Image-based modeling using a two-step camera calibration method

Conference Paper · November 1998

DOI: 10.1109/SIBGRA.1998.722777 · Source: IEEE Xplore

3 authors:

Paulo de Carvalho
University of Coimbra
325 PUBLICATIONS **2,651** CITATIONS
SEE PROFILE

Flavio Szenberg
Pontifícia Universidade Católica do Rio de Janeiro
12 PUBLICATIONS **65** CITATIONS
SEE PROFILE

Marcelo Gattass
TECGRAF / PUC-Rio
299 PUBLICATIONS **3,282** CITATIONS
SEE PROFILE

Image-based Modeling Using a Two-step Camera Calibration Method

PAULO CEZAR PINTO CARVALHO[1]
FLÁVIO SZENBERG[2]
MARCELO GATTASS[2]

[1]IMPA- Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina, 110, 22460-320, Rio de Janeiro, RJ, Brazil
pcezar@visgraf.impa.br

[2]TeCGraf - Grupo de Tecnologia em Computação Gráfica, PUC-RIO
Rua Marquês de São Vicente, 255, 22453-900, Rio de Janeiro, RJ, Brazil
{szenberg, gattass}@tecgraf.puc-rio.br

Abstract. We present a two-step camera calibration process based on linear least squares formulations. We show how to apply this process to build an integrated modeling environment, in which a scene is modeled by having the user specify, on a given image, a set of reference points (used for camera calibration) and object points (used for positioning the objects in the scene). Once this has been done, synthetic information can be superimposed onto the image and arbitrary views can be produced. Using this methodology, we developed *Virtual Referee*, an application to analyze soccer plays from still images, generating an OpenGL camera over a 3D field model.

Keywords: image-based modeling, camera calibration, computer vision.

1. Introduction

In this paper we present a technique for integrating information given by an image containing known reference points into a geometric modeling system. Examples of applications are some modeling systems used in the broadcasting of sport events. In such systems, a three-dimensional model of the scene is built from images, with the purpose of checking whether a given play was legal according to the sport regulations or obtaining other views of the same scene (as seen by one of the players, for instance).

The techniques proposed here are based on appropriately calibrating the camera that took the picture. This means finding a transformation **T** that describes how points in a scene (represented using a 3D reference frame) are mapped into pixels in an image (represented using a 2D reference frame). Finding the 3D positions of the objects in the scene, using the corresponding points on the image, involves inverting this transformation. Since **T** maps 3D space into 2D space, this requires using additional conditions. In our case, for instance, we use the fact that most objects lie on the playing field. Once the objects are located in the 3D scene, one can change the visualization parameters, in order to place the camera in the desired position.

We assume that the user specifies both the reference points used for calibration and the points used for positioning the objects. User-assisted methods for image-based modeling have been studied in papers such as [Debevec+96], [Horry+97] and [Heckbert86].

One of the contributions of our paper is a new 2-step technique for camera calibration using linear least square methods. We also show how to integrate camera information into the OpenGL environment, in order to produce synthetic scenes corresponding to the image. Using this methodology, we developed *Virtual Referee* [JuizVirtual98], an application that helps analyzing soccer plays from still images, by allowing the user to change camera position in order to better observe a given play.

The camera calibration presented here is a variation of the affine method for camera calibration, described for instance in [Faugeras93] or [Gonzalez+92].

2. Camera calibration

Calibrating a camera consists in, given a sample of N points $\boldsymbol{M}_i = (x_i, y_i, z_i)$ in 3D-space and their corresponding projections $\boldsymbol{m}_i=(u_i, v_i)$ on the image, finding a transformation T that maps each point $\boldsymbol{M}_i$ into the corresponding image point $\boldsymbol{m}_i$. In other words, this involves determining the external parameters (position

and orientation of the focal point) and internal parameters (focal distance and lenses) of the camera.

In this paper, we consider images from TV broadcast soccer matches, as the one shown in Fig. 1. The image shows several points of the playing field that have relative positions specified by the game regulations, as shown the symbol X in Fig. 2 (see also [FIFA]). Corresponding points in the two images provide the sample for camera calibration.

The set of parameters to take into account for camera calibration depends on the particular camera model. The simplest model is a camera with no lenses, in which the image is obtained by projecting the object onto a planar screen through a "pin-hole" (the optical center of the camera). This model is adequate for our purposes, since our goal is to integrate image information into a modeling environment, such as, where rendering is done using a simple synthetic camera that follows the same model.

Fig. 1 – An image from a soccer game

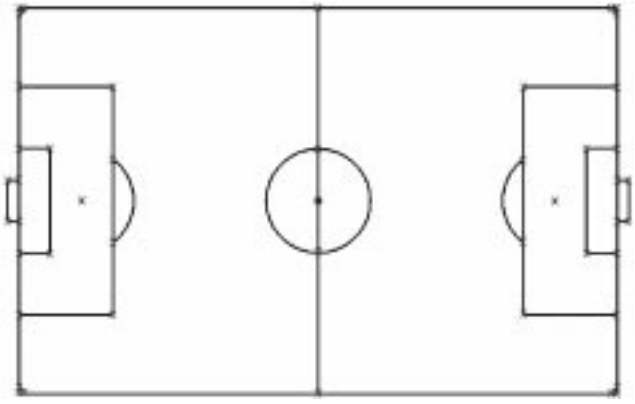


Fig. 2 – The field of play

The calibration method used in this paper is a modification of the method given in [Faugeras93], which is briefly explained below.

Let us consider a three-dimensional coordinate system CXYZ, with its origin at the optical center of the camera, its Z axis perpendicular to the plane on which the image is formed and its X and Y axis parallel to the edges of the image, as shown in Fig. 3. Projecting such a system onto the image plane induces a two-dimensional coordinate system C'X'Y' in that plane. Relative to these coordinate systems, the perspective projection (X', Y') of a point (X, Y, Z) space is given by

$$X' = fX/Z \qquad Y' = fY/Z \tag{1}$$

where f is the distance between the optical center of the camera and the image plane.

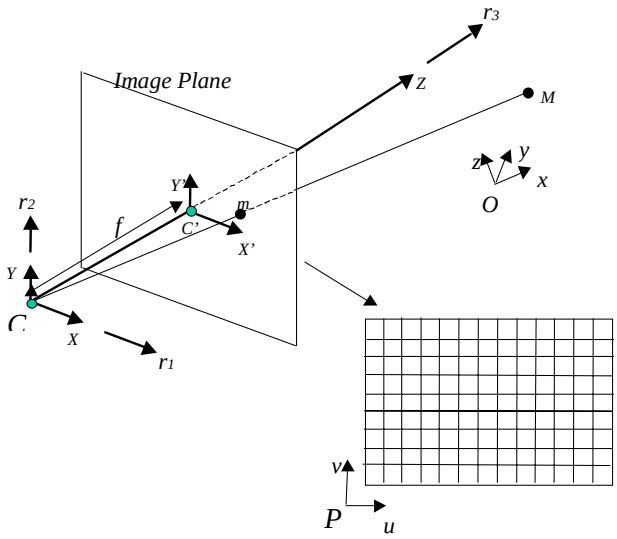


Fig. 3 – Pin-hole camera model

In a camera calibration problem, however, camera position and orientation are unknown. The reference points are given in a reference system that is not associated with the camera and the corresponding image points are usually specified in pixel units. Therefore, the transformation from 3D to image coordinates involves three steps: the conversion from a general coordinate system Oxyz to the camera-based system CXYZ (which is described by a translation followed by rotation); the perspective projection itself, given by (1); and, finally, the conversion, in the image plane, from the C'X'Y' system

to the pixel-based system Puv (which consists of a translation followed by scaling along each axis). An expression for a general camera results from concatenating those three transformations. Using homogeneous coordinates, the position [*u*, *v*, *w*] in the image corresponding to point [*x*, *y*, *z*, *1*] in 3D space is given by [Faugeras93]:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \alpha_u \mathbf{r}_1 + u_0 \mathbf{r}_3 & \alpha_u t_x + u_0 t_z \\ \alpha_v \mathbf{r}_2 + v_0 \mathbf{r}_3 & \alpha_v t_y + v_0 t_z \\ \mathbf{r}_3 & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \qquad (2)$$

where $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$, t_x, t_y and t_z are the extrinsic parameters, three orthonormal vectors for rotation (see Fig. 3) and three scalars for translations, which define the transformation from the system Oxyz to CXYZ. The parameters α_u , α_v, u_0, v_0 are intrinsic to the camera that relate C'X'Y' to Puv by:

$$u = \alpha_u X' + u_0$$
$$v = \alpha_v Y' + v_0$$

This model includes non-linear relationships involving the camera parameters. This presents disadvantages, when compared to a linear model formulation. In particular, one needs to use iterative solution methods, whereas closed expressions are available for linear models. As an alternative, one may use a more general model, given by a generic projective transformation of the form:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \qquad (3)$$

obtained by grouping parameters in (2). It should be stressed that equation (3) above describes a more general transformation than the one induced by a camera. In order to represent a true camera, it must satisfy the following (nonlinear) condition ([Faugeras93]):

$$(\boldsymbol{q}_1 \times \boldsymbol{q}_3).(\boldsymbol{q}_2 \times \boldsymbol{q}_3) = 0 \qquad (4)$$

For many purposes, however, the camera model given by (3) is appropriate. In our method, it will provide a first approximation for the camera.

3. Linear method for camera calibration

Let us consider the problem of finding a matrix

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix}$$

that associates, through equation (3), each sample point $\boldsymbol{M}_i$ to its respective image $\boldsymbol{m}_i$. All scalar multiples of $\boldsymbol{Q}$ determines the same transformation. Thus, finding $\boldsymbol{Q}$ involves 11 independent parameters. This immediately shows that at least 6 points are needed to determine $\boldsymbol{Q}$. It is also necessary that these points be in "general position", a notion which is made precise in [Faugeras93]. It requires, for instance, that the 6 points do not lie on the same plane. In our case, this implies that at least one the reference points must not lie on the field (the top corners of the goal posts are used for this purpose).

Determining $\boldsymbol{Q}$ requires solving the system given by the following linear equations:

$$\begin{aligned} \boldsymbol{q}_1 \boldsymbol{M}_1 + q_{14} &= u_i (\boldsymbol{q}_3 \boldsymbol{M}_i + q_{34}) \\ \boldsymbol{q}_2 \boldsymbol{M}_1 + q_{24} &= v_i (\boldsymbol{q}_3 \boldsymbol{M}_i + q_{34}) \end{aligned} \qquad (i = 1, \ldots, N) \qquad (5)$$

In general, equation (5) has no exact solution, due to errors in the sampling process, and has to be approximated by using a least square approach. That is, $\boldsymbol{Q}$ is determined in such a way as to minimize the sum of the square errors obtained by comparing the sample image points with the image, by (3), of the corresponding space points.

Ideally, one should minimize the total error given by

$$\sum (u_i - u'_i)^2 + (v_i - v'_i)^2 \qquad (6)$$

where $\boldsymbol{m}'_i = (u'_i, v'_i)$ is the image of $\boldsymbol{M}_i$ by the projective transformation given by $\boldsymbol{Q}$. That is:

$$u'_i = \frac{q_1 M_i + q_{14}}{q_3 M_i + q_{34}} \text{ and } v'_i = \frac{q_2 M_i + q_{24}}{q_3 M_i + q_{34}} \qquad (7)$$

However, the resulting least squares problem is nonlinear. To reduce computational effort, we may consider solving, instead, the problem of minimizing the total square error in equations (5), which is given by:

$$\sum (q_1 M_i + q_{14} - u_i q_3 M_i - u_i q_{34})^2 + (q_2 M_i + q_{24} - v_i q_3 M_i - v_i q_{34})^2 \qquad (8)$$

This problem has the trivial solution $\boldsymbol{Q} = 0$. This comes from the fact that all matrices of the form $\lambda\boldsymbol{Q}$, where $\lambda \in \Re$, give the same projective transformation. Thus, it is necessary to add a normalization constraint. [Faugeras93] suggests using the constraint $\|\boldsymbol{q}_3\| = 1$. The result is a nonlinear constrained minimization problem that can be solved by eigenvalue methods.

In the next sections we present a modified calibration method that uses exclusively linear least squares and that yields a true camera, through a second calibration step.

4. First calibration step

The first step in our method consists of minimizing (8), but adding a linear constraint instead of the nonlinear constraint $\|\boldsymbol{q}_3\| = 1$. If we were truly minimizing (6), all such normalizations would be equivalent, since scalar multiples of $\boldsymbol{Q}$ give the same projective transformation. In the linear version, however, different normalizations lead to different estimates for $\boldsymbol{Q}$.

We adopt the constraint

$$\boldsymbol{q}_3\boldsymbol{M'} + q_{34} = 1,$$

where $\boldsymbol{M'}$ is the centroid of the sample points $\boldsymbol{M}_i$.

The idea behind this choice is setting the denominator of the projective transformation to 1 for a "typical" point of the scene. Since the remaining points are not to far from their centroid, one may expect that the error incurred when using the linear model instead of the nonlinear one will not be too serious.

Therefore, the problem to solve is of the form:

$$\begin{aligned} &\text{Min} && \| \boldsymbol{Aq} \| \\ &\text{subject to} && \boldsymbol{a}^{\mathrm{T}}\boldsymbol{q} = 1, \end{aligned}$$

where $\boldsymbol{a}$ and $\boldsymbol{q}$ are 12×1 vectors and $\boldsymbol{A}$ is a $2N \times 12$ matrix, built from the sample points.

The problem can be solved using Lagrange multipliers, leading to the following linear system:

$$\begin{cases} \mathbf{A}^T\mathbf{A}\mathbf{q} - \lambda\mathbf{a} = 0 \\ \mathbf{a}^T\mathbf{q} = 1 \end{cases}$$

where λ is the Lagrange multiplier associated with the linear constraint.

The quality of the resulting estimate $\boldsymbol{Q}$ can be improved through a sequential process. Using $\boldsymbol{Q}$, obtained as indicated above, we compute the denominators

$$d_i = \mathbf{q}_3\mathbf{M}_i + \mathbf{q}_{34}$$

of the expressions in (7). Next, we solve the following least squares problem:

$$\min \sum \left(\frac{q_1 M_i + q_{14} - u_i q_3 M_i - u_i q_{34}}{d_i} \right)^2 + \left(\frac{q_2 M_i + q_{24} - v_i q_3 M_i - v_i q_{34}}{d_i} \right)^2$$

$$\text{subject to} \quad \mathbf{q}_3\mathbf{M}' + \mathbf{q}_{34} = 1.$$

This yields a second approximation for $\boldsymbol{Q}$, and so on. In fact, this provides a sequential linear least squares method to solve the nonlinear problem of minimizing (6). In the tests performed with *Virtual Referee*, however, we found that the numerical changes resulting from using this sequential improvement process were very small and the perceptual changes were practically non existent.

The result obtained in this step, with or without the sequential optimization method, is a projective transformation that is not necessarily a camera transformation, since condition (4) was not imposed. This may have little importance for some applications (for instance, when one just wishes to locate the players on the field). But it has very serious implications in our case, where the results will be used to build a synthetic camera in some graphics system. Although most of these systems allow the user to specify a camera directly through an arbitrary $\boldsymbol{Q}$ matrix (thus allowing the use of projections that do not correspond to the "pin-hole" camera model), this makes it harder to change visualization parameters in a natural way. For instance, if the camera is moved to another position, the corresponding image may present unwanted distortions, as discussed in section 7.

The solution that we propose for this problem is using matrix $\boldsymbol{Q}$ obtained in the first step as a starting point for a restricted calibration method, as explained in the next section.

5. Second calibration step

In this second step, we obtain a true camera (i.e., a matrix $\overline{\mathbf{Q}}$ satisfying condition (4)), starting from matrix $\boldsymbol{Q}$

obtained in the previous step. To do so, we first set the camera **position** and the **direction orthogonal to the image plane** based on information provided by ***Q***. Next, we readjust the camera parameters, using the same sample points, but ensuring that we have a true camera transformation.

We first find the position $\mathbf{C} = (x_0, y_0, z_0)$ of the camera optical center, according to ***Q***. In a camera transformation, the optical center is the only point for which there is no corresponding image (proper or improper). Our estimate for **C** is the point in 3D space that has the same property regarding the projective transformation given by ***Q***. Therefore we choose **C** so that the homogeneous coordinates of its image by that transformation are all equal to zero, which is done by solving

$$\begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

In order to obtain an estimate for the camera main axis, we first determine the point on the field whose projection, according to ***Q***, is the center of the screen. This is the point **R** that has homogeneous coordinates $[x, y, 0, w]$ satisfying:

$$\begin{bmatrix} \boldsymbol{q}_1 & q_{14} \\ \boldsymbol{q}_2 & q_{24} \\ \boldsymbol{q}_3 & q_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ w \end{bmatrix} = \begin{bmatrix} u_0 \\ v_0 \\ 1 \end{bmatrix} \tag{9}$$

where u_o and v_0 are the center screen coordinates. Vector $\mathbf{r}_3$, that results from normalizing **R**–**C**, is our estimate for the direction orthogonal to the image plane.

Once we have set the above parameters, the camera model given by equation (2) becomes much simpler. First, we translate the reference systems for 3D space and for the image, moving their origins to **C** and to the center of the image. In these reference systems, the coordinates of the sample points and their respective images are given by:

$$(\bar{x}, \bar{y}, \bar{z}) = (x - x_0, y - y_0, z - z_0) \quad \text{and} \tag{10}$$

$$(\bar{u}, \bar{v}) = (u - u_0, v - v_0)$$

Now, using homogeneous coordinates, the relationship between 3D points and their images can be expressed by:

$$\begin{bmatrix} \bar{u} \\ \bar{v} \\ \bar{w} \end{bmatrix} = \begin{bmatrix} \alpha_u \mathbf{r}_1 & 0 \\ \alpha_v \mathbf{r}_2 & 0 \\ \mathbf{r}_3 & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{z} \\ 1 \end{bmatrix} \tag{11}$$

Since $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$ determine an orthonormal basis, with known $\mathbf{r}_3$, our problem becomes one of estimating $\mathbf{q}_1 = \alpha_u$ and $\mathbf{q}_2 = \alpha_v$, subject to constraints:

$$\mathbf{q}_1 . \mathbf{r}_3 = 0 \tag{12a}$$

$$\mathbf{q}_2 . \mathbf{r}_3 = 0 \tag{12b}$$

$$\mathbf{q}_1 . \mathbf{q}_2 = 0 \tag{12c}$$

The problem becomes simpler if we assume, in addition, that the image aspect ratio is 1 (that is, the lengths corresponding to the sides of an image pixel are equal). Under these conditions, we have $\alpha_u = \alpha_v$, which allows avoiding the nonlinear constraint (12c). Indeed, in this case we have $\mathbf{q}_2 = \mathbf{r}_3 \times \mathbf{q}_1$. As a consequence, the coordinates of $\mathbf{q}_2$ are expressed linearly in terms of the coordinates of $\mathbf{q}_1$. These, in turn, may be parameterized in terms of only two components, due to constraint (12a).

Let $\mathbf{r}_3 = (a, b, c)$. If we designate by s and *t* the first two components of $\mathbf{q}_1 = (s, t, w)$, the third component *w* satisfies $as + bt + cw = 0$. Thus $w = -(a.s + b.t)/c$. Since $\mathbf{q}_2 = \mathbf{r}_3 \times \mathbf{q}_1$, its components are given by $\boldsymbol{q}_2 = (b.w - t.c, s.c - a.w, a.t - s.b)$.

Using these expressions, together with the expressions in (10) and (11) we obtain the following expressions for the coordinates (*u*', *v*') of the image point corresponding to point (*x*, *y*, *z*):

$$u' = \frac{r_1(x - x_0, y - y_0, z - z_0)}{r_3(x - x_0, y - y_0, z - z_0)} =$$

$$= \frac{s.(x - x_0) + t.(y - y_0) - \dfrac{a.s + b.t}{c} \times (z - z_0)}{a.(x - x_0) + b.(y - y_0) + c.(z - z_0)}$$

$$v' = \frac{r_2(x - x_0, y - y_0, z - z_0)}{r_3(x - x_0, y - y_0, z - z_0)} =$$

$$= \frac{\left(b.\dfrac{a.s + b.t}{c} - t.c\right).(x - x_0) + \left(s.c - a.\dfrac{a.s + b.t}{c}\right).(y - y_0) + (a.t - s.b).(z - z_0)}{a.(x - x_0) + b.(y - y_0) + c.(z - z_0)}$$

Both expressions are linear in *s* and *t*, which are the only unknown quantities in them. Estimates for *s* and *t* can then be obtained minimizing the error given by:

$$\sum (u_i - u'_i)^2 + (v_i - v'_i)^2 .$$

Matrix $\overline{\mathbf{Q}}$ resulting from the process provides a true camera transformation.

6. An integrated modeling environment

As explained before, the starting point for the modeling process is an image, as the one shown in Fig. 1. In such images, there are two classes of points that provide input for the modeling process: reference points (marked ×) and object points (marked +), as shown in Fig. 4. Reference points (field markings, goal posts) have known 3D positions and are used in the camera calibration process. Object points are used to locate objects (players, referee, and the ball) relative to the field.

Fig. 4 – Reference and object points

Using the reference points, we execute the calibration process described in the previous sections. With the information provided by the output of the process, we are able to generate a synthetic image of the soccer field, with the objects and field markings, and superimpose it to the original image. To do so, we have to determine the visualization parameters in the graphics system to be used. For the OpenGL ([Neider+93]) graphics library, for instance, this implies defining the camera position and the view volume (frustum).

Positioning the camera involves setting the observer's position, the view reference point and an up vector. The observer's position is the optical center $\mathbf{C} = (x_0, y_0, z_0)$ found before. The view reference point is $\mathbf{R}$. Vector $\mathbf{q}_2$ provides an up vector.

In order to determine the frustum (view volume), we must find the vertical viewing angle, the window aspect ratio and the near and far clipping values. To find the viewing angle, we determine two 3D points that project onto the middle points of the top and bottom lines of the image, respectively. That is, we determine, by solving a system of linear equations, points of the form $(x, y, 0)$ whose projections are located at those positions. The two points thus determined, together with the observer's position, define the desired vertical viewing angle. The window aspect ratio is directly given by the image dimensions and the clipping values are chosen in such a way that the entire scene is comprised in the view volume.

After establishing the viewing parameters, we can superimpose synthetic information onto the original image, as shown in Fig. 5, or generate a completely synthetic image, as shown in Fig. 6.

Fig. 5 – Superimposing synthetic information

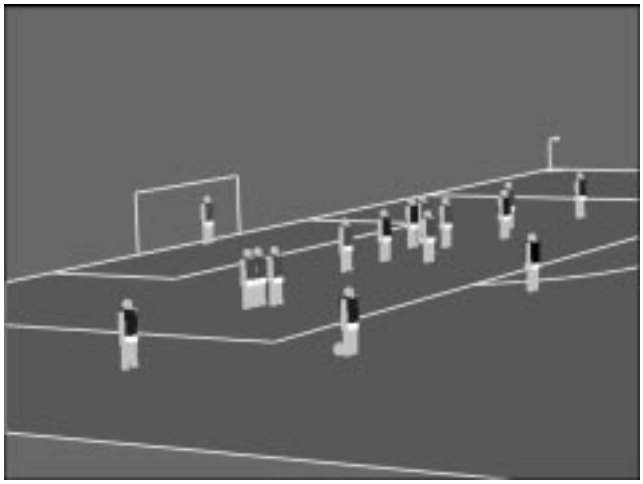


Fig. 6 – A purely synthetic image

Finally, the viewing parameters can now be changed in order to provide different views of the scene, as in Fig. 7.

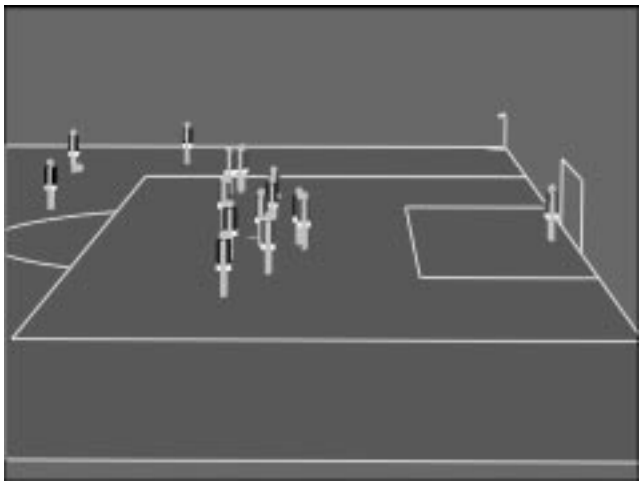


Fig. 7 – A different view

The images in figures 6 and 7 were produced using a publicly available application called Juiz Virtual [JuizVirtual98]. The application's interface allows the user to load an image and specify both reference and object points. Besides generating the corresponding 3D model, Virtual Referee is also able to perform measurements on the field, directly in the 2D image.

7. Comparing camera models

In Fig. 8 we compare the results obtained with the general projective transformation computed at the end of the first calibration step and the true camera at the end of the second step. For each case, we exhibit an image obtained by superimposing the field markings onto the original image and a synthetic view from a different vantage point, obtained by rotating the objects around an axis which is perpendicular to the field and passes through the point that projects onto the center of the screen. As one should expect, we have a better adjustment between the lines on the field and their synthetic counterpart when using only the first step, since it uses a more general model. For this reason, positioning the objects (players, ball and referee) in the 3D model is done using this transformation. However, the corresponding synthetic image presents severe distortions, because the transformation is not a true camera. The original image was produced by a camera oriented in such a way that its lines are very near the horizontal position. Thus, in the image corresponding to the user being positioned behind one of the goals, lines parallel to the goal line should be at least approximately parallel. It is easy to see that this is not the case in (a). Applying the second calibration step corrects this problem, as shown in (b).

We also observe that neither method exactly fits the lines on the image. This is due to distortions, caused by lenses or by the processing of the video signal. A better adjustment to the image would result from applying methods that take into account nonlinear lens behavior ([Tsai86]). Such a camera, however, would be harder to integrate into a synthetic camera environment.

8. Conclusions and Future Work

We have described a method for camera calibration that is suitable for integrating image information into a modeling environment. Since it uses a simple “pin-hole” camera model, it provides information that is easily used in an environment using simple synthetic cameras. The method presented here is easily implemented, since it uses only linear least squares formulations.

When the camera is at its most common position (close to the center of the field), our method produces renderings of the synthetic 3D scenes that are very close to the original image, providing a good departure point for user navigation. This adjustment is not so good when the camera is near the goal area. In this case, the 3D model is still accurate (since object positioning is done using the general projective model), but the initial synthetic image may not be as similar to the original image as desired. To circumvent this problem, we have implemented a local search procedure that checks for better adjustment in a neighborhood of the initial camera position. We are currently investigating efficient ways to perform this search that seems to provide good results.

The use of Juiz Virtual during the world cup of 98 have proved relevant to use other information present in a broadcasted image of sport events such as circles or field patterns. We intend to improve our calibration process to take into account filed information other than individual points.

We also plan to investigate techniques to make the procedure more automatic, by recognizing reference points without user assistance.

9. Acknowledgments

This work was developed in Visgraf/IMPA and TeCGraf/PUC-Rio and was partially funded by CNPq. TeCGraf is a Laboratory mainly funded by PETROBRAS/CENPES.

10. References

[Faugeras93] Faugeras, O., "*Three-Dimensional Computer Vision: a Geometric ViewPoint*", MIT Press, 1993.

[Neider+93] Neider, J. et al. "*OpenGL Programming Guide: the Official Guide to Learning OpenGL*", release 1, Addison-Wesley Publishing Company, 1993.

[Debevec+96] Debevec, P. et al, "*Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach*", SIGGRAPH 96 Conference Proceedings, 1996, pages 11-20.

[Tsai86] Tsai, R., "*An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision*", Proceedings of IEEE Conference on Computer Vision and Pattern Reconigtion, Miami Beach, FL, 1986, pages 364-374.

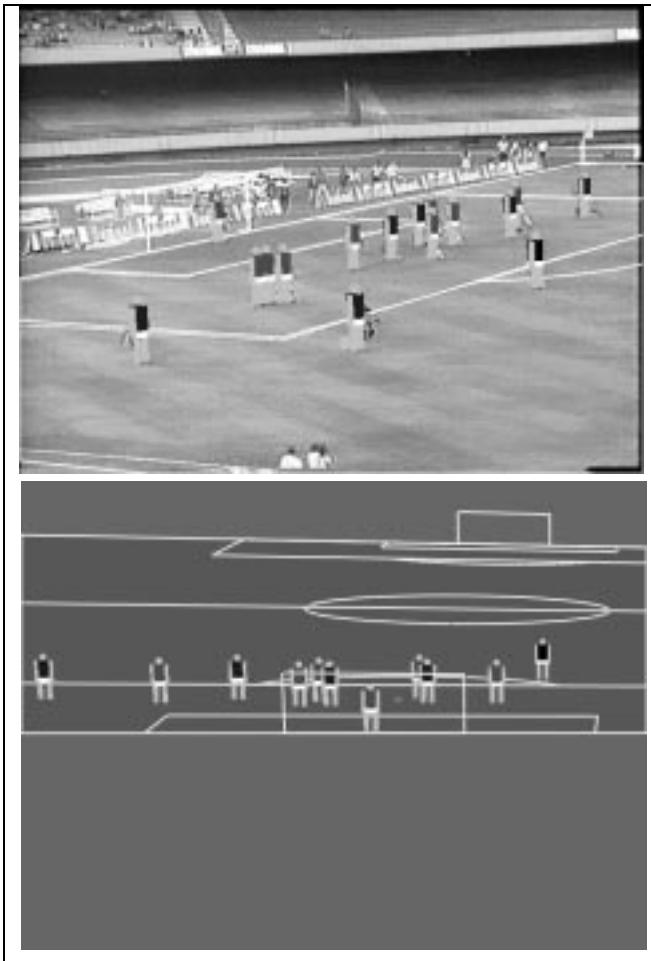
[Horry+97] Horry, Y. et al, "*Tour Into the Picture: Using a Spidery Mesh Interface to Make Animation from a Single Image*", SIGGRAPH 97 Conference Proceedings, 1997, pages 225-232.

[Heckbert86] Heckbert, P, "*Survey of Texture Mapping*", IEEE Computer Graphics and Applications, November, 1986, pages 56-67.

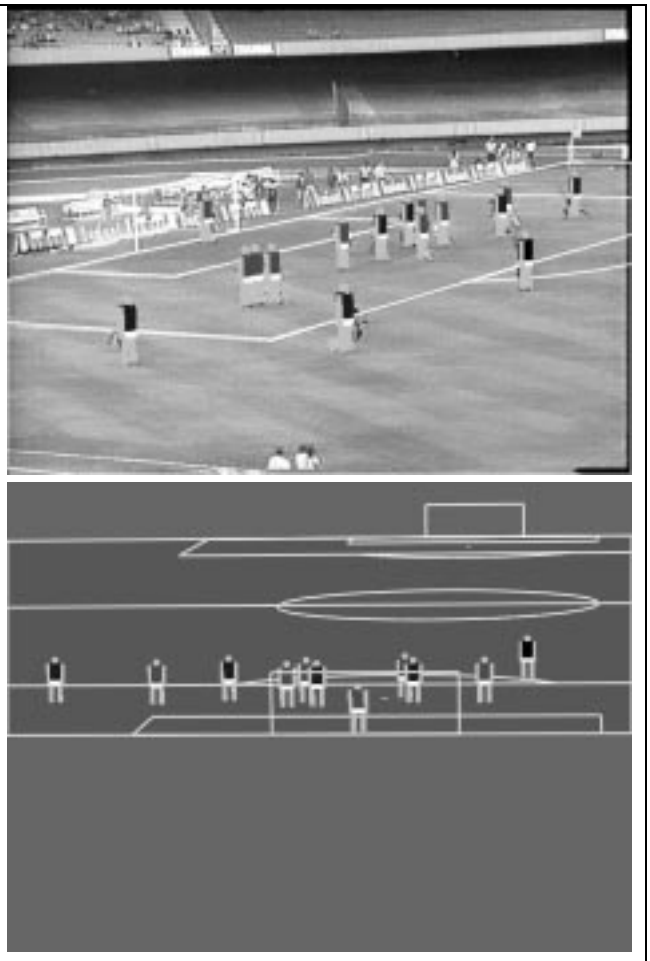
[Gonzalez+92] Gonzalez, R. et al, "*Digital Image Processing*", Addison-Wesley Publishing Company, 1992.

[FIFA] http://www.fifa.com/fifa/handbook/laws/law.1.frame.html

[JuizVirtual98] http://www.tecgraf.puc-rio.br/juizvirtual or http://www.visgraf.impa.br/juizvirtual



(a) After first step



(b) After second step

Fig. 8 – Comparing camera models