An Augmented Reality System with Camera Self-calibration

Norma Irene Serna Rodríguez and Luis Gerardo de la Fraga
Cinvestav. Computer Science Department.
Av. Instituto Politécnico Nacional 2508. 07360 México, D.F., México
E-mail: fraga@cs.cinvestav.mx

***Abstract*— We present an augmented reality system in which interact digital video and a virtual ball. The system is simple but functional, and consists of a box and a virtual ball. The ball's movement follows the direction of the box according to the Newton's Physics, and the friction force. The box is recognized by its corners, and in case of failing to do so, the camera is again self-calibrated which makes the system suitable to a camera's zoom adjust. To self-calibrate the camera, a numerical-stable algorithm is used which is based on cuboids.**

Keywords: augmented reality, camera self-calibration, physically based simulation.

I. Introduction

An augmented reality (AR) system generates a composed view for the user. It is a combination of real scenes viewed by the user, and virtual scenes generated by the computer, resulting in augmented scenes by the addition of information. The AR's aim is to create a system such that the user cannot notice the difference between the real world and a virtual augmentation of it [1], [2]. An AR system involves digital image processing techniques, computer vision, and real-time systems.

Computer vision is the study and application of methods that allow computers to *understand* the content of images or the multidimensional content of data. The term *understand* means that specific information from the image data is extracted for a specific purpose. This field obtains significant and explicit descriptions of the world represented by an image, and helps to define the model of objects represented in a scene, the relation between a observer (the camera) and a scene, and the 3-D structure of the space shown in the scene.

A camera's calibration process is one of the most important problems in computer vision, its purpose is to obtain through a camera, an estimation of the parameters to transform a point in the real world to a point in an image. To carry out the camera's calibration, some techniques based on vanishing points [3], [4], plannar patterns and homographies [5], one-dimensional objects [6], and single images from parallelepipeds [7] are considered; we used the last technique due to its numerical stability, and because it require one single image of the scene.

A camera's self-calibration means that there is no previous necessity to calibrate the system with a special known object. In the process of self-calibration the camera's parameters, and the geometry of a scene, from the correspondences of points in the image are obtained. With the [7] technique, with the knowledge that the box has three right angles, it is possible to use a single image to obtain the needed parameters from six corner points of the box.

The remaining of this paper is organized as follows. In Section 2, the system and the processing of digital frames to obtain the corners of the box are described. In addition, the autocalibration process, and the restriction for the physic-based simulation are stated. In Section 3, the application and results are presented. Finally, in Section 4 the work is concluded.

II. System description

Our AR system has a real object (a box) which is captured by a digital video camera, and is displayed in real time on a computer screen. In the box's corners identification process the camera is self-calibrated. With the geometric information obtained, a virtual object is drawn (a ball) to complement the scene. The ball is moved according to the physic laws of movement and taking account the friction force. In this way, the real object's visualization, and the virtual object interact in the same scene. The box in the video is slightly manipulated in all the possible directions, and the real time video is shown at 30 frames per second. This video displays not only the box, but also the virtual ball whose movement follows the box's inclination.

The system is constructed by a semi-profesional video camera *Canon model GL2*. The interface between the camera and the computer is carried out by a Firewire port (IEEE1394).

The application is controlled by the algorithm 1 as follows. A frame is extracted from the video, then the box's corners are obtained, and the camera calibration is carried out, subsequently the box orientation is calculated, finally a virtual ball is drawn. In the subsequent frames, points around of the previous ones are sought (i.e. corners are tracking). It is worthy note that there is no necessity to re-calibrate the camera anymore. However, in the case of not being able to follow the corners (as when a camera's zoom adjust occurs), the camera calibration is performed.

Algorithm 1 AR system algorithm.

```
Require: Digital video.
Ensure: The AR system visualization.
 1: corners ← 0; calibrating ← 0
 2: while there be frames in the video do
 3:   Extracting a frame
 4:   if !corners then
 5:     corners ← searching_corners();
 6:   end if
 7:   if corners then
 8:     if corner_tracking() then
 9:       if !calibrating then
10:         Camera calibration process;
11:         calibrating ← 1;
12:       end if
13:       Calculating the orientation of the box;
14:       Drawing the virtual ball;
15:     else
16:       corners ← 0; calibrating ← 0;
17:     end if
18:   end if
19: end while
```

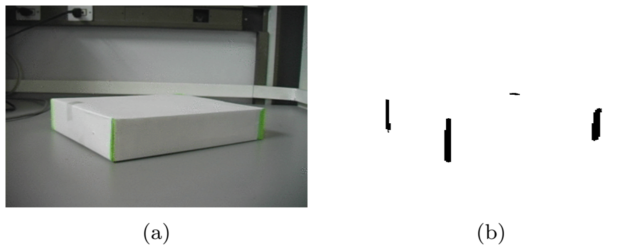

Fig. 1. Color process segmentation. a) frame with the real object with colored lines; b) segmented frame.

A. Video frames processing

To extract the box's corners, several digital image processing techniques on the video frame are used: smoothing, color segmentation, morphological operations such as erosion and dilation. In addition, a very simple method is implemented to obtain the box's corners.

For the smoothing process, data pixels are averaged with respect to a series of 3×3 neighbour pixels. In the segmentation case, it is advantageous to consider a RGB format video to paint the box's edges, and segment them by a global threshold. Two erosion and dilation steps are applied with a structural element of 3×3 pixels size to eliminate isolated pixels and single lines (noise). Fig. 1.a shows an example of a frame; the result of the smoothing, and color segmentation is depicted in Fig. 1.b. To obtain the corners, the maximum and minimum points are found on each segmented lines depicted in this figure.

B. Camera calibration

The pinhole camera model provides a relation of the perspective projection from the 3-D space to the 2-D image plane. It is expressed by $\lambda \mathbf{p} = M\mathbf{P}$, where M is a matrix of 3×4, $\mathbf{p}$ are the homogeneous point coordinates $[x, y, 1]^T$ in the image plane , $\mathbf{P}$ are the homogeneous point coordinates $[x_w, y_w, z_w, 1]^T$ in the 3-D space, and λ is a scale factor. The matrix M can be decomposed as $M = K \cdot [Rt]$. $[Rt]$ is the 3×4 matrix determining the relative orientation of R and position $-Rt$ of the camera with respect to the world coordinates. In this work K, the matrix that defines the intrinsic camera parameters with square pixels is defined as

$$K = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

f is the camera's focal length and (u_0, v_0) is the intersection point of the optical axis with the image plane. On this phase the camera calibration is carried out by using a single cuboid image. A cuboind is a parallelepiped with three right angles. The general calibration process using parallelepipeds is described in [7].

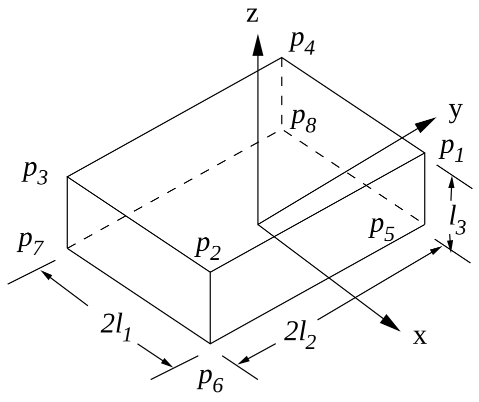

Fig. 2. Cuboid's parametrics coordinates.

The used cuboid's parameterization is represented in Fig. 2. In this way, the base of the box is the center of the world coordinates, and for example, the P_1 point is equal to $\widetilde{\Lambda}(1, 1, 1, 1)^T$, where $\widetilde{\Lambda}$ is equal to

$$\widetilde{\Lambda} = \begin{pmatrix} l_1 & 0 & 0 & 0 \\ 0 & l_2 & 0 & 0 \\ 0 & 0 & l_3 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

$2l_1$, $2l_2$, and l_3 are the cuboid's three edge lengths.

The points in the image $p_i = [u_i, v_i]$, with $i = 1 \ldots 8$, satisfy the equation:

$$\begin{pmatrix} \alpha_1 u_1 & \alpha_2 u_2 & \cdots & \alpha_8 u_8 \\ \alpha_1 v_1 & \alpha_2 v_2 & \cdots & \alpha_8 v_8 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_8 \end{pmatrix} = \widetilde{X} \cdot \begin{pmatrix} 1 & 1 & \cdots & -1 \\ 1 & -1 & \cdots & -1 \\ 1 & 1 & \cdots & 0 \\ 1 & 1 & \cdots & 1 \end{pmatrix} \tag{2}$$

$\widetilde{X}$ is the projection matrix of 3×4 size which is defined up to a scale factor as:

$$\widetilde{X} \sim M \cdot \Lambda \sim K \cdot [R|t] \cdot \widetilde{\Lambda} \tag{3}$$

The projection matrix $\widetilde{X}$ captures all the geometric information contained in the cuboid's image projection. By considering the relation $X \sim K \cdot R \cdot \Lambda$, where X and

Λ are the matrices of the first three lines and the columns of $\widetilde{X}$ and $\widetilde{\Lambda}$, respectively. And based on the last relation, $K^{-1}X \sim KR$, squaring it, and considering $R^TR = I$ it is obtained:

$$X^T \cdot K^{-T} \cdot K^{-1} \cdot X \sim \Lambda^T \cdot \Lambda \tag{4}$$

The $\widetilde{X}$ matrix can be estimated by six points (in this work six or seven points are extracted from the box). From Eq. (2), it is possible to obtain 14 equations as long as seven points are available. The result is two equations per each point by the procedure of removing the scale factor α_i per each point. A least squares estimation of $\widetilde{X}$ for the resulting homogeneous equations system is obtained by using singular value decomposition (SVD).

1) Estimating intrinsic camera parameters: The "camera calibration" step in the Algorithm 1 means to estimate the camera's intrinsic parameters. From Eq. (4) we define $\omega = K^{-T} \cdot K^{-1}$. Three parameters have to be estimated, that is, by calculating the matrix ω, it is only necessary to obtain ω_{11}, ω_{13}, ω_{23}, and ω_{33}. It is possible to define ω_{33} in terms of the others three parameters, but this relation is non linear. The matrix ω can be calculated by using the linear restrictions on Eq. (4), where X_i is the i-th column of X:

1) $X_i^T \cdot w \cdot X_j = 0$.
2) For each length value $r_{ij} = l_i/l_j$:
   $X_i^T \cdot w \cdot X_i - r_{ij}^2 X_j^T \cdot w \cdot X_j = 0$.

With the first three restrictions and one of (2), it is possible to establish the homogeneous linear system equations and an estimation can be obtained by SVD. The knowledge of a relation between the length edges, removes the ambiguity of the scale factor.

2) Extrinsic parameters: The "Calculating the camera orientation" step in Algorithm 1 calculates the orientation parameters, that is, the three rotation angles, and the camera translation. The parameters are calculated from Eq. (4), and thus the rotation matrix R is,

$$R = \frac{1}{\lambda} K^{-1} X \Lambda^{-1}$$

Note that the estimated matrix R do not exactly achieve the orthogonal matrix properties ($R^{-1}R = R^TR = I$), therefore to reach this property, it is necessary to force the orthogonality restrictions by using SVD again [8]. The matrix R can be defined by three rotations in each axis, thus $R = R_x(\alpha)R_y(\beta)R_z(\gamma)$. Once that R is obtained, it is easy to calculate α, β, and γ [8].

The translation vector t determines the position of the world coordinate system with respect to the camera. It is possible to calculate it by Eq. (3) as,

$$\widetilde{X} = \lambda K \cdot [R|t]\widetilde{\Lambda}, \quad \text{and}$$
$$\lambda[R|t] = K^{-1}\widetilde{X}\widetilde{\Lambda}^{-1}$$

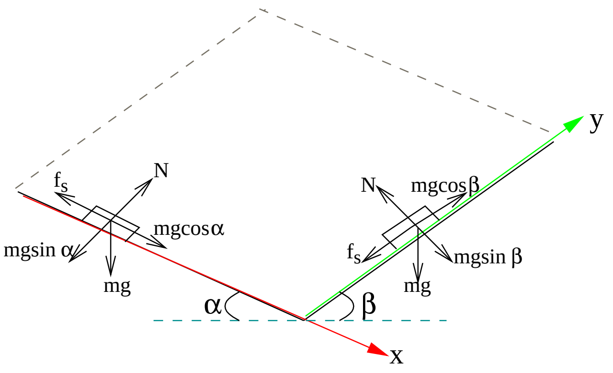

Fig. 3. Physics movement of the virtual object.

here λ is a scale factor which is $\lambda = 1/\|\mathbf{r}_1\| = 1/\|\mathbf{r}_2\| = 1/\|\mathbf{r}_2\|$, where $\mathbf{r}_i$ is the i-th column of matrix R. The translation vector t is given by the last column of the matrix $[R|t]$.

To obtain any position of the world with respect to the camera, it is only necessary to calculate $p \sim M \cdot P$. In this work, the points of an image are calculated by the cuboid's parameterization which are composed by the coordinates (P_5, P_6, P_7, P_8), and M is equal to $K \cdot [R|t]$.

C. Movement of the virtual object

The movement of the virtual object is implemented by the theory of the *physics on a inclined plane* [9], [10]. In this work two components of movement are calculated, one for the x and a second for the y axis that corresponds to the rotation angles α and β, that define the inclination on the x and y axes, respectively.

For example, consider *block*$_1$ of volume m over a incline plane's surface that forms an angle α with the horizontal, and *block*$_2$ of volume m over a incline plane's surface that forms an angle β with the horizontal, as depicted in Fig. 3. The forces that are acting on these blocks are, the normal (N) that takes effect parallel to the surface of the incline plane, and the weight (mg) that takes effect perpendicular to the surface of the incline plane. To achieve a real motion of the virtual object, a coefficient of friction is implemented. The produced effect when the surface of a body is sliding over another one, is known as *frictional force*. The relation between magnitude of the maximum static friction, and the magnitude of the normal force is known as *coefficient of the static friction* on the involved surfaces. If f_s represents the magnitude of the static friction force, then it is satisfied that $f_s \leqslant \mu_s N$, where μ_s is the coefficient of the static friction and N is the magnitude of the normal force. The equal sign is only achieved when f_s reaches the maximum value.

By considering both movements on the axes x, y, it is concluded that:

⁜ The acceleration along the inclined plane's surface is given by the weight component along to the component inclined plane's surface: $a_x = g(\sin\alpha - f_s)$, $a_y = g(f_s - \sin\beta)$.

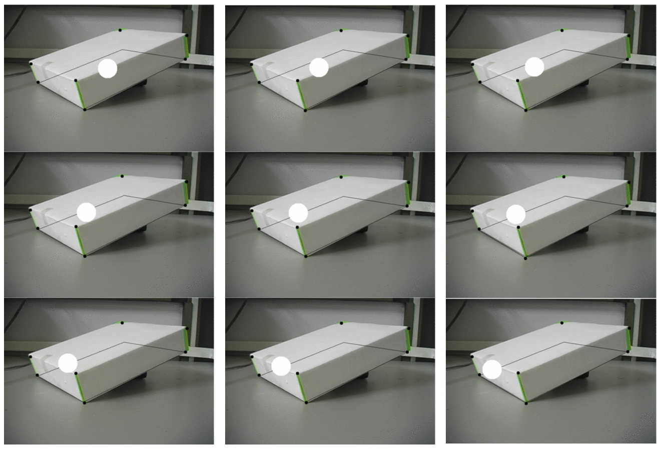
Fig. 4. An example of movement sequence of the virtual object.

- ⁕ The weight component that acts perpendicular to the surface of the inclined plane is equilibrated for the normal force $N = mg\cos\alpha = mg\cos\beta$.
- ⁕ The acceleration a_x, a_y is independent to the volume of the block, and only depends on the angle (α or β) and g.
- ⁕ The acceleration is constant.

On each frame, the cuboid's rotation angles are calculated. Therefore, according to the equations previously defined, a decision about if the object has to move or not is made. To calculate the new position of the virtual object, the parametric coordinates of the object are uploaded with respect to the movement produced by the equations 5 $x'_{t+1} = x'_t + x$, $y'_{t+1} = y'_t + y$. After this, the coordinates are send to the function that calculates the new position in terms of the image coordinates,

$$\begin{array}{l} x = x_0 + v_x t + \frac{1}{2}a_x t^2 \\ y = y_0 + v_y t + \frac{1}{2}a_y t^2 \end{array} \tag{5}$$

To obtain a point $\mathbf{p}$ on the plane in the image, we calculate it as $\mathbf{p} = M\mathbf{P}$, where there is only necessary to define P as $(x', y', 0, 1)$. This representation is shown in the next relation:

$$p \sim M \cdot [x' \; y' \; 0 \; 1]^T \tag{6}$$

III. Results

In Fig. 4 a sequence of images is shown. The movement of the virtual object in the box can be observed for nine frames. The detected rotation angles on this sequence of frames are in the next average: $\alpha = -49.136°$, $\beta = 10.268°$. The AR system provides the next results:

- The real time video processing is suitable due to the averaging transmission of frames with respect to a frame every 33 milliseconds. This process is not affected by others stages of the system such as the digital processing frames, and the algorithms used to obtain the geometric cuboid's features.
- The color segmentation applied to the frames provides worthy results to find the cuiboid's corners and it is computationally cheap.
- The camera self-calibration uses the information contained in the cuboid (projection matrix, angles and length's edges) which allows to effectively obtain the geometric relation between the camera and the image. As a result, with the precision of this information is possible to carry out the movement of the virtual object with respect to the real movement of the box in real time.
- The interaction between the virtual object and the cuboid is correctly performed, and allows to simulate physics events with strong similarity to the same interaction between real objects.

IV. Conclusions

This work presents an experimental AR system which shows the interaction between virtual and real objects.

We presented a AR system composed by a real object, a box, it showed in digital video, and a virtual ball moving inside the box according to the box's inclination. The systes has three main stages: video processing, camera calibration, and the ball's physics movement.

In the digital video frame processing stage, several techniques to improve the video frame quality are used. Such techniques include, smoothing algorithm, color segmentation, corner detection, and some morphological functions. This process is important in the detection of parameters that defines the geometrics characteristics of a real object. The digital video frame processing provides suitable results with a minimum performance time, therefore, the video visualization is not delayed.

In the system the calibration process is automatically carried out. Therefore, there is no intervention of the user to establish the characteristics that could help on purpose, the interaction between the virtual object and the real scene.

The physics movement of the virtual object is fairly similar to that of the objects interacting in the real world. This movement is defined by the estimation of geometric components which establish the orientation and inclination of the real object. The estimation can be affected by noise included in the acquired frame by the video processing stage. However, for this AR system the selected process to improve the video frames strongly reduce this problem. The specification of this system is based on a perspective view, so the system is restricted to a maximum and minimum view in which is possible to calculate the necessary components, and helps to define the geometric body of the object. The movement of the virtual object is displayed perfectly on the different possible box's inclinations.

References

[1] Jim Vallino, Department of Software Engineering, Rochester Institute of Technology, 134 Lomb Memorial Drive, Rochester,

NY 14623-5608. *Introduction to Augmented Reality*, 2002. http://www.se.rit.edu/~jrv/research/ar/.

[2] Ronald T. Azuma. Augmented reality: Approaches and technical challenges. In Woodrow Barfield and Thomas Caudell, editors, *In Fundamentals of Wearable Computers and Augmented Reality*, pages 27–63. Lawrence Erlbaum Associates, 2001.

[3] Bruno Caprile and Vincent Torre. Using vanishing points for camera calibration. *International Journal of Computer Vision.*, 4(2):127–140, 1900.

[4] R. Cipolla, T. Drummond, and D. Robertson. Camera calibration from vanishing points in images of architectural scenes. *BMVC99*, page 10, 1999.

[5] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans on Patt Anal & Mach Intel*, 22:1330–1334, Nov 2000.

[6] Z. Zhang. Camera calibration with one-dimensional objects. *IEEE Trans on Patt Anal & Mach Intel*, 26(7):892–899, 2004.

[7] Wilczkowiak Marta, Boyer Edmond, and Sturm Peter. Camera calibration and 3d reconstruction from single images using parallelepipeds. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, volume 1, pages 142–148. IEEE Computer Society Press, Jul 2001.

[8] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision.* Prentice Hall PTR, 1998.

[9] Serway R. *Física Vol. 1.* McGraw-Hill, 1992.

[10] Resnick Robert, Halliday David, and Krane Kenneth S. *Física Vol. 1.* Continental de México, 1992.