An Optimal Solution for Mobile Camera Calibration

P. Puget, T. Skordas
ITMI, Filiale de CAP SESA
ZIRST, 11 Chemin des Prés, BP 87, 38243 Meylan cédex - France
e-mail: puget@itmi.uucp, skordas@itmi.uucp

Abstract
This paper addresses the problem of determining the intrinsic and extrinsic parameters of a mobile camera. We present an optimal solution which consists of the following steps: first, the camera is calibrated in several working positions and for each position, the corresponding transformation matrix is computed using a method developed by Faugeras and Toscani [1]; next, optimal intrinsic parameters are searched for all positions; finally, for each separate position, optimal extrinsic parameters are computed by minimizing a mean square error through a closed form solution. Experimental results show that such a technique yields a spectacular reduction of calibration errors and a considerable gain relative to other existing on-site calibration techniques.

1 Introduction

This paper addresses the problem of determining the optical (internal) camera parameters (called *intrinsic parameters*) and the three-dimensional (3-D) position and orientation of the camera frame relative to some predefined world coordinate frame (*extrinsic parameters*).

This problem is a major part of the general problem of camera calibration and is the starting point for several important applications in Computer Vision and Robotics. 3-D object tracking and dynamic scene reconstruction, stereo vision, stereo calibration and stereo reconstruction, object recognition and localization from a single view, and sensory based navigation, are just a few situations where camera calibration is explicitly needed.

Calibrating a camera is determining the relationship between the 3-D coordinates of a point and the corresponding 2-D coordinates of its image. Such a relationship is usually expressed in terms of a 3×4 matrix M, which is called the *perspective transformation matrix* [2,7]. In other words, a camera is considered being calibrated if for each of its working positions, the corresponding matrix M can be derived. If the camera is fixed, the calibration problem is reduced to the problem of computing a single matrix M. If the camera moves arbitrarily, calibration involves computation of both intrinsic and extrinsic parameters, the latter being necessary for the hand/eye calibration [5,9,10].

Numerous techniques tackling the general calibration problem exist in the literature. An excellent survey of these techniques is given by Tsai in [8]. There are two main approaches for calibrating a moving camera. One approach is to compute the perspective transformation matrix from which the camera parameters are derived, [1,2,6]. In a second approach, that of Tsai [7], and Tsai and Lenz [9], no perspective transformation matrix is explicitly calculated. This method seeks

the solution of a linear matrix equation in five unknowns merely consisting of extrinsic parameters, from which, the rest of the parameters can be easily recovered. This method is based on *a priori* knowledge of two intrinsic parameters. In [9], a method is proposed to compute all intrinsic camera parameters by performing a preliminary off-site calibration stage. This in turn, is likely to cause additional difficulties, since the camera parameters may be altered accidentally when fixing the camera on its working place. Their results in terms of accuracy are similar to those reported by Faugeras and Toscani.

We have performed camera calibration on several camera positions using the technique proposed by Faugeras and Toscani [1]. Among different camera positions, significant differences between homologous intrinsic camera parameters occurred. While the method turns out to be particularly accurate in the computation of the perspective matrix for a given camera position, nevertheless, when the intrinsic and extrinsic parameters are derived from this matrix, the results are disappointing. This is due to the noisy 2-D and 3-D data and to the instability of equations allowing to compute these parameters (see Section 3). Note also, the fact that the perspective matrix is accurate and the intrinsic parameters are sometimes inaccurate, implies that the extrinsic parameters are also affected by errors.

If one particular set of intrinsic parameters (corresponding to a given position) is assumed to be valid everywhere, and is associated with the extrinsic parameters corresponding to another camera position, the resulting errors are enormous (see section 5).

Our approach

Our technique can be summarized as follows: a camera is calibrated at several (N) positions and in each position, a perspective transformation matrix is computed by using the direct method (without applying extended Kalman filtering) of Faugeras and Toscani. Once the N matrices are available, the problem of computing correct intrinsic and extrinsic parameters is broken down into three successive steps: 1) a minimization criterion is formulated leading to the computation of optimal intrinsic parameters; 2) an optimal rotation is computed for each camera position; 3) an optimal translation is computed for each camera position.

The optimization of the rotational and translational parameters is achieved by minimizing a mean square error. All these optimizations are performed using closed-form algorithms. Such a technique yielded a spectacular reduction of errors and a gain in precision of a factor up to 50, relatively to the Faugeras-Toscani approach.

2 The camera model

The camera model that we use is the *pinhole* model. This is the most frequently used model existing in the bibliography [1,2,6,7]. The underlying mathematical model is the perspective transformation: a 3-D point $P = (X, Y, Z)^t$ is projected on a 2-D point $p = (u, v)^t$ in the image plane (see figure 1) where:

$$u = su/s \quad and \quad v = sv/s \tag{1}$$

with:

$$(su,\ sv,\ s)^t = M\ (X,\ Y,\ Z,\ 1)^t \tag{2}$$

Coordinates of P are expressed relatively to some scene coordinate frame, and coordinates of p are expressed relatively to the image frame in pixels. $M = (m_{ij})$ is a 3×4 matrix called the *perspective transformation matrix* and is defined up to a scale factor. Coefficients m_{ij} depend on 10 independent parameters which are the following:

- α_u and α_v, the scale factors on axes u and v of the image frame;

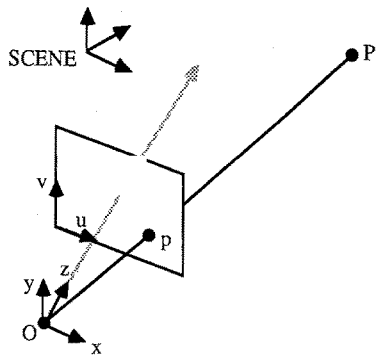

Figure 1: Mathematical model of the camera

- u_0 and v_0, the coordinates in the image frame of the intersection of the optical axis with the image plane;
- t_x, t_y and t_z, the coordinates of a translation vector $\vec{t}$;
- r_x, r_y and r_z, the coordinates of a rotation vector $\vec{r}$.

The first four parameters depend only on the physical characteristics of the camera and are called the *intrinsic camera parameters*. The last six parameters define the geometric transform T between the camera frame and a given scene coordinate frame. They depend on the position of the camera relative to the scene coordinate frame. They are called *extrinsic camera parameters*. Let us define a few notations:

$$M = \begin{pmatrix} \vec{m}_1 & m_{14} \\ \vec{m}_2 & m_{24} \\ \vec{m}_3 & m_{34} \end{pmatrix} \qquad T = \begin{pmatrix} \vec{r}_1 & t_x \\ \vec{r}_2 & t_y \\ \vec{r}_3 & t_z \end{pmatrix}$$

where $\vec{m}_i$ and $\vec{r}_i$ are (1×3) row vectors. Vectors $\vec{r}_i$ define a rotation matrix ρ. Let θ be the angle, $\vec{n} = (n_x, n_y, n_z)^t$ the unit vector defining the axis and $\vec{r} = (r_x, r_y, r_z)^t$ the vector of rotation. These variables are related by the expressions (see [3]):

$$\rho = \begin{pmatrix} \cos\theta + r_x^2 g(\theta) & r_x r_y g(\theta) - r_z f(\theta) & r_x r_z g(\theta) + r_y f(\theta) \\ r_x r_y g(\theta) + r_z f(\theta) & \cos\theta + r_y^2 g(\theta) & r_y r_z g(\theta) - r_x f(\theta) \\ r_x r_z g(\theta) - r_y f(\theta) & r_y r_z g(\theta) + r_x f(\theta) & \cos\theta + r_z^2 g(\theta) \end{pmatrix} \tag{3}$$

$$f(\theta) = \frac{\sin\theta}{\theta}, \qquad g(\theta) = \frac{1 - \cos\theta}{\theta^2}, \qquad \theta = \sqrt{r_x^2 + r_y^2 + r_z^2}, \qquad \vec{r} = \theta\vec{n}$$

Notice that $\vec{r}_i$ as rows of a rotation matrix define an orthonormal frame.

The expression of M as a function of the intrinsic and extrinsic parameters is:

$$M = \omega \begin{pmatrix} \alpha_u \vec{r}_1 + u_0 \vec{r}_3 & \alpha_u t_x + u_0 t_z \\ \alpha_v \vec{r}_2 + u_0 \vec{r}_3 & \alpha_v t_y + u_0 t_z \\ \vec{r}_3 & t_z \end{pmatrix} \tag{4}$$

ω is the scale factor of matrix M. As $\vec{r}_3$ is a unit vector, ω can be identified with $sqrt\vec{m}_3.\vec{m}_3^t$. In the sequel, M is considered with ω equal to 1.

The intrinsic and extrinsic parameters can be calculated by identifying coefficients m_{ij} with their expression given by (4), and by using properties of vectors $\vec{r}_i$. This method is presented and analyzed in the following section.

3 Instability in the computation of the parameters

The following equalities can be derived from the orthogonality of vectors $\vec{r}_i$:

$$\vec{r}_3 = \vec{m}_3 \tag{5}$$

$$\begin{cases} u_0 &= \vec{m}_1.\vec{m}_3 \\ v_0 &= \vec{m}_2.\vec{m}_3 \end{cases} \tag{6}$$

$$\begin{cases} \alpha_u &= \|\vec{m}_1 \times \vec{m}_3\| \\ \alpha_v &= \|\vec{m}_2 \times \vec{m}_3\| \end{cases} \tag{7}$$

$$\begin{cases} \vec{r}_1 &= 1/\alpha_u(\vec{m}_1 - u_0\vec{m}_3) \\ \vec{r}_2 &= 1/\alpha_v(\vec{m}_2 - v_0\vec{m}_3) \end{cases} \tag{8}$$

$$\begin{cases} t_x &= 1/\alpha_u(m_{14} - u_0 m_{34}) \\ t_y &= 1/\alpha_v(m_{24} - v_0 m_{34}) \\ t_z &= m_{34} \end{cases} \tag{9}$$

The equations above are non linear and give rather unstable results, especially the first four ones which allow the computation of the intrinsic parameters and the rotational part of the extrinsic parameters.

A geometrical interpretation of the above calculation shows this instability. Figure 2 depicts the geometrical situation of the vectors $\vec{m}_i$ and $\vec{r}_i$. In this figure, we volunteerly magnified vectors $\vec{r}_i$. For the cameras commonly in use, the order of magnitude of α_u and α_v is 1000 and the order of magnitude of u_0 and v_0 is 250. The problem of finding the rotation and the intrinsic parameters is equivalent to the problem of properly "placing" the orthonormal vectors $\vec{r}_1$, $\vec{r}_2$ and $\vec{r}_3$ relatively to $\vec{m}_1$, $\vec{m}_2$ and $\vec{m}_3$ and of calculating the projections of $\vec{m}_1$ and $\vec{m}_2$ onto the vectors $\vec{r}_i$. This is done through the above equalities by:

1. identifying $\vec{r}_3$ with $\vec{m}_3$ (equality (5));
2. calculating $\vec{r}_1$ and $\vec{r}_2$ as the vectors orthogonal to $\vec{r}_3$ in the planes defined by $\vec{r}_3$ and $\vec{m}_1$ and $\vec{r}_3$ and $\vec{m}_2$, respectively (equality (8));
3. calculating the projections of $\vec{m}_1$ and $\vec{m}_2$ onto the vectors $\vec{r}_i$ (eq (6) and (7)).

The consequences of such a method are:

- even a small error on $\vec{m}_3$ badly propagates and can cause very significant errors in the calculation of vectors $\vec{r}_i$ and of the intrinsic parameters, (notice that especially u_0 and v_0 are extremely sensitive to $\vec{m}_3$),
- the resulting vectors $\vec{r}_i$ are not necessarily orthogonal; notice here that the "c" factor introduced by Toscani [1] not only reflects the non-orthogonality of the u and v axes of the image frame but also the errors produced in vectors $\vec{m}_i$ and $\vec{r}_i$ as well;
- even if the values of the parameters are wrong, they are mutually compatible: if the matrix M is recomputed from these values, results are quite good.

In order to give quantitative estimations of the numerical instability, the intuitive geometrical interpretation given above can be completed with calculations of error propagation through the equations (5) to (7). This is done in appendix A.

4 An optimal solution for computing the parameters

The basic principles of our method are the following:

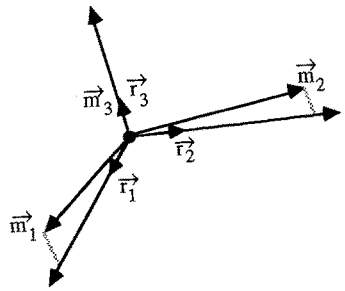

Figure 2: Geometrical interpretation of the calculation of the parameters

1. the intrinsic parameters are computed after having computed the perspective transformation matrix M on *several* positions. The underlying idea is to compute a unique set of intrinsic parameters that is correct for every position of the moving camera;
2. using the formerly computed intrinsic parameters for each calibration position, the extrinsic parameters are computed by two successive optimization algorithms: one for the rotational and one for the translational parameters.

4.1 Computing the intrinsic parameters

We suppose that we have computed N perspective transformation matrices $M^{(i)}$ corresponding to N different positions i of the camera:

$$M^{(i)} = \begin{pmatrix} \alpha_u \vec{r}_1^{(i)} + u_0 \vec{r}_3^{(i)} & \alpha_u t_x^{(i)} + u_0 t_z^{(i)} \\ \alpha_v \vec{r}_2^{(i)} + u_0 \vec{r}_3^{(i)} & \alpha_v t_y^{(i)} + u_0 t_z^{(i)} \\ \vec{r}_3^{(i)} & t_z^{(i)} \end{pmatrix} \tag{10}$$

The principle of the calculation is to look for invariants which are independent of the extrinsic parameters and which allow one to calculate the intrinsic parameters. The most obvious possibility is to choose for these non varying quantities the intrinsic parameters that we could calculate for each separate position. For each position i, we can write:

$$\begin{cases} u_0^{(i)} = \vec{m}_1^{(i)}.\vec{m}_3^{(i)} \\ v_0^{(i)} = \vec{m}_2^{(i)}.\vec{m}_3^{(i)} \end{cases} \qquad \begin{cases} \alpha_u^{(i)} = \|\vec{m}_1^{(i)} \times \vec{m}_3^{(i)}\| \\ \alpha_v^{(i)} = \|\vec{m}_2^{(i)} \times \vec{m}_3^{(i)}\| \end{cases}$$

The optimal value for u_0 is that which minimizes the criterion:

$$C = \sum_{i=1}^{N} (u_0 - u_0^{(i)})^2$$

This is the mean value of $u_0^{(i)}$:

$$u_0 = 1/N \sum_{i=1}^{N} u_0^{(i)}$$

The same reasoning holds for the other three intrinsic parameters.

4.2 Computing the extrinsic parameters

For each camera position, a perspective transformation matrix M is available, and a set of intrinsic parameters have been computed. Now we focus on the calculation of the extrinsic parameters.

Let X_j denote the 3-D points used for the calibration (expressed in the scene coordinate frame) and $(u_j, v_j)^t$ the 2-D coordinates of their projections onto the image plane. Eliminating s in equations (1) and (2) gives:

$$\begin{cases} (u_j\vec{m}_3 - \vec{m}_1).X_j + (u_j m_{34} - m_{14}) = 0 \\ (v_j\vec{m}_3 - \vec{m}_2).X_j + (v_j m_{34} - m_{24}) = 0 \end{cases} \tag{11}$$

Expressing $\vec{m}_j$ as functions of intrinsic and extrinsic parameters leads to:

$$\begin{cases} \alpha_u(\vec{r}_1.X_j + t_x) + (u_0 - u_j)(\vec{r}_3.X_j + t_z) = 0 \\ \alpha_v(\vec{r}_2.X_j + t_y) + (v_0 - v_j)(\vec{r}_3.X_j + t_z) = 0 \end{cases} \tag{12}$$

The problem is now to find $\vec{r}_1$, $\vec{r}_2$, $\vec{r}_3$, t_x, t_y, t_z so that equations (12) are verified for all points X_j. To separate the calculation of the translational part from the rotational part, we must first state the following lemma:

Lemma 1 *If X_j is considered as the unknown in equations (11), then the point $X_0 = -\rho^{-1}\vec{t}$ is a solution for all j.*

Proof: $(\vec{r}_1.X_0 + t_x)$, $(\vec{r}_2.X_0 + t_y)$, $(\vec{r}_3.X_0 + t_z)$ are the coordinates of the point $\rho X_0 + \vec{t}$. This vector is equal to $(-\rho\rho^{-1}\vec{t} + \vec{t})$, or the zero vector. The three coordinates are equal to zero and X_0 is then a solution of equations (12) for all j, and consequently a solution of the equations (11) which are equivalent. □

Putting all equations (11) together gives a linear system. Its solution X_0 can be calculated by a least squares procedure.

The geometrical interpretation of lemma 1 is that X_0 represents the coordinates of the optical center of the camera, expressed in the scene coordinate frame. The equalities (11) considered as equations whose unknown is X_j are the equations defining the locus which is projected in $(u_j, v_j)^t$. This locus is the straight line passing through the optical center and the point $(u_j, v_j)^t$ of the image plane. When considering all equations for all j, the unique solution is the point which is the intersection of all these lines: this is the optical center.

4.2.1 Computing the rotation

Equalities (11) are now equivalent to:

$$\begin{cases} (u_j\vec{m}_3 - \vec{m}_1).X_j + (u_j m_{34} - m_{14}) = (u_j\vec{m}_3 - \vec{m}_1).X_0 + (u_j m_{34} - m_{14}) \\ (v_j\vec{m}_3 - \vec{m}_2).X_j + (v_j m_{34} - m_{24}) = (v_j\vec{m}_3 - \vec{m}_2).X_0 + (v_j m_{34} - m_{24}) \end{cases} \tag{13}$$

which leads to:

$$\begin{cases} \alpha_u\vec{r}_1.(X_j - X_0) + (u_0 - u_j)\vec{r}_3.(X_j - X_0) = 0 \\ \alpha_v\vec{r}_2.(X_j - X_0) + (v_0 - v_j)\vec{r}_3.(X_j - X_0) = 0 \end{cases} \tag{14}$$

As $\vec{r}_i.(X_j - X_0)$ is the i^{th} coordinate of the vector $\rho(X_j - X_0)$, the first of equations (14) means that $\rho(X_j - X_0)$ is orthogonal to the vector $(\alpha_u, 0, (u_0 - u_j))$. For the same reason the second equation means that the vector $\rho(X_j - X_0)$ is orthogonal to the vector $(0, \alpha_v, (v_0 - v_j))$. This implies that the direction of the vector $\rho(X_j - X_0)$ is given by the cross-product of the two previous vectors:

$$N_j = \begin{pmatrix} \alpha_v(u_0 - u_j) \\ \alpha_u(v_0 - v_j) \\ -\alpha_u\alpha_v \end{pmatrix}$$

The geometrical interpretation of this result is that for every point X_j, the line joining the optical center and X_j must be colinear to the line joining the optical center and the image point (u_j, v_j).

Let us define:

$$S_j = \frac{X_j - X_0}{\|X_j - X_0\|} \qquad \text{and} \qquad Q_j = \frac{N_j}{\|N_j\|}$$

The problem of finding the rotation is finally equivalent to finding ρ which minimizes the criterion:

$$C_\rho = \sum_{j=1}^{M} \|\rho S_j - Q_j\|^2$$

To solve this problem, we consider the unit quaternion q associated with ρ. It is shown in appendix B that q is the unit eigenvector associated with the smallest eigenvalue of the positive symmetric matrix B where:

$$B = \sum_{j=1}^{M} (B_j^t B_j)$$

with B_j given by:

$$B_j = \begin{pmatrix} 0 & -S_{jx} + Q_{jx} & -S_{jy} + Q_{jy} & -S_{jz} + Q_{jz} \\ S_{jx} - Q_{jx} & 0 & S_{jz} + Q_{jz} & -S_{jy} - Q_{jy} \\ S_{jy} - Q_{jy} & -S_{jz} - Q_{jz} & 0 & S_{jx} + Q_{jx} \\ S_{jz} - Q_{jz} & S_{jy} + Q_{jy} & -S_{jx} - Q_{jx} & 0 \end{pmatrix} \tag{15}$$

4.2.2 Computing the translation

Equalities (12) can now be considered as equations whose unknowns are t_x, t_y, t_z:

$$\begin{cases} (\alpha_u \vec{r}_1 . X_j) t_x + (u_0 - u_j) t_z + \alpha_u \vec{r}_1 . X_j + (u_0 - u_j) \vec{r}_3 . X_j = 0 \\ (\alpha_v \vec{r}_2 . X_j) t_y + (v_0 - v_j) t_z + \alpha_v \vec{r}_2 . X_j + (v_0 - v_j) \vec{r}_3 . X_j = 0 \end{cases} \tag{16}$$

These equations are linear and we have two of them available for each calibration point X_j. The optimal value for $\vec{t} = (t_x, t_y, t_z)^t$ is computed with a classical least-squares algorithm.

5 Experimental results

Experimental setup

The experimental setup is shown on figure 3. A Pulnix TM 560 R camera is mounted on a robot wrist. This camera has a 2/3 inches, 500 × 582 elements, CCD sensor with a 16mm TV lens. We use a VME board for image acquisition. The calibration pattern is a grid of lines drawn with a laser printer. A 3-D calibration point is determined as the intersection of two lines. The calibration pattern is fixed on an horizontal metal plate moving along the z axis. The images are taken at an approximate distance from 40 to 50 cm. The number of points used for the calibration is about 100, lying on two to four distinct planes. The 2-D calibration points in the image are determined by extracting junctions. These junctions are found as intersections of lines segments detected in the image.

Error evaluation

To evaluate and compare our method with other methods, we used two criteria which are now presented. For each 3-D point X_j, let $p_j = (u_j, v_j)^t$ denote its observed projection on the image, and $p_j^{mod} = (u_j^{mod}, v_j^{mod})^t$ its projection given by the camera model corresponding to the parameters

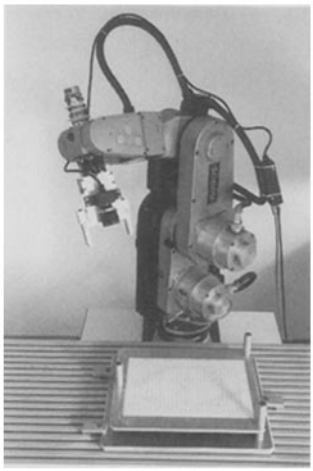

Figure 3: The experimental setup

that have been found. For each point X_j, this gives an error vector $\vec{\delta_j} = (u_j^{mod} - u_j, v_j^{mod} - v_j)^t$. These vectors $\vec{\delta_j}$ constitute one criterion. Another criterion is the mean module of vectors $\vec{\delta_j}$.

Results

The following table reports the mean error module for eight different calibration positions, using five different methods:

	pos 1	pos 2	pos 3	pos 4	pos 5	pos 6	pos 7	pos 8
method 1	0.87	5.48	48.26	10.86	57.59	5.52	2.60	13.78
method 2	0.82	0.93	4.83	1.38	6.16	0.98	0.84	1.62
method 3	1.82	1.22	4.81	1.15	5.77	1.33	2.22	2.81
method 4	0.88	0.81	2.86	1.01	3.76	0.93	0.87	1.14
method 5	0.85	0.79	1.11	0.89	1.38	0.87	0.82	0.85

Method 1: This is the method of Faugeras and Toscani applied to each of the eight positions. Intrinsic parameters of position 1 are chosen as unique values for all positions and we associate them with the extrinsic parameters corresponding to each position. We see that the mean error module for position 1 is quite good: this shows that associating intrinsic and extrinsic parameters computed in the same position yields an excellent perspective transformation matrix. This is not the case when intrinsic and extrinsic parameters from different positions are associated.

Method 2: Intrinsic parameters computed in position 1 (also with Faugeras and Toscani's method) are again chosen, and optimal *extrinsic* parameters are computed with these selected values, using the method presented in section 4.2. The errors are dramatically reduced. This shows that, for a given set of intrinsic parameters, even if they are noisy, it is possible to find compatible extrinsic parameters which yield a positive result, i.e., give a relatively accurate perspective transformation matrix. The associated extrinsic parameters are also somewhat inaccurate.

Method 3: Optimal intrinsic parameters are computed taking into account all eight positions with the method presented in section 4.1. The extrinsic parameters are simply computed using equations (5), (8) and (9). With reference to method 2, the errors are reduced to a similar level.

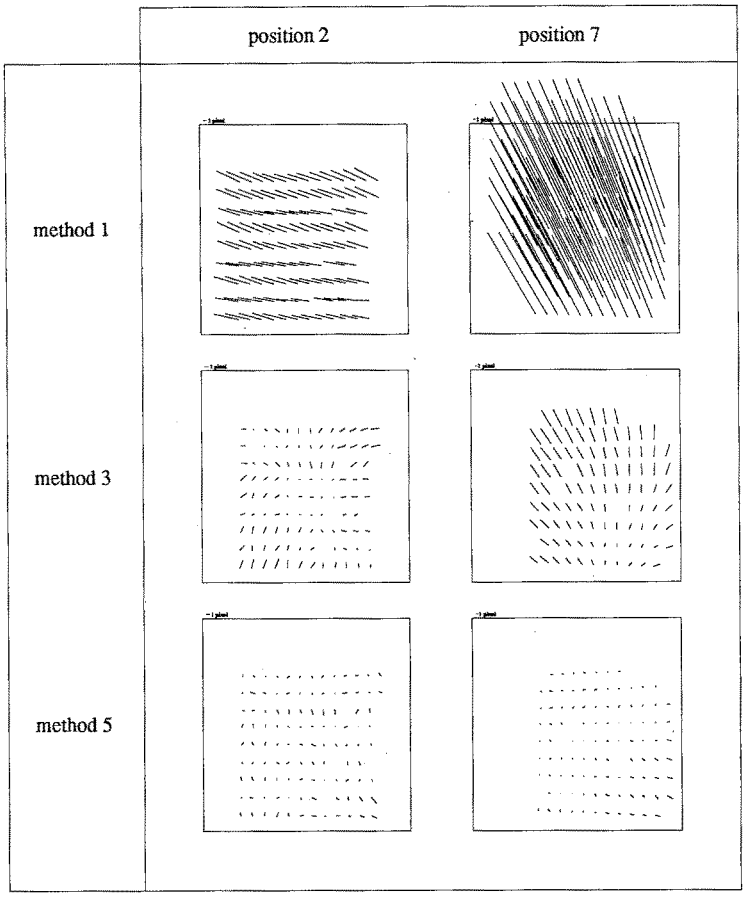

Figure 4: Error vectors for various positions and methods

Here, intrinsic and extrinsic parameters are less corrupted than for the first two methods. Notice also that the mean error in position 1 is worse here than the two methods above. The new values for intrinsic parameters are globally optimal for all the positions but not for this specific position itself.

Method 4: Intrinsic parameters are the same as in method 3 and extrinsic *translational* parameters are optimized. *Rotational* parameters are computed using equations (5) and (8).

Method 5: We use the complete optimal method that has been presented in this paper. This produces the best results. Compared with method 1, we have a gain factor varying between 1.02 (pos 1) and 43.47 (pos 3). The average gain factor is 16.4 .

A qualitative comparison of the various methods can be done by observing the error vectors. These are shown in figure 4 for positions 2 and 7, and methods 1, 3 and 5 respectively. The error vectors are magnified by a factor of 10 for position 2 and a factor of 5 for position 7.

The CPU time needed for a whole camera calibration process (computation of the perspective transformation matrices and optimization of parameters) is 13 seconds on a SUN 4/110 workstation

for eight positions and 100 calibration points per position. This time does not include the low level image processing and the computation of 2-D calibration points.

Acknowledgements

This work has been partially supported by Esprit project P940. The authors would like to thank Philippe Bobet from LIFIA and the members of ITMI's research department for their contribution to the implementation of this work, and Radu Horaud from LIFIA for his insightful comments.

References

[1] O. D. Faugeras and G. Toscani. Camera Calibration for 3D Computer Vision. In *Proc of Int. Workshop on Machine Vision and Machine Intelligence*, Tokyo, Japan, February 1987.

[2] S. Ganapathy. Decomposition of transformation matrices for robot vision. *Pattern Recognition Letters*, 2:401–412, December 1984.

[3] R. P. Paul. *Robot Manipulators: Mathematics, Programming, and Control.* The MIT Press, 1981.

[4] E. Previn and J. A. Webb. *Quaternions in computer vision and robotics.* Technical Report CS-82-150, Carnegie-Mellon University, 1982.

[5] Y. C. Shiu and S. Ahmad. Finding the Mounting Position of a Sensor by Solving a Homogeneous Transform Equation of the Form AX=XB. In *IEEE Conference on Robotics and Automation*, pages 1666–1671, Raleygh, North Carolina, USA, April 1987.

[6] T. M. Strat. *Recovering the Camera Parameters from a Transformation Matrix*, pages 93–100. Morgan Kaufmann Publishers, Inc, 1987.

[7] R. Y. Tsai. A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323–344, August 1987.

[8] R. Y. Tsai. Synopsis of Recent Progress on Camera Calibration for 3D Machine Vision. In Oussama Khatib, John J. Craig, and Tomás Lozano-Pérez, editors, *The Robotics Review*, pages 147–159, The MIT Press, 1989.

[9] R. Y. Tsai and R. K. Lenz. Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 10(5):713–720, september 1988.

[10] R.Y. Tsai and R.K. Lenz. Real Time Versatile Robotics Hand/Eye Calibration using 3D Machine Vision. In *IEEE International Conference on Robotics and Automation*, Philadelphia, Penn, USA, April 1988.

A Numerical estimation of errors in the computation of camera parameters

In this appendix, Δu_0, Δv_0, $\Delta \alpha_u$, $\Delta \alpha_v$ denote the maximal errors on u_0, v_0, α_u and α_v respectively. $\Delta \vec{m}_i$ denotes the maximal error vector on $\vec{m}_i$, *i.e.* the vector $(\Delta m_{i1}, \Delta m_{i2}, \Delta m_{i3})^t$. $|\vec{m}_i|$ denotes the vector $(|m_{i1}|, |m_{i2}|, |m_{i3}|)^t$. As for any two vectors $\vec{u}$ and $\vec{v}$, the following relationship holds:

$$\Delta(\vec{u}.\vec{v}) = \Delta\vec{u}.|\vec{v}| + \vec{u}.|\Delta\vec{v}|$$

we get the expression of Δu_0 and Δv_0 from equations (6):

$$\begin{cases} \Delta u_0 &= \Delta\vec{m}_1.|\vec{m}_3| + |\vec{m}_1|.\Delta\vec{m}_3 \\ \Delta v_0 &= \Delta\vec{m}_2.|\vec{m}_3| + |\vec{m}_2|.\Delta\vec{m}_3 \end{cases} \tag{17}$$

We have also:

$$\begin{cases} \Delta(\|\vec{u}\|) &= \frac{1}{\|\vec{u}\|}(|\vec{u}|.\Delta\vec{u}) \\ \Delta(\vec{u}\times\vec{v}) &= \Delta\vec{u}\otimes|\vec{v}| + |\vec{u}|\otimes\Delta\vec{v} \end{cases}$$

with $\otimes$ being the operation:

$$\vec{u}\otimes\vec{v} = \begin{pmatrix} u_y v_z + u_z v_y \\ u_z v_x + u_x v_z \\ u_x v_y + u_y v_x \end{pmatrix}$$

This leads to the expression of $\Delta\alpha_u$ and $\Delta\alpha_v$:

$$\begin{cases} \Delta\alpha_u &= \frac{1}{\alpha_u}[|\vec{m}_1\times\vec{m}_3|.(\Delta\vec{m}_1\otimes|\vec{m}_3| + |\vec{m}_1|\otimes\Delta\vec{m}_3)] \\ \Delta\alpha_v &= \frac{1}{\alpha_v}[|\vec{m}_2\times\vec{m}_3|.(\Delta\vec{m}_2\otimes|\vec{m}_3| + |\vec{m}_2|\otimes\Delta\vec{m}_3)] \end{cases}$$

These formulae allow one to calculate the maximal errors on the intrinsic parameters when the errors Δm_{ij} are known. However, due to the algorithm used to compute the perspective transformation matrix, it is difficult to calculate Δm_{ij} as a function of the errors on the 3-D points X_j and their projections p_j. Consequently, to estimate the errors Δm_{ij}, we did a Monte-Carlo simulation. Random errors on 2-D data whose standard deviation is 1.0 pixel leads to the following results:

calibration type	Δu_0	Δv_0	$\Delta\alpha_u$	$\Delta\alpha_v$
4 planes	7.42	12.54	6.42	11.99
2 upper planes	8.80	19.62	16.03	26.41
2 lower planes	17.81	25.02	24.94	40.19

These results show that the noise on 2-D data is amplified. With real data, we found differences of about 20.0 pixels on u_0 and v_0 for different camera positions.

B Finding the optimal rotation

We saw that the optimal rotation ρ is the rotation that minimizes the criterion:

$$C_\rho = \sum_{j=1}^{M} \|\rho S_j - Q_j\|^2 \tag{18}$$

Let q be the quaternion associated with the rotation ρ (notations were introduced in section 2):

$$q = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}(n_x i + n_y j + n_z k)$$

It is possible also to consider a vector or a point as S_j as a quaternion (see [4] for more details):

$$S_j = S_{jx} i + S_{jy} j + S_{jz} k$$

Our demonstration is based on the three following properties:

Property 1 *The quaternion q associated with the rotation ρ is a unit quaternion. This means that:*

$$\|q\|^2 = q * \bar{q} = 1$$

where $*$ denotes the product of quaternions and $\bar{q}$ the conjugate of q:

$$\bar{q} = \cos\frac{\theta}{2} - \sin\frac{\theta}{2}(n_x i + n_y j + n_z k)$$

This implies that q^{-1} is equal to $\bar{q}$.
For the proof, calculate $\|q\|^2$ using the fact that $(n_x, n_y, n_z)^t$ is a unit vector and $\cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2}$ is equal to 1.
Property 2 *Representing both point S_j and rotation ρ with quaternions, the result of applying ρ to S_j is represented by the quaternion:*

$$\rho S_j = q * S_j * q^{-1}$$

For the proof, see [4].
Property 3 *For any quaternion p and any* unit *quaternion q, we have:*

$$\|p * q\|^2 = \|p * \bar{q}\|^2 = \|p\|^2$$

Proof:

$$\|p * q\|^2 = (p * q) * \overline{(p * q)} = p * q * \bar{q} * \bar{p} = p * \bar{p} = \|p\|^2 \square$$

Using successively properties 2, 1, and 3, (18) leads to:

$$C_\rho = \sum_{j=1}^{M} \|q * S_j * \bar{q} - Q_j\|^2 = \sum_{j=1}^{M} \|(q * S_j - Q_j * q) * \bar{q}\|^2 = \sum_{j=1}^{M} \|(q * S_j - Q_j * q)\|^2$$

$(q * S_j - Q_j * q)$ is a quaternion which is a linear function of q. More precisely, this quaternion is equal to $B_j q$ where B_j is the matrix introduced in equation (15). This leads to:

$$C_\rho = \sum_{j=1}^{M} (B_j q)^t (B_j q) = q^t (\sum_{j=1}^{M} B_j)^t (\sum_{j=1}^{M} B_j) q = q^t B q$$

$q^t B q$ is minimal for q being equal to the unit eigenvector associated with the smallest eigenvalue of matrix B. Notice that B, as a 4×4 symmetric positive matrix has four positive eigenvalues.