Computer Vision
A Modern Approach

David A. Forsyth
University of California at Berkeley

Jean Ponce
University of Illinois at Urbana-Champaign

An Alan R. Apt Book

Prentice Hall
Upper Saddle River, New Jersey 07458

3

Geometric Camera Calibration

This chapter addresses the problem of estimating the intrinsic and extrinsic parameters of a camera, a process known as *geometric camera calibration*. We assume throughout that the camera observes a set of features such as points or lines with known positions in some fixed world coordinate system (Figure 3.1): In this context, camera calibration can be modeled as an optimization process, where the discrepancy between the observed image features and their theoretical positions (as predicted by the perspective projection equations derived in chapter 2) is minimized with respect to the camera's intrinsic and extrinsic parameters.

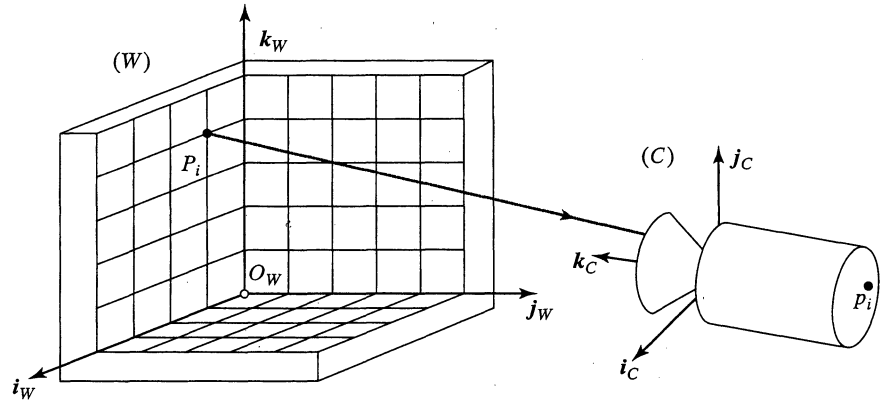


Figure 3.1 Camera calibration setup: In this example, the calibration rig is formed by three grids drawn in orthogonal planes. Other patterns could be used as well, and they may involve lines or other geometric figures.

We start with an overview of *least-squares* techniques aimed at solving this type of optimization problems before presenting several linear and nonlinear approaches to calibration. Once a camera has been calibrated, it is possible to associate with any image point a well-defined ray passing through this point and the camera's optical center as well as perform accurate three-dimensional measurements from digitized pictures. An application to mobile robot localization is briefly discussed at the end of the chapter.

3.1 LEAST-SQUARES PARAMETER ESTIMATION

As already mentioned, calibrating a camera amounts to estimating the intrinsic and extrinsic parameters that minimize the mean-squared deviation from predicted to observed image features. This section introduces a class of optimization techniques, known as least-squares methods, for solving this kind of problem. They prove useful on several other occasions in the rest of this book.

3.1.1 Linear Least-Squares Methods

Let us first consider a system of p linear equations in q unknowns:

$$\begin{cases} u_{11}x_1 + u_{12}x_2 + \cdots + u_{1q}x_q = y_1 \\ u_{21}x_1 + u_{22}x_2 + \cdots + u_{2q}x_q = y_2 \\ \cdots \\ u_{p1}x_1 + u_{p2}x_2 + \cdots + u_{pq}x_q = y_p \end{cases} \iff \mathcal{U}\boldsymbol{x} = \boldsymbol{y}. \tag{3.1}$$

In this equation,

$$\mathcal{U} = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1q} \\ u_{21} & u_{22} & \cdots & u_{2q} \\ \cdots & \cdots & \cdots & \cdots \\ u_{p1} & u_{p2} & \cdots & u_{pq} \end{pmatrix}, \quad \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_q \end{pmatrix} \quad \text{and} \quad \boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_p \end{pmatrix}.$$

We know from linear algebra that (in general)

- when $p < q$, the set of solutions to this equation forms a $(q - p)$-dimensional vector subspace of $\mathbb{R}^q$;
- when $p = q$, there is a unique solution;
- when $p > q$, there is no solution.

This statement is true when the *rank* (i.e., the maximum number of independent rows or columns) of $\mathcal{U}$ is maximal—that is, equal to $\min(p, q)$ (this is what we mean by *in general*). When the rank is smaller than $\min(p, q)$, the existence of solutions to Eq. (3.1) depends on the value of $\boldsymbol{y}$ and whether it belongs to the *range* of $\mathcal{U}$ (i.e., the subspace of $\mathbb{R}^p$ spanned by its columns).

Normal Equations and the Pseudoinverse The rest of this section focuses on the overconstrained case $p > q$ and assumes that $\mathcal{U}$ has maximal rank q. Since there is no exact solution in this case, we content ourselves with finding the vector $\boldsymbol{x}$ that minimizes the error measure

$$E \stackrel{\text{def}}{=} \sum_{i=1}^{p} (u_{i1}x_1 + \cdots + u_{iq}x_q - y_i)^2 = |\mathcal{U}\boldsymbol{x} - \boldsymbol{y}|^2.$$

E is proportional to the mean-squared error associated with the equations, hence the name of least-squares methods given to techniques for minimizing it.

Now, we can write $E = \boldsymbol{e} \cdot \boldsymbol{e}$, where $\boldsymbol{e} \stackrel{\text{def}}{=} \mathcal{U}\boldsymbol{x} - \boldsymbol{y}$. To find the vector $\boldsymbol{x}$ minimizing E, we write that the derivatives of this error measure with respect to the coordinates x_i $(i = 1, \ldots, q)$ of $\boldsymbol{x}$ must be zero—that is,

$$\frac{\partial E}{\partial x_i} = 2\frac{\partial \boldsymbol{e}}{\partial x_i} \cdot \boldsymbol{e} = 0 \quad \text{for} \quad i = 1, \ldots, q.$$

But if the columns of $\mathcal{U}$ are the vectors $\boldsymbol{c}_j = (u_{1j}, \ldots, u_{mj})^T$ $(j = 1, \ldots, q)$, we have

$$\frac{\partial \boldsymbol{e}}{\partial x_i} = \frac{\partial}{\partial x_i}\left[\begin{pmatrix} \boldsymbol{c}_1 & \cdots & \boldsymbol{c}_q \end{pmatrix}\begin{pmatrix} x_1 \\ \cdots \\ x_q \end{pmatrix} - \boldsymbol{y}\right] = \frac{\partial}{\partial x_i}(x_1\boldsymbol{c}_1 + \cdots + x_q\boldsymbol{c}_q - \boldsymbol{y}) = \boldsymbol{c}_i.$$

In particular, writing that $\partial E/\partial x_i = 0$ implies that $\boldsymbol{c}_i^T(\mathcal{U}\boldsymbol{x} - \boldsymbol{y}) = 0$, and stacking the constraints associated with the q coordinates of $\boldsymbol{x}$ yields the *normal equations* associated with our least-squares problem—that is,

$$\boldsymbol{0} = \begin{pmatrix} \boldsymbol{c}_1^T \\ \cdots \\ \boldsymbol{c}_q^T \end{pmatrix}(\mathcal{U}\boldsymbol{x} - \boldsymbol{y}) = \mathcal{U}^T(\mathcal{U}\boldsymbol{x} - \boldsymbol{y}) \Longleftrightarrow \mathcal{U}^T\mathcal{U}\boldsymbol{x} = \mathcal{U}^T\boldsymbol{y}.$$

When $\mathcal{U}$ has maximal rank q, the matrix $\mathcal{U}^T\mathcal{U}$ is easily shown to be invertible, and the solution of the normal equations is $\boldsymbol{x} = \mathcal{U}^\dagger\boldsymbol{y}$ with $\mathcal{U}^\dagger \stackrel{\text{def}}{=} [(\mathcal{U}^T\mathcal{U})^{-1}\mathcal{U}^T]$. The $q \times q$ matrix $\mathcal{U}^\dagger$ is called the *pseudoinverse* of $\mathcal{U}$. It coincides with $\mathcal{U}^{-1}$ when the matrix $\mathcal{U}$ is square and nonsingular. Linear least-squares problems can be solved without explicitly computing the pseudoinverse, using, for example, QR decomposition or singular value decomposition (more on the latter in chapter 12), which are known to be better behaved numerically.

Homogeneous Systems and Eigenvalue Problems Let us now consider a variant of our original problem, where we have again a system of p linear equations in q unknowns, but the vector $\boldsymbol{y}$ is zero—that is,

$$\left\{\begin{array}{l} u_{11}x_1 + u_{12}x_2 + \cdots + u_{1q}x_q = 0 \\ u_{21}x_1 + u_{22}x_2 + \cdots + u_{2q}x_q = 0 \\ \cdots \\ u_{p1}x_1 + u_{p2}x_2 + \cdots + u_{pq}x_q = 0 \end{array}\right. \Longleftrightarrow \mathcal{U}\boldsymbol{x} = \boldsymbol{0}. \tag{3.2}$$

This is a *homogeneous* equation in $\boldsymbol{x}$ (i.e., if $\boldsymbol{x}$ is a solution, so is $\lambda\boldsymbol{x}$ for any $\lambda \neq 0$). When $p = q$ and the matrix $\mathcal{U}$ is nonsingular, Eq. (3.2) admits as a unique solution $\boldsymbol{x} = \boldsymbol{0}$. Conversely, when $p \geq q$, nontrivial (i.e., nonzero) solutions may only exist when $\mathcal{U}$ is singular with rank strictly smaller than q. In this context, minimizing $E = |\mathcal{U}\boldsymbol{x}|^2$ only makes sense when some additional constraint is imposed on $\boldsymbol{x}$ since the value $\boldsymbol{x} = \boldsymbol{0}$ yields the zero global minimum of E. By homogeneity, we have $E(\lambda\boldsymbol{x}) = \lambda^2 E(\boldsymbol{x})$, and it is reasonable to choose the constraint $|\boldsymbol{x}|^2 = 1$, which avoids the trivial solution and forces the uniqueness of the result.

The error E can be rewritten as $|\mathcal{U}\boldsymbol{x}|^2 = \boldsymbol{x}^T(\mathcal{U}^T\mathcal{U})\boldsymbol{x}$. The $q \times q$ matrix $\mathcal{U}^T\mathcal{U}$ is by construction symmetric positive semidefinite (i.e., its eigenvalues are all positive or zero), and it can be diagonalized in an orthonormal basis of eigenvectors $\boldsymbol{e}_i$ $(i = 1, \ldots, q)$ associated with the eigenvalues $0 \leq \lambda_1 \leq \cdots \leq \lambda_q$. Thus we can write any unit vector as $\boldsymbol{x} = \mu_1\boldsymbol{e}_1 + \cdots + \mu_q\boldsymbol{e}_q$ with $\mu_1^2 + \cdots + \mu_q^2 = 1$. In particular,

$$E(\boldsymbol{x}) - E(\boldsymbol{e}_1) = \boldsymbol{x}^T(\mathcal{U}^T\mathcal{U})\boldsymbol{x} - \boldsymbol{e}_1^T(\mathcal{U}^T\mathcal{U})\boldsymbol{e}_1 = \lambda_1^2\mu_1^2 + \cdots + \lambda_q^2\mu_q^2 - \lambda_1^2$$
$$\geq \lambda_1^2(\mu_1^2 + \cdots + \mu_q^2 - 1) = 0.$$

It follows that the unit vector $\boldsymbol{x}$ minimizing E is the eigenvector $\boldsymbol{e}_1$ associated with the minimum eigenvalue of $\mathcal{U}^T\mathcal{U}$, and the corresponding minimum value of E is λ_1^2. Various methods are available for computing the eigenvectors and eigenvalues of a symmetric matrix, including Jacobi transformations and reduction to tridiagonal form followed by QR decomposition. Singular value decomposition can also be used to compute the eigenvectors and eigenvalues without actually constructing the matrix $\mathcal{U}^T\mathcal{U}$.

Before illustrating the use of homogeneous linear least-squares techniques with an example, let us pause for a minute to consider the slightly more general problem of minimizing $|\mathcal{U}\boldsymbol{x}|^2$ under the constraint $|\mathcal{V}\boldsymbol{x}|^2 = 1$, where $\mathcal{V}$ is an $r \times q$ matrix (this reduces to homogeneous linear least squares when $\mathcal{V} = \mathrm{Id}$). A vector $\boldsymbol{x}$ and a scalar λ such that

$$\mathcal{U}^T\mathcal{U}\boldsymbol{x} = \lambda\mathcal{V}^T\mathcal{V}\boldsymbol{x}$$

are called a *generalized eigenvector* and the corresponding *generalized eigenvalue* of the $q \times q$ symmetric matrices $\mathcal{U}^T\mathcal{U}$ and $\mathcal{V}^T\mathcal{V}$. As shown in the exercises, the solution of our constrained optimization problem is precisely the unit generalized eigenvector associated with the minimum generalized eigenvalue (which is in this case guaranteed to be positive or zero by construction). As before, effective methods for computing the generalized eigenvectors and eigenvalues of a pair of symmetric matrices are available.

Example 3.1 Fitting a line to points in a plane.

Consider n points p_i $(i = 1, \ldots, n)$ in a plane, with coordinates (x_i, y_i) in some fixed coordinate system (Figure 3.2). What is the straight line that best fits these points? To answer this question, we must first quantify how well a line δ fits a set of points or, equivalently, define some error function E measuring the discrepancy between this line and the points. The best-fitting line can then be found by minimizing E.

A reasonable choice for the error function is the mean-squared distance between the points and the line (Figure 3.2). We saw in chapter 2 that the equation of a line with unit normal $\boldsymbol{n} = (a, b)^T$ lying at a distance d from the origin is $ax + by = d$. It is in fact easy to show that the perpendicular distance between a point with coordinates $(x, y)^T$ and this line is $|ax + by - d|$. We can therefore

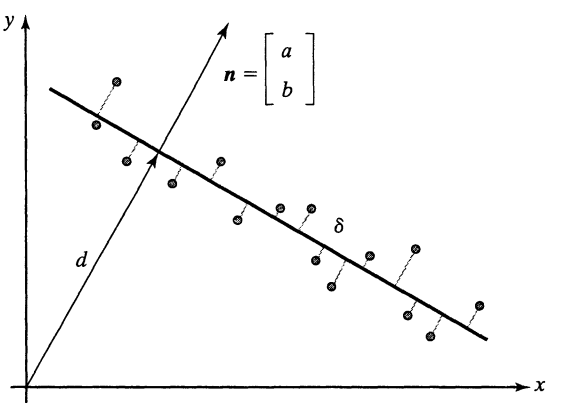


Figure 3.2 The line that best fits n points in the plane can be defined as the line δ that minimizes the mean-squared perpendicular distance to these points (i.e., in this diagram, the mean-squared length of the short parallel line segments joining δ to the points).

use

$$E(a, b, d) = \sum_{i=1}^{n} (ax_i + by_i - d)^2$$

as our error measure, and the line-fitting problem reduces to the minimization of E with respect to a, b, and d under the constraint $a^2 + b^2 = 1$. Differentiating E with respect to d shows that, at a minimum of this function, we must have $0 = \partial E/\partial d = -2\sum_{i=1}^{n}(ax_i + by_i - d)$, thus

$$d = a\bar{x} + b\bar{y}, \quad \text{where} \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{and} \quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i, \tag{3.3}$$

and the two scalars $\bar{x}$ and $\bar{y}$ are simply the coordinates of the center of mass of the input points. Substituting this expression for d in the definition of E yields

$$E = \sum_{i=1}^{n} [a(x_i - \bar{x}) + b(y_i - \bar{y})]^2 = |\mathcal{U}\boldsymbol{n}|^2 \quad \text{where} \quad \mathcal{U} = \begin{pmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \cdots & \cdots \\ x_n - \bar{x} & y_n - \bar{y} \end{pmatrix},$$

and our original problem finally reduces to minimizing $|\mathcal{U}\boldsymbol{n}|^2$ with respect to $\boldsymbol{n}$ under the constraint $|\boldsymbol{n}|^2 = 1$. We recognize a homogeneous linear least-squares problem, whose solution is the unit eigenvector associated with the minimum eigenvalue of the 2×2 matrix $\mathcal{U}^T\mathcal{U}$. Once a and b have been computed, the value of d is immediately obtained from Eq. (3.3). Note that $\mathcal{U}^T\mathcal{U}$ is easily shown to be equal to

$$\begin{pmatrix} \sum_{i=1}^{n} x_i^2 - n\bar{x}^2 & \sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y} \\ \sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y} & \sum_{i=1}^{n} y_i^2 - n\bar{y}^2 \end{pmatrix},$$

that is, the matrix of second moments of inertia of the points p_i. In fact, the line best fitting these points in the sense defined in this section is simply their axis of least inertia as defined in elementary mechanics.

3.1.2 Nonlinear Least-Squares Methods

Let us now consider a general system of p equations in q unknowns:

$$\begin{cases} f_1(x_1, x_2, \ldots, x_q) = 0 \\ f_2(x_1, x_2, \ldots, x_q) = 0 \\ \cdots \\ f_p(x_1, x_2, \ldots, x_q) = 0 \end{cases} \iff \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}. \tag{3.4}$$

Here, f_i denotes, for $i = 1, \ldots, p$, a (possibly nonlinear) differentiable function from $\mathbb{R}^p$ to $\mathbb{R}$, and we take $\boldsymbol{f} = (f_1, \ldots, f_p)^T$ and $\boldsymbol{x} = (x_1, \ldots, x_q)^T$. In general,

- when $p < q$, the solutions form a $(q - p)$-dimensional *subset* of $\mathbb{R}^q$;
- when $p = q$, there is a *finite set* of solutions;
- when $p > q$, there is no solution.

Let us emphasize the main differences with the linear case: In general, the dimension of the solution set is still $q - p$ in the underconstrained case, but this set does not form a vector space anymore. Its structure depends on the nature of the functions f_i. Likewise, there is usually a finite number of solutions instead of a unique one in the case $p = q$. A precise definition of

the general conditions that a family of functions f_i $(i = 1, \ldots, p)$ has to satisfy for the prior statement to be true is unfortunately beyond the scope of this book.

There is no general method for finding all the solutions of Eq. (3.4) when $p = q$ or for finding the global minimum of the least-squares error

$$E(\boldsymbol{x}) \stackrel{\text{def}}{=} |\boldsymbol{f}(\boldsymbol{x})|^2 = \sum_{i=1}^{p} f_i^2(\boldsymbol{x})$$

when $p > q$. Instead, we present next a number of iterative methods that linearize the problem in hope of finding at least one suitable solution. They all rely on a first-order Taylor expansion of the functions f_i in the neighborhood of a point $\boldsymbol{x}$:

$$f_i(\boldsymbol{x} + \delta\boldsymbol{x}) = f_i(\boldsymbol{x}) + \delta x_1 \frac{\partial f_i}{\partial x_1}(\boldsymbol{x}) + \cdots + \delta x_q \frac{\partial f_i}{\partial x_q}(\boldsymbol{x}) + O(|\delta\boldsymbol{x}|^2) \approx f_i(\boldsymbol{x}) + \nabla f_i(\boldsymbol{x}) \cdot \delta\boldsymbol{x}.$$

Here, $\nabla f_i(\boldsymbol{x}) = (\partial f_i/\partial x_1, \ldots, \partial f_i/\partial x_q)^T$ is the *gradient* of f_i at the point $\boldsymbol{x}$, and we have neglected the second-order term $O(|\delta\boldsymbol{x}|^2)$. It follows immediately that

$$\boldsymbol{f}(\boldsymbol{x} + \delta\boldsymbol{x}) \approx \boldsymbol{f}(\boldsymbol{x}) + \mathcal{J}_{\boldsymbol{f}}(\boldsymbol{x})\delta\boldsymbol{x}, \tag{3.5}$$

where $\mathcal{J}_{\boldsymbol{f}}(\boldsymbol{x})$ is the *Jacobian* of $\boldsymbol{f}$—that is, the $p \times q$ matrix

$$\mathcal{J}_{\boldsymbol{f}}(\boldsymbol{x}) \stackrel{\text{def}}{=} \begin{pmatrix} \nabla f_1^T(\boldsymbol{x}) \\ \cdots \\ \nabla f_p^T(\boldsymbol{x}) \end{pmatrix} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial f_1}{\partial x_q}(\boldsymbol{x}) \\ \cdots & \cdots & \cdots \\ \frac{\partial f_p}{\partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial f_p}{\partial x_q}(\boldsymbol{x}) \end{pmatrix}.$$

Newton's Method: Square Systems of Nonlinear Equations As mentioned earlier, Eq. (3.4) admits (in general) a finite number of solutions when $p = q$. Although there is no general method for finding all of these solutions when $\boldsymbol{f}$ is arbitrary, Eq. (3.5) can be used as the basis for a simple iterative algorithm for finding one of these solutions: Given some current estimate $\boldsymbol{x}$ of the solution, the idea is to compute a perturbation $\delta\boldsymbol{x}$ of this estimate such that $\boldsymbol{f}(\boldsymbol{x} + \delta\boldsymbol{x}) \approx \boldsymbol{0}$, or, according to Eq. (3.5),

$$\mathcal{J}_{\boldsymbol{f}}(\boldsymbol{x})\delta\boldsymbol{x} = -\boldsymbol{f}(\boldsymbol{x}).$$

When the Jacobian is nonsingular, $\delta\boldsymbol{x}$ is easily found as the solution of this $q \times q$ system of linear equations, and the process is repeated until convergence.

Newton's method converges rapidly once close to a solution: It has a *quadratic convergence rate* (i.e., the error at step $k + 1$ is proportional to the square of the error at step k). When started far from a solution, Newton's method as presented here may be unreliable. Various strategies can be used to improve its robustness, but their discussion is beyond the scope of this book.

Newton's Method: Overconstrained Systems of Nonlinear Equations When p is greater than q, we seek a local minimum of the least-squares error E. Newton's method can be adapted to this case by noting that such a minimum is a zero of the error's gradient. More precisely, we introduce $\boldsymbol{F}(\boldsymbol{x}) = \frac{1}{2}\nabla E(\boldsymbol{x})$ and use Newton's method to find the desired minimum as a solution of the $q \times q$ system of nonlinear equations $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$. Differentiating E shows that

$$\boldsymbol{F}(\boldsymbol{x}) = \mathcal{J}_{\boldsymbol{f}}^T(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}), \tag{3.6}$$

and differentiating this expression shows in turn that the Jacobian of $\boldsymbol{F}$ is

$$\mathcal{J}_F(\boldsymbol{x}) = \mathcal{J}_f^T(\boldsymbol{x})\mathcal{J}_f(\boldsymbol{x}) + \sum_{i=1}^{p} f_i(\boldsymbol{x})\mathcal{H}_{f_i}(\boldsymbol{x}). \tag{3.7}$$

In this equation, $\mathcal{H}_{f_i}(\boldsymbol{x})$ denotes the *Hessian* of f_i—that is, the $q \times q$ matrix of second derivatives

$$\mathcal{H}_{f_i}(\boldsymbol{x}) \stackrel{\text{def}}{=} \begin{pmatrix} \frac{\partial^2 f_i}{\partial x_1^2}(\boldsymbol{x}) & \cdots & \frac{\partial^2 f_i}{\partial x_1 x_q}(\boldsymbol{x}) \\ \cdots & \cdots & \cdots \\ \frac{\partial^2 f_i}{\partial x_1 x_q}(\boldsymbol{x}) & \cdots & \frac{\partial^2 f_i}{\partial x_q^2}(\boldsymbol{x}) \end{pmatrix}.$$

The term $\delta\boldsymbol{x}$ in Newton's method satisfies $\mathcal{J}_F(\boldsymbol{x})\delta\boldsymbol{x} = -\boldsymbol{F}(\boldsymbol{x})$. Equivalently, combining Eqs. (3.6) and (3.7) shows that $\delta\boldsymbol{x}$ is the solution of

$$\left[\mathcal{J}_f^T(\boldsymbol{x})\mathcal{J}_f(\boldsymbol{x}) + \sum_{i=1}^{p} f_i(\boldsymbol{x})\mathcal{H}_{f_i}(\boldsymbol{x})\right]\delta\boldsymbol{x} = -\mathcal{J}_f^T(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}). \tag{3.8}$$

The Gauss–Newton and Levenberg–Marquardt Algorithms Newton's method requires computing the Hessians of the functions f_i, which may be difficult and/or expensive. We discuss here two other approaches to nonlinear least-squares that do not involve the Hessians. Let us first consider the Gauss–Newton algorithm: In this approach, we use again a first-order Taylor expansion of $\boldsymbol{f}$ to minimize E, but this time we seek the value of $\delta\boldsymbol{x}$ that minimizes $E(\boldsymbol{x} + \delta\boldsymbol{x})$ for a given value of $\boldsymbol{x}$. Substituting Eq. (3.5) into Eq. (3.4) yields

$$E(\boldsymbol{x} + \delta\boldsymbol{x}) = |\boldsymbol{f}(\boldsymbol{x} + \delta\boldsymbol{x})|^2 \approx |\boldsymbol{f}(\boldsymbol{x}) + \mathcal{J}_f(\boldsymbol{x})\delta\boldsymbol{x}|^2.$$

At this point, we are back in the linear least-squares setting, and the adjustment $\delta\boldsymbol{x}$ can be computed as the solution of $\mathcal{J}_f^\dagger(\boldsymbol{x})\delta\boldsymbol{x} = -\boldsymbol{f}(\boldsymbol{x})$ or, equivalently, according to the definition of the pseudoinverse,

$$\mathcal{J}_f^T(\boldsymbol{x})\mathcal{J}_f(\boldsymbol{x})\delta\boldsymbol{x} = -\mathcal{J}_f^T(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}). \tag{3.9}$$

Comparing Eqs. (3.8) and (3.9), we see that the Gauss–Newton algorithm can be thought of as an approximation of Newton's method where the term involving the Hessians $\mathcal{H}_{f_i}$ has been neglected. This is justified when the values of the functions f_i at a solution (the *residuals*) are small since the matrices $\mathcal{H}_{f_i}$ are multiplied by these residuals in Eq. (3.8). In this case, the performance of the Gauss–Newton algorithm is comparable to that of Newton's method, with (nearly) quadratic convergence close to a solution. When the residuals at the solution are too large, however, it may converge slowly or not at all.

When Eq. (3.9) is replaced by

$$[\mathcal{J}_f^T(\boldsymbol{x})\mathcal{J}_f(\boldsymbol{x}) + \mu\mathrm{Id}]\delta\boldsymbol{x} = -\mathcal{J}_f^T(\boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}), \tag{3.10}$$

where the parameter μ is allowed to vary at each iteration, we obtain the Levenberg–Marquardt algorithm, popular in computer vision circles. This is another variant of Newton's method where the term involving the Hessians is this time approximated by a multiple of the identity matrix. The Levenberg–Marqardt algorithm has convergence properties comparable to its Gauss–Newton cousin, but it is more robust: For example, unlike that algorithm, it can be used when the Jacobian $\mathcal{J}_f$ does not have maximal rank and its pseudoinverse does not exist.

3.2 A LINEAR APPROACH TO CAMERA CALIBRATION

It is now time to go back to geometric camera calibration. We assume in this section that a calibration rig is observed by a camera and that the image positions (u_i, v_i) of n points P_i $(i = 1, \ldots, n)$ with known homogeneous coordinate vectors $\boldsymbol{P}_i$ have been found in a picture of the rig, either automatically or by hand. We decompose the calibration process into (a) the computation of the perspective projection matrix $\mathcal{M}$ associated with the camera in this coordinate system, followed by (b) the estimation of the intrinsic and extrinsic parameters of the camera from this matrix. Degenerate point configurations for which the first step of this process may fail are identified in Section 3.2.3. As shown shortly, writing that the points p_i are the perspective images of the points P_i imposes a set of n linear constraints on the 11 independent coefficients of the corresponding projection matrix. When $n > 11$, these equations generally do not admit a common root, but the techniques introduced in Section 3.1.1 can be used to effectively construct their solution in the least-squares sense.

3.2.1 Estimation of the Projection Matrix

Let us assume that our camera has nonzero skew. According to Theorem 1 from chapter 2, the matrix $\mathcal{M}$ is not singular, but otherwise arbitrary. Clearing the denominators in the perspective projection Eq. (2.16) yields

$$\begin{cases} (\boldsymbol{m}_1 - u_i \boldsymbol{m}_3) \cdot \boldsymbol{P} = 0, \\ (\boldsymbol{m}_2 - v_i \boldsymbol{m}_3) \cdot \boldsymbol{P} = 0. \end{cases}$$

Collecting the constraints associated with our n points yields a system of $2n$ homogeneous linear equations in the twelve coefficients of the matrix $\mathcal{M}$—namely, $\mathcal{P}\boldsymbol{m} = 0$, where

$$\mathcal{P} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{P}_1^T & \boldsymbol{0}^T & -u_1 \boldsymbol{P}_1^T \\ \boldsymbol{0}^T & \boldsymbol{P}_1^T & -v_1 \boldsymbol{P}_1^T \\ \cdots & \cdots & \cdots \\ \boldsymbol{P}_n^T & \boldsymbol{0}^T & -u_n \boldsymbol{P}_n^T \\ \boldsymbol{0}^T & \boldsymbol{P}_n^T & -v_n \boldsymbol{P}_n^T \end{pmatrix} \quad \text{and} \quad \boldsymbol{m} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \\ \boldsymbol{m}_3 \end{pmatrix} = 0.$$

When $n \geq 6$, homogeneous linear least-squares can be used to compute the value of the unit vector $\boldsymbol{m}$ (hence the matrix $\mathcal{M}$) that minimizes $|\mathcal{P}\boldsymbol{m}|^2$ as the solution of an eigenvalue problem.

3.2.2 Estimation of the Intrinsic and Extrinsic Parameters

Once the projection matrix $\mathcal{M}$ has been estimated, its expression in terms of the camera intrinsic and extrinsic parameters (Eq. [2.17] in chapter 2) can be used to recover these parameters as follows: We write as before $\mathcal{M} = (\mathcal{A} \quad \boldsymbol{b})$, with $\boldsymbol{a}_1^T$, $\boldsymbol{a}_2^T$, and $\boldsymbol{a}_3^T$ denoting the rows of $\mathcal{A}$. We obtain

$$\rho(\mathcal{A} \quad \boldsymbol{b}) = \mathcal{K}(\mathcal{R} \quad \boldsymbol{t}) \Longleftrightarrow \rho \begin{pmatrix} \boldsymbol{a}_1^T \\ \boldsymbol{a}_2^T \\ \boldsymbol{a}_3^T \end{pmatrix} = \begin{pmatrix} \alpha \boldsymbol{r}_1^T - \alpha \cot\theta \boldsymbol{r}_2^T + u_0 \boldsymbol{r}_3^T \\ \dfrac{\beta}{\sin\theta} \boldsymbol{r}_2^T + v_0 \boldsymbol{r}_3^T \\ \boldsymbol{r}_3^T \end{pmatrix},$$

where ρ is an unknown scale factor introduced here to account for the fact that the recovered matrix $\mathcal{M}$ has unit Frobenius form since $|\mathcal{M}| = |\boldsymbol{m}| = 1$.

In particular, using the fact that the rows of a rotation matrix have unit length and are perpendicular to each other yields immediately

$$\begin{cases} \rho = \varepsilon/|\boldsymbol{a}_3|, \\ \boldsymbol{r}_3 = \rho \boldsymbol{a}_3, \\ u_0 = \rho^2(\boldsymbol{a}_1 \cdot \boldsymbol{a}_3), \\ v_0 = \rho^2(\boldsymbol{a}_2 \cdot \boldsymbol{a}_3), \end{cases} \quad \text{where} \quad \varepsilon = \mp 1. \tag{3.11}$$

Since θ is always in the neighborhood of $\pi/2$ with a positive sine, we have

$$\begin{cases} \rho^2(\boldsymbol{a}_1 \times \boldsymbol{a}_3) = -\alpha \boldsymbol{r}_2 - \alpha \cot\theta \boldsymbol{r}_1, \\ \rho^2(\boldsymbol{a}_2 \times \boldsymbol{a}_3) = \dfrac{\beta}{\sin\theta}\boldsymbol{r}_1, \end{cases} \quad \text{and} \quad \begin{cases} \rho^2|\boldsymbol{a}_1 \times \boldsymbol{a}_3| = \dfrac{|\alpha|}{\sin\theta}, \\ \rho^2|\boldsymbol{a}_2 \times \boldsymbol{a}_3| = \dfrac{|\beta|}{\sin\theta}. \end{cases} \tag{3.12}$$

Thus,

$$\begin{cases} \cos\theta = -\dfrac{(\boldsymbol{a}_1 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3)}{|\boldsymbol{a}_1 \times \boldsymbol{a}_3||\boldsymbol{a}_2 \times \boldsymbol{a}_3|}, \\ \alpha = \rho^2|\boldsymbol{a}_1 \times \boldsymbol{a}_3| \sin\theta, \\ \beta = \rho^2|\boldsymbol{a}_2 \times \boldsymbol{a}_3| \sin\theta, \end{cases} \tag{3.13}$$

since the sign of the magnification parameters α and β is normally known in advance and can be taken to be positive.

We can now compute $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ from the second part of Eq. (3.12) as

$$\begin{cases} \boldsymbol{r}_1 = \dfrac{\rho^2 \sin\theta}{\beta}(\boldsymbol{a}_2 \times \boldsymbol{a}_3) = \dfrac{1}{|\boldsymbol{a}_2 \times \boldsymbol{a}_3|}(\boldsymbol{a}_2 \times \boldsymbol{a}_3), \\ \boldsymbol{r}_2 = \boldsymbol{r}_3 \times \boldsymbol{r}_1. \end{cases} \tag{3.14}$$

Note that there are two possible choices for the matrix $\mathcal{R}$ depending on the value of ε. The translation parameters can now be recovered by writing $\mathcal{K}\boldsymbol{t} = \rho \boldsymbol{b}$, and hence $\boldsymbol{t} = \rho \mathcal{K}^{-1}\boldsymbol{b}$. In practical situations, the sign of t_z is often known in advance (this corresponds to knowing whether the origin of the world coordinate system is in front or behind the camera), which allows the choice of a unique solution for the calibration parameters.

3.2.3 Degenerate Point Configurations

We now examine the *degenerate configurations* of the points P_i $(i = 1, \ldots, n)$ that may cause the failure of the camera calibration process. We focus on the (ideal) case where the data points $\boldsymbol{p}_i$ $(i = 1, \ldots, n)$ can be measured with zero error, and we identify the *nullspace* of the matrix $\mathcal{P}$ (i.e., the subspace of $\mathbb{R}^{12}$ formed by the vectors $\boldsymbol{l}$ such that $\mathcal{P}\boldsymbol{l} = \boldsymbol{0}$).

Let $\boldsymbol{l}$ be such a vector. Introducing the vectors formed by successive quadruples of its coordinates (i.e., $\boldsymbol{\lambda} = (l_1, l_2, l_3, l_4)^T$, $\boldsymbol{\mu} = (l_5, l_6, l_7, l_8)^T$, and $\boldsymbol{\nu} = (l_9, l_{10}, l_{11}, l_{12})^T$) allows us to write

$$\boldsymbol{0} = \mathcal{P}\boldsymbol{l} = \begin{pmatrix} \boldsymbol{P}_1^T & \boldsymbol{0}^T & -u_1\boldsymbol{P}_1^T \\ \boldsymbol{0}^T & \boldsymbol{P}_1^T & -v_1\boldsymbol{P}_1^T \\ \cdots & \cdots & \cdots \\ \boldsymbol{P}_n^T & \boldsymbol{0}^T & -u_n\boldsymbol{P}_n^T \\ \boldsymbol{0}^T & \boldsymbol{P}_n^T & -v_n\boldsymbol{P}_n^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\nu} \end{pmatrix} = \begin{pmatrix} \boldsymbol{P}_1^T\boldsymbol{\lambda} - u_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \boldsymbol{P}_1^T\boldsymbol{\mu} - v_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \cdots \\ \boldsymbol{P}_n^T\boldsymbol{\lambda} - u_n\boldsymbol{P}_n^T\boldsymbol{\nu} \\ \boldsymbol{P}_n^T\boldsymbol{\mu} - v_n\boldsymbol{P}_n^T\boldsymbol{\nu} \end{pmatrix}. \tag{3.15}$$

Combining Eqs. (2.16) and (3.15) yields

$$\begin{cases} \boldsymbol{P}_i^T \boldsymbol{\lambda} - \dfrac{\boldsymbol{m}_1^T \boldsymbol{P}_i}{\boldsymbol{m}_3^T \boldsymbol{P}_i} \boldsymbol{P}_i^T \boldsymbol{\nu} = 0, \\ \boldsymbol{P}_i^T \boldsymbol{\mu} - \dfrac{\boldsymbol{m}_2^T \boldsymbol{P}_i}{\boldsymbol{m}_3^T \boldsymbol{P}_i} \boldsymbol{P}_i^T \boldsymbol{\nu} = 0, \end{cases} \quad \text{for} \quad i = 1, \ldots, n.$$

After clearing the denominators and rearranging the terms, we finally obtain

$$\begin{cases} \boldsymbol{P}_i^T (\boldsymbol{\lambda} \boldsymbol{m}_3^T - \boldsymbol{m}_1 \boldsymbol{\nu}^T) \boldsymbol{P}_i = 0, \\ \boldsymbol{P}_i^T (\boldsymbol{\mu} \boldsymbol{m}_3^T - \boldsymbol{m}_2 \boldsymbol{\nu}^T) \boldsymbol{P}_i = 0, \end{cases} \quad \text{for} \quad i = 1, \ldots, n. \tag{3.16}$$

As expected, the vector $\boldsymbol{l}$ associated with $\boldsymbol{\lambda} = \boldsymbol{m}_1$, $\boldsymbol{\mu} = \boldsymbol{m}_2$, and $\boldsymbol{\nu} = \boldsymbol{m}_3$ is a solution of these equations. Are there other solutions?

Let us first consider the case where the points P_i ($i = 1, \ldots, n$) all lie in some plane Π—that is, according to Eq. (2.2), $\boldsymbol{\Pi} \cdot \boldsymbol{P}_i = 0$ for some 4-vector $\boldsymbol{\Pi}$. Clearly, choosing $(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu})$ equal to $(\boldsymbol{\Pi}, \boldsymbol{0}, \boldsymbol{0})$, $(\boldsymbol{0}, \boldsymbol{\Pi}, \boldsymbol{0})$, or $(\boldsymbol{0}, \boldsymbol{0}, \boldsymbol{\Pi})$, or any linear combination of these vectors yields a solution of Eq. (3.16). In other words, the nullspace of $\mathcal{P}$ contains the four-dimensional vector space spanned by these vectors and $\boldsymbol{m}$. In practice, this means that the fiducial points P_i should not all lie in the same plane.

In general, for a given nonzero value of the vector $\boldsymbol{l}$, the points P_i that satisfy Eq. (3.16) must lie on the curve where the two quadric surfaces defined by the corresponding equations intersect. A closer look at Eq. (3.16) reveals that the straight line where the planes defined by $\boldsymbol{m}_3 \cdot \boldsymbol{P} = 0$ and $\boldsymbol{\nu} \cdot \boldsymbol{P} = 0$ intersect lies on both quadrics. It can be shown that the intersection curve of these two surfaces consists of this line and a *twisted cubic* curve Γ passing through the origin. A twisted cubic is entirely determined by six points lying on it, and it follows that seven points chosen at random do not fall on Γ. Since this curve passes through the origin, choosing $n \geq 6$ random points generally guarantees that the matrix $\mathcal{P}$ has rank 11 and the projection matrix can be recovered in a unique fashion.

3.3 TAKING RADIAL DISTORTION INTO ACCOUNT

We have assumed so far that our camera is equipped with a perfect lens. As shown in chapter 1, real lenses suffer from a number of aberrations. In this section, we show how to account for *radial distortion*, a type of aberration that depends on the distance separating the optical axis from the point of interest. We assume that the image center is known so that we can take $u_0 = v_0 = 0$ and model the projection process as

$$\boldsymbol{p} = \frac{1}{z} \begin{pmatrix} 1/\lambda & 0 & 0 \\ 0 & 1/\lambda & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathcal{M} \boldsymbol{P}, \tag{3.17}$$

where λ is a polynomial function of the squared distance d^2 between the image center and the image point p. In most applications, it is sufficient to use a low-degree polynomial (e.g., $\lambda = 1 + \sum_{p=1}^{q} \kappa_p d^{2p}$, with $q \leq 3$) and the *distortion coefficients* κ_p ($p = 1, \ldots, q$) are normally assumed to be small. Note that d^2 is naturally expressed in terms of the *normalized* image coordinates of the point p (i.e., $d^2 = \hat{u}^2 + \hat{v}^2$). Substituting $u_0 = 0$ and $v_0 = 0$ in Eq. (2.13) allows us, after some algebraic manipulation, to rewrite d^2 as a function of u and v instead—namely,

$$d^2 = \frac{u^2}{\alpha^2} + \frac{v^2}{\beta^2} + 2\frac{uv}{\alpha\beta} \cos\theta. \tag{3.18}$$

Using Eq. (3.18) to write λ as an explicit function of u and v in Eq. (3.17) yields highly nonlinear constraints on the $q + 11$ camera parameters. Although these parameters in principle can all be found using the general *nonlinear least-squares* techniques introduced in the next section, we prefer here a two-stage approach tailored to the calibration problem: Eliminating λ from Eq. (3.17) first allows us to use *linear* least squares to estimate nine of the camera parameters. The q+2 remaining ones are then computed from Eqs. (3.17) and (3.18) by a simple nonlinear process.

3.3.1 Estimation of the Projection Matrix

Geometrically, radial distortion changes the distance between the image center and the image point p, but it does not affect the direction of the vector joining these two points. This is the *radial alignment constraint* introduced by Tsai (1987*a*), and it can be expressed algebraically by writing

$$\lambda \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \dfrac{\boldsymbol{m}_1 \cdot \boldsymbol{P}}{\boldsymbol{m}_3 \cdot \boldsymbol{P}} \\ \dfrac{\boldsymbol{m}_2 \cdot \boldsymbol{P}}{\boldsymbol{m}_3 \cdot \boldsymbol{P}} \end{pmatrix} \Longrightarrow v(\boldsymbol{m}_1 \cdot \boldsymbol{P}) - u(\boldsymbol{m}_2 \cdot \boldsymbol{P}) = 0. \tag{3.19}$$

Given n fiducial points, we obtain n linear equations in the eight coefficients of the vectors $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$—namely,

$$\mathcal{Q}\boldsymbol{n} = 0, \quad \text{where} \quad \mathcal{Q} \stackrel{\text{def}}{=} \begin{pmatrix} v_1 \boldsymbol{P}_1^T & -u_1 \boldsymbol{P}_1^T \\ \cdots & \cdots \\ v_n \boldsymbol{P}_n^T & -u_n \boldsymbol{P}_n^T \end{pmatrix} \quad \text{and} \quad \boldsymbol{n} = \begin{pmatrix} \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \end{pmatrix}. \tag{3.20}$$

Note the similarity with the previous case. When $n \geq 8$, this system of equations is in general overconstrained, and a solution with unit norm can be found using linear least squares.

3.3.2 Estimation of the Intrinsic and Extrinsic Parameters

Once $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ have been estimated, we can define as before the corresponding values of $\boldsymbol{a}_1, \boldsymbol{a}_2$ and write

$$\rho \begin{pmatrix} \boldsymbol{a}_1^T \\ \boldsymbol{a}_2^T \end{pmatrix} = \begin{pmatrix} \alpha \boldsymbol{r}_1^T - \alpha \cot\theta \boldsymbol{r}_2^T + u_0 \boldsymbol{r}_3^T \\ \dfrac{\beta}{\sin\theta} \boldsymbol{r}_2^T + v_0 \boldsymbol{r}_3^T \end{pmatrix}.$$

Calculating the norm and dot product of the vectors $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ immediately yields the aspect ratio and the skew of the camera as

$$\frac{\beta}{\alpha} = \frac{|\boldsymbol{a}_2|}{|\boldsymbol{a}_1|} \quad \text{and} \quad \cos\theta = -\frac{\boldsymbol{a}_1 \cdot \boldsymbol{a}_2}{|\boldsymbol{a}_1||\boldsymbol{a}_2|}. \tag{3.21}$$

Using the fact that $\boldsymbol{r}_2^T$ is the second row of a rotation matrix, and thus has unit norm, now yields

$$\alpha = \varepsilon\rho|\boldsymbol{a}_1|\sin\theta \quad \text{and} \quad \beta = \varepsilon\rho|\boldsymbol{a}_2|\sin\theta, \tag{3.22}$$

where, as before, $\varepsilon = \mp 1$. After some simple algebraic manipulation, we obtain

$$\begin{cases} \boldsymbol{r}_1 = \dfrac{\varepsilon}{\sin\theta}\left(\dfrac{1}{|\boldsymbol{a}_1|}\boldsymbol{a}_1 + \dfrac{\cos\theta}{|\boldsymbol{a}_2|}\boldsymbol{a}_2\right), \\ \boldsymbol{r}_2 = \dfrac{\varepsilon}{|\boldsymbol{a}_2|}\boldsymbol{a}_2. \end{cases}$$

Using these equations and $\boldsymbol{r}_3 = \boldsymbol{r}_1 \times \boldsymbol{r}_2$ allows us to recover the rotation matrix $\mathcal{R}$ up to a twofold ambiguity. Two of the translation parameters can also be recovered by writing

$$\begin{pmatrix} \alpha t_x - \alpha\cot\theta t_y \\ \dfrac{\beta}{\sin\theta} t_y \end{pmatrix} = \rho\begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$

where b_1 and b_2 are the first two coordinates of the vector $\boldsymbol{b}$, which in turn allows us to compute these parameters t_x and t_y as

$$\begin{cases} t_x = \dfrac{\varepsilon}{\sin\theta}\left(\dfrac{b_1}{|\boldsymbol{a}_1|} + \dfrac{b_2\cos\theta}{|\boldsymbol{a}_2|}\right), \\ t_y = \dfrac{\varepsilon b_2}{|\boldsymbol{a}_2|}. \end{cases}$$

Without further constraints, it is impossible to recover t_z and the absolute scale of the magnification parameters, or equivalently, the value of ρ, from the values of $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ only. To estimate these parameters, it is necessary to go back to the original projection equations: We rewrite the left side of Eq. (3.19) as

$$\begin{cases} (\boldsymbol{m}_1 - \lambda u\boldsymbol{m}_3)\cdot\boldsymbol{P} = 0, \\ (\boldsymbol{m}_2 - \lambda v\boldsymbol{m}_3)\cdot\boldsymbol{P} = 0, \end{cases} \tag{3.23}$$

Here $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ are known and, according to Eq. (2.17), $\boldsymbol{m}_3^T = (\boldsymbol{r}_3^T \quad t_z)$, where $\boldsymbol{r}_3$ is also known. Now, combining the expression for d^2 given in Eq. (3.18) with the expressions for α, β, and $\cos\theta$ given in Eqs. (3.21) and (3.22) yields

$$d^2 = \frac{1}{\rho^2}\frac{|u\boldsymbol{a}_2 - v\boldsymbol{a}_1|^2}{|\boldsymbol{a}_1 \times \boldsymbol{a}_2|^2},$$

and substituting this value in Eq. (3.23) yields a nonlinear equation in ρ, t_z, and the distortion parameters κ_p ($p = 1, \ldots, q$). Given enough data points, the nonlinear least-squares techniques that have been presented in Section 3.1.2 can be used to solve for these parameters. These methods are iterative and require initial guesses for all unknowns. Here, a reasonable estimate for ρ and t_z can be found using linear least squares by first assuming that $\lambda = 1$. Likewise, zero values are reasonable initial guesses for the distortion parameters. As before, the twofold ambiguity can be resolved when the sign of t_z is known in advance.

3.3.3 Degenerate Point Configurations

Let us determine the degenerate point configurations for which the vectors $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ cannot be uniquely determined. Given a vector $\boldsymbol{l}$ in the nullspace, we define the vectors $\boldsymbol{\lambda} = (l_1, l_2, l_3, l_4)^T$ and $\boldsymbol{\mu} = (l_5, l_6, l_7, l_8)^T$ and write

$$\boldsymbol{0} = \mathcal{Q}\boldsymbol{l} = \begin{pmatrix} v_1\boldsymbol{P}_1^T & -u_1\boldsymbol{P}_1^T \\ \cdots & \cdots \\ v_n\boldsymbol{P}_n^T & -u_n\boldsymbol{P}_n^T \end{pmatrix}\begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \end{pmatrix} = \begin{pmatrix} v_1\boldsymbol{P}_1^T\boldsymbol{\lambda} - u_1\boldsymbol{P}_1^T\boldsymbol{\mu} \\ \cdots \\ v_n\boldsymbol{P}_n^T\boldsymbol{\lambda} - u_n\boldsymbol{P}_n^T\boldsymbol{\mu} \end{pmatrix}.$$

Taking into account the values of u_i and v_i, rearranging the terms, and clearing the denominators yields

$$\boldsymbol{P}_i^T(\boldsymbol{m}_2\boldsymbol{\lambda}^T - \boldsymbol{m}_1\boldsymbol{\mu}^T)\boldsymbol{P}_i = 0 \quad \text{for} \quad i = 1, \ldots, n. \tag{3.24}$$

The vector $\boldsymbol{l}$ associated with $\boldsymbol{\lambda} = \boldsymbol{m}_1$ and $\boldsymbol{\mu} = \boldsymbol{m}_2$ is of course a solution of these equations (in the noise-free case; i.e., when all image positions are exact). When the points P_i ($i = 1, \ldots, n$) all lie in some plane Π or, equivalently, $\boldsymbol{\Pi} \cdot \boldsymbol{P}_i = 0$ for some 4-vector $\boldsymbol{\Pi}$, we can choose $(\boldsymbol{\lambda}, \boldsymbol{\mu})$ equal to $(\boldsymbol{\Pi}, \boldsymbol{0})$, $(\boldsymbol{0}, \boldsymbol{\Pi})$, or any linear combination of these two vectors, and construct a solution of Eq. (3.24). The nullspace of $\mathcal{Q}$ contains the three-dimensional vector space spanned by these vectors and $\boldsymbol{l}$. Thus, as before, points that all lie in the same plane cannot be used in this calibration method.

More generally, for a given value of $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, the points P_i form a degenerate configuration when they lie on the quadric surface defined by Eq. (3.24). Note that this surface contains the four straight lines defined by $\boldsymbol{\lambda} \cdot \boldsymbol{P} = \boldsymbol{\mu} \cdot \boldsymbol{P} = 0$, $\boldsymbol{\lambda} \cdot \boldsymbol{P} = \boldsymbol{m}_1 \cdot \boldsymbol{P} = 0$, $\boldsymbol{\mu} \cdot \boldsymbol{P} = \boldsymbol{m}_2 \cdot \boldsymbol{P} = 0$ and $\boldsymbol{m}_1 \cdot \boldsymbol{P} = \boldsymbol{m}_2 \cdot \boldsymbol{P} = 0$. Therefore, it must consist of two planes or be a cone, hyperboloid of one sheet, or hyperbolic paraboloid. In any case, for a large enough number of points in general position, our least-squares problem admits a unique solution.

3.4 ANALYTICAL PHOTOGRAMMETRY

The techniques presented so far ignore some of the constraints associated with the calibration process. For example, the camera skew was assumed to be arbitrary instead of (very close to) zero in Section 3.2. We present in this section a nonlinear approach to camera calibration that takes into account *all* the relevant constraints. This approach is borrowed from *photogrammetry*—an engineering field whose aim is to recover quantitative geometric information from one or several pictures, with applications in cartography, military intelligence, city planning, and so on. For many years, photogrammetry relied on a combination of geometric, optical, and mechanical methods to recover three-dimensional information from pictures, but the advent of computers in the 1950s has made a purely computational approach to this problem feasible. This is the domain of *analytical photogrammetry*, where the intrinsic parameters of a camera define its *interior orientation*, and the extrinsic parameters define its *exterior orientation*.

In this setting, we assume once again that we observe n fiducial points P_i ($i = 1, \ldots, n$) whose positions in some world coordinate system are known, and we minimize the mean-squared distance between the measured positions (u_i, v_i) of their images and the positions $(\tilde{u}_i, \tilde{v}_i)$ predicted by the perspective projection equation with respect to a vector of camera parameters $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_q)^T$ ($q \geq 11$) that may include various distortion coefficients in addition to the usual intrinsic and extrinsic parameters. The least-squares error can be written as

$$E(\boldsymbol{\xi}) = \sum_{i=1}^{n} [(\tilde{u}_i(\boldsymbol{\xi}) - u_i)^2 + (\tilde{v}_i(\boldsymbol{\xi}) - v_i)^2],$$

where

$$\tilde{u}_i(\boldsymbol{\xi}) \stackrel{\text{def}}{=} \frac{\boldsymbol{m}_1(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i} \quad \text{and} \quad \tilde{v}_i(\boldsymbol{\xi}) \stackrel{\text{def}}{=} \frac{\boldsymbol{m}_2(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i}.$$

Contrary to the cases studied so far, the dependency of each error term on the unknown parameters $\boldsymbol{\xi}$ is not linear. Instead, it involves a combination of polynomial and trigonometric functions, and minimizing the overall error measure involves the use of the nonlinear least squares algorithms discussed in Section 3.1.2. To follow the notation introduced in that section,

we rewrite our error function as

$$E(\xi) = |f(\xi)|^2 = \sum_{j=1}^{2n} f_j^2(\xi), \text{ where } \begin{cases} f_{2i-1}(\xi) = \tilde{u}_i(\xi) - u_i \\ f_{2i}(\xi) \quad = \tilde{v}_i(\xi) - v_i \end{cases} \text{ for } i = 1, \ldots, n.$$

The Gauss–Newton and Levenberg–Marquard techniques described in Section 3.1.2 require the gradient of the functions f_j, and Newton's method requires both their gradient and Hessian. Here we only calculate the gradient or, equivalently, the Jacobian of f. Let us drop the ξ argument for conciseness and define $\tilde{x}_i = \boldsymbol{m}_1 \cdot \boldsymbol{P}_i$, $\tilde{y}_i = \boldsymbol{m}_2 \cdot \boldsymbol{P}_i$ and $\tilde{z}_i = \boldsymbol{m}_3 \cdot \boldsymbol{P}_i$ $(i = 1, \ldots, n)$, so $\tilde{u}_i = \tilde{x}_i / \tilde{z}_i$ and $\tilde{v}_i = \tilde{y}_i / \tilde{z}_i$. We have

$$\begin{cases} \dfrac{\partial f_{2i-1}}{\partial \xi_j} = \dfrac{\partial \tilde{u}_i}{\partial \xi_j} = \dfrac{1}{z_i}\dfrac{\partial \tilde{x}_i}{\partial \xi_j} - \dfrac{\tilde{x}_i}{\tilde{z}_i^2}\dfrac{\partial \tilde{z}_i}{\partial \xi_j} = \dfrac{1}{\tilde{z}_i}\left(\dfrac{\partial}{\partial \xi_j}(\boldsymbol{m}_1 \cdot \boldsymbol{P}_i) - \tilde{u}_i \dfrac{\partial}{\partial \xi_j}(\boldsymbol{m}_3 \cdot \boldsymbol{P}_i)\right), \\ \dfrac{\partial f_{2i}}{\partial \xi_j} \quad = \dfrac{\partial \tilde{v}_i}{\partial \xi_j} = \dfrac{1}{\tilde{z}_i}\dfrac{\partial \tilde{y}_i}{\partial \xi_j} - \dfrac{\tilde{y}_i}{\tilde{z}_i^2}\dfrac{\partial \tilde{z}_i}{\partial \xi_j} = \dfrac{1}{\tilde{z}_i}\left(\dfrac{\partial}{\partial \xi_j}(\boldsymbol{m}_2 \cdot \boldsymbol{P}_i) - \tilde{v}_i \dfrac{\partial}{\partial \xi_j}(\boldsymbol{m}_3 \cdot \boldsymbol{P}_i)\right), \end{cases}$$

which is easily rewritten as

$$\begin{pmatrix} \dfrac{\partial f_{2i-1}}{\partial \xi_j} \\ \dfrac{\partial f_{2i}}{\partial \xi_j} \end{pmatrix} = \frac{1}{\tilde{z}_i} \begin{pmatrix} \boldsymbol{P}_i^T & \boldsymbol{0}^T & -\tilde{u}_i \boldsymbol{P}_i^T \\ \boldsymbol{0}^T & \boldsymbol{P}_i^T & -\tilde{v}_i \boldsymbol{P}_i^T \end{pmatrix} \mathcal{J}_{\boldsymbol{m}},$$

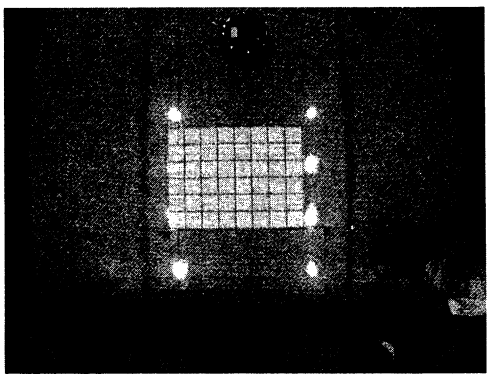
where $\boldsymbol{m}$ is as before the vector of $\mathbb{R}^{12}$ associated with $\mathcal{M}$, and $\mathcal{J}_{\boldsymbol{m}}$ denotes its Jacobian with respect to ξ. We finally obtain the Jacobian of f as

$$\mathcal{J}_f = \begin{pmatrix} \dfrac{1}{\tilde{z}_1}\boldsymbol{P}_1^T & \boldsymbol{0}^T & -\dfrac{\tilde{u}_1}{\tilde{z}_1}\boldsymbol{P}_1^T \\ \boldsymbol{0}^T & \dfrac{1}{\tilde{z}_1}\boldsymbol{P}_1^T & -\dfrac{\tilde{v}_1}{\tilde{z}_1}\boldsymbol{P}_1^T \\ \cdots & \cdots & \cdots \\ \dfrac{1}{\tilde{z}_n}\boldsymbol{P}_n^T & \boldsymbol{0}^T & -\dfrac{\tilde{u}_n}{\tilde{z}_n}\boldsymbol{P}_n^T \\ \boldsymbol{0}^T & \dfrac{1}{\tilde{z}_n}\boldsymbol{P}_n^T & -\dfrac{\tilde{v}_n}{\tilde{z}_n}\boldsymbol{P}_n^T \end{pmatrix} \mathcal{J}_{\boldsymbol{m}}.$$

In this expression, $\tilde{u}_i$, $\tilde{v}_i$, $\tilde{z}_i$, and $\boldsymbol{P}_i$ depend on the point considered, but $\mathcal{J}_{\boldsymbol{m}}$ only depends on the intrinsic and extrinsic parameters of the camera. Note that this method requires an explicit parameterization of the matrix $\mathcal{R}$. Such a parameterization in terms of three elementary rotations about coordinate axes was mentioned in chapter 2. Many other parameterizations can be used as well (see Exercises and chapter 21).

3.5 AN APPLICATION: MOBILE ROBOT LOCALIZATION

The calibration methods presented in this chapter can be used in a variety of applications, from metrology to stereo vision and object localization in robotic tasks. Here we briefly describe the nonlinear approach to camera calibration proposed by Devy *et al.* (1997) and its application to mobile robot localization. Unlike the techniques discussed so far, this method uses several images (up to 20 in the experiments presented here) of a planar rectangular grid to calibrate a



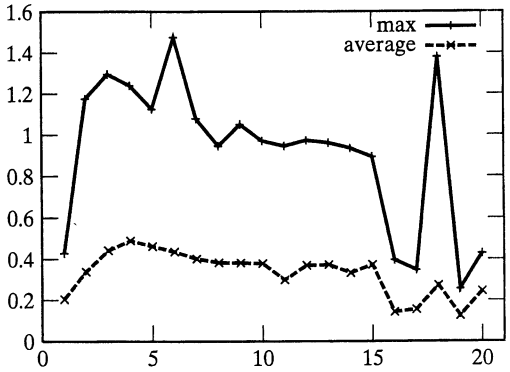


Figure 3.3 Calibration experiments. Left: One of the 20 input pictures; note the strong radial distortion. Note the mobile robot at the top of the photograph with its characteristic LED pattern. Right: Average and maximum reprojections errors (in pixels) in the 20 images. *Calibration and localization software courtesy of Michel Devy. Experiments courtesy of Fred Rothganger.*

static camera (Figure 3.3). One of these pictures is taken with the grid lying on the ground, and it is used to define the world coordinate system. After a rough manual localization, the corners of the grid are found in each picture with a precision of 1/10 pixel using a parametric model of the gray-level surface in the neighborhood of a corner.

The imaging geometry is modeled as in Section 3.3, with three radial distortion coefficients and zero skew. The calibration algorithm recovers a single set of intrinsic parameters, and one set of extrinsic parameters per input image. An initial guess for the intrinsic parameters can be obtained from information supplied by the camera and frame-grabber manufacturers. An initial guess for the extrinsic parameters can be obtained for each image using a variant of Tsai's (1987*a*) algorithm. Briefly, the projection matrix is estimated via linear least squares by choosing the z coordinate axis of the world reference frame perpendicular to the calibration grid. Accordingly, Eq. (3.20) now becomes

$$\mathcal{Q}'\boldsymbol{n}' = 0, \quad \text{where} \quad \mathcal{Q}' = \begin{pmatrix} v_1x_1 & v_1y_1 & v_1 & -u_1x_1 & -u_1y_1 & -u_1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ v_nx_n & v_ny_n & v_n & -u_nx_n & -u_1y_n & -u_n \end{pmatrix}$$

and $\boldsymbol{n}' = (m_{11}, m_{12}, m_{14}, m_{21}, m_{22}, m_{24})^T$.

Note that explicitly imposing that $z_i = 0$ avoids the degeneracies encountered by the previously discussed algorithms for coplanar points. Once $\boldsymbol{n}'$ is known, since the intrinsic parameters are assumed to be (roughly) known, it is a simple matter to compute the extrinsic parameters (see Exercises). Once initial guesses for both the intrinsic and extrinsic parameters are available, nonlinear optimization (in this case, the Levenberg–Marquardt algorithm) can be used to minimize as usual the mean-squared distance between predicted and observed image features.

Figure 3.3 shows the result of experiments conducted with a 576 × 768 camera equipped with a 4.5 mm lens and plots the errors found when reprojecting the corners of the calibration grid model into the 20 images. Once the camera has been calibrated, it can be used to monitor the position and orientation of mobile robots in the coordinate system attached to the ground reference image. Each robot carries an array of infrared LEDs forming a distinctive pattern (Figure 3.3). During localization experiments, the camera is equipped with an infrared filter that effectively blocks out all incoming light except for that from the LEDs. Each robot is identified

using a simple pattern matching algorithm, and its position and orientation are deduced from the image position of the LEDs and the camera parameters. With the camera mounted 4 m above the ground, typical localization errors within the entire field of view of the camera are below 2 cm in position and 1° in orientation, with maximum errors that may reach 5 cm and 5°.

3.6 NOTES

The linear calibration technique described in Section 3.2 is detailed in Faugeras (1993). Its variant that takes radial distortion into account is adapted from Tsai (1987*a*). Haralick and Shapiro (1992) present a concise introduction to analytical photogrammetry. The *Manual of Photogrammetry* is of course the gold standard, and newcomers (such as the authors of this book) will probably find the ingenious mechanisms and rigorous methods described in its various editions fascinating (Thompson *et al.*, 1966, Slama *et al.*, 1980). We come back to photogrammetry in the context of multiple images in chapter 10. General optimization techniques are discussed in Luenberger (1985), Bertsekas (1995), and Heath (2002), for example. An excellent survey and discussion of least-squares methods in the context of analytical photogrammetry can be found in Triggs *et al.* (2000). The output of least-squares methods admits a statistical interpretation in maximum-likelihood terms when the coordinates of the data points are modeled as random variables obeying a normal distribution. We come back to this interpretation in chapter 15, where we also revisit the problem of fitting a straight line to a set of points in the plane.

PROBLEMS

3.1. Show that the vector $\boldsymbol{x}$ that minimizes $|\mathcal{U}\boldsymbol{x}|^2$ under the constraint $|\mathcal{V}\boldsymbol{x}|^2 = 1$ is the unit generalized eigenvector associated with the minimum generalized eigenvalue of the symmetric matrices $\mathcal{U}^T\mathcal{U}$ and $\mathcal{V}^T\mathcal{V}$.

Hint: This problem is equivalent to minimizing (without constraints) the error $E = |\mathcal{U}\boldsymbol{x}|^2/|\mathcal{V}\boldsymbol{x}|^2$ with respect to $\boldsymbol{x}$.

3.2. Show that the 2×2 matrix $\mathcal{U}^T\mathcal{U}$ involved in the line-fitting example from Section 3.1.1 is the matrix of second moments of inertia of the points p_i $(i = 1, \ldots, n)$.

3.3. Extend the line-fitting method presented in Section 3.1.1 to the problem of fitting a plane to n points in $\mathbb{E}^3$.

3.4. Derive an expression for the Hessian of the functions $f_{2i-1}(\boldsymbol{\xi}) = \tilde{u}_i(\boldsymbol{\xi}) - u_i$ and $f_{2i}(\boldsymbol{\xi}) = \tilde{v}_i(\boldsymbol{\xi}) - v_i$ $(i = 1, \ldots, n)$ introduced in Section 3.4.

3.5. Euler angles. Show that the rotation obtained by first rotating about the z axis of some coordinate frame by an angle α, then rotating about the y axis of the new coordinate frame by an angle β and finally rotating about the z axis of the resulting frame by an angle γ can be represented in the original coordinate system by

$$\begin{pmatrix} \cos\alpha\cos\beta\cos\gamma - \sin\alpha\sin\gamma & -\cos\alpha\cos\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta \\ \sin\alpha\cos\beta\cos\gamma + \cos\alpha\sin\gamma & -\sin\alpha\cos\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta \\ -\sin\beta\cos\gamma & \sin\beta\sin\gamma & \cos\beta \end{pmatrix}.$$

3.6. The Rodrigues formula. Consider a rotation $\mathcal{R}$ of angle θ about the axis $\boldsymbol{u}$ (a unit vector). Show that $\mathcal{R}\boldsymbol{x} = \cos\theta\boldsymbol{x} + \sin\theta\,\boldsymbol{u} \times \boldsymbol{x} + (1 - \cos\theta)(\boldsymbol{u} \cdot \boldsymbol{x})\boldsymbol{u}$.

Hint: A rotation does not change the projection of a vector $\boldsymbol{x}$ onto the direction $\boldsymbol{u}$ of its axis and applies a planar rotation of angle θ to the projection of $\boldsymbol{x}$ into the plane orthogonal to $\boldsymbol{u}$.

3.7. Use the Rodrigues formula to show that the matrix associated with $\mathcal{R}$ is

$$\begin{pmatrix} u^2(1-c)+c & uv(1-c)-ws & uw(1-c)+vs \\ uv(1-c)+ws & v^2(1-c)+c & vw(1-c)-us \\ uw(1-c)-vs & vw(1-c)+us & w^2(1-c)+c \end{pmatrix}.$$

where $c = \cos\theta$ and $s = \sin\theta$.

3.8. Assuming that the intrinsic parameters of a camera are known, show how to compute its extrinsic parameters once the vector $\boldsymbol{n}'$ defined in Section 3.5 is known.

Hint: Use the fact that the rows of a rotation matrix have unit norm.

3.9. Assume that n fiducial lines with known Plücker coordinates are observed by a camera.

(a) Show that the line projection matrix $\tilde{\mathcal{M}}$ introduced in the exercises of chapter 2 can be recovered using linear least squares when $n \geq 9$.

(b) Show that once $\tilde{\mathcal{M}}$ is known, the projection matrix $\mathcal{M}$ can also be recovered using linear least squares.

Hint: Consider the rows $\boldsymbol{m}_i$ of $\mathcal{M}$ as the coordinate vectors of three planes Π_i and the rows $\tilde{\boldsymbol{m}}_i$ of $\tilde{\mathcal{M}}$ as the coordinate vectors of three lines, and use the incidence relationships between these planes and these lines to derive linear constraints on the vectors $\boldsymbol{m}_i$.

Programming Assignments

3.10. Use linear least-squares to fit a plane to n points $(x_i, y_i, z_i)^T$ $(i = 1, \ldots, n)$ in $\mathbb{R}^3$.

3.11. Use linear least-squares to fit a conic section defined by $ax^2 + bxy + cy^2 + dx + ey + f = 0$ to n points $(x_i, y_i)^T$ $(i = 1, \ldots, n)$ in $\mathbb{R}^2$.

3.12. Implement the linear calibration algorithm presented in Section 3.2.

3.13. Implement the calibration algorithm that takes into account radial distortion and that was presented in Section 3.3.

3.14. Implement the nonlinear calibration algorithm from Section 3.4.