Self-calibration of a pair of stereo cameras in general position

Raúl Rojas

Institut für Informatik
Freie Universität Berlin
Takustr. 9, 14195 Berlin, Germany

Abstract. This paper shows that it is possible to calibrate a pair of stereo video cameras using an object in parabolic flight which is visible to both cameras. No point correspondences are needed, the cameras can but do not need to be triggered simultaneously. The world coordinates of the imaged object are not needed. Using our method it is possible to compute the relative 3D rotation and displacement of each camera with respect to a world coordinate system, and among themselves. This method can be used for performing fast self-calibration of static video cameras whose actual pose is unknown and is difficult to adjust. This method goes beyond self-calibration techniques for stereo camera rigs in which three images from a displaced rig, and eight point correspondences are needed.

1 Introduction

Stereoscopic vision algorithms match the images from two or more cameras overlooking the same scene. Knowing the position and orientation of the cameras, it is possible to reconstruct the 3D coordinates of objects from their relative parallax in the different camera views. Estimating the pose and position of each camera is done in an initial calibration step, in which reference points with known 3d coordinates are used. Once the orientation and position of the cameras is known, standard methods of stereoscopy can be applied for three-dimensional scene reconstruction.

There has been much interest in the calibration of stereo cameras without having to use fixed and known reference points.

Brooks proposed to let a robot with stereoscopic cameras drive. The robot tracks salient points, and from the point correspondences from frame to frame and knowledge aboutv the movement of the robot, it is possible to calibrate the stereo cameras.

Luong and Faugeras generalized this result. The displace a stereo camera rig and take three images, in which at least eight point correspondences have to be found. from this information, the relative orientation and displacement of the stereo cameras follows.

Kim et al. [3] studied a related problem: the reconstruction of the parabolic flight of a ball from a video of a soccer game. However, their method is based on using the two extremes of the parabola (when the ball touches the ground, at the start and at the end of the ballistic motion), and is not suitable for parabolic motion without a reference plane. A variation of their method, in which they adjust a quadratic function to many alternative planes of motion, searching among them for an optimal fit, is too cumbersome and inefficient. The method described here, by contrast, is direct, does not require any search driven computation, and can be used for forecasting future motion using only three video frames just after the kick.

2 Projective Transformation and Reconstructed Path

We adopt the following conventions. The world has its own system of coordinates, as well as each camera. Both cameras are in general position. Figure 1 shows how the three coordinate systems (field and cameras) are related. In the general case, the three axis of the camera's system of coordinates are rotated relative to the world's coordinate axis. Let us denote by R_1 (R_2) the rotation matrix needed for transforming from world to the first (second) camera coordinates, and by ℓ_1 (ℓ_2) the translation vector from the origin of the world system to the first (to the second) camera coordinate system. Therefore, a point with world coordinates $p = (x, y, z)^{\mathrm{T}}$ has coordinates $q = R_1(p - \ell_1)$ in the first camera's coordinate system. The inverse transformation, from camera to world coordinates, is therefore $p = R_1^{-1}q + \ell_1$.

We first discuss one camera. In what follows the words"the camera" refer to any one of the two cameras.

We assume, for the sake of the computation, and without losing generality, that the camera imaging chip is at a unit distance from the pinhole. The point where parabolic flight starts to be measured has camera coordinates (x_0, y_0, z_0). The velocity of the ball, after the kick, is given by the vector $v = (v_x, v_y, v_z)$. The parabolic flight of the ball is then described by the following parameterized path,

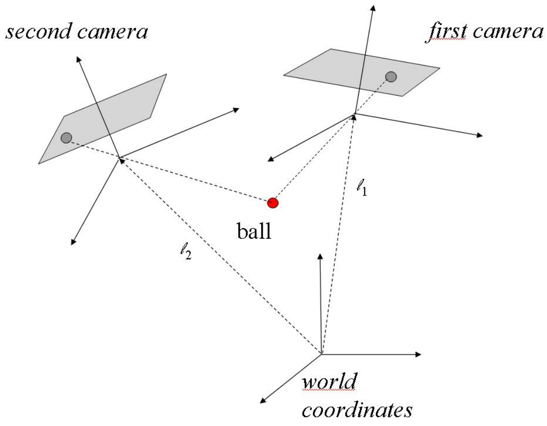

Fig. 1. Coordinate system of each camera, and world coordinates

in the camera's coordinate system:

$$(x, y, z) = \left(x_0 + v_x t + \frac{1}{2} g_x t^2,\ y_0 + v_y t + \frac{1}{2} g_y t^2,\ z_0 + v_z t + \frac{1}{2} g_z t^2\right)$$

where t is the time elapsed, starting with the first data point ($t = 0$ for that point), and (g_x, g_y, g_z) are the components of the earth's acceleration in the camera's coordinate system (which can be tilted with respect to the vertical).[1] The components of the earth's acceleration with respect to the camera system could be computed if we knew the rotation matrix R:

$$(g_x, g_y, g_z)^{\mathrm{T}} = R(0, 0, -9.8)^{\mathrm{T}}$$

but in what follows (g_x, g_y, g_z) is unknown. Let us assume that the path is never such that $z = 0$ (in order to avoid numerical exceptions in what follows). The projection of the position of the ball in the image plane of the camera (at a unit distance from the pinhole) is then

$$\left(\frac{x_0 + v_x t + \frac{1}{2} g_x t^2}{z_0 + v_z t + \frac{1}{2} g_z t^2}, \frac{y_0 + v_y t + \frac{1}{2} g_y t^2}{z_0 + v_z t + \frac{1}{2} g_z t^2}\right)$$

Assume now that m points in m images are given, where the "virtual" ball has been detected at times $t_1, t_2, \ldots, t_m$ (setting $t_1 = 0$). Let us denote the coordinates of the m points with respect to the camera system, and on the

[1] Gravitational acceleration varies with the latitude, because the earth is not perfectly spherical. The Geodetic Reference formula of 1967, used by geographers, is given by $g = -9.7803185(1 + 0.005278895\sin^2\phi - 0.000023462\sin^4\phi)$ m/s. Surface features of the earth are not considered in the formula.

imaging plane, by $(\alpha_1, \beta_1), \ldots, (\alpha_m,\, \beta_m)$. Then, since "virtual ball" and real ball have the same projection on the camera chip, we have in general:

$$(\alpha_i, \beta_i) = \left(\frac{x_0 + v_x\, t_i + \frac{1}{2} g_x t_i^2}{z_0 + v_z t_i + \frac{1}{2} g_z t_i^2}, \frac{y_0 + v_y\, t_i + \frac{1}{2} g_y t_i^2}{z_0 + v_z t_i + \frac{1}{2} g_z t_i^2} \right) \qquad \text{Eq.1}$$

Note that α_i and β_i are measured in meters. That means that the pixel position in the image has to be multiplied by a constant which relates pixels to metric units (which is a parameter known for each camera, or which can be computed from the focal distance, lens mount, and chip size). From the expression above (and for the i-th point) we can derive two linear equations:

$$z_0 \alpha_i + v_z \alpha_i t_i - x_0 - v_x t_i - \frac{1}{2} g_x t_i^2 + 0 \cdot y_0 + 0 \cdot v_y t_i + 0 \cdot \frac{1}{2} g_y t_i^2 = -\frac{1}{2} g_z \alpha_i t_i^2$$

and

$$z_0 \beta_i + v_z \beta_i t_i + 0 \cdot x_0 + 0 \cdot v_x t_i + 0 \cdot \frac{1}{2} g_x t_i^2 - y_0 - v_y t_i - \frac{1}{2} g_y t_i^2 = -\frac{1}{2} g_z \beta_i t_i^2$$

We have two linear equations for nine variables. Therefore, five points on the parabolic flight path provide enough equations (ten) which can be used to solve the system. If we use more than 5 points, let us say m, then the general form of the system of equations we obtain is

$$D(z_0, v_z, x_0, v_x, g_x, y_0, v_y, g_y)^{\mathrm{T}} = d$$

where D (for data) is a $2m \times 8$ matrix and d is a $2m$-dimensional vector. Since the system of equations is homogeneous (remember that (g_x, g_y, g_z) is unknown), we transform it into a non-homogeneous system by setting tentatively $g_z = 1$ (we are assuming that $g_z = 0$ does not occur, and if it occurs it can be detected). Using the pseudoinverse D^+ of D we find the solution

$$(z_0, v_z, x_0, v_x, g_x, y_0, v_y, g_y)^{\mathrm{T}} = D^+ d$$

where $D^+ = (D^T D)^{-1} D^T$. The pseudoinverse allows us to use as many points for the calculation as we have already measured, because we are interested in producing an estimate of the flight trajectory of the ball as precise as possible.

Since we solved the homogeneous system of equations setting $g_z = 1$, we need to normalize the length of the (g_x, g_y, g_z) vector to 9.8. Setting $c = ||(g_x, g_y, g_z)||$, we obtain the final result

$$(z'_0, v'_z, x'_0, v'_x, g'_x, y'_0, v'_y, g'_y) = -\frac{9.8}{c}(z_0, v_z, x_0, v_x, g_x, y_0, v_y, g_y)$$

and $g'_z = -\frac{9.8}{c}$. Now we compute the rotation matrix for one of the two cameras. Let us call it R (it can be R_1 or R_2).

For the rotation matrix $R = \{r_{ij}\}$, for $i, j = 1, 2, 3$, it must be true that $R(g_x, g_y, g_z)^{\mathrm{T}} = (0, 0, -9.8)$. This means that we can set the third column of R as:

$$(r_{13}, r_{23}, r_{33})^{\mathrm{T}} = -\frac{1}{9.8}(g'_x, g'_y, g'_z)^{\mathrm{T}}$$

We can now set the second column of R to a vector orthogonal to $-(g_x, g_y, g_z)^{\mathrm{T}}$ and to the vector $(v'_x, v'_y, v'_z)^{\mathrm{T}}$. We choose $(v'_x, v'_y, v'_z)^{\mathrm{T}}$ because we can obtain this same vector for both cameras. We also assume that the object in parabolic flight was not just thrown in an up-down trajectory (in which case the vector $(v'_x, v'_y, v'_z)^{\mathrm{T}}$ would be parallel to the gravitation vector $(g_x, g_y, g_z)^{\mathrm{T}}$ We therefore set

$$(r_{12}, r_{22}, r_{32})^{\mathrm{T}} = w^{\mathrm{T}}/||w||$$

where

$$w^{\mathrm{T}} = -(g_x, g_y, g_z)^{\mathrm{T}} \times (v'_x, v'_y, v'_z)$$

and × is the vector product operator.

The first column of R is then set to the vector product of the second and third columns. The resulting matrix has all the properties of a rotation matrix.

What we have done is just to find a system of world coordinates in which the z axis is parallel to the vertical direction (as given by the gravity vector), the second axis runs in the direction of the parabolic flight (projected on a plane normal to the vertical direction), and the third axis is normal to the first two axis.

We repeat this computation for each camera and this provides us with two rotation matrices R_1 and R_2. Given a vector p in the first (second) camera system of coordinates, we can transform to the common world system in which gravitation points downward, by computing $R_1^{\mathrm{T}} p$ ($R_2^{\mathrm{T}} p$).

Fig. 2 shows an example of the computation. The world reference system is on the lower left corner of the image. A simulated parabolic path is shown in the image, and the direction of the vector (v'_x, v'_y) on the "floor". The systems of coordinates found for two cameras are shown. The pinhole of each camera is at the origin of the center of coordinates. The rotation matrix R_1^{T} maps coordinates of the first camera to the coordinate system shown in the middle of the figure (with vertical z-axis, and another axis parallel to the direction of motion), whereas the matrix R_2^{T} maps to the second coordinate system shown in the upper right of the figure.

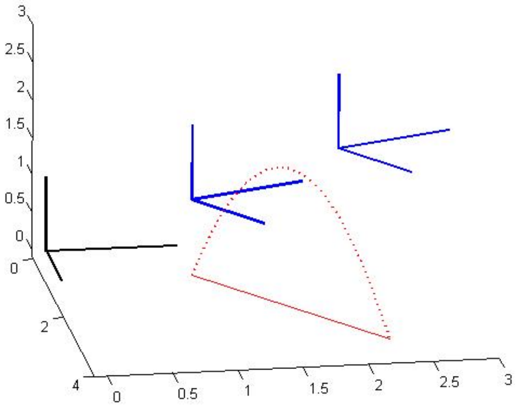

Fig. 2. Computed world coordinate system for each camera, and world coordinates

3 Common clock and camera translation

In the previous section we handle each camera separately. However, there is a common clock for both cameras. That is, we assume that each image from a camera gets a time stamp t when it is received by the same computer. If one computer is handling each camera, we have to make them agree on a common clock.

The advantage of having a common clock is that the point $p_0 = (x'_0, y'_0, z'_0)$ (the start of parabolic flight at $t = 0$) is the same for both cameras (in the coordinate system of each camera). Let us call this point in the first camera system p_0^1 and p_0^2 in the second. The relative displacement of the camera pinholes (the origin of each system of coordinates), in the world system of coordinates, can be computed then as:

$$\ell = R_1^{\mathrm{T}} p_0^1 - R_2^{\mathrm{T}} p_0^2$$

Having ℓ, it is easy to compute the coordinates of a point p relative to the second camera, in the frame of reference of the first camera. We first transform from the second camera to the world system of coordinates:

$$p' = R_2^{\mathrm{T}} p$$

and from this to the second camera system, adding the relative displacement between the cameras:

$$p'' = R_1 R_2^{\mathrm{T}} p + \ell$$

4 Conclusions

The numeric for our stereoscopic calibration method is much simpler than that used by other groups [4], [5], does not require matching points in objects nor the world coordinates of the corresponding points in space.

Our method has the disadvantage of requiring cameras capable of shooting several frames per second (so that enough points can be found along a parabolic trajectory). However, modern cameras for robotic or computer vision applications are capable of this and also of being externally triggered. Detecting a flying object introduces also some noise, which can be minimized by using as many frames as possible in the self-calibration process (that is, as many frames per second as possible given the illumination conditions).

There are several variations of the method described here. One is to use a pendulum for the calibration and not a parabolic path. The equations of motion are different and somewhat more complex, but the advantage is that a pendulum can be started once and its movement can be reused many times until the calibration is finished.

Another variation is to move the robot along a path and use the odometry and acceleration sensors for describing its change of pose. Tracking fixed points in a scene, it is possible to recover the pose of two cameras mounted on the robot, as well as their relative translation. (Brooks)

Another alternative is to throw the robot in the air in parabolic flight.

References

1. I. P. Howard, A. P. Howard, and B. Rogers, *Binocular Vision and Stereopsis*, Oxford University Press, Oxford, 1995.
2. D. Forsythe, and J. Ponce, *Computer Vision: A Modern Approach*, Prentice-Hall, 2003.
3. T.Kim, Y.Seo, and K-S Hong, “Physics-based 3D position analysis of a soccer ball from monocular image sequence”, ICCV 1998, Bombay, India, January 1998, pp. 721–726.
4. R. Hartley, R. Gupta, T. Chang, “Stereo from Uncalibrated Cameras”, *IEEE Conference on Computer Vision and Pattern Recognition*, 1992, pp. 761-764.
5. R. I. Hartley, A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, 2000.