Composition of 3D Graphic Objects and Panorama

Chu-Song Chen [1], Wen-Ten Hsieh [1,2]

[1] *Institute of Information Science, Academia Sinica, Taipei, Taiwan*

[2] *Department of Computer Science and Information Engineering, National Taiwan University*

Email: song@iis.sinica.edu.tw

Abstract

In this paper, we propose an efficient and easy-to-implement method for the interactive placement of virtual objects in a panorama. In particular, we developed a systematic approach for estimation of the camera parameters using a single panorama with reasonable human-computer interactions.

Key words: 3D/2D Composition, Camera Pose Estimation, Panorama, Augmented Reality, Virtual Reality.

1. Introduction

There are two common approaches to build a VR world: the *image-based approach* (e.g., Quick-Time VR, Surround Video, Real VR, and IPIX) and the *model-based approach* (e.g., AutoCAD, 3D Studio). The panorama is the most popular image-based approach which creates an omni-directional view by seaming photographs. Image-based approach can generate photo-realistic scenes. However, it is difficult to allow the user to view the scene from arbitrary viewing directions. Model-based approaches construct the 3D models of the real world objects and then generate views by rendering the 3D models. It allows the users to interactively view the virtual world from arbitrary viewing directions. However, most model-based approaches use manually-created virtual objects, and thus the generated virtual world is usually not realistic enough for sophisticated objects.

A hybrid VR system is a good solution to exploit both the advantages of these two types of approaches. In this paper, we proposed a simple and systematic method to combine the panorama generated by an image-based approach and the virtual objects generated with a model-based approach. To solve this image-composition problem, two major issues have to be considered: (1) *geometry consistency*, and (2) *photometry consistency*. In this paper, we focus on the problem of geometry consistency. However, our system can also generate realistic shadows of the virtual objects by setting light positions manually. In particular, we allow a user to interactively place the 3D graphic objects in arbitrary positions of the 3D world photographed in a single panorama in a geometrically-reasonable way.

Many methods have been proposed for the composition of virtual objects and images or videos [1][4][8]. However, no approaches are suitable for composition of virtual 3D objects and panoramas because most of the above approaches have to use the disparity information generated by the point correspondences among images. Nevertheless, a panorama is a wide-angle *static* image, while there is no disparity information allowed to be used in a static image. Although some methods can extract 3D structures from panoramas [7][9], at least two panoramas are required.

2. Criterions of Specific Shape

In this paper, we developed a method which can insert virtual 3D objects in a single panorama. To insert 3D graphic objects into a panorama while maintaining their geometry consistency, it is necessary to know the rigid transformation between the object coordinate system and the coordinate system defined by the panorama. This problem is referred to as the *camera pose estimation* in the computer vision community. Basically, estimation of the camera parameters from a single image is ill-posed if there is no additional constraints on the reference objects.

What we try to solve in this paper is to estimate camera parameters using a single panorama. To provide suitable geometrical constraints, our basic idea is to allow the users to draw an appearance of the exemplar shape on the panorama via his/her own perception to the scene. Based on the exemplar shape drawn by human, the camera parameters can be computed by using the related geometrical constraints. In principle, we hope that the exemplar shape satisfies the following criterions:

I. It can provide *sufficient* constraints for computing the camera parameters.
II. It is as simple as possible, so as to release user's burden for drawing it. That is, the constraints provided by it are also *not redundant*.
III. It is intuitive and *easy to be perceived* by human.

To find an exemplar shape satisfying the above criterions, the shapes with metric information are not considered because that they are not easy to be perceived by human. Standard camera calibration [5] or pose estimation methods [3] use the 3D control points with metric information that the distances between each pair of the

control points have to be given in advance, and thus they are not suitable for our work. In this paper, we use the shape *without metric information*. In particular, what we need is to use the geometric information less constraining to obtain to human perception, such as parallelism, orthogonality of lines, and so on. Inspired by a previous work [2], we select the specified shape to be *three lines joining at a single point and are orthogonal to each other*. It can also be treated quite naturally as *the origin and the three axes of a 3D Euclidean coordinate system*. In fact, such a coordinate system may appear in many natural scenes (for example, the one shown in Figure 4(b)). It is also easy and intuitive for the users to hallucinate such orthogonal axes (for example, the one shown in Figure 5(b).

3. Camera Parameter Estimation with A Single Panorama

In this section, we show that the exemplar shape selected above provides sufficient constraints for computing the camera parameters. There are usually two types of data structures for storing a panorama: the cylindrical type and the spherical type. Without lost of generality, we use the cylindrical type for the illustration in the sequel. Nevertheless, our method can be easily generalized to the spherical type.

3.1. Intrinsic Parameters

The intrinsic parameters (e.g., focal point and focal length) of any de-warped views of a panorama can be computed directly from the de-warping process for either cylindrical or spherical types of panoramas. In fact, the focal point (i.e., the point which is the orthogonal projection of the lens center in the image plane) of a de-warped image is set to be in its center in almost all cases. The focal length (in pixels) of a de-warped view can be approximately computed by P/(2π) where P is the number of pixels of the width of the panorama.

The intrinsic parameters can also be computed more accurately. In fact, an important property of a panorama is that the intrinsic camera calibrations are recovered as part of the panorama construction [9]. That is, the intrinsic parameters can be directly computed from the panorama. More precisely, consider the panorama recorded in the surface of a cylinder as shown in Figure 1(a). A panorama viewer allows the user to see the contents of the panorama from arbitrary viewing directions specified by the user. The panorama viewer de-warps the panorama recorded in a cylinder to an image in a plane, as shown in Figures 1(a) and 1(b). The de-warped image (DI) is photographed in a rectangular plane tangential to the cylinder. The perspective imaging equation of a DI can be written as follows:

$$\lambda p = K[R_{3\times3} \mid t_{3\times1}]P \qquad (1)$$

where P is the homogeneous coordinate of a 3D point, p is the homogeneous coordinates of its 2D image point, R and t are rotation and translation with respect to the world coordinate system, and K is an upper-triangular matrix consisting of the intrinsic parameters, where

$$K = \begin{bmatrix} f_u & s & u \\ 0 & f_v & v \\ 0 & 0 & 1 \end{bmatrix}.$$

In most cases, the coordinate system selected by the panorama viewer to represent the pixel grids in DI is orthogonal, and thus $s=0$. In addition, the Panorama viewer usually de-warps the panorama to a square patch, and hence the aspect ratio $\frac{f_u}{f_v}=1$. Also, the tangential point is always set to be the center of the de-warped image in a panorama viewer, as shown in Figure 1(c). Therefore, the image center of DI, (u, v), is $(0, 0)$. If there are N pixels in a horizontal scan-line of DI, as shown in Figure 1(c), then the pixel resolution in DI is $du = 2f\tan(\frac{\theta}{2})/N$. Consequently,

$$f_u = du/f = 2\tan(\frac{\theta}{2})/N. \qquad (2)$$

3.2. Extrinsic Parameters

Once the intrinsic parameters of a de-warped image are obtained, what we need is to compute the extrinsic parameters of it, i.e., the rotation and translation between the camera coordinate system and the Euclidean coordinate system drawn by the user. This problem is referred to as the *camera pose estimation* in the computer vision community. Given a trihedral with the angles between each pair of lines being 90°. By using the results shown in [6], we can compute the camera pose by solving a second-degree polynomial equation system. In this paper, we derive this result in another way. The detailed procedure of computation is shown in the following.

Given three lines joining at a point P_0 and orthogonal to each other, as shown in Figure 2. We select three control points, P_1, P_2, P_3, in the three lines, respectively. Assume that the homogeneous coordinates of their image points are p_0, p_1, p_2, p_3 respectively. Based on these three lines, we define an orthogonal object coordinate system that the origin is P_0, and the **X, Y,** and **Z** axes are defined to be along the directions from P_0 to P_1, P_0 to P_2, and P_0 to P_3, respectively. Let $\|P_0P_1\|=a$, $\|P_0P_2\|=b$, $\|P_0P_3\|=c$, then the coordinates of P_0, P_1, P_2, P_3 are $[0\ 0\ 0]^T$, $[a\ 0\ 0]^T$, $[0\ b\ 0]^T$, $[0\ 0\ c]^T$, respectively. From (1), we can list the following four equations:

$$\lambda_0 p_0 = KRP_0 + Kt = Kt \qquad (4)$$

$$\lambda_1 p_1 = KRP_1 + Kt \qquad (5)$$

$$\lambda_2 p_2 = KRP_2 + Kt \qquad (6)$$

$$\lambda_3 p_3 = KRP_3 + Kt \qquad (7)$$

Since there always exists a scale factor which can not be computed, we set λ_0 =1 (i.e., the distance from the lens center to P_0 is the *unit length*) and it will not affect the camera pose estimation results. Hence, (4) becomes

$$Kt = p_0$$

From the above equation, we can solve the translation vector,

$$t = K^{-1}p_0 \qquad (8)$$

where K is given in (3).

Substituting (8) to (5), (6), (7) and multiplying K^{-1} to the left side of (5), (6), (7), we can obtain the following equations:

$$K^{-1}(\lambda_1 p_1 - p_0) = RP_1 \qquad (9)$$

$$K^{-1}(\lambda_2 p_2 - p_0) = RP_2 \qquad (10)$$

$$K^{-1}(\lambda_3 p_3 - p_0) = RP_3 \qquad (11)$$

Since the three vectors $\vec{P_0P_1}$, $\vec{P_0P_2}$, $\vec{P_0P_3}$ are orthogonal to each other. By computing the inner products of each of the two equations of (9), (10), and (11), we can obtain the following equations:

$$(\lambda_1 p_1 - p_0)^T K^{-T} K^{-1}(\lambda_2 p_2 - p_0) = 0 \qquad (12)$$

$$(\lambda_2 p_2 - p_0)^T K^{-T} K^{-1}(\lambda_3 p_3 - p_0) = 0 \qquad (13)$$

$$(\lambda_1 p_1 - p_0)^T K^{-T} K^{-1}(\lambda_3 p_3 - p_0) = 0 \qquad (14)$$

where

$$K^{-T}K^{-1} = \begin{bmatrix} f_u^{-2} & 0 & 0 \\ 0 & f_u^{-2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (15)$$

There are three unknowns, $\lambda_1, \lambda_2, \lambda_3$ in (12) - (14). Since the left side of (12)-(14) are bilinear forms, expanding (12)-(14) yields the following three bilinear equations:

$$a_{11}\lambda_1\lambda_2 + a_{12}\lambda_1 + a_{13}\lambda_2 + a_{14} = 0 \qquad (16)$$

$$a_{21}\lambda_2\lambda_3 + a_{22}\lambda_2 + a_{23}\lambda_3 + a_{24} = 0 \qquad (17)$$

$$a_{31}\lambda_1\lambda_3 + a_{32}\lambda_1 + a_{33}\lambda_3 + a_{34} = 0 \qquad (18)$$

where a_{ij} are the coefficients computed from K and P_0, P_1, P_2, P_3 by expanding (12)-(14). The equations (17) and (18) yield that

$$\lambda_2 = \frac{-a_{23}\lambda_3 - a_{24}}{a_{21}\lambda_3 + a_{22}} \qquad (19)$$

and

$$\lambda_1 = \frac{-a_{33}\lambda_3 - a_{34}}{a_{31}\lambda_3 + a_{32}} \qquad (20)$$

By substituting (19) and (20) to (16), we can obtain a quadratic equation in terms of λ_3:

$$\begin{aligned} &a_{11}(a_{33}\lambda_3 + a_{34})(a_{23}\lambda_3 + a_{24}) \\ &- a_{12}(a_{33}\lambda_3 + a_{34})(a_{21}\lambda_3 + a_{22}) \\ &- a_{13}(a_{23}\lambda_3 + a_{24})(a_{31}\lambda_3 + a_{32}) \\ &+ a_{14}(a_{21}\lambda_3 + a_{22})(a_{31}\lambda_3 + a_{32}) = 0 \end{aligned} \qquad (21)$$

Hence, λ_3 can be obtained by solving (21). After solving $\lambda_1, \lambda_2, \lambda_3$, the rotation matrix R can be obtained using (9) - (11) because the three columns of R are the unit vectors of $K^{-1}(\lambda_1 p_1 - p_0)$, $K^{-1}(\lambda_2 p_2 - p_0)$, $K^{-1}(\lambda_3 p_3 - p_0)$, respectively. In addition, *a*, *b*, *c* are the lengths of these three vectors, respectively.

4. Experimental Results

We have implemented a user interface which allows the users to draw an appearance of the exemplar shape and composite the virtual graphic objects in a geometrically-consistent way. Figure 3(a) shows an example of the three axes of a Euclidean coordinate system drawn by the users. In particular, a cuboid will appear in our interface if the user drawing makes the solution of (21) exist, as shown in Figure 3(b).

Some experimental results are shown in Figures 4 and 5 to clarify the effectiveness of our method. Notice that in both experiments we only have to estimate the camera parameters from a single de-warped view, the same parameters can then be used for other views while maintaining highly-convincing geometric consistencies of the generated composition views.

5. Summary

In summary, we developed a simple and intuitive approach in this paper for inserting geometrically consistent virtual 3D objects in a single panorama. To provide sufficient and not redundant geometrical constraints, what a user required to do is simply to draw the three axes of a 3D Euclidean coordinate system in the de-warped image according to his (or her) perception to the scene. Then, our method allows the user to interactively place 3D graphic objects in arbitrary positions of the 3D world photographed in the panorama.

Acknowledgement

This work was supported in part by the National Science Council, Republic of China under Grant NSC-89-2218-E-001-004.

References

[1] C. S. Chen, et al. “Integrating Virtual Objects into Real Images for Augmented Reality,” *Proceedings of ACM Symposium on Virtual Reality Software and Technology, VRST’98*, pp. 1-8, 1998.

[2] C. S. Chen, C. K. Yu, and Y. P. Hung, “New Calibration-free Approach for Augmented Reality Based on Parameterized Cuboid Structure,” *Proceedings of International Conference on Computer Vision, ICCV’99*, Corfu, Greece, September 1999.

[3] D. F. Dementhon and L. S. Davis, “Exact and Approximate Solutions of the Perspective-Three-Point Problem,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, pp. 1100-1105, 1992.

[4] O. Faugeras, “From Geometry to Variational Calculus: Theory and Applications of Three-Dimensional Vision,” *Proceedings of IEEE and ATR Workshop on Computer Vision for Virtual Reality Based Human Communications, CVVRHC’98*, Bombay, India, pp. 52-71, January 1998.

[5] R. K. Lenz and R. Y. Tsai, “Techniques for Calibration of the Scale Factor and Image Center for High Accurate 3-D Machine Vision Metrology,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 713-719, 1988.

[6] Y. Wu, S. Iyengar, R. Jain, “A New Generalized Computational Framework for Finding Object Orientation Using Perspective Trihedral Angle Constraint,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 10, pp. 961-975, 1994.

[7] S. B. Kang and R. Szeliski, “3D Scene Data Recovery Using Omnidirectional Multibaseline Stereo,” *International Journal of Computer Vision*, 25(2).

[8] K. N. Kutulakos, J. R. Vallino, “Calibration-Free Augmented Reality,” *IEEE Transactions on Visualization and Computer Graphics*, Vol. 4, pp. 3-20, 1998.

[9] H. Y. Shum, et al., “Interactive 3D Modeling from Multiple Images Using Scene Regularities,” *Proceedings of SMILE’98, Lecture Notes in Computer Science, Vol. 1506*, pp. 236-252, 1998.

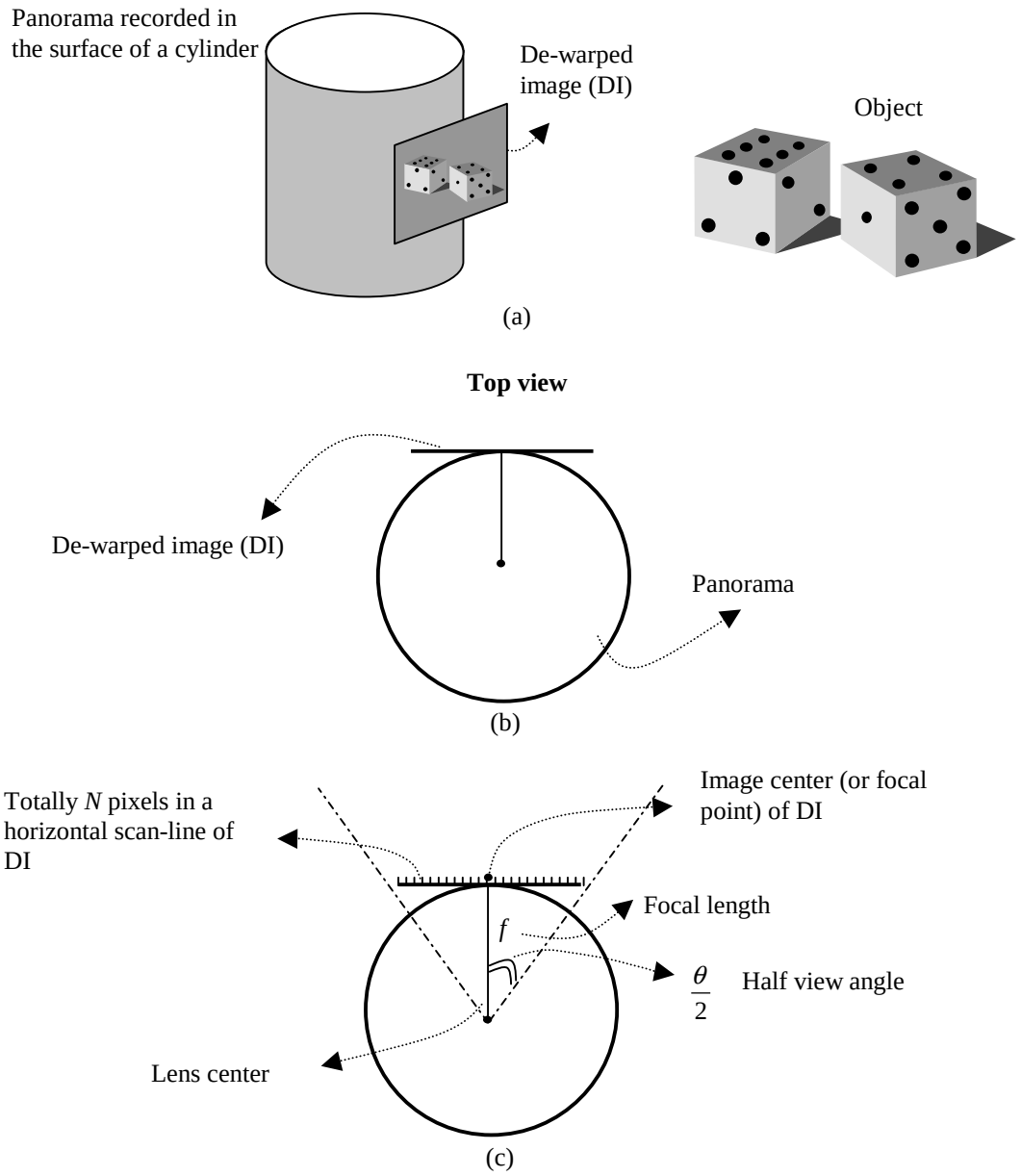
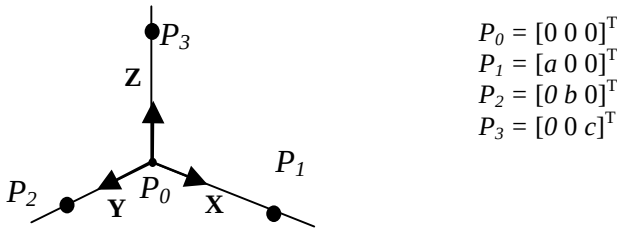
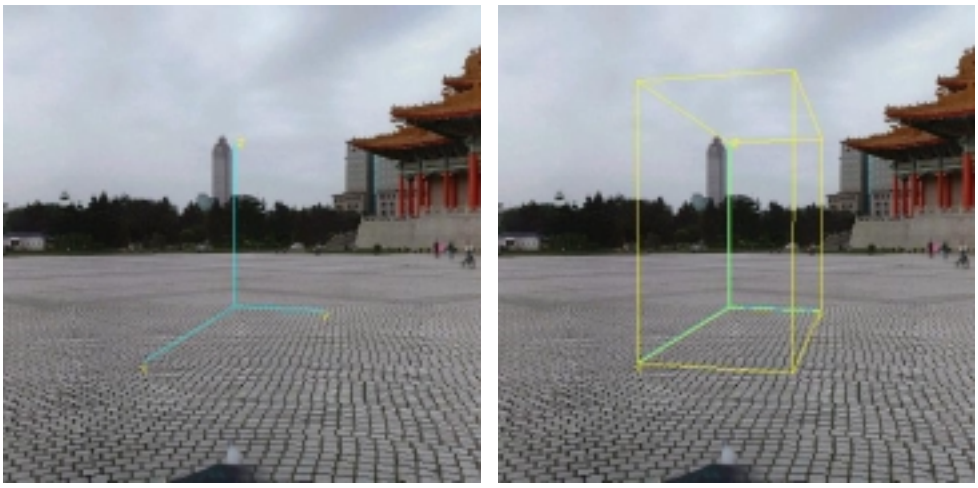


Figure 1. (a) A panorama recorded in the surface of a cylinder. A panorama viewer de-warps the panorama to a planar image (DI). (b) The top view of (a). (c) The intrinsic parameters of DI.



$P_0 = [0\ 0\ 0]^T$
$P_1 = [a\ 0\ 0]^T$
$P_2 = [0\ b\ 0]^T$
$P_3 = [0\ 0\ c]^T$

Figure 2. The object coordinate system defined in three orthogonal lines.



(a) (b)

Figure 3. (a) An appearance of the three axes of a 3D Euclidean system drawn by a user. (b) A cuboid will appear in our interface if the user drawings allow the solutions to exist.

(a)

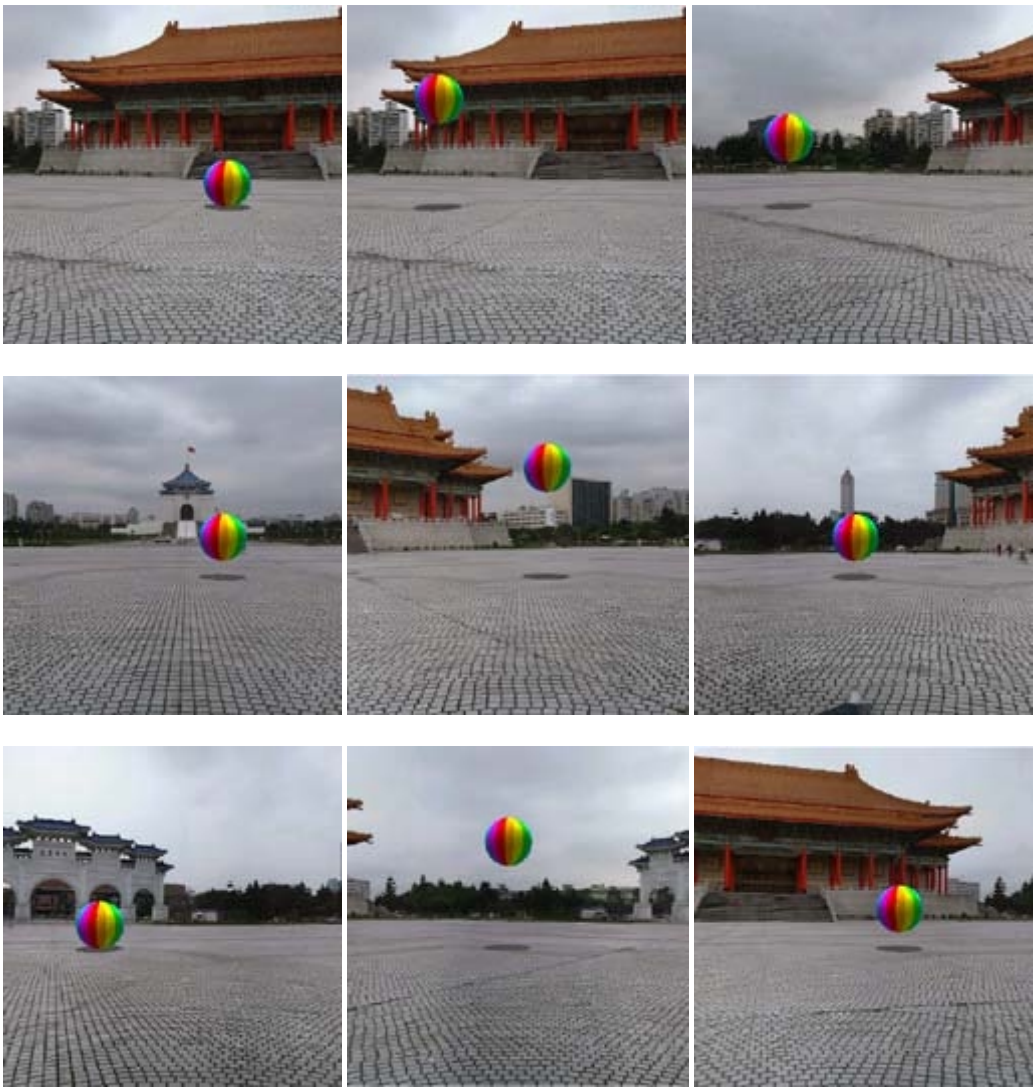
(b) (c) (d)

Figure 4. (a) A panorama. (b) Camera parameter estimation with the "front view" using the three axes of an Euclidean coordinate system drawn by human. Notice that such a coordinate system exists in the scene *explicitly*, and a user can easily identify it easily via his (or her) perception. A virtual chair is then inserted in (c) the "front view" and (d) the "side view" (using the same set of estimated camera parameters).

(a)

(b)

Figure 5. (a) A panorama of the Chung-Cheng Memorial Hall. (b) Three axes of the Euclidean coordinate system drawn by human for camera parameter estimation. Notice that although such a coordinate system does not exist explicitly in the scene, a user can still draw it *implicitly* via his (or her) perception.



(c)

Figure 5 (continue). (c) The insertion of an animation of a bouncing ball in the panorama. Also, the same set of camera parameters estimated from the Euclidean coordinate system drawn in Figure (b) is used for all views.