유클리드 카메라 보정을 하지 않는 비디오 기반 증강현실

서 용덕

서강대학교 영상대학원 미디어공학과

요 약

An algorithm is developed for augmenting a real video with virtual graphics objects without computing Euclidean information. Real motion of the camera is obtained in affine space by a direct linear method using image matches. Then, virtual camera is provided by determining the locations of four basis points in two input images as initialization process. The four pairs of 2D location and its 3D affine coordinates provide Euclidean orthographic projection camera through the whole video sequence. Our method has the capability of generating views of objects shaded by virtual light sources, because we can make use of all the functions of the graphics library written on the basis of Euclidean geometry. Our novel formulation and experimental results with real video sequences are presented.

제 1 절 Introduction

In video based augmented reality, the major topic is to synthesize views of a 3D graphic model according to the motion of a *real* camera and mixing them with a real scene video in real time [1]. The virtual objects are generally 3D graphics models and their rendered views by graphics machines are overlayed on real-video frames. Applications include image-guided or assisted surgery or its training [2, 3], assembly, maintenance and repair [4, 5], simulating light conditions of an outdoor building [6], virtual studio systems for commercial/public broadcasting [7], etc. These systems enrich reality with computer-generated views, cooperating with computer vision techniques: estimating the camera parameters [8], resolving occlusion between a virtual object and a real object [9, 10], and correcting the registered location of a graphic object dynamically [2]. An extensive survey can be found in [11]. Among the techniques, camera calibration has been a prerequisite for embedding virtual objects into video frames because the geometric relationship among physical objects, virtual objects and the camera needs be established to get correct results.

On the other hand, the pioneering work of Kutulakos and Vallino [12] employed a computer vision technology to bring forth the weakly-calibrated method or calibration-free method for augmenting real video *without* explicit camera calibration and Euclidean reconstruction. Camera motion and scene structure are recovered by affine structure from motion algorithm. An information similar to Euclidean is supplied by specifying a set of virtual fiducial points on the basis of the image contents. Then the image locations are directly utilized in the virtual view synthesis. However, the system had a limitation that it could not generate a shaded view of graphics objects nor specify a lighting source through a ready-made standard graphics library because theories and packages of computer graphics are all written on the basis of *Euclidean geometry*. Indeed, this is one of the critical reasons that previous augmented reality systems have relied on the calibration of their cameras. Chen *et. al.* [13] proposed a method for overlaying graphics on the images. A cuboid, specified through a user-interface, is utilized to connect real camera to computer graphics world. However, they only dealt with a fixed (not moving) camera. Seo and Hong [14] extended the method of [12] to the case of perspective camera. It proposed a method of virtual camera decomposition so that the process of virtual perspective view synthesis can be done under the Euclidean geometry.

In this paper, we further extend the work of [14] so that an orthographic camera model can be used as the *virtual* camera. We attach a virtual camera of the weak-perspective model to the real *affine* camera so that the virtual camera moves in the same way as does the real camera. Our method decomposes the virtual camera into Euclidean motion components and internal calibration components, which enables us to make full use of a well-developed computer graphics package and to synthesize graphics objects that are shaded and colored by a lighting source.

Section 2 provides some notations and preliminaries of camera models and a brief review of affine structure from motion algorithm. Section 3 presents a short description of our algorithm. Section 4 illustrates how the virtual world coordinate system is embedded into the recovered affine space. It gives the method of computing a weak-perspective virtual camera by specifying three/four basis points in two control images. Section 5 shows how a corresponding virtual camera is computed, given a video image, and graphics views are rendered using a general graphics machine . Section 6 illustrates our experimental results. Finally, concluding remarks are given in Section 7.

제 2 절 Preliminaries

There are two kinds of orthographic cameras: affine camera and weak-perspective camera. The weak-perspective camera retains the features of the Euclidean projection model;

it consists of rotation, translation and scaling. On the contrary, the affine camera assumes affine geometry and affine ambient space.

Every vector is denoted by a column vector. We represent an image point by $\boldsymbol{x} = [u, v]^{\mathrm{T}}$ and a 3D point by $\boldsymbol{X} = [X, Y, Z, 1]$. A 3D point $\boldsymbol{X}$ projects to the image point $\boldsymbol{x}$:

$$\boldsymbol{x} = \boldsymbol{K}_{2\times2}\boldsymbol{I}_{2\times4}\boldsymbol{D}_{4\times4}\boldsymbol{X} \quad (1)$$

$$= \begin{bmatrix} f_X & 0 \\ 0 & f_Y \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}_3^{\mathrm{T}} & 1 \end{bmatrix} \boldsymbol{X}, \quad (2)$$

where $\boldsymbol{R} = [\boldsymbol{r}_1\, \boldsymbol{r}_2\, \boldsymbol{r}_3]^{\mathrm{T}}$, $\boldsymbol{r}_i^{\mathrm{T}}$ being i-th row vector, and $\boldsymbol{t} = [t_X, t_Y, t_Z]^{\mathrm{T}}$ denote the transformation from the world coordinate system to the camera coordinate system.Notice that $\boldsymbol{X}$ is represented in the world coordinate system. Thus, we have the following:

Definition 1 *A weak perspective camera* $\boldsymbol{P}_{\mathrm{wp}}$ *is defined by a* 2×4 *matrix as follows:*

$$\boldsymbol{P}_{\mathrm{wp}} = \begin{bmatrix} f_X & 0 \\ 0 & f_Y \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_1^{\mathrm{T}} & t_X \\ \boldsymbol{r}_2^{\mathrm{T}} & t_Y \end{bmatrix} \quad (3)$$

$$= \begin{bmatrix} \boldsymbol{b}_1^{\mathrm{T}} & t'_X \\ \boldsymbol{b}_2^{\mathrm{T}} & t'_Y \end{bmatrix} \quad (4)$$

$$= \begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \boldsymbol{a}_3 & \boldsymbol{a}_4 \end{bmatrix}. \quad (5)$$

Notice that $\boldsymbol{a}_i$ is the i-th column of $\boldsymbol{P}_{\mathrm{wp}}$, and $\boldsymbol{b}_1^{\mathrm{T}} = f_X \boldsymbol{r}_1^{\mathrm{T}}$ and $\boldsymbol{b}_2^{\mathrm{T}} = f_Y \boldsymbol{r}_2^{\mathrm{T}}$ satisfy the **orthogonality constraint:**

$$\boldsymbol{b}_1 \cdot \boldsymbol{b}_2 = 0, \quad (6)$$

where $\cdot$ represents the operation of the inner product. Equation (1) can now be simply written as $\boldsymbol{x} = \boldsymbol{P}_{\mathrm{wp}}\boldsymbol{X}$, where $\boldsymbol{X}$ is represented in the world coordinate system.

The ratio of the two focal lengths f_X and f_Y are called the *aspect ratio*: $\gamma = \frac{f_X}{f_Y}$. When $\gamma = 1$, the camera is denoted by $\boldsymbol{P}_{\mathrm{wp1}}$ in this paper. Notice that the orthogonal constraint is still satisfied for $\boldsymbol{P}_{\mathrm{wp1}}$.

Affine camera model is a generalization of the weak-perspective camera model and may be defined as follows:

Definition 2 *An affine camera is defined by a general* 2×4 *matrix of rank 2 and denoted by* $\boldsymbol{P}_A$.

Replacing $\boldsymbol{K}_{2\times2}$ with a non-singular 2×2 matrix and $\boldsymbol{D}$ with a matrix of general 3D affine transformation in equations (1) and (2), we obtain the general form of the affine camera $\boldsymbol{P}_A$.

2.1 Affine Structure from Motion

Since the recovery of affine structure and motion is not main topic of this paper, here we briefly introduce affine structure from motion algorithms. It is well known that given point matches from two orthographic or scaled orthographic views, affine structure of a scene can be found [15]. Affine epipolar geometry has been studied [16] and affine structure and motion can be obtained through factorization methods [17, 18]. Details can also be found in various literatures [19, 20, 21, 22, 23].

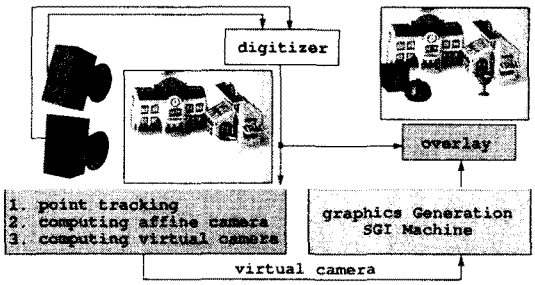

그림 1: Overall configuration. Point matches are obtained for the recovery of affine motion and structure. The basis points specified in the initial stereo-pair images yield our virtual weak-perspective cameras, through which a shaded graphics objects by a lighting source can be rendered.

Let us suppose that we have point matches from input images and let $\boldsymbol{x}_{ki}$ be the i-th point of k-th image. Then from the theory of affine structure from motion, we can estimate the k-th *affine* camera matrix $\boldsymbol{P}_{A,k}$ for each of the input images and the i-th *affine* coordinate vector $\boldsymbol{X}_A$ that satisfy the following projection equation:

$$\boldsymbol{x}_{ki} = \boldsymbol{P}_{A,k}\boldsymbol{X}_{A,i} \quad \text{for} \quad i = 1, ..., N,\ k = 1, ..., K, \quad (7)$$

where N is the number of point correspondences and K is the number of input images. In this paper $K = 2$ because of the stereo image input. Note that this reconstruction is up to an unknown affine transformation [15].

제 3 절 Algorithm Overview

Here our algorithm is roughly introduced when the virtual camera is modeled to be $\boldsymbol{P}_{\mathrm{wp}}$. The case of $\boldsymbol{P}_{\mathrm{wp1}}$ is slightly different and explained in Section 4. Figure 1 briefly shows our method. Notice that we have a stereo camera system.

1. The two affine camera matrices and affine structure are computed from the image correspondences.

2. Embedding: We insert the graphics world coordinate system into the recovered affine space by specifying four basis points in the two initial images, respectively, using affine epipolar geometry.

   Virtual graphics objects are then placed with respect to the world coordinate system. Corresponding virtual camera matrices are computed, too, using the basis points. Notice that this initialization step is performed just once.

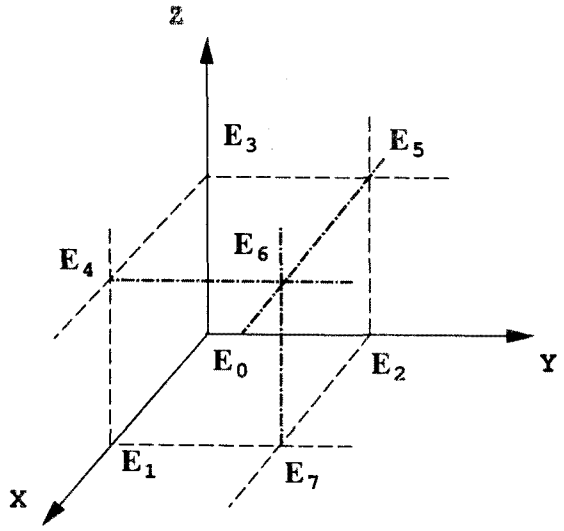

그림 2: Vertices of the world coordinate system.

3. Rendering: As the real camera moves, the affine camera pair is computed, the virtual camera pair is updated, and the views of graphics objects are rendered and overlayed on the real video images.

Because the virtual cameras are of Euclidean form, they enables us to utilize the full functionality of our graphics package. That is, we can define various characteristics of light sources including their colors and positions even though we recover only the affine camera and structure of the real camera pair.

제 4 절 Embedding Algorithm

The first thing we have to do for augmentation is to define the weak-perspective virtual camera attached to the real video camera by specifying some basis points of the graphics world coordinate system. Sections 4.1 - 4.6 provide an image-based embedding method for $P_{\rm wp}$ in order to insert the world coordinate system by specifying its $3\frac{1}{2}$ basis points in the first image pair on the ground of affine epipolar geometry. Then, Section 4.7 provides how to embed the world coordinate system and compute the virtual camera by specifying three basis points in the image pair. Note that in [12] they specified *four* points in the image pair using only affine epipolar constraint.

4.1 Embedding in the Left Image

We first specify *three* image locations $\{\boldsymbol{x}_0^{\rm b}, \boldsymbol{x}_1^{\rm b}, \boldsymbol{x}_2^{\rm b}\}$ of the basis points $\{\boldsymbol{E}_0, \boldsymbol{E}_1, \boldsymbol{E}_2\}$ in the left image of the first stereo pair for initialization. Specifically, $\boldsymbol{E}_0 = [0,0,0,1]^{\rm T}$ is the origin and $\boldsymbol{E}_1 = [1.0,0,1]^{\rm T}$, $\boldsymbol{E}_2 = [0,1,0,1]^{\rm T}$ and $\boldsymbol{E}_3 = [0,0,1,1]^{\rm T}$ are three basis points on the x, y and z axes, respectively, of the world coordinate system. Figure 2 illustrates this. Using the notation of Equation (5), each column $\boldsymbol{a}_i$ of the left virtual camera $\boldsymbol{P}_{\rm wp}^{L}$ is given from the relationship $\boldsymbol{x}_k^{\rm b} = \boldsymbol{P}_{\rm wp}^{L}\boldsymbol{E}_k$ as follows:

$$\boldsymbol{a}_4 = \boldsymbol{x}_0^{\rm b}, \tag{8}$$

$$\boldsymbol{a}_j = \boldsymbol{x}_j^{\rm b} - \boldsymbol{x}_0^{\rm b} \quad \text{for } j = 1,2. \tag{9}$$

The third column, $\boldsymbol{a}_3$, is still left unknown. If an arbitrary image point is given for $\boldsymbol{E}_3$, it will make our virtual camera a general affine camera like [12]. In order to force the virtual camera to become a weak-perspective camera, we utilize the *orthogonality constraint* of Equation (6). Now we use the notation of Equation (4). Since we have determined $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$, we may write $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ as

$$\boldsymbol{b}_1 = [a_{11}, a_{12}, a_{13}]^{\rm T} \quad \text{and} \quad \boldsymbol{b}_2 = [a_{21}, a_{22}, a_{23}]^{\rm T}, \tag{10}$$

where a_{ki} is the k-th element of $\boldsymbol{a}_i$. Then from the orthogonality constraint, we have

$$0 = \boldsymbol{b}_1 \cdot \boldsymbol{b}_2 = a_{11}a_{21} + a_{12}a_{22} + a_{13}a_{23}, \tag{11}$$

which gives a constraint equation for the elements of the third column $\boldsymbol{a}_3 = [a_{13}, a_{23}]^{\rm T}$:

$$a_{23} = -\frac{a_{11}a_{21} + a_{12}a_{22}}{a_{13}}. \tag{12}$$

Since $\boldsymbol{x}_3^{\rm b} = \boldsymbol{P}_{\rm wp}^{L}\boldsymbol{E}_3 = \boldsymbol{a}_3 + \boldsymbol{x}_0^{\rm b}$, the coordinates of the last image point $\boldsymbol{x}_3^{\rm b} = [u_3^{\rm b}, v_3^{\rm b}]^{\rm T}$ must satisfy the hyperbolic equation

$$v_3^{\rm b} - v_0^{\rm b} = \frac{a_{11}a_{21} + a_{12}a_{22}}{u_3^{\rm b} - u_0^{\rm b}}. \tag{13}$$

Finally, we choose an appropriate value of $u_3^{\rm b}$ on the hyperbola and compute $\boldsymbol{P}_{\rm wp}^{L}$ to complete the embedding step for the left image:

$$\boldsymbol{P}_{\rm wp}^{L} = \begin{bmatrix} \boldsymbol{x}_1^{\rm b} - \boldsymbol{x}_0^{\rm b} & \boldsymbol{x}_2^{\rm b} - \boldsymbol{x}_0^{\rm b} & \boldsymbol{x}_3^{\rm b} - \boldsymbol{x}_0^{\rm b} & \boldsymbol{x}_0^{\rm b} \end{bmatrix}. \tag{14}$$

Figure 3 shows an example of the result of our embedding procedure. Figure 3(a) is the first left image where three image locations were specified as explained. The three points provided a hyperbolic equation, which is also shown in the figure. Finally, the u coordinate of the last image point $\boldsymbol{x}_3^{\rm b}$ was selected. The wire-cube is the projection of a graphics unit cube through the computed virtual camera.

4.2 Embedding in the Right Image

In the right image of the first stereo pair, we also have to specify the image locations of the basis points of the world coordinate system. In the embedding step for the left image, we are provided with just one constraint for building up a weak-perspective virtual camera. However, in the second

image, we have an additional constraint – affine epipolar geometry.

Let $P_A^L = [c_1, c_2, c_3, c_4]^\top$ and $P_A^R = [c_1', c_2', c_3', c_4']^\top$ be the left and right affine camera matrices for the two initializing images. For a point x in the left image, its epipolar line in the right has the equation [16]

$$x'(\zeta) = (G(x - c_4) + c_4') + \zeta\,(c_3' - Gc_3)\,, \qquad (15)$$

where $G = [c_1', c_2'][c_1, c_2]^{-1}$ is a 2×2 matrix and ζ is a scalar parameter. This equation implies that a corresponding point x' of x must be on its epipolar line. The four lines in Figure 3(b) are the epipolar lines of the four points $\{x_0^b, x_1^b, x_2^b, x_3^b\}$ specified in Section 4.1.

What we now have to do is to choose *three* image locations $\{x_0^{b'}, x_1^{b'}, x_2^{b'}\}$ on their epipolar lines $\{l_0, l_1, l_2\}$. As soon as the three points are determined, they also provide a hyperbolic curve in the second image as shown in Figure 3(b). At this time, we need not choose $u_3^{b'}$ because the last point $x_3^{b'}$ is determined as the intersection point of the hyperbola and the epipolar line of x_3^b. When there are two intersection points, we just need to select one of them. If there is no intersection, then we have to adjust the locations of the three specified pairs in order to get at least one.

Computing P_{wp}^R with the four image locations of the basis points finishes our embedding step (Equation (14)). Figure 3 also shows a wire-cube, which is the image of the unit cube through the computed weak-perspective virtual camera.

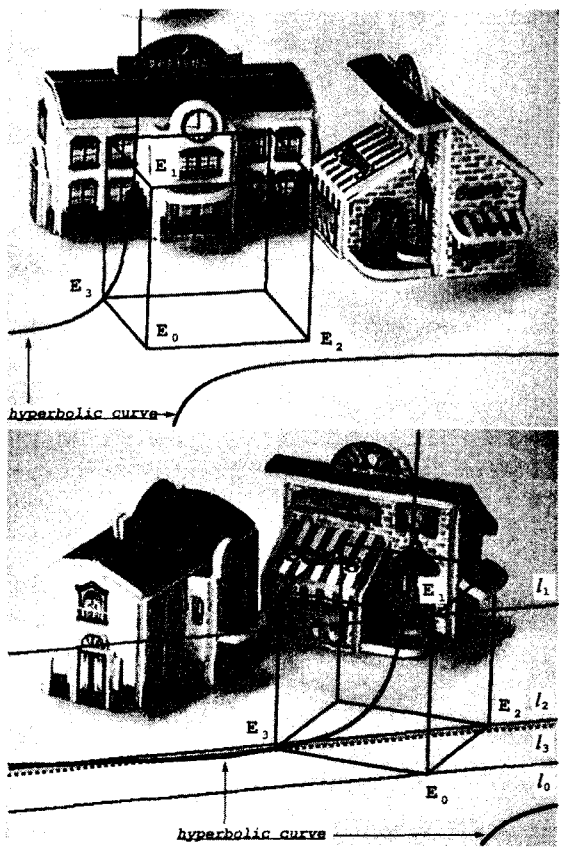

그림 3: Embedding step. (Left image) Three basis points $\{E_0, E_1, E_2\}$ were selected. In selecting these points, we tried to place the basis frame on the same floor that the toy houses existed. (Right image) On the corresponding epipolar lines, three basis points were selected.

4.3 Affine Structure for the Control Points

We compute the affine coordinates $\{X_i^b\}_{i=0}^3$ of the four pairs of the control points $\{(x_i^b, x_i^{b'})\}_{i=0}^3$ in order to enable the virtual camera to move according to the motion of the real camera. In other words, the affine coordinates form an imaginary link between the real camera and the virtual camera.

Let ζ_i be the selected parameter of Equation (15) for the i-th pair $(x_i^b, x_i^{b'})$. Then, corresponding 3D affine coordinates $X_i^b = [X_i^b, Y_i^b, Z_i^b, 1]^\top$ are given as follows:

$$Z_i^b = \zeta_i \qquad (16)$$

$$[X_i^b \quad Y_i^b]^\top = G^{-1}\left(x_i^b - c_4 - Z_i c_3\right). \qquad (17)$$

The weak-perspective virtual camera at an arbitrary time k will be determined in Section 5.1 using the affine coordinates $\{X_i^b\}_{i=0}^3$.

4.4 Virtual Camera Decomposition

To use the matrices of the virtual cameras P_{wp}^L and P_{wp}^R in graphics rendering, we decompose each of them into three parts – K, R and t – as in equations (1) and (2). We will use the notations of equations (3) - (5). The following formulation is for one of the two virtual cameras.

The scale components f_X and f_Y are given as follows:

$$f_X = \|b_1\| \quad \text{and} \quad f_Y = \|b_2\|, \qquad (18)$$

which determine the calibration matrix K. Then we have an orthonormal pair

$$r_1 = b_1/f_X \quad \text{and} \quad r_2 = b_2/f_Y, \qquad (19)$$

which define a rotation matrix $R = [r_1, r_2, r_1 \times r_2]^\top$. What is left is the translation vector $t = [t_X, t_Y, t_Z]^\top$. The first two elements, t_X and t_Y, of t are given by $K^{-1}a_4$ but the last element t_Z can be determined at our disposal. Note that algebraically speaking, a view through an orthographic camera has nothing to do with the depth component. However, in a physical situation, one strict constraint is that every object must be in front of the camera, and according to our camera model we choose a positive depth t_Z in practice so that every graphics object lie in front of the virtual camera.

The calibration part $\boldsymbol{K}$ will be utilized in setting the viewing volume of the virtual camera in graphics machine and the motion parts $\boldsymbol{R}$ and $\boldsymbol{t}$ are for modelview transformation. Details are given in Section 5.2.

4.5 Projection through Virtual Camera

During the procedure of embedding, we specified basis points of the world coordinate system. To facilitate this procedure, we need to see the results of our embedding. Given a virtual camera $\boldsymbol{P}_{\text{wp}}$ we compute the image locations $\boldsymbol{x}_i^{\text{b}} = \boldsymbol{P}_{\text{wp}}\boldsymbol{E}_i$ of the vertices of the unit cube and examine our embedding result; Here, $\boldsymbol{P}_{\text{wp}}$ denotes one of $\boldsymbol{P}_{\text{wp}}^L$ and $\boldsymbol{P}_{\text{wp}}^R$. An example (the wire-cube) is shown in Figure 3.

4.6 Placing Graphics Objects

The location and pose of a 3D graphics objects are defined with respect to the world coordinate system that is now embedded in the affine camera coordinate system. In this way, the characteristics of the graphics objects, such as color, specularity of surfaces, etc, can be defined by usual graphics modeling. The light sources are also configured with respect to the embedded world coordinate system.

4.7 Decomposition of $\boldsymbol{P}_{\text{wp1}}$

When the aspect ratio is one ($f_X = f_Y$), the number of necessary basis points becomes three. As we did in the previous sections, let us specify in the left image three basis points for $\boldsymbol{E}_0, \boldsymbol{E}_1$, and $\boldsymbol{E}_2$. Then the camera $\boldsymbol{P}_{\text{wp1}}^L$ can be written as

$$\boldsymbol{P}_{\text{wp1}}^L = \begin{bmatrix} u_1^b - u_0^b & u_2^b - u_0^b & a_{31} & u_0 \\ v_1^b - v_0^b & v_2^b - v_0^b & a_{32} & v_0 \end{bmatrix}, \quad (20)$$

where a_{31} and a_{32} are left to be computed. From the orthogonality constraint, we have one equation for a_{31} and a_{32}

$$(u_1^b - u_0^b)(v_1^b - v_0^b) + (u_2^b - u_0^b)(v_2^b - v_0^b) + a_{13}a_{23} = 0. \quad (21)$$

Another constraint equation is from the fact that the aspect ratio is one:

$$(u_1^b - u_0^b)^2 + (u_2^b - u_0^b)^2 + a_{13}^2 = (v_1^b - v_0^b)^2 + (v_2^b - v_0^b)^2 + a_{23}^2. \quad (22)$$

Therefore, we are end up with four candidate set of solutions for a_{13} and a_{23}.

$$a_{13} = \pm\left(\frac{-(\beta-\gamma) \pm \sqrt{(\beta-\gamma)^2 + 4\alpha^2}}{2}\right)^{1/2} \quad (23)$$

$$a_{23} = -\frac{\alpha}{a_{13}}, \quad (24)$$

where

$$\alpha = (u_1^b - u_0^b)(v_1^b - v_0^b) + (u_2^b - u_0^b)(v_2^b - v_0^b) \quad (25)$$

$$\beta = (u_1^b - u_0^b)^2 + (u_2^b - u_0^b)^2 \quad (26)$$

$$\gamma = (v_1^b - v_0^b)^2 + (v_2^b - v_0^b)^2. \quad (27)$$

Among the four set of candidates, we need to choose an appropriate one. Then, the camera is decomposed to be used in Euclidean rendering pipeline as explained in Section 4.4. For the right side camera, we choose a set of three basis points on the corresponding epipolar lines, respectively.

One advantage of this approach is that one needs specify only *three* basis points rather than $3\frac{1}{2}$ with the help of epipolar geometry. However, it provides four candidate sets of solutions among which one must be selected.

제 5 절 Rendering Algorithm

Graphics rendering for the k-th time step consists of two steps:

1. Computation of the k-th virtual cameras.
2. Generation of a corresponding view of graphics objects and overlaying it on the video image.

5.1 Transferring the Virtual Camera

As one of the video camera (or the stereo-rig) moves, its affine camera matrix changes and, accordingly, its virtual camera must be computed or transferred to the k-th time step. The procedure is as follows:

1. Find the image matches $\{\boldsymbol{x}_{A,ki}\}_{i=1}^N$ in $\mathcal{I}_k^L$ and $\mathcal{I}_k^R$ through a tracking.
2. Compute the affine camera matrix $\boldsymbol{P}_{A,k}^L$ and $\boldsymbol{P}_{A,k}^R$ using the affine re-projection equation:

   $$\boldsymbol{x}_{A,ki}^n = \boldsymbol{P}_{A,k}^n \boldsymbol{X}_{A,i},\ i = 1, ..., N,\ n \in \{L, R\}. \quad (28)$$

   Note that the affine coordinates $\boldsymbol{X}_{A,i}$ were already obtained at the initialization step using affine structure from motion algorithm.
3. Project the 3D affine coordinates $\{\boldsymbol{X}_i^{\text{b}}\}_{i=0}^3$ of the control points into the k-th image pair through $\boldsymbol{P}_{A,k}^L$ and $\boldsymbol{P}_{A,k}^R$, respectively:

   $$\boldsymbol{x}_i^{\text{b}} = \boldsymbol{P}_{A,k}^L \boldsymbol{X}_i^{\text{b}}, \boldsymbol{x}_i^{\text{b}'} = \boldsymbol{P}_{A,k}^R \boldsymbol{X}_i^{\text{b}},\ i = 0, ..., 3. \quad (29)$$

4. Compute the virtual cameras $\boldsymbol{P}_{\text{wp},k}^L$ and $\boldsymbol{P}_{\text{wp},k}^R$ using the projections $\{\boldsymbol{x}_i^{\text{b}}\}_{i=0}^3$ and $\{\boldsymbol{x}_i^{\text{b}'}\}_{i=0}^3$, respectively, of the control points (Equation (14)).
5. Decompose $\boldsymbol{P}_{\text{wp},k}^L$ into $\boldsymbol{K}_k^L$, $\boldsymbol{R}_k^L$ and $\boldsymbol{t}_k^L$, and $\boldsymbol{P}_{\text{wp},k}^R$ into $\boldsymbol{K}_k^R$, $\boldsymbol{R}_k^R$ and $\boldsymbol{t}_k^R$.

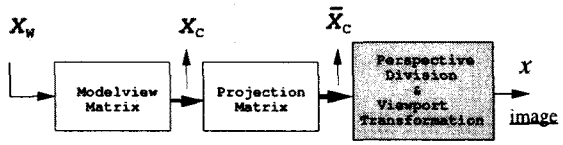

그림 4: Stages of coordinate transformation in a graphics machine.

5.2 Graphics Rendering

Figure 4 shows the three stages of coordinate transformation in our graphics machine for making a view of graphics objects through the virtual camera $P_{\mathrm{wp},k}$. The rotation R_k and translation t_k define the modeling transformation and K_k gives the orthographic viewing volume. The modelview matrix M_k for modeling transformation is given as $M_k = \begin{bmatrix} R_k & t_k \\ 0_3^\top & 1 \end{bmatrix}$, which converts a 3D point X_W, defined with respect to the world coordinate system, to the point X_C with respect to the camera coordinate system. After that, a projection matrix is applied to yield clip coordinates $\bar{X}_C$ by removing any part of the graphic objects outside the viewing volume which is a rectangular parallelepiped defined by K_k.

제 6 절 Experiments

We implemented and tested our method and Figure 5 shows the augmented images. The wire-cubes are the result of the embedding shown in Figure 3 of Section 4. With respect to the embedded world coordinate system, we inserted three graphics objects and the images in Figure 5 show the results.

Figure 6 shows another augmentation result. The video sequence consisted of 251 images of 720 × 486 size. We selected eight lines in the first image of the sequence and the lines were then tracked automatically in the sequence. Eight intersection points (indicated by arrows in Figure 6) were then used for affine structure and motion computation. In the embedding step, basis points were specified using the affine epipolar geometry (epipolar lines) and the orthogonality constraint (hyperbolas) as illustrated in Figure 6. Graphics objects were located with respect to the embedded world coordinate system. In addition, we made the sphere move linearly in the y direction. Three frames of the augmentation result are shown.

Figure 7 illustrates another experimental result.

제 7 절 Conclusion

We proposed a method for weakly calibrated augmented reality of an orthographic projection camera. In order to make full use of computer graphics package, we designed a weak-perspective virtual camera which was obtained by applying

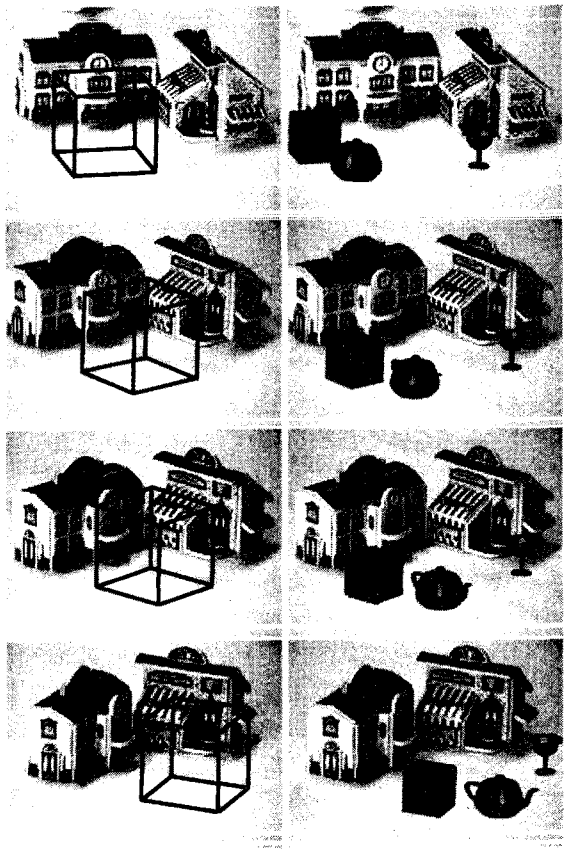

그림 5: Experiment 1.

the orthogonality constraint. The image-based embedding enabled us to compute the (virtual) intrinsic and extrinsic motion of the virtual camera, without metric camera calibration. The orthographic projection model is useful especially because all the numerical computation can be done in a linear formulation, but also the result is robust for the practical level of image corner point detection.

참고 문헌

[1] Ronald T. Azuma. A survey of augmented reality. *PRESENCE: Teleoperations and Virtual Environments*, 6(4):355–385, August 1997.

[2] Michael Bajura and Ulrich Neumann. Dynamic registration correction in video-based augmented reality systems. *Computer Graphics and Applications*, pages 52–60, 1995.

[3] W. Grimson, G. Ettinger, S. White, P. Gleason, T. Lozano-Perez, W. Wells, and R. Kikinis. Evaluating and validating an automated registration system for

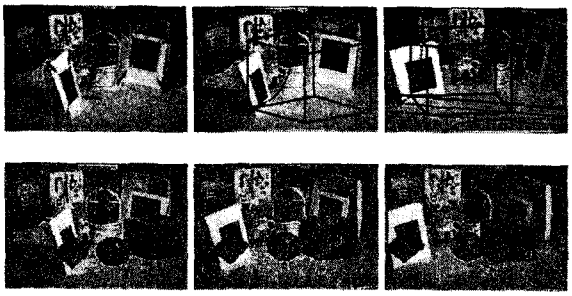

그림 6: Experiment 2.

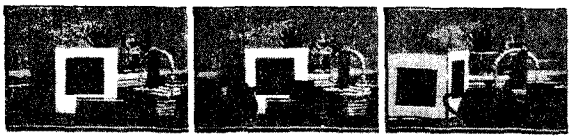

그림 7: Input and output of experiment 3.

enhanced reality visualization in srugery. In *Computer Vision, Virtual Reality and Robotics in Medicine '95 (CVRMed'95)*, pages 3–12, April 1995.

[4] M. Tuceryan, D. Greer, R. Whitaker, D. Breen, C. Crampton, E. Rose, and K. Ahlers. Calibration requirements and procedures for a monitor-based augmented reality system. *IEEE Transactions on Visualization and Computer Graphics*, 1(3):255–273, September 1995.

[5] Rajeev Sharma and Jose Molineros. Computer vision-based AR for guiding manual assembly. *Presence: Teleoperators and Virtual Environments*, 6(3):292–317, June 1997.

[6] M.-O. Berger, G. Simon, S. Petitjean, and B. Wrobel-Dautcourt. Mixing synthesis and video images of outdoor environments: Application to the bridges of paris. In *ICPR'96*, pages 90–94, 1996.

[7] S.W. Park, Yongduek. Seo, and Ki-Sang Hong. Real-time camera calibration for virtual studio. *Real-Time Imaging*, (6):433–448, Dec. 2000.

[8] Hagen Schumann, Silviu Burtescu, and Frank Siering. Applying augmented reality techniques in the field of interactive collaborative design. In *SMILE Workshop on 3D structure from multiple images of large-scale environments, in conjunction with ECCV'98*, June 1998.

[9] M.-O. Berger. Resolving occlusion in augmented reality: a contour based approach without 3D reconstruction. In *CVPR'97*, 1997.

[10] Stefan Grosskopf and Peter Neugebauer. The use of reality models in augmented reality applications. In *SMILE Workshop on 3D structure from multiple images of large-scale environments, in conjunction with ECCV'98*, June 1998.

[11] Y. Ohta and H. Tamura, editors. *Mixed Reality - Merging real and virtual worlds*. Springer Verlag, 1999.

[12] K. N. Kutulakos and J.R. Vallino. Calibration-free augmented reality. *IEEE Trans. Visualization and Computer Graphics*, 4(1):1–20, 1998.

[13] C.-S. Chen, C.-K. Yu, and Y. Hung. New calibration-free approach for augmented reality based on parameterized cuboid structure. In *Proc. 7th Int. Conf. on Computer Vision*, 1999.

[14] Y. Seo and K.S. Hong. Calibration-free augmented reality in perspective. *IEEE Trans. Visualization and Computer Graphics*, pages 346–359, 2000.

[15] J.J. Koenderink and A.J. Van Doorn. Affine structure from motion. *J. of the Optical Society of America*, 8:377–385, 1991.

[16] L.S. Shapiro, A. Zisserman, and M. Brady. Motion from point matches using affine epipolar geometry. In *ECCV'94*, 1994.

[17] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9:137–154, 1992.

[18] S. Seitz and C.Dyer. Complete scene structure from four point correspondences. In *Proc. 5th Int. Conf. on Computer Vision*, pages 330–337, 1995.

[19] J. M. Lawn and R. Cipolla. Robust egomotion estimation from affine motion-parallax. In *Proc. 3rd European Conf. on Computer Vision*, volume I, pages 205–210. Springer–Verlag, 1994.

[20] L. Quan. Self-calibration of an affine camera from multiple views. *Int. Journal of Computer Vision*, 19(1):93–105, 1996.

[21] L. Quan and T. Kanade. Affine structure from line correspondences with uncalibrated affine cameras. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(8), August 1997.

[22] F. Kahl and A. Heyden. Structure and motion from points, lines and conics with affine cameras. In *Proc. 6th European Conf. on Computer Vision*, 2000.

[23] K. Åström, A. Heyden, F. Kahl, and M. Oskarsson. Structure and motion from lines under affine projections. In *Proc. 7th Int. Conf. on Computer Vision, Kerkyra, Greece*. IEEE Computer Society Press, 1999.