).

(a) Frame 0
Rot. (± 0, ± 0, ± 0)
Trans. (± 0, ± 0, ± 0)

(b) Frame 33
Rot. (+0.95, +1.14, -5.36)
Trans. (+2.81, -0.33, -0.02)

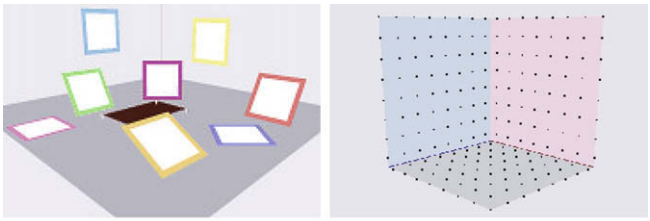
(c) Frame 66
Rot. (+2.17, +2.97, -0.16)
Trans. (-3.73, +0.29, -0.56)

(d) Frame 70
Rot. (+1.56, +5.59, -0.59)
Trans. (-4.52, -0.02, -1.70)

(e) Frame 74
Rot. (+0.11, +4.40, +2.93)
Trans. (-5.18, -0.31, +0.38)

(f) Frame 99
Rot. (+0.38, +3.12, +8.56)
Trans. (-9.75, -2.39, +3.81)

Fig. 9. Overlaid image sequence of a virtual object. Camera's rotation and translation with respect to Frame 0 are represented as Rot. (X, Y, Z) and Trans. (X, Y, Z).



(a) 8 planes (b) 3 planes

Fig. 10. Synthesized images with OpenGL.

the planes can be successfully estimated by our method without manual measuring tasks and constraints of the arrangement.

We also evaluate the proposed method by comparing with a related work presented by Simon [10], in which multiple planes must be perpendicular to the reference plane (that is, one of the multiple planes). We use the image sequence as shown in Fig. 10b that includes three perpendicular planes and apply it to Simon's method and our method. In both methods, the feature points on the planes are tracked and used for computing the homography of each plane. Then, both the registration results (estimated $x-y$ coordinates which are projection of known 3D points) are compared with the ground truth for 40 frames. Simon's method uses the information that the planes are perpendicular each other. Our method does not know the geometrical relationship information. The results are shown in Fig. 11b. Even though our method does not use information about geometrical relationships of the planes, our results are very similar to the ground truth and to those of Simon's method, in which the orthogonality of the planes is used for computation.

6. Discussion

In this section, we will discuss the effect of the designation of a 3D coordinate system by manual clicking. For designating a 3D coordinate system on each plane, a user manually clicks four points on each plane. If a user clicks a different position on the plane, the computed projection matrix also becomes a different one. Therefore we evaluate how much errors will be caused by a user's clicking and how much recovered errors will be caused by the computed projection matrix from the clicked points.

We will also discuss the selection of the reference images. Because epipolar geometry of the reference images defines the 3D projective space, the difference caused in the selection of a pair of reference images is much bigger than the difference in the defining 3D coordinate systems by clicking four points on the reference images. Therefore the selecting a pair of reference images is more important issue. Here, we introduce the evaluation algorithm for selecting the best pair of reference images.

6.1. Designation of 3D coordinate systems

When designating a 3D coordinate system on each plane, four points on each plane should be clicked on the reference images A and B by a user. Then a 3D coordinate system is defined so that the clicked four points composes a square on X–Y axes and Z-axis is perpendicular to the $X-Y$ square.

Here, we evaluate errors of clicked coordinates and recovered coordinates by using synthesized images as shown in Fig. 12a whose size is 640×480 [pixel]. As for the clicking errors, four corners of a plane are clicked by a user as shown in Fig. 12a. Then the clicked coordinates are compared to the true positions of the plane's corners. As for the recovered errors, first, the projection matrix is computed by using the clicked four points and projects a cube on the plane. Then eight vertices of the recovered cube are used for the evaluation. Since the size of the cube is known, the 2D coordinates of the eight vertices of the recovered cube are compared to the theoretical positions as shown in Fig. 12b. The errors are computed as described in the following equations.

$$\text{clickng error} = \sum_{v=1}^{4} \sqrt{(x_v - xt_v)^2 + (y_v - yt_v)^2} \tag{17}$$

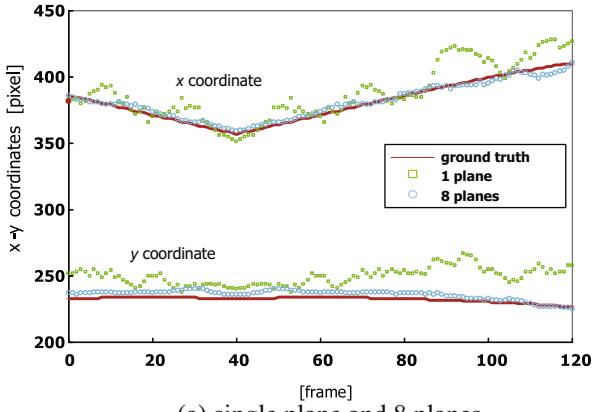
$$\begin{bmatrix} (x_v, y_v): & \text{2D position of clicked point for } v\text{th corner} \\ (xt_v, yt_v): & \text{2D position of theoretical point for } v\text{th corner} \end{bmatrix}$$

$$\text{recovered error} = \sum_{v=1}^{8} \sqrt{(x_v - xt_v)^2 + (y_v - yt_v)^2} \tag{18}$$

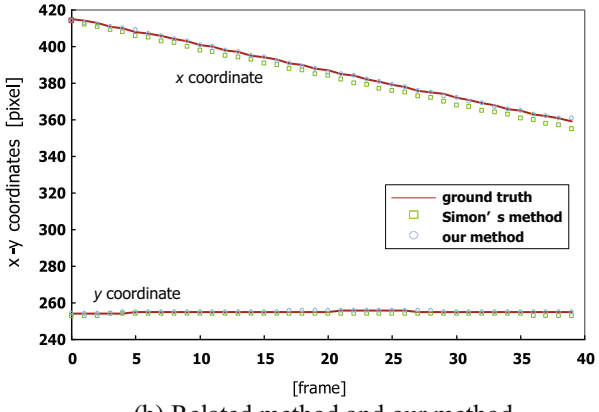
$$\begin{bmatrix} (x_v, y_v): & \text{2D position of recovered point for } v\text{th vertex} \\ (xt_v, yt_v): & \text{2D position of theoretical point for } v\text{th vertex} \end{bmatrix}$$

In this evaluation, a single user clicked four corners of the plane (base plane), and then a cube is recovered by using the clicked points. The same procedure is repeated 15 times with about 1 min interval.

Table 1 shows average, minimum, maximum and variance values of the differences between the results in 15 trials and the ground truth, respectively. Upper row of Table 1 describes the clicking errors for four corners on the image (Fig. 12a). In the 15 trials, the maximum error is only 3 pixels and the variance is 0.6.

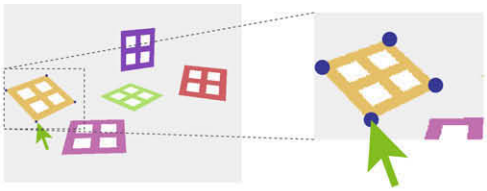


(a) single plane and 8 planes.

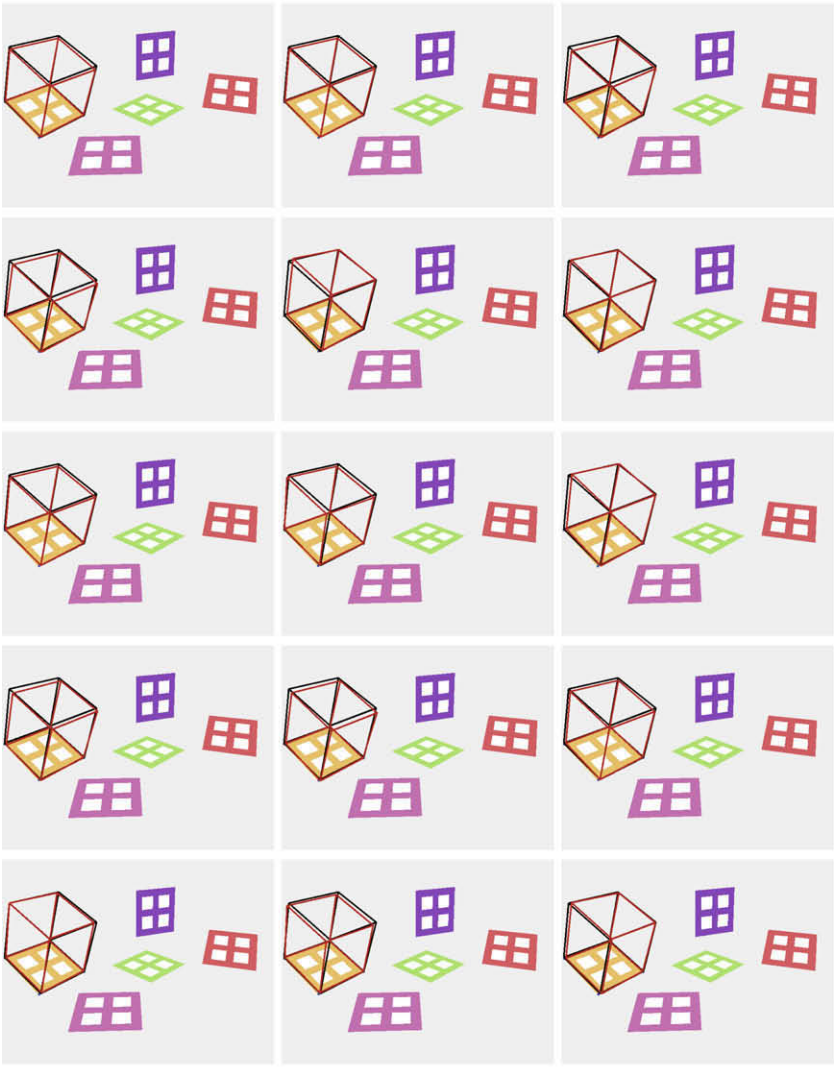


(b) Related method and our method.

Fig. 11. Comparison of $x-y$ coordinates accuracy with theoretical value. (a) Ground truth vs. results using a single plane and 8 planes. (b) Ground truth, related work, and proposed method.



(a) Clicked four points on a plane in the reference image.



(b) Recovered cubes by the clicked four points for 15 times. (Red cube is the recovered cube, black cube is the ground truth.)

Fig. 12. Designating 3D coordinate system of a plane on the reference image.

Table 1
Manual clicking errors of plane's corners and recovered errors of cube's vertices.

[Pixel]	Ave.	Min.	Max.	Var.
Manual clicking errors	1.0	0	3	0.6
Recovered errors	3.8	0	18	17.0

Therefore to click the specified points in the image manually is not difficult procedure.

The Lower row of Table 1 describes describes the recovered errors for the cube's vertices which are projected by the projection matrix computed by the clicked four corners of the plane. Fig. 12b also shows the recovered cubes (red cube) and the theoretical cubes (black cube) for 15 times trials. Although the projection matrix is computed depending on the 2D coordinates of the clicked points, the difference among 15 times trials is very small as shown in Fig. 12b. Therefore there is no significant error on the designation of 3D coordinate systems on the reference images by manual clicking.

6.2. Selection of reference images

When defining Projective Space, two reference images are selected. Selection of the reference images depends on the

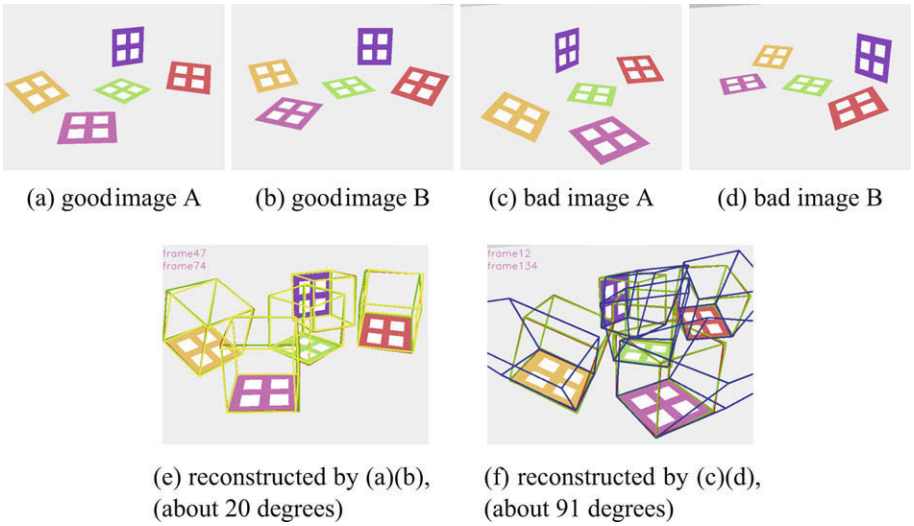


Fig. 13. (a and b) show a good pair of reference images (The angle between A and B is about 20 degrees). (c and d) show a bad pair of reference images (The angle between A and B is about 91 degrees). (e) shows cubes on each plane projected by using (a and b), (f) are using (c and d).

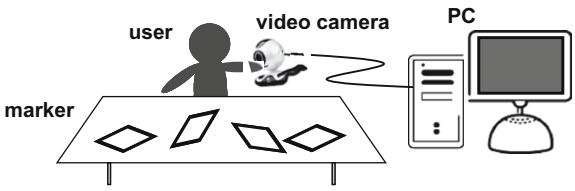


Fig. 14. On-line AR System.

accuracy of the fundamental matrix and epipole between the reference images. Therefore we have to select a reasonable pair of the reference images. To evaluate whether the pair is good or not, we introduce the following scheme.

First we select two input images as a temporary choice of reference images. Using the pair of the reference images, the fundamental matrix and epipole are computed and the Projective Space is defined by Eqs. (3) and (4). $\boldsymbol{T}_i^{WP}$ and $\boldsymbol{P}_i^{PI}$ are also computed for each plane and all the $\boldsymbol{P}_i^{PI}$ are integrated into $\boldsymbol{P}^{PI}$. Then we project two cubes onto each plane and compare the coordinates of vertices of the cubes. The cubes are projected by the following equations.

$$\boldsymbol{x}_i = \boldsymbol{P}_i^{WI}\boldsymbol{X}_W, \quad \boldsymbol{x}'_i = \left(\boldsymbol{P}^{PI}\boldsymbol{T}_i^{WP}\right)\boldsymbol{X}_W$$

$\boldsymbol{X}_W$ is a 3D coordinate of a vertex of a cube which is arbitrary size. $\boldsymbol{x}_i$ is a projected result of a projection matrix which is computed from the homography of the plane i. $\boldsymbol{x}'_i$ is a projected result of a integrated projection matrix. If the fundamental matrix and epipole are accurately computed, $\boldsymbol{x}'_i$ equals $\boldsymbol{x}_i$. Therefore when the difference between $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$ about all vertices of the cubes becomes less than 5 pixels, we select the pair of images as the reference images.

Fig. 13 shows two example pairs of reference images. (a) and (b) show a good pair of reference images and (c) and (d) show a bad pair of reference images. In (e) and (f), reconstructed cubes by (a,b) and (c,d), respectively, are shown. Green cubes (representing $\boldsymbol{x}_i$) in (e) and (f) are projected by the each plane's projection matrix. Yellow cubes in (e) and blue cubes in (f) (representing $\boldsymbol{x}'_i$) are projected by the integrated projection matrix. In (e), the green cubes and yellow cubes are almost overlapping. In contrast, in (f), the blue cubes are very different from the green cubes. In this way, we can evaluate all pairs of reference images and select the reasonable reference images.

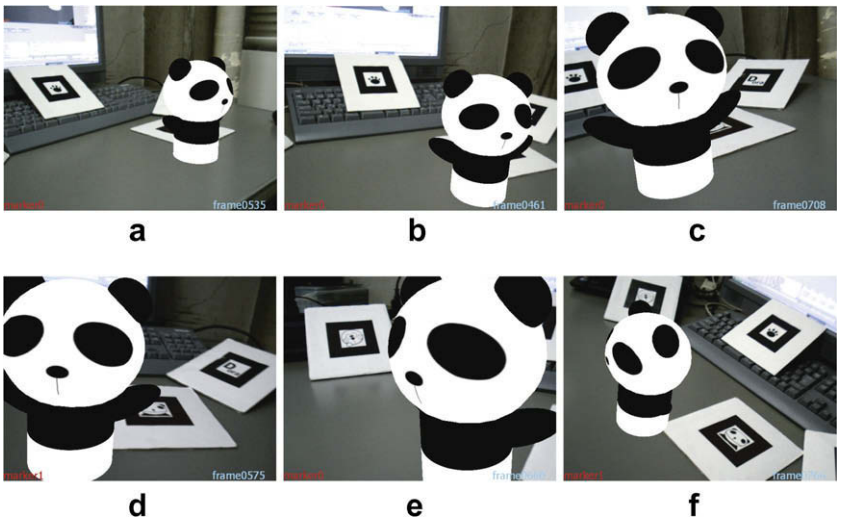


Fig. 15. A virtual object maintains its position from any viewpoints.

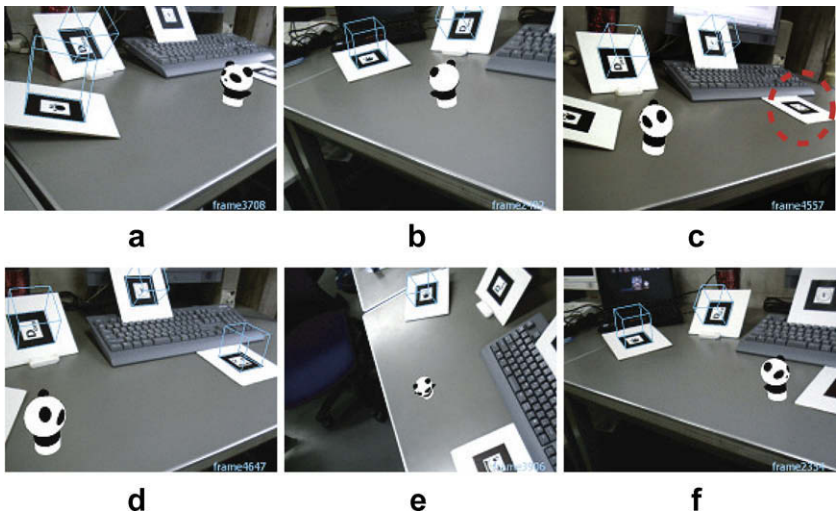


Fig. 16. A virtual object moves in a straight line on tabletop.

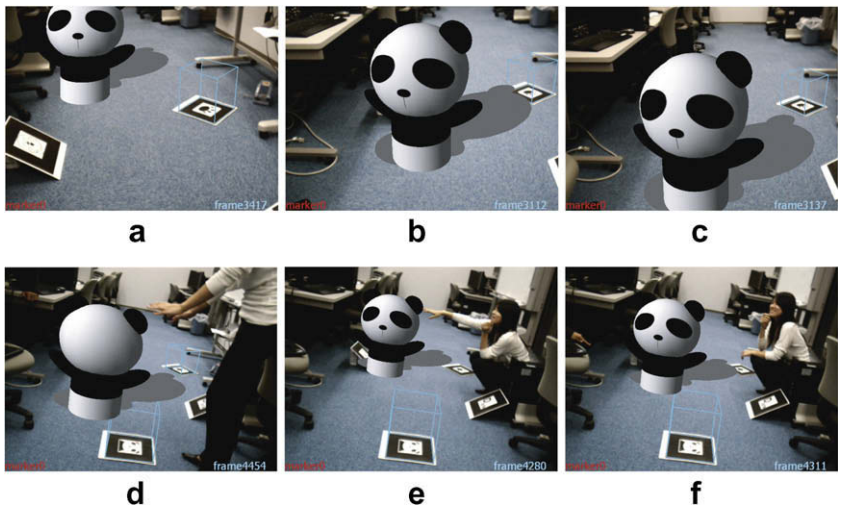


Fig. 17. A big virtual object moves in real room with a person.

Since the proposed method in the past sections performs off-line, we can take time to select the reasonable reference images. However, it is difficult to select reasonable reference images when applying our method to an on-line application. In the on-line application introduced in Section 7, therefore, two reasonable images are automatically selected by this evaluation scheme.

7. Applications

We extended the proposed method to an on-line AR application that uses multiple planar markers [14,15] as shown in Fig. 14. Multiple markers can be distributed in the real scene without manual measurement or special devices (ex. a high resolution camera), which contrasts with related works [16–19]. Therefore, a user can freely place the multiple markers in a wide area and move around inside it. The system can be implemented in a small space, such as a tabletop, and in a large space such, as a room.

User activities are listed as follows. First, the user places multiple markers in the real scene and captures the scene for a few seconds with a video camera. Then, the system automatically selects the two reference images, which are used to define a Projective Space from the captured images, as described in Section 4.3. After it selects the images, the system uses them to define the Projective Space.

This is all done automatically. When the Projective Space is ready, the system proceeds to the augmentation process. The user freely moves the camera to favorite viewpoints and watches through a display as virtual objects stay in place or move around the real scene.

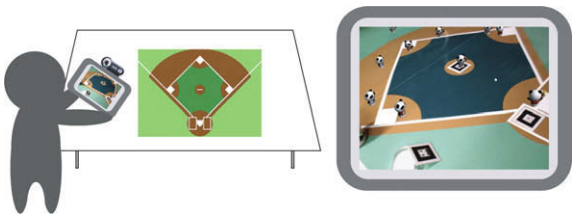


Fig. 18. AR Baseball Presentation System.

Fig. 19. Images of baseball game scenes played on tabletop field model. Batter hits and all runners move up a base.

User's activities

(1) **Preparations**
- Place multiple markers in the real scene;
- Capture the object scene as a candidate for two reference images;

(Auto-selection of two reference images from the captured images)

(2) **Augmentation**

Watch virtual objects from favorite viewpoints by moving the camera;

Some output images generated from the on-line system are shown in Figs. 15 and 16. The multiple markers are freely placed over a wide area on the tabletop in the real scene, but their positions and poses are not measured. The virtual object is then successfully overlaid on the on-line images. Here, the angle of the camera relative to the tabletop is too small to detect the markers on the tabletop plane in Fig. 16c. When this happens, the markers on the tabletop plane cannot be detected. Therefore, if all markers are on the same plane as in related works [16,18,19], the system may fail to detect most of them, and registration would be discontinued. In contrast, if the markers are faced in various directions as in our system, at least one marker will be detected in every frame. Since our system requires no measurement, a free arrangement of the planes is possible. This is a big advantage of our system.

The system also processes at a frame rate of 10 fps during the augmentation process. Therefore, our system easily enables on-line implementation in a wide space simply by distributing multiple markers.

Fig. 17 shows the results of an experiment conducted in a large room. The virtual object, which is as big as a person, is moving around the room. By using the system in a space as large as a human's living space, as shown in Fig. 17, the system creates the impression that the virtual object lives together in the same space.

Finally, we introduce "AR Baseball Presentation System", to which our on-line AR System is extended. This system enables the user to watch a virtual baseball game scene on a tabletop model baseball field through a web-camera attached to a hand-held LCD monitor, as shown in Fig. 18. The virtual baseball scene is synthesized from the input history data of an actual baseball game. Visualizing the input history data can help the user to understand the game. To align the coordinates of the virtual baseball game with the coordinates of the field model, multiple planar markers are distributed in the real field model. Manual measurement of the geometrical relationship among the markers is not required, so the user can easily start and enjoy this system. Fig. 19 shows images of the baseball game scenes. The user can observe such scenes from favorite viewpoints using this system.

8. Conclusion

We proposed a geometrical registration method for Augmented Reality with multiple planes in arbitrary positions and directions in the real world. In contrast with related works, the proposed method requires no time-consuming measurement of the planes or constraints on arranging the planes. Therefore, this system's algorithm can be implemented in a small space, like a tabletop, or in a large space. Furthermore, camera motion can be tracked frame-by-frame without using all frames of the input image sequence. This means the algorithm can be applied to on-line AR systems.

Appendix A. Stereo algorithm in designating 3d coordinate systems

$\boldsymbol{P_A}$ and $\boldsymbol{P_B}$ are projection matrices which are computed from four clicked points on the base plane and relates the base plane to the reference image A and B, respectively.

$$\boldsymbol{P_A} = \begin{bmatrix} p^A_{11} & p^A_{12} & p^A_{13} & p^A_{14} \\ p^A_{21} & p^A_{22} & p^A_{23} & p^A_{24} \\ p^A_{31} & p^A_{32} & p^A_{33} & 1 \end{bmatrix}, \quad \boldsymbol{P_B} = \begin{bmatrix} p^B_{11} & p^B_{12} & p^B_{13} & p^B_{14} \\ p^B_{21} & p^B_{22} & p^B_{23} & p^B_{24} \\ p^B_{31} & p^B_{32} & p^B_{33} & 1 \end{bmatrix} \tag{A.1}$$

We assume that a 2D coordinate of a point which is clicked on other plane in the reference image A and B are (x_A, y_A) and (x_B, y_B), respectively. The 3D coordinate of the point (X, Y, Z) can be computed by following equation.

$$\begin{bmatrix} (p^A_{31}x_A - p^A_{11}) & (p^A_{32}x_A - p^A_{12}) & (p^A_{33}x_A - p^A_{13}) \\ (p^A_{31}y_A - p^A_{11}) & (p^A_{32}y_A - p^A_{12}) & (p^A_{33}y_A - p^A_{13}) \\ (p^B_{31}x_B - p^B_{11}) & (p^B_{32}x_B - p^B_{12}) & (p^B_{33}x_B - p^B_{13}) \\ (p^B_{31}y_B - p^B_{11}) & (p^B_{32}y_B - p^B_{12}) & (p^B_{33}y_B - p^B_{13}) \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} p^A_{14} - p^A_{34}x_A \\ p^A_{14} - p^A_{34}y_A \\ p^B_{14} - p^B_{34}x_B \\ p^B_{14} - p^B_{34}y_B \end{bmatrix} \tag{A.2}$$

In the same way, 3D coordinates of the other clicked points on the plane can be computed. Then we define a 3D coordinate system on the plane so that one of the four points becomes the origin of the coordinate system. Therefore, we can define the 3D coordinates of the four points which are used for computing Homography in the same scale as the base plane's coordinate system.

Table A.1
Estimated size of the plane by stereo algorithm for 10 times.

	Ave.	Min.	Max.	Var.
Estimated size	20.28060668	18.45728195	21.60286124	0.713196008

For example, we arbitrary select a plane in Fig. 13(a, b), define the size of a plane as 20 and estimate the size of the other plane by above algorithm for 10 times. The true size of the plane is also 20. Table A.1 shows the results. We can find that the size of the other plane can correctly be estimated in the same scale as the base plane.

Appendix B. Computation of P_i^{WI} (between plane's coordinates and image coordinate system)

Since a 3D coordinate system is projected onto a 2D coordinate system by a 3×4 projection matrix, each 3D coordinate system based on plane i is projected to the image coordinate system by each projection matrix $\boldsymbol{P}_i^{WI}$.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \simeq \boldsymbol{P}_i^{WI} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad \boldsymbol{P}_i^{WI} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \tag{B.1}$$

As descried above, a 3D coordinate system is designated to each plane so that each plane becomes $Z = 0$. Therefore, Eq. (B.1) can be written as the following equation,

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \simeq \boldsymbol{P}_i^{WI} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \simeq \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \simeq \hat{\boldsymbol{P}}_i \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \simeq \boldsymbol{H}_i \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{B.2}$$

$$\therefore \hat{\boldsymbol{P}}_i \simeq \boldsymbol{H}_i$$

where, $\hat{\boldsymbol{P}}_i$ is a 3×3 matrix that lacks the third column vector of $\boldsymbol{P}_i^{WI}$. $\boldsymbol{H}_i$ is a homography of plane i between the $X - Y$ plane and the image plane ($x - y$ plane). As described in Eq. (B.2), $\hat{\boldsymbol{P}}_i$ is equivalent to homography $\boldsymbol{H}_i$. Thus, we estimate the third column vector of $\hat{\boldsymbol{P}}_i$ from $\boldsymbol{H}_i$. Actually, $\hat{\boldsymbol{P}}_i$ is divided into intrinsic and extrinsic parameters for the estimation.

$$\boldsymbol{P}_i^{WI} = \boldsymbol{A}[\boldsymbol{R}|\boldsymbol{t}] = \boldsymbol{A}[\boldsymbol{r}_1 \boldsymbol{r}_2 \boldsymbol{r}_3 \boldsymbol{t}] \tag{B.3}$$

$$\hat{\boldsymbol{P}}_i = \boldsymbol{A}[\boldsymbol{r}_1 \boldsymbol{r}_2 \boldsymbol{t}] = \boldsymbol{H}_i = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{B.4}$$

B.1. Estimation of intrinsic parameters

If an uncalibrated camera is used to capture input images, intrinsic parameters of the camera are estimated by homography $\boldsymbol{H}_i$. In our method, the matrix of intrinsic parameters $\boldsymbol{A}$ is defined as

$$\boldsymbol{A} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{B.5}$$

where, f is the focal length and (c_x, c_y) is the center point of the image. The skew and aspect ratios are fixed to 0 and 1, respectively. Therefore, we need to estimate f. Eq. (B.4) can be written as follows.

$$\boldsymbol{A}^{-1}\boldsymbol{H}_i = [\boldsymbol{r}_1 \boldsymbol{r}_2 \boldsymbol{t}] \tag{B.6}$$

The first and second vectors of $\boldsymbol{A}^{-1}\boldsymbol{H}_i$ corresponds to $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$.

Based on the property of rotation matrix $\boldsymbol{R}$, which is that the inner product of $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ is equal to 0, we can calculate the focal length f.

$$f^2 = \frac{(h_{11} - c_x h_{31})(h_{12} - c_x h_{32}) + (h_{21} - c_y h_{31})(h_{22} - c_y h_{32})}{-h_{31} h_{32}} \tag{B.7}$$

B.2. Estimation of extrinsic parameters

Extrinsic parameters of a camera consist of a rotation matrix $\boldsymbol{R}$ and a translation vector $\boldsymbol{t}$. Since $\boldsymbol{r}_1, \boldsymbol{r}_2$ (the first and second column vectors of $\boldsymbol{R}$), and $\boldsymbol{t}$ are already known, as shown in Eq. (B.6), we should estimate only $\boldsymbol{r}_3$. Then, also according to the property of $\boldsymbol{R}$, which is that the cross product of $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ becomes $\boldsymbol{r}_3$, we compute $\boldsymbol{r}_3$. Therefore, $\boldsymbol{R}$ is

$$\boldsymbol{R} = [\boldsymbol{r}_1 \boldsymbol{r}_2 (\boldsymbol{r}_1 \times \boldsymbol{r}_2)] = [\boldsymbol{r}_1 \boldsymbol{r}_2 \boldsymbol{r}_3] \tag{B.8}$$

References

[1] R.T. Azuma, A survey of augmented reality, Presence (1997) 355–385.
[2] R.T. Azuma, Recent advances in augmented reality, IEEE Computer Graphics and Applications 21 (6) (2001) 34–47.
[3] M. Billinghurst, S. Cambell, I. Poupyrev, K. Takahashi, H. Kato, W. Chinthammit, D. Hendrickson, Magic book: Exploring transitions in collaborative AR interfaces, Proceedings of SIGGRAPH 2000 (2000) 87.
[4] K. Satoh, S. Uchiyama, H. Yamamoto, H. Tamura, Robust vision-based registration utilizing bird's-eye with user's view, in: Proceedings of the ISMAR, 2003, pp. 46–55.
[5] T. Drummond, R. Cipolla, Real-time tracking of complex structures with online camera calibration, in: Proceedings of the BMVC, 1999, pp. 574–583.
[6] A.I. Comport, F. Marchand, E. Chaumette, A real-time tracker for markerless augmented reality, in: Proceedings of the ISMAR, 2003, pp. 36–45.
[7] K. Klein, T. Drummond, Sensor fusion and occlusion refinement for tablet-based AR, in: Proceedings of the ISMAR, 2004, pp. 38–47.
[8] K.W. Chia, A. Cheok, S.J.D. Prince, Online 6 DOF augmented reality registration from natural features, in: Proceedings of the ISMAR, 2002, pp. 305–313.
[9] G. Simon, A. Fitzgibbon, A. Zisserman, Markerless tracking using planar structures in the scene, in: Proceedings of the ISAR, 2000, pp. 120–128.
[10] G. Simon, M. Berger, Reconstructing while registering: a novel approach for markerless augmented reality, in: Proceedings of the ISMAR, 2002, pp. 285–294.
[11] G. Simon, M.O. Berger, Real time registration known or recovered multi-planar structures: application to AR, in: Proceedings of the BMVC, 2002, pp. 567–576.
[12] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2000.
[13] J. Shi, C. Tomasi, Good features to track, IEEE Conference on CVPR (1994) 593–600.
[14] Y. Uematsu, H. Saito, AR registration by merging multiple planar markers at arbitrary positions and poses via Projective Space, in: Proceedings of ICAT2005, 2005, pp. 48–55.
[15] Y. Uematsu, H. Saito, AR baseball presentation system based on registration with multiple markers, in: Proceedings of ACM SIGCHI International Conference on Advanced Computer Entertainment technology (ACE2006), 2006, pp. 48–55.
[16] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, K. Tachibana, Virtual object manipulation on a table-top AR environment, in: Proceedings of the ISAR, 2000, pp. 111–119.
[17] Y. Genc, S. Riedel, F. Souvannavong, C. Akinlar, N. navab, Marker-less tracking for AR: a learning-based approach, in: Proceedings of the ISMAR, 2002, pp. 295–304.
[18] A. Henrysson, M. Billinghurst, M. Ollila, Virtual object manipulation using a mobile phone, in: Proceedings of the ICAT, 2005, pp. 164–171.
[19] D. Wagner, M. Billinghurst, D. Schmalstieg, How real should virtual characters be ? in: Proceedings of ACM SIGCHI International Conference on Advanced Computer Entertainment technology (ACE2006), 2006.