ANALYTICAL IMAGE FEATURES

Chapter 1 laid the geometric foundations of image formation. This chapter uses analytical geometry to quantify more precisely the relationship between a camera, the objects it observes, and the pictures of these objects. We start by briefly recalling elementary notions of analytical Euclidean geometry, including dot and cross products, norms and distances, rigid transformations and homogeneous coordinates: this machinery will allow us in the rest of the book to reason about geometric objects like points, lines and planes, and geometric transformations like rotations, translations and projections in terms of linear algebra constructs such as vectors and matrices. We then introduce the various physical parameters that relate the world and camera coordinate frames, and present as an application various methods for estimating these parameters, a process known as geometric camera calibration. We also present along the way some linear and non-linear least-squares techniques for parameter estimation that will prove useful on several occasions in the rest of the book.

6.1 Elements of Analytical Euclidean Geometry

We assume that the reader has some familiarity with elementary analytical Euclidean geometry and linear algebra. This section will serve to fix the notation used in the book and introduce informally some useful notions such as coordinate systems, homogeneous coordinates, rotation matrices, etc.

Notation. We will use the following notation in the rest of this chapter and throughout the book: points, lines and planes will be denoted by Roman or Greek letters in italic font, e.g., P, Δ or Π. Vectors will be denoted by Roman or Greek bold-italic letters, e.g., $\boldsymbol{v}$, $\boldsymbol{P}$, or $\boldsymbol{\xi}$, although the vector joining two points P and Q will often be denoted by $\overrightarrow{PQ}$. Matrices will be denoted by Roman letters in calligraphic font, e.g., $\mathcal{M}$. The familiar three-dimensional Euclidean space will be denoted by $\mathbb{E}^3$ and the vector space formed by n-tuples of real numbers with the usual laws of addition and multiplication by a scalar will be denoted by $\mathbb{R}^n$. El-

ements of $\mathbb{R}^n$ will be considered as column vectors or $n \times 1$ matrices, and the transpose of the $m \times n$ matrix $\mathcal{A}$ with coefficients a_{ij} will be the $n \times m$ matrix denoted by $\mathcal{A}^T$ with coefficients a_{ji}.

We will denote the dot product (or inner product) between two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ as $\boldsymbol{u} \cdot \boldsymbol{v}$. When these two vectors are elements of $\mathbb{R}^n$ given by $\boldsymbol{u} = (u_1, \ldots, u_n)^T$ and $\boldsymbol{v} = (v_1, \ldots, v_n)$, we have of course

$$\boldsymbol{u} \cdot \boldsymbol{v} = u_1 v_1 + \ldots + u_n v_n,$$

and we will often use the fact that in this case the dot product can be rewritten as a matrix product, i.e., $\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u}^T \boldsymbol{v} = \boldsymbol{v}^T \boldsymbol{u}$. We will denote by $|\boldsymbol{v}|^2 = \boldsymbol{v} \cdot \boldsymbol{v}$ the square of the Euclidean norm of the vector $\boldsymbol{v}$, and denote by d the distance function induced by the Euclidean norm in $\mathbb{E}^3$, i.e., $d(P, Q) = |\overrightarrow{PQ}|$.

The symbol "$\times$" will be used to denote the cross product (or outer product) operator that associates with two vectors $\boldsymbol{u} = (u_1, u_2, u_3)^T$ and $\boldsymbol{v} = (v_1, v_2, v_3)^T$ the vector

$$\boldsymbol{u} \times \boldsymbol{v} \stackrel{\text{def}}{=} \begin{pmatrix} u_2 v_3 - u_3 v_2 \\ u_3 v_1 - u_1 v_3 \\ u_1 v_2 - u_2 v_1 \end{pmatrix}.$$

When $\boldsymbol{u}$ has unit norm, the dot product $\boldsymbol{u} \cdot \boldsymbol{v}$ is equal to the (signed) length of the projection of $\boldsymbol{v}$ onto $\boldsymbol{u}$, and two vectors are orthogonal when their dot product is zero. On the other hand, the cross product of two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ in $\mathbb{R}^3$ is orthogonal to these two vectors, and a necessary and sufficient condition for $\boldsymbol{u}$ and $\boldsymbol{v}$ to have the same direction is that $\boldsymbol{u} \times \boldsymbol{v} = 0$. We will also use the identities

$$\begin{cases} (\boldsymbol{u} \cdot \boldsymbol{v})^2 = |\boldsymbol{u}|^2 |\boldsymbol{v}|^2 \cos^2 \theta, \\ |\boldsymbol{v} \times \boldsymbol{v}|^2 = |\boldsymbol{u}|^2 |\boldsymbol{v}|^2 \sin^2 \theta, \end{cases}$$

where θ denotes the angle between the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$.

6.1.1 Coordinate Systems and Homogeneous Coordinates

We already used three-dimensional coordinate systems in Chapter 1. Let us introduce them a bit more formally: we assume a fixed system of units, say meters, or inches, so unit length is well defined; picking a point O in $\mathbb{E}^3$ and three unit vectors $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ orthogonal to each other defines an *orthonormal coordinate frame* (F) as the quadruple $(O, \boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$. The point O is the *origin* of the coordinate system (F), and $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ are its *basis vectors*. We will restrict our attention to *right-handed* coordinate systems, such that the vectors $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ can be thought of as being attached to fingers of your right hand, with the thumb pointing up, index pointing straight, and middle finger pointing left as shown in Figure 6.1.[1]

[1] This is the traditional way of defining right-handed coordinate systems. One of the authors, who is left-handed, has always found it a bit confusing, and prefers to identify these coordinate systems using the fact that when one looks down the $\boldsymbol{k}$ axis at the $(\boldsymbol{i}, \boldsymbol{j})$ plane, the vector $\boldsymbol{i}$ is mapped onto the vector $\boldsymbol{j}$ by a *counterclockwise* 90° rotation (Figure 6.1). Left-handed coordinate systems correspond to *clockwise* rotations. Left- and right-handed readers alike may find this characterization useful as well.

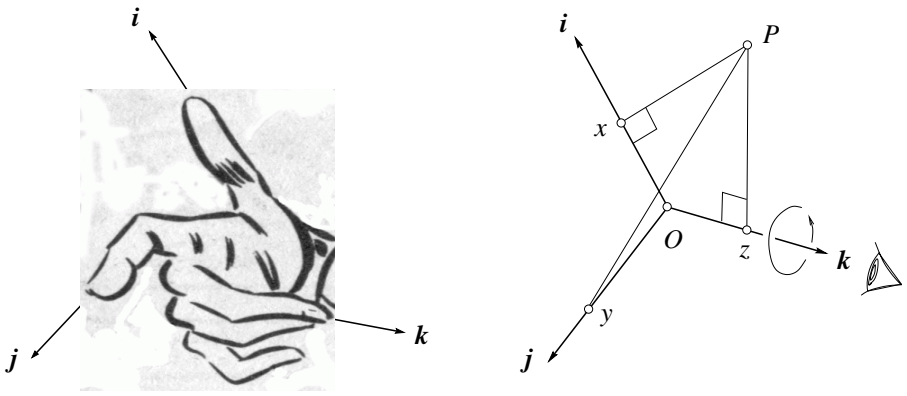

Figure 6.1. A right-handed coordinate system and the Cartesian coordinates x, y, z of a point P.

The Cartesian coordinates x, y and z of a point P in this coordinate frame are defined as the (signed) lengths of the orthogonal projections of the vector $\overrightarrow{OP}$ onto the vectors $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ (Figure 6.1), with

$$\begin{cases} x = \overrightarrow{OP} \cdot \boldsymbol{i} \\ y = \overrightarrow{OP} \cdot \boldsymbol{j} \\ z = \overrightarrow{OP} \cdot \boldsymbol{k} \end{cases} \iff \overrightarrow{OP} = x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}.$$

The column vector

$$\boldsymbol{P} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3$$

is called the *coordinate vector* of the point P in (F). We can also define the coordinate vector associated with any free vector $\boldsymbol{v}$ by the lengths of its projections onto the basis vectors of (F), and these coordinates are of course independent of the choice of the origin O.

Let us now consider a plane Π, an arbitrary point A in Π and a unit vector $\boldsymbol{n}$ perpendicular to the plane. The points lying in Π are characterized by

$$\overrightarrow{AP} \cdot \boldsymbol{n} = 0.$$

In a coordinate system (F) where the coordinates of the point P are x, y, z and the coordinates of $\boldsymbol{n}$ are a, b and c, this can be rewritten as $\overrightarrow{OP} \cdot \boldsymbol{n} - \overrightarrow{OA} \cdot \boldsymbol{n} = 0$ or

$$ax + by + cz - d = 0, \tag{6.1.1}$$

where $d \stackrel{\text{def}}{=} \overrightarrow{OA} \cdot \boldsymbol{n}$ is independent of the choice of the point A in Π and is simply the (signed) distance between the origin O and the plane Π (Figure 6.2)

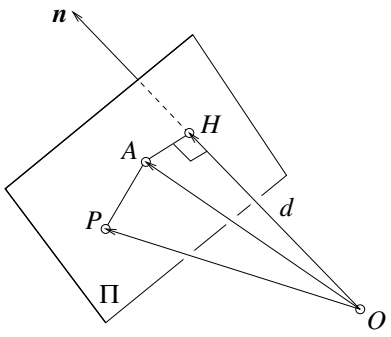

Figure 6.2. The geometric definition of the equation of a plane. The distance d between the origin and the plane is reached at the point H where the normal vector passing through the origin pierces the plane.

At times, it is useful to use *homogeneous coordinates* to represent points, vectors, and planes. We will justify formally their definition later in this book, when we introduce notions of affine and projective geometry, but for the time being, let us just note that (6.1.1) can be rewritten as

$$(a, b, c, -d) \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = 0$$

or, more concisely, as

$$\boldsymbol{\Pi} \cdot \boldsymbol{P} = 0, \quad \text{where} \quad \boldsymbol{\Pi} \stackrel{\text{def}}{=} \begin{pmatrix} a \\ b \\ c \\ -d \end{pmatrix} \quad \text{and} \quad \boldsymbol{P} \stackrel{\text{def}}{=} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \tag{6.1.2}$$

The vector $\boldsymbol{P}$ is called the *vector of homogenous coordinates* of the point P in the coordinate system (F), and it is simply obtained by adding a fourth coordinate equal to 1 to the ordinary coordinate vector of P. Likewise, the vector $\boldsymbol{\Pi}$ is the vector of homogeneous coordinates of the plane Π in the coordinate frame (F) and (6.1.2) is called the equation of Π in that coordinate system. Note that $\boldsymbol{\Pi}$ is only defined up to scale since multiplying this vector by any nonzero constant does not change the solutions of (6.1.2).

We will use the convention that homogeneous coordinates are only defined up to scale, whether they represent points or planes (this will be established more formally for points later). To go back to the ordinary non-homogenous coordinates of points, one just divides all coordinates by the fourth one. Among other things, homogeneous coordinates will allow us shortly to express changes of coordinate in terms of vectors of matrices, but first, we have to understand how coordinates change between two frames.

6.1.2 Coordinate System Changes and Rigid Transformations

When several different coordinate systems are considered at the same time, it is convenient to follow Craig [1989] and denote by ${}^F P$ (resp. ${}^F \boldsymbol{v}$) the coordinate vector of the point P (resp. vector $\boldsymbol{v}$) in the frame (F), i.e.,[2]

$$ {}^F P = {}^F \overrightarrow{OP} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \Longleftrightarrow \overrightarrow{OP} = x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}. $$

Let us now consider two coordinate systems $(A) = (O_A, \boldsymbol{i}_A, \boldsymbol{j}_A, \boldsymbol{k}_A)$ and $(B) = (O_B, \boldsymbol{i}_B, \boldsymbol{j}_B, \boldsymbol{k}_B)$. The rest of this section will allow us to express ${}^B P$ as a function of ${}^A P$. Let us suppose first that the basis vectors of both coordinate systems are parallel to each other, i.e., $\boldsymbol{i}_A = \boldsymbol{i}_B$, $\boldsymbol{j}_A = \boldsymbol{j}_B$ and $\boldsymbol{k}_A = \boldsymbol{k}_B$, but the origins O_A and O_B are distinct (Figure 6.3). We say that the two coordinate systems are separated by a *pure translation*, and we have $\overrightarrow{O_B P} = \overrightarrow{O_B O_A} + \overrightarrow{O_A P}$, thus

$$ {}^B P = {}^A P + {}^B O_A. $$

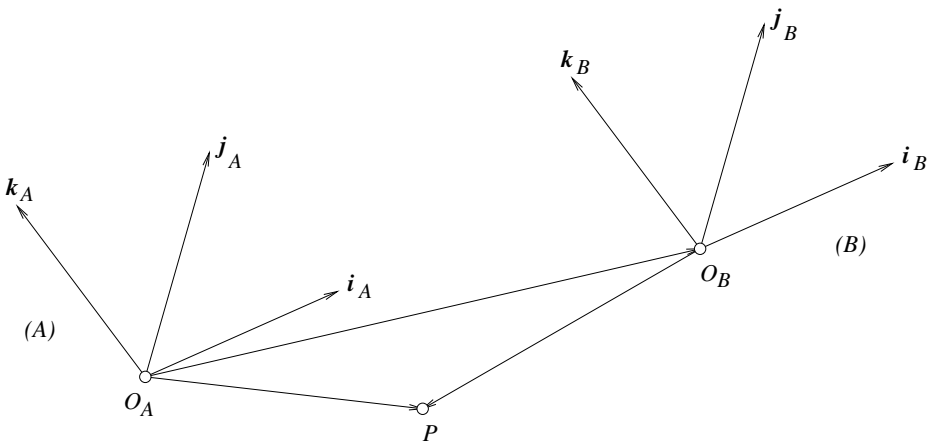

Figure 6.3. Coordinate change between two frames: pure translation.

When the origins of the two frames coincide, i.e., $O_A = O_B = O$, we say that the frames are separated by a *pure rotation* (Figure 6.4). Let us define the *rotation matrix* ${}^B_A\mathcal{R}$ as the 3×3 array of numbers

$$ {}^B_A\mathcal{R} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{i}_A \cdot \boldsymbol{i}_B & \boldsymbol{j}_A \cdot \boldsymbol{i}_B & \boldsymbol{k}_A \cdot \boldsymbol{i}_B \\ \boldsymbol{i}_A \cdot \boldsymbol{j}_B & \boldsymbol{j}_A \cdot \boldsymbol{j}_B & \boldsymbol{k}_A \cdot \boldsymbol{j}_B \\ \boldsymbol{i}_A \cdot \boldsymbol{k}_B & \boldsymbol{j}_A \cdot \boldsymbol{k}_B & \boldsymbol{k}_A \cdot \boldsymbol{k}_B \end{pmatrix}. $$

[2] The superscripts and subscripts preceding points, vectors and matrices in Craig's notation may appear awkward at first, but the rest of this section should clearly demonstrate their utility. Please stick with us for a little while..

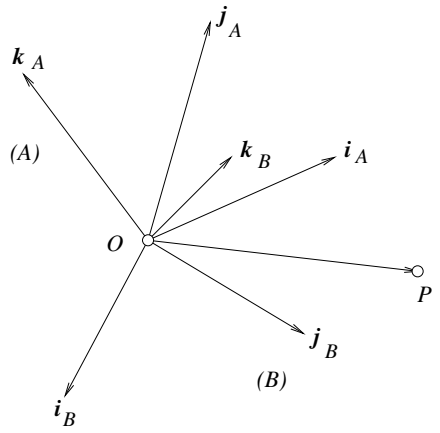

Figure 6.4. Coordinate change between two frames: pure rotation.

Note that the first column of ${}^B_A\mathcal{R}$ is formed by the coordinates of $\boldsymbol{i}_A$ in the basis $(\boldsymbol{i}_B, \boldsymbol{j}_B, \boldsymbol{k}_B)$. Likewise, the third row of this matrix is formed by the coordinates of $\boldsymbol{k}_B$ in the basis $(\boldsymbol{i}_A, \boldsymbol{j}_A, \boldsymbol{k}_A)$, etc. More generally, the matrix ${}^B_A\mathcal{R}$ can be written in a more compact fashion using a combination of three column vectors or three row vectors:

$$ {}^B_A\mathcal{R} = \left({}^B\boldsymbol{i}_A \quad {}^B\boldsymbol{j}_A \quad {}^B\boldsymbol{k}_A \right) = \begin{pmatrix} {}^A\boldsymbol{i}_B{}^T \\ {}^A\boldsymbol{j}_B{}^T \\ {}^A\boldsymbol{k}_B{}^T \end{pmatrix}, $$

and it follows that ${}^A_B\mathcal{R} = {}^B_A\mathcal{R}^T$.

As noted earlier, all these subscripts and superscripts may be somewhat confusing at first. To keep everything straight, it is useful to remember that in a change of coordinates, subscripts refer to the object being described, while superscripts refer to the coordinate system in which the object is described. For example ${}^AP$ refers to the coordinate vector of the point P in the frame (A), ${}^B\boldsymbol{j}_A$ is the coordinate vector of the vector $\boldsymbol{j}_A$ in the frame (B), and ${}^B_A\mathcal{R}$ is the rotation matrix describing the frame (A) in the coordinate system (B).

Let us give an example of pure rotation: suppose that $\boldsymbol{k}_A = \boldsymbol{k}_B = \boldsymbol{k}$, and denote by θ the angle such that the vector $\boldsymbol{i}_B$ is obtained by applying to the vector $\boldsymbol{i}_A$ a counterclockwise rotation of angle θ about $\boldsymbol{k}$ (Figure 6.5). The angle between the vectors $\boldsymbol{j}_A$ and $\boldsymbol{j}_B$ is also θ in this case, and we have

$$ {}^B_A\mathcal{R} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{6.1.3} $$

Similar formulas can be written when the two coordinate systems are deduced from each other via rotations about the $\boldsymbol{i}_A$ or $\boldsymbol{j}_A$ axes (see exercises). In general,

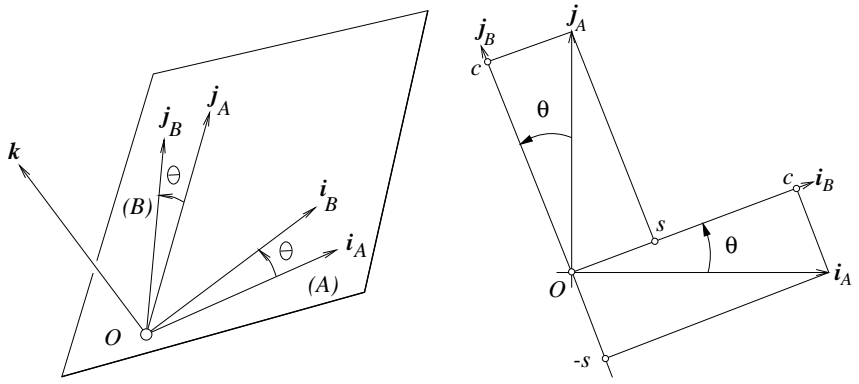

Figure 6.5. Two coordinate frames separated by a rotation of angle θ about their common $\boldsymbol{k}$ basis vector. As shown in the right of the figure, $\boldsymbol{i}_A = c\boldsymbol{i}_B - s\boldsymbol{j}_B$ and $\boldsymbol{i}_A = s\boldsymbol{i}_B + c\boldsymbol{j}_B$, where $c = \cos\theta$ and $s = \sin\theta$.

it can be shown that any rotation matrix can be written as the product of three elementary rotations about the $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ vectors of some coordinate system.

Let us go back to characterizing the change of coordinates associated with an arbitrary rotation matrix. Writing

$$\overrightarrow{OP} = \begin{pmatrix} \boldsymbol{i}_A & \boldsymbol{j}_A & \boldsymbol{k}_A \end{pmatrix} \begin{pmatrix} {}^Ax \\ {}^Ay \\ {}^Az \end{pmatrix} = \begin{pmatrix} \boldsymbol{i}_B & \boldsymbol{j}_B & \boldsymbol{k}_B \end{pmatrix} \begin{pmatrix} {}^Bx \\ {}^By \\ {}^Bz \end{pmatrix}$$

in the frame (B) yields immediately

$${}^BP = {}^B_A\mathcal{R}\,{}^AP$$

since the rotation matrix ${}^B_B\mathcal{R}$ is obviously the identity. Note how the subscript matches the following superscript. This property remains true for more general coordinate changes and it can be used after some practice to reconstruct the corresponding formulas without calculations.

It is easy to show (see exercises) that rotation matrices are characterized by the following properties: (1) the inverse of a rotation matrix is equal to its transpose, and (2) its determinant is equal to 1. By definition, the columns of a rotation matrix form a right-handed orthonormal coordinate system. It follows from property (1) that their rows also form such a coordinate system.

It should be noted that the set of rotation matrices, equipped with the matrix product, forms a *group*, i.e., the product of two rotation matrices is also a rotation matrix (this is intuitively obvious and easily verified analytically); the matrix product is associative; there is a unit element, the 3×3 identity matrix Id; and every rotation matrix $\mathcal{R}$ admits an inverse $\mathcal{R}^{-1} = \mathcal{R}^T$ such that $\mathcal{R}\mathcal{R}^{-1} = \mathcal{R}^{-1}\mathcal{R} = \mathrm{Id}$.

When the origins and the basis vectors of the two coordinate systems are different, we say that that the frames are separated by a general *rigid transformation* (Figure 6.6), and we have

$$ {}^{B}P = {}^{B}_{A}\mathcal{R}\,{}^{A}P + {}^{B}O_A. \tag{6.1.4}$$

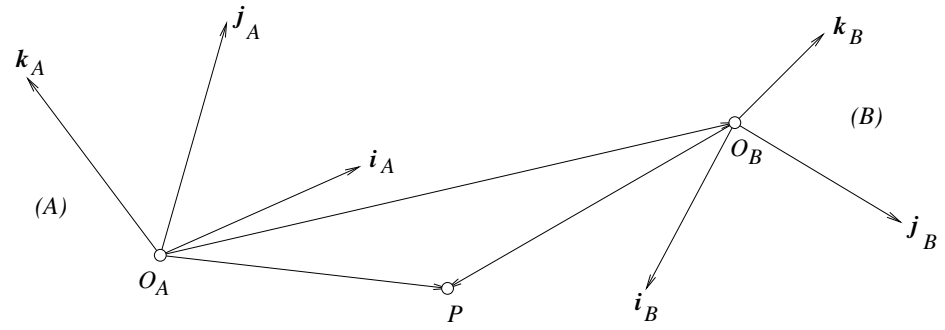

Figure 6.6. Coordinate changes between two frames: general rigid transformation.

Homogeneous coordinates can be used to rewrite (6.1.4) as a matrix product: let us first note that matrices can be multiplied in blocks, i.e., if

$$ \mathcal{A} = \begin{pmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ \mathcal{A}_{21} & \mathcal{A}_{22} \end{pmatrix} \quad \text{and} \quad \mathcal{B} = \begin{pmatrix} \mathcal{B}_{11} & \mathcal{B}_{12} \\ \mathcal{B}_{21} & \mathcal{B}_{22} \end{pmatrix}, \tag{6.1.5}$$

where the number of columns of the sub-matrices $\mathcal{A}_{11}$ and $\mathcal{A}_{21}$ (resp. $\mathcal{A}_{12}$ and $\mathcal{A}_{22}$) is equal to the number of rows of $\mathcal{B}_{11}$ and $\mathcal{B}_{12}$ (resp. $\mathcal{B}_{21}$ and $\mathcal{B}_{22}$), then

$$ \mathcal{A}\mathcal{B} = \begin{pmatrix} \mathcal{A}_{11}\mathcal{B}_{11} + \mathcal{A}_{12}\mathcal{B}_{21} & \mathcal{A}_{11}\mathcal{B}_{12} + \mathcal{A}_{12}\mathcal{B}_{22} \\ \mathcal{A}_{21}\mathcal{B}_{11} + \mathcal{A}_{22}\mathcal{B}_{21} & \mathcal{A}_{21}\mathcal{B}_{12} + \mathcal{A}_{22}\mathcal{B}_{22} \end{pmatrix}. $$

For example, we have

$$ \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{pmatrix} = \begin{pmatrix} r_{11}c_{11} + r_{12}c_{21} + r_{13}c_{31} & r_{11}c_{12} + r_{12}c_{22} + r_{13}c_{32} \\ r_{21}c_{11} + r_{22}c_{21} + r_{23}c_{31} & r_{21}c_{12} + r_{22}c_{22} + r_{23}c_{32} \\ r_{31}c_{11} + r_{32}c_{21} + r_{33}c_{31} & r_{31}c_{12} + r_{32}c_{22} + r_{33}c_{32} \end{pmatrix} $$

$$ = \left[\begin{array}{ccc|cc} r_{11} & r_{12} & r_{13} & c_{11} & c_{12} \\ r_{21} & r_{22} & r_{23} & c_{21} & c_{22} \\ \hline r_{31} & r_{32} & r_{33} & c_{31} & c_{32} \end{array}\right] = \begin{pmatrix} \begin{pmatrix} r_{11}\,r_{12}\,r_{13} \\ r_{21}\,r_{22}\,r_{23} \end{pmatrix} \begin{pmatrix} c_{11} \\ c_{21} \\ c_{31} \end{pmatrix} & \begin{pmatrix} r_{11}\,r_{12}\,r_{13} \\ r_{21}\,r_{22}\,r_{23} \end{pmatrix} \begin{pmatrix} c_{12} \\ c_{22} \\ c_{32} \end{pmatrix} \\ \begin{pmatrix} r_{31}\,r_{32}\,r_{33} \end{pmatrix} \begin{pmatrix} c_{11} \\ c_{21} \\ c_{31} \end{pmatrix} & \begin{pmatrix} r_{31}\,r_{32}\,r_{33} \end{pmatrix} \begin{pmatrix} c_{12} \\ c_{22} \\ c_{32} \end{pmatrix} \end{pmatrix}. $$

In particular, (6.1.5) allows us to rewrite the change of coordinates (6.1.4) as

$$ \begin{pmatrix} {}^{B}P \\ 1 \end{pmatrix} = {}^{B}_{A}\mathcal{T} \begin{pmatrix} {}^{A}P \\ 1 \end{pmatrix}, \quad \text{where} \quad {}^{B}_{A}\mathcal{T} \stackrel{\text{def}}{=} \begin{pmatrix} {}^{B}_{A}\mathcal{R} & {}^{B}O_A \\ \mathbf{0}^T & 1 \end{pmatrix} \tag{6.1.6}$$

and $\mathbf{0} = (0,0,0)^T$. In other words, using homogeneous coordinates allows us to write a general change of coordinates as the product of a 4×4 matrix and a 4-vector. It is easy to show that the set of rigid transformations defined by (6.1.6), equipped with the matrix product operation is also a group (see exercises).

A rigid transformation maps a coordinate system onto another one. In a given coordinate frame (F), a rigid displacement can also be considered as a mapping between points, i.e., a point P is mapped onto the point P' such that

$$^F P' = \mathcal{R}\, ^F P + \boldsymbol{t} \Longleftrightarrow \begin{pmatrix} ^F P' \\ 1 \end{pmatrix} = \begin{pmatrix} \mathcal{R} & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} ^F P \\ 1 \end{pmatrix}, \tag{6.1.7}$$

where $\mathcal{R}$ is a rotation matrix and $\boldsymbol{t}$ is an element of $\mathbb{R}^3$ (Figure 6.7). The set of rigid transformations considered as mappings of $\mathbb{E}^3$ onto itself and equipped with the law of composition is once again easily shown to form a group. It is also easy to show that rigid transformations preserve the distance between points and the angle between vectors. On the other hand, the 4×4 matrix associated with a rigid transformation depends on the choice of (F) (see exercises).

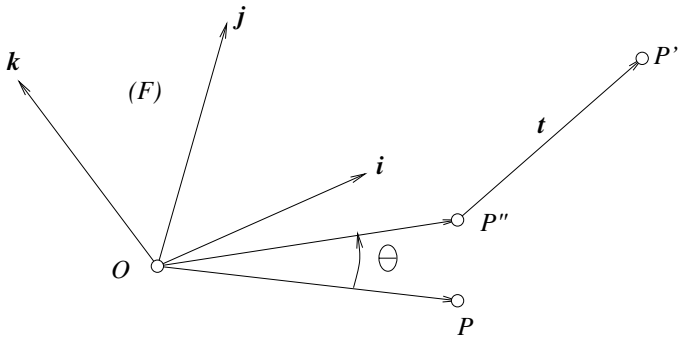

Figure 6.7. A rigid transformation maps the point P onto the point P'' through a rotation $\mathcal{R}$ before mapping P'' onto P' via a translation $\boldsymbol{t}$. In the example shown in this figure, $\mathcal{R}$ is a rotation of angle θ about the $\boldsymbol{k}$ axis of the coordinate system (F).

For example, let us consider the rotation of angle θ about the $\boldsymbol{k}$ axis of the frame (F). As shown in the exercises, this mapping can be represented by

$$^F P' = \mathcal{R}\, ^F P, \quad \text{where} \quad \mathcal{R} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

In particular, if (F') is the coordinate system obtained by applying this rotation to (F), we have, according to (6.1.3), $^{F'} P = {}_F^{F'}\mathcal{R}\, ^F P$, and $\mathcal{R} = {}_F^{F'}\mathcal{R}^{-1}$. More generally, the matrix representing the change of coordinates between two frames is the inverse of the matrix mapping the first frame onto the second one (see exercises).

What happens when $\mathcal{R}$ is replaced by an arbitrary 3×3 matrix $\mathcal{A}$? Equation (6.1.7) still represents a mapping between points (or a change of coordinates between frames), but this time lengths and angles may not be preserved anymore (equivalently, the new coordinate system does not necessarily have orthogonal axes with unit length). We say that the 4×4 matrix

$$\mathcal{T} = \begin{pmatrix} \mathcal{A} & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

represents an *affine transformation*. When $\mathcal{T}$ is allowed to be completely arbitrary, we say that we have a *projective transformation*. Affine and projective transformations also form groups, and they will be given a more thorough treatment later in the book.

6.2 Geometric Camera Parameters

We saw in Chapter 1 that the coordinates (x, y, z) of a scene point P observed by a pinhole camera are related to its image coordinates (x', y') by the perspective equation (1.1.1). In reality, this equation is only valid when all distances are measured in the camera's reference frame, and image coordinates have their origin at the principal point where the axis of symmetry of the camera pierces its retina. In practice, the world and camera coordinate systems are related by a set of physical parameters, such as the focal length of the lens, the size of the pixels, the position of the principal point, and the position and orientation of the camera.

This section identifies these parameters. We will distinguish the *intrinsic* parameters, that relate the camera's coordinate system to the idealized coordinate system used in Chapter 1, from the *extrinsic* parameters, that relate the camera's coordinate system to a fixed world coordinate system and specify its position and orientation in space.

Before proceeding, let us note that we will ignore in the rest of this chapter the fact that for cameras equipped with a lens, a point will only be in focus when its depth and the distance betweem the optical center of the camera and its image plane obey the thin lens equation (1.2.4). Likewise, the non-linear aberrations associated with real lenses are not taken into account by (1.1.1). We will neglect these aberrations in most of the chapter but will consider radial distortion in Section 6.3.2.

6.2.1 Intrinsic Parameters

We can associate with a camera two different image planes: the first one is a normalized plane located at a unit distance from the pinhole. We attach to this plane its own coordinate system with an origin located at the point $\hat{C}$ where the optical axis pierces it (Figure 6.8). The perspective projection equation (1.1.1) can be written

in this normalized coordinate system as

$$\begin{cases} \hat{u} = \dfrac{x}{z} \\ \hat{v} = \dfrac{y}{z} \end{cases} \iff \hat{\boldsymbol{p}} = \frac{1}{z} \begin{pmatrix} \text{Id} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{P} \\ 1 \end{pmatrix}, \tag{6.2.1}$$

where $\hat{\boldsymbol{p}} \stackrel{\text{def}}{=} (\hat{u}, \hat{v}, 1)^T$ is the vector of homogeneous coordinates of the projection $\hat{p}$ of the point P into the normalized image plane.

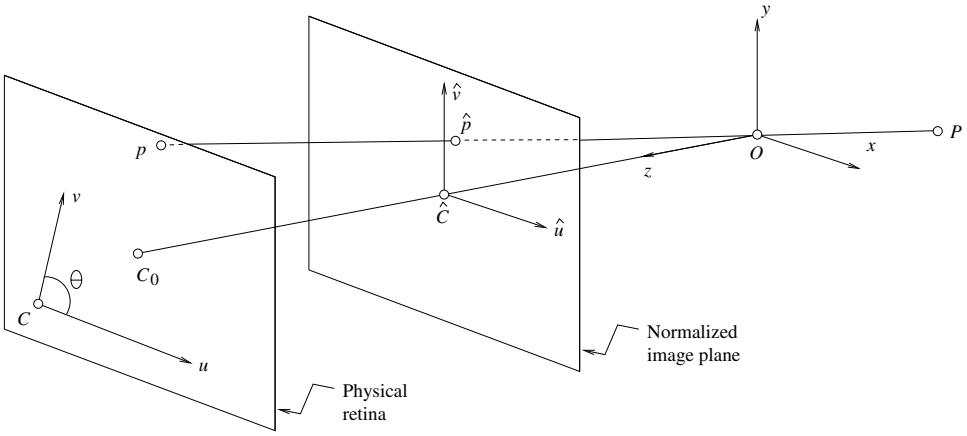

Figure 6.8. Physical and normalized image coordinate systems.

The physical retina of the camera is in general different (Figure 6.8): it is located at a distance $f \neq 1$ from the pinhole,[3] and the image coordinates (u, v) of the image point p are usually expressed in pixel units (instead of, say, meters). In addition, pixels are normally rectangular instead of square, so the camera has two additional scale parameters k and l, and

$$\begin{cases} u = kf\dfrac{x}{z}, \\ v = lf\dfrac{y}{z}. \end{cases} \tag{6.2.2}$$

Let us talk units for a second: f is a distance, expressed in meters for example, and a pixel will have dimensions $\frac{1}{k} \times \frac{1}{l}$, where k and l are expressed in pixel $\times\, m^{-1}$. The parameters k, l and f are not independent, and they can be replaced by the magnifications $\alpha = kf$ and $\beta = lf$ expressed in pixel units.

Now, in general, the actual origin of the camera coordinate system is at a corner C of the retina (e.g., in the case depicted in Figure 6.8, the lower-left corner,

[3] From now on we will assume that the camera is focused at infinity so the distance between the pinhole and the image plane is equal to the focal length.

or sometimes the upper-left corner, when the image coordinates are the row and column indices of a pixel) and not at its center, and the center of the CCD matrix usually does not coincide with the principal point C_0. This adds two parameters u_0 and v_0 that define the position (in pixel units) of C_0 in the retinal coordinate system. Thus, (6.2.2) is replaced by

$$\begin{cases} u = \alpha \dfrac{x}{z} + u_0, \\ v = \beta \dfrac{y}{z} + v_0. \end{cases} \tag{6.2.3}$$

Finally, the camera coordinate system may also be skewed, due to some manufacturing error, so the angle θ between the two image axes is not equal to (but of course not very different from either) 90 degrees. In this case, it is easy to show (see exercises) that (6.2.3) transforms into

$$\begin{cases} u = \alpha \dfrac{x}{z} - \alpha \cot\theta \dfrac{y}{z} + u_0, \\ v = \dfrac{\beta}{\sin\theta} \dfrac{y}{z} + v_0. \end{cases} \tag{6.2.4}$$

Combining (6.2.1) and (6.2.4) now allows us to write the change in coordinates between the physical image frame and the normalized one as a planar affine transformation:

$$\boldsymbol{p} = \mathcal{K}\hat{\boldsymbol{p}}, \quad \text{where} \quad \boldsymbol{p} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \quad \text{and} \quad \mathcal{K} \stackrel{\text{def}}{=} \begin{pmatrix} \alpha & -\alpha\cot\theta & u_0 \\ 0 & \dfrac{\beta}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Putting it all together, we obtain

$$\boldsymbol{p} = \frac{1}{z}\mathcal{M}\boldsymbol{P}, \quad \text{where} \quad \mathcal{M} \stackrel{\text{def}}{=} (\,\mathcal{K} \quad \mathbf{0}\,) \tag{6.2.5}$$

and $\boldsymbol{P}$ denotes this time the *homogeneous* coordinate vector of P in the camera coordinate system: homogeneous coordinates have allowed us to represent the perspective projection mapping by the 3×4 matrix $\mathcal{M}$.

Note that the physical size of the pixels and the skew are always fixed for a given camera and frame grabber, and they can in principle be measured during manufacturing (this information may of course not be available, in the case of stock film footage for example, or when the frame grabber's digitization rate is unknown and different from 1). For zoom lenses, the focal length and possibly the optical center may vary with time. As mentioned earlier, simply changing the focus of the camera will also affect the magnification since it will change the lens-to-retina distance, but we will ignore this effect in the sequel.

6.2.2 Extrinsic Parameters

We consider in this section the case where the camera frame (C) is distinct from the world frame (W). Noting that

$$ {}^{C}P = \begin{pmatrix} {}^{C}_{W}\mathcal{R} & {}^{C}O_W \end{pmatrix} \begin{pmatrix} {}^{W}P \\ 1 \end{pmatrix} $$

and substituting in (6.2.5) yields

$$ \boldsymbol{p} = \frac{1}{z}\mathcal{M}\boldsymbol{P}, \quad \text{where} \quad \mathcal{M} = \mathcal{K}\begin{pmatrix} \mathcal{R} & \boldsymbol{t} \end{pmatrix}, \tag{6.2.6} $$

$\mathcal{R} = {}^{C}_{W}\mathcal{R}$ is a rotation matrix, $\boldsymbol{t} = {}^{C}O_W$ is a translation vector, and $\boldsymbol{P}$ denotes the vector of homogeneous coordinates of P in the frame (W).

We will often write the general perspective projection equation as $z\boldsymbol{p} = \mathcal{M}\boldsymbol{P}$, or even, slightly abusing the notation, as $\boldsymbol{p} = \mathcal{M}\boldsymbol{P}$, with the convention that a vector of homogeneous coordinates is only defined up to scale, and the actual image coordinates of the image point p being defined as u/w and v/w if $\boldsymbol{p} = (u, v, w)^T$. In this setting, the matrix $\mathcal{M}$ is also defined up to scale, with 11 free coefficients. Note that there are 5 intrinsic parameters (α, β, u_0, v_0 and θ) and 6 extrinsic parameters (the three angles defining $\mathcal{R}$ and the three coordinates of $\boldsymbol{t}$), which matches the number of independent coefficients of $\mathcal{M}$.

The matrix $\mathcal{M}$ can of course be rewritten explicitly as a function of the intrinsic and extrinsic parameters of the camera, namely

$$ \mathcal{M} = \begin{pmatrix} \alpha\boldsymbol{r}_1^T - \alpha\cot\theta\boldsymbol{r}_2^T + u_0\boldsymbol{r}_3^T & \alpha t_x - \alpha\cot\theta t_y + u_0 t_z \\ \dfrac{\beta}{\sin\theta}\boldsymbol{r}_2^T + v_0\boldsymbol{r}_3^T & \dfrac{\beta}{\sin\theta}t_y + v_0 t_z \\ \boldsymbol{r}_3^T & t_z \end{pmatrix}, \tag{6.2.7} $$

where $\boldsymbol{r}_1^T$, $\boldsymbol{r}_2^T$ and $\boldsymbol{r}_3^T$ denote the three rows of the matrix $\mathcal{R}$ and t_x, t_y and t_z are the coordinates of the vector $\boldsymbol{t}$ in the frame attached to the camera. If $\mathcal{R}$ is written as the product of three elementary rotations, the vectors $\boldsymbol{r}_i$ $(i = 1, 2, 3)$ can of course be written explicitly in terms of the corresponding three angles.

It is worth noting that the matrix $\mathcal{M}$ determines the coordinate vector $\boldsymbol{C}$ of the camera's optical center in the world coordinate system. Indeed, as shown in the exercises, $\boldsymbol{C}$ verifies

$$ \mathcal{M}\begin{pmatrix} \boldsymbol{C} \\ 1 \end{pmatrix} = 0. $$

(Intuitively this is rather obvious since the optical center is the only point whose image is not uniquely defined.) In particular, if $\mathcal{M} = \begin{pmatrix} \mathcal{A} & \boldsymbol{b} \end{pmatrix}$ then $\boldsymbol{C} = -\mathcal{A}^{-1}\boldsymbol{b}$.

6.2.3 A Characterization of Perspective Projection Matrices

We say that a 3×4 matrix that can be written (up to scale) as (6.2.6) or equivalently (6.2.7) for some set of intrinsic and extrinsic parameters is a *perspective projection*

matrix. It is of practical interest to put some restrictions on the intrinsic parameters of a camera since, as noted earlier, some of these parameters will be fixed and may be known. In particular, we will say that a 3×4 matrix is a *zero-skew perspective projection matrix* when it can be rewritten (up to scale) as (6.2.7) with $\theta = \pi/2$, and that it is a *perspective projection matrix with zero skew and unit aspect-ratio* when it can be rewritten (up to scale) as (6.2.7) with $\theta = \pi/2$ and $\alpha = \beta$. Of course, a camera with *known* non-zero skew and non-unit aspect-ratio can be transformed into a camera with zero skew and unit aspect-ratio by an appropriate change of image coordinates. Are arbitrary 3×4 matrices perspective projection matrices? The following theorem answers this question.

Theorem 2: *Let* $\mathcal{M} = (\mathcal{A} \quad \boldsymbol{b})$ *be a* 3×4 *matrix and let* $\boldsymbol{a}_i^T$ *(*$i = 1, 2, 3$*) denote the rows of the matrix* $\mathcal{A}$ *formed by the three leftmost columns of* $\mathcal{M}$*.*

- *A necessary and sufficient condition for* $\mathcal{M}$ *to be a perspective projection matrix is that* $\mathrm{Det}(\mathcal{A}) \neq 0$*.*
- *A necessary and sufficient condition for* $\mathcal{M}$ *to be a zero-skew perspective projection matrix is that* $\mathrm{Det}(\mathcal{A}) \neq 0$ *and*

$$(\boldsymbol{a}_1 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3) = 0.$$

- *A necessary and sufficient condition for* $\mathcal{M}$ *to be a perspective projection matrix with zero skew and unit aspect-ratio is that* $\mathrm{Det}(\mathcal{A}) \neq 0$ *and*

$$\begin{cases} (\boldsymbol{a}_1 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3) = 0, \\ (\boldsymbol{a}_1 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_1 \times \boldsymbol{a}_3) = (\boldsymbol{a}_2 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3). \end{cases}$$

The conditions of the theorem are clearly necessary: according to (6.2.6), we have $\mathcal{A} = \mathcal{K}\mathcal{R}$, thus the determinants of $\mathcal{A}$ and $\mathcal{K}$ are the same and $\mathcal{A}$ is non-singular. Further, a simple calculation shows that the rows of $\mathcal{K}\mathcal{R}$ in (6.2.7) satisfy the conditions of the theorem under the various assumptions imposed by its statement. Proofs that they are also sufficient can be found in [Faugeras, 1993; Heyden, 1995] and in the exercises. Note that when the conditions of the theorem are satisfied, there are exactly four sets of intrinsic and extrinsic parameters satisfying (6.2.7), see [Faugeras, 1993; Heyden, 1995] and Section 6.3.1.

6.3 Calibration Methods

This section introduces various techniques for estimating the intrinsic and extrinsic parameters of a camera, a process known as geometric camera calibration. Specifically, suppose that a camera observes n geometric features such as points or lines with known positions in some fixed world coordinate system. This section addresses the problem of (1) computing the perspective projection matrix $\mathcal{M}$ associated with the camera in this coordinate system, then (2) computing the intrinsic and extrinsic

parameters of the camera from this matrix. 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, and to conduct quantitative three-dimensional measurements from digitized pictures [Tsai, 1987a].

6.3.1 A Linear Approach to Camera Calibration

Let us first assume that our camera has non-zero skew. According to Theorem 2, the matrix $\mathcal{M}$ is not singular but otherwise arbitrary. If the 4-vectors $\boldsymbol{P}_i$ $(i = 1, \ldots, n)$ and $\boldsymbol{m}_j^T$ $(j = 1, 2, 3)$ denote respectively the homogeneous coordinate vectors of the points P_i and the rows of the matrix $\mathcal{M}$, we can express the position of the image of each point as

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

Collecting these constraints for all points yields a system of $2n$ homogeneous linear equations in the twelve coefficients of the matrix $\mathcal{M}$, namely,

$$\mathcal{P}\boldsymbol{m} = 0, \quad \text{where} \quad \mathcal{P} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{P}_1^T & \boldsymbol{0}^T & -u_1 \boldsymbol{P}_1^T \\ \boldsymbol{0}^T & \boldsymbol{P}_1^T & -v_1 \boldsymbol{P}_1^T \\ \ldots & \ldots & \ldots \\ \boldsymbol{P}_n^T & \boldsymbol{0}^T & -u_n \boldsymbol{P}_n^T \\ \boldsymbol{0}^T & \boldsymbol{P}_n^T & -v_n \boldsymbol{P}_n^T \end{pmatrix} \quad \text{and} \quad \boldsymbol{m} \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \\ \boldsymbol{m}_3 \end{pmatrix} = 0. \tag{6.3.1}$$

When $n \geq 6$, the system of equations (6.3.1) is in general *overconstrained*, i.e., there is no non-zero vector $\boldsymbol{m} \in \mathbb{R}^{12}$ that satisfies exactly these equations. On the other hand, the zero vector is always a solution. The *linear least-squares* literature, as briefly discussed in the insert next page, provides methods for computing the value of the *unit* vector $\boldsymbol{m}$ that minimizes $|\mathcal{P}\boldsymbol{m}|^2$. In particular, estimating the vector $\boldsymbol{m}$ (hence the matrix $\mathcal{M}$) reduces to computing the eigenvectors and eigenvalues of the 12×12 matrix $\mathcal{P}^T\mathcal{P}$.

Technique: Linear Least Squares Methods

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

$$\left\{\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1p}x_p = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2p}x_p = b_2 \\ \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{np}x_p = b_n \end{array}\right. \Leftrightarrow \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1p} \\ a_{21} & a_{22} & \ldots & a_{2p} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{np} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_p \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \ldots \\ b_n \end{pmatrix}. \tag{6.3.2}$$

Let $\mathcal{A}$ denote the $n \times p$ matrix with coefficients a_{ij}, and let $\boldsymbol{x} = (x_1, \ldots, x_p)^T$ and $\boldsymbol{b} = (b_1, \ldots, b_n)^T$. We know from linear algebra that (in general):

1. when $n < p$, there exists an $(p-n)$-dimensional vector space of vectors $\boldsymbol{x}$ that are solutions of (6.3.2);
2. when $n = p$, there is a unique solution;
3. when $n > p$, there is no solution.

This statement is true when the rank of $\mathcal{A}$ is maximal, i.e., equal to $\min(n, p)$ (this is what we mean by "in general"). When the rank is lower, there exists a higher-dimensional set of solutions.

Here we will consider the overconstrained case $n > p$. Since there is no exact solution in this case, we will content ourselves with finding the vector $\boldsymbol{x}$ that minimizes the error measure

$$E \stackrel{\text{def}}{=} \sum_{i=1}^{n} (a_{i1}x_1 + \ldots + a_{ip}x_p - b_i)^2 = |\mathcal{A}\boldsymbol{x} - \boldsymbol{b}|^2.$$

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

Now, we can write $E = |\boldsymbol{e}^T\boldsymbol{e}|$, where $\boldsymbol{e} \stackrel{\text{def}}{=} \mathcal{A}\boldsymbol{x} - \boldsymbol{b}$. To find the vector $\boldsymbol{x}$ minimizing E, we write that the derivatives of this error measure with respect to the coordinates x_i $(i = 1, \ldots, p)$ of $\boldsymbol{x}$ must be zero, i.e.,

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

But if the vectors $\boldsymbol{c}_i$ $(i = 1, \ldots, p)$ denote the columns of $\mathcal{A}$, we have

$$\frac{\partial \boldsymbol{e}}{\partial x_i} = \frac{\partial}{\partial x_i}\left[\begin{pmatrix} \boldsymbol{c}_1 & \ldots & \boldsymbol{c}_p \end{pmatrix} \begin{pmatrix} x_1 \\ \ldots \\ x_p \end{pmatrix} - \boldsymbol{b}\right] = \frac{\partial}{\partial x_i}(x_1\boldsymbol{c}_1 + \ldots + x_p\boldsymbol{c}_p - \boldsymbol{b}) = \boldsymbol{c}_i.$$

In particular, the constraint $\partial E/\partial x_i = 0$ implies that $\boldsymbol{c}_i^T(\mathcal{A}\boldsymbol{x} - \boldsymbol{b}) = 0$, and stacking the constraints associated with the p coordinates of $\boldsymbol{x}$ yields

$$\boldsymbol{0} = \begin{pmatrix} \boldsymbol{c}_1^T \\ \ldots \\ \boldsymbol{c}_p^T \end{pmatrix} (\mathcal{A}\boldsymbol{x} - \boldsymbol{b}) = \mathcal{A}^T(\mathcal{A}\boldsymbol{x} - \boldsymbol{b}) \Longleftrightarrow \mathcal{A}^T\mathcal{A}\boldsymbol{x} = \mathcal{A}^T\boldsymbol{b}.$$

The equations in this linear system are called the normal equations. When $\mathcal{A}$ has maximal rank p, the matrix $\mathcal{A}^T\mathcal{A}$ is easily shown to be invertible, and the solution of the least-squares problem can be written as

$$\boldsymbol{x} = \mathcal{A}^\dagger\boldsymbol{b} \quad \text{where} \quad \mathcal{A}^\dagger \stackrel{\text{def}}{=} [(\mathcal{A}^T\mathcal{A})^{-1}\mathcal{A}^T].$$

The $p \times p$ matrix $\mathcal{A}^\dagger$ is called the pseudoinverse of $\mathcal{A}$. It coincides with $\mathcal{A}^{-1}$ when the matrix $\mathcal{A}$ is square and non-singular. Linear least-squares problems can be solved without explicitly computing the pseudoinverse, using for example QR decomposition or singular value decomposition techniques, which are known to be better behaved numerically.

Let us now consider a slightly different problem, where we have a system of n *homogeneous* linear equations in p unknowns:

$$\left\{ \begin{array}{l} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1p}x_p = 0 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2p}x_p = 0 \\ \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{np}x_p = 0 \end{array} \right. \Leftrightarrow \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1p} \\ a_{21} & a_{22} & \ldots & a_{2p} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{np} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_p \end{pmatrix} = \mathbf{0}. \quad (6.3.3)$$

As before, we denote by $\mathcal{A}$ the $n \times p$ matrix with coefficients a_{ij}, and define $\boldsymbol{x} = (x_1, \ldots, x_p)^T$. When $n = p$ and the matrix $\mathcal{A}$ is non-singular, the system (6.3.3) admits as a unique solution $\boldsymbol{x} = \mathbf{0}$. Conversely, when $n \geq p$, non-trivial (i.e., non-zero) solutions may only exist when $\mathcal{A}$ is singular.

In this context, minimizing the error measure

$$E \stackrel{\text{def}}{=} |\mathcal{A}\boldsymbol{x}|^2 = \sum_{i=1}^{n} [\boldsymbol{a}_i \cdot \boldsymbol{x}]^2$$

only makes sense when some constraint is imposed on the solution $\boldsymbol{x}$ since $\boldsymbol{x} = \mathbf{0}$ yields the zero global minimum of E.

Since, by homogeneity, $E(\lambda\boldsymbol{x}) = \lambda^2 E(\boldsymbol{x})$, it is reasonable to minimize E under the constraint $|\boldsymbol{x}|^2 = 1$, which avoids the trivial solution and forces the uniqueness of the result.

Let us have another look at the error $E = \boldsymbol{x}^T(\mathcal{A}^T\mathcal{A})\boldsymbol{x}$. The $p \times p$ matrix $\mathcal{A}^T\mathcal{A}$ is symmetric positive semidefinite, and it can be diagonalized in an orthonormal basis of eigenvectors $\boldsymbol{e}_i$ $(i = 1, \ldots, p)$ associated with the eigenvalues $0 \leq \lambda_1 \leq \ldots \leq \lambda_p$. Now we can write any unit vector $\boldsymbol{x}$ as $\boldsymbol{x} = \mu_1\boldsymbol{e}_1 + \ldots + \mu_p\boldsymbol{e}_p$ for some μ_i $(i = 1, \ldots, p)$ such that $\mu_1^2 + \ldots + \mu_p^2 = 1$. We have

$$E(\boldsymbol{x}) - E(\boldsymbol{e}_1) = \boldsymbol{x}^T(\mathcal{A}^T\mathcal{A})\boldsymbol{x} - \boldsymbol{e}_1^T(\mathcal{A}^T\mathcal{A})\boldsymbol{e}_1 = \lambda_1\mu_1^2 + \ldots + \lambda_p\mu_p^2 - \lambda_1 \geq \lambda_1(\mu_1^2 + \ldots + \mu_p^2 - 1) = 0.$$

It follows that the unit vector $\boldsymbol{x}$ minimizing the least-squares error E is the eigenvector $\boldsymbol{e}_1$ associated with the minimum eigenvalue of $\mathcal{A}^T\mathcal{A}$ and the corresponding minimum value of E is λ_1.

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.

It should finally be noted that least-squares minimization admits a statistical interpretation in terms of maximum likelihood when the coordinates of the data points are modelled as random variables obeying a normal distribution. We will come back to this interpretation in a latter chapter.

In the noise-free case, there will be a unique solution for the matrix $\mathcal{M}$ as long at the rank of the matrix $\mathcal{P}$ is equal to its maximum value of 11 (the matrix $\mathcal{P}$ is singular since by construction $\mathcal{P}\boldsymbol{m} = 0$). A degenerate point configuration will correspond to the case where the matrix has rank 10 or less, or equivalently the nullspace of the matrix has dimension two or greater. Let us consider a vector $\boldsymbol{l}$ in the nullspace and introduce 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 $\boldsymbol{\nu} = (l_9, l_{10}, l_{11}, l_{12})^T$. Since $\boldsymbol{l}$ belongs to the nullspace we have

$$\mathbf{0} = \mathcal{P}\boldsymbol{l} = \begin{pmatrix} \boldsymbol{P}_1^T & \mathbf{0}^T & -u_1\boldsymbol{P}_1^T \\ \mathbf{0}^T & \boldsymbol{P}_1^T & -v_1\boldsymbol{P}_1^T \\ \dots & \dots & \dots \\ \boldsymbol{P}_n^T & \mathbf{0}^T & -u_n\boldsymbol{P}_n^T \\ \mathbf{0}^T & \boldsymbol{P}_n^T & -v_n\boldsymbol{P}_n^T \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\mu} \\ \boldsymbol{\nu} \end{pmatrix} = \begin{pmatrix} \boldsymbol{P}_1^T\boldsymbol{\lambda} - u_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \boldsymbol{P}_1^T\boldsymbol{\mu} - v_1\boldsymbol{P}_1^T\boldsymbol{\nu} \\ \dots \\ \boldsymbol{P}_n^T\boldsymbol{\lambda} - u_n\boldsymbol{P}_n^T\boldsymbol{\nu} \\ \boldsymbol{P}_n^T\boldsymbol{\mu} - v_n\boldsymbol{P}_n^T\boldsymbol{\nu} \end{pmatrix},$$

or, equivalently, taking into account the values of u_i and v_i yields

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

We finally obtain after clearing the denominators and rearranging the terms:

$$\begin{cases} \boldsymbol{P}_i^T(\boldsymbol{m}_3\boldsymbol{\lambda}^T - \boldsymbol{m}_1\boldsymbol{\nu}^T)\boldsymbol{P}_i = 0, \\ \boldsymbol{P}_i^T(\boldsymbol{m}_3\boldsymbol{\mu}^T - \boldsymbol{m}_2\boldsymbol{\nu}^T)\boldsymbol{P}_i = 0, \end{cases} \quad \text{for} \quad i = 1, \dots, n. \tag{6.3.4}$$

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 $(i = 1, \dots, n)$ all lie in some plane Π, or equivalently, $\boldsymbol{\Pi} \cdot \boldsymbol{P}_i = 0$ for some 4-vector $\boldsymbol{\Pi}$. Clearly, choosing $(\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu})$ equal to $(\boldsymbol{\Pi}, \mathbf{0}, \mathbf{0})$, $(\mathbf{0}, \boldsymbol{\Pi}, \mathbf{0})$, or $(\mathbf{0}, \mathbf{0}, \boldsymbol{\Pi})$, or any linear combination of these vectors will yield a solution of (6.3.4). 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 coplanar points should not be used in calibration tasks.

In general, for a given non-zero value of the vector $\boldsymbol{l}$, the points P_i that satisfy (6.3.4) must lie on the curve where the two quadric surfaces defined by the corresponding equations intersect. A closer look at (6.3.4) 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 of a twisted cubic curve Γ passing through the origin [Faugeras, 1993]. A twisted cubic is entirely determined by six points lying on it, and it follows that seven points chosen at random will not fall on Γ. Since, in addition, this curve passes through the origin, choosing $n \geq 6$ random points will in general guarantee that the matrix $\mathcal{P}$ has rank 11 and that the projection matrix can be recovered in a unique fashion.

Once the projection matrix $\mathcal{M}$ has been estimated, (6.2.7) can be used to recover the intrinsic and extrinsic parameters as follows. If we write as before $\mathcal{M} = (\mathcal{A} \quad \boldsymbol{b})$, we have

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

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

$$\begin{cases} \rho = \varepsilon / |\boldsymbol{a}_3|, \\ \boldsymbol{r}_3 = \rho \boldsymbol{a}_3, \\ u_0 = \rho^2 (\boldsymbol{a}_1 \cdot \boldsymbol{a}_3), \\ v_0 = \rho^2 (\boldsymbol{a}_2 \cdot \boldsymbol{a}_3), \end{cases} \tag{6.3.5}$$

where $\varepsilon = \mp 1$.

In addition, we have

$$\begin{cases} \rho^2 (\boldsymbol{a}_1 \times \boldsymbol{a}_3) = -\alpha \boldsymbol{r}_2 - \alpha \cot\theta \boldsymbol{r}_1, \\ \rho^2 (\boldsymbol{a}_2 \times \boldsymbol{a}_3) = \dfrac{\beta}{\sin\theta} \boldsymbol{r}_1, \end{cases} \quad \text{and} \quad \begin{cases} \rho^2 |\boldsymbol{a}_1 \times \boldsymbol{a}_3| = \dfrac{|\alpha|}{\sin\theta}, \\ \rho^2 |\boldsymbol{a}_2 \times \boldsymbol{a}_3| = \dfrac{|\beta|}{\sin\theta}, \end{cases} \tag{6.3.6}$$

since θ is always in the neighborhood of $\pi/2$ with a positive sine, and it follows that

$$\begin{cases} \cos\theta = -\varepsilon_u \varepsilon_v \dfrac{(\boldsymbol{a}_1 \times \boldsymbol{a}_3) \cdot (\boldsymbol{a}_2 \times \boldsymbol{a}_3)}{|\boldsymbol{a}_1 \times \boldsymbol{a}_3| |\boldsymbol{a}_2 \times \boldsymbol{a}_3|}, \\ \alpha = \varepsilon_u \rho^2 |\boldsymbol{a}_1 \times \boldsymbol{a}_3| \sin\theta, \\ \beta = \varepsilon_v \rho^2 |\boldsymbol{a}_2 \times \boldsymbol{a}_3| \sin\theta, \end{cases} \tag{6.3.7}$$

where $\varepsilon_u = \alpha/|\alpha|$ and $\varepsilon_v = \beta/|\beta|$.

We can now compute $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ from the second equation in (6.3.6) as

$$\begin{cases} \boldsymbol{r}_1 = \dfrac{\rho^2 \sin\theta}{\beta} (\boldsymbol{a}_2 \times \boldsymbol{a}_3) = \dfrac{1}{|\boldsymbol{a}_2 \times \boldsymbol{a}_3|} (\boldsymbol{a}_2 \times \boldsymbol{a}_3), \\ \boldsymbol{r}_2 = \boldsymbol{r}_3 \times \boldsymbol{r}_1. \end{cases} \tag{6.3.8}$$

Note that there are four possible choices for the matrix $\mathcal{R}$ depending on the values of ε and ε_v.

Finally, the translation parameters are recovered by writing

$$\rho \begin{pmatrix} \alpha t_x - \alpha \cot\theta t_y + u_0 t_z \\ \dfrac{\beta}{\sin\theta} t_y + v_0 t_z \\ t_z \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}.$$

6.3.2 Taking Radial Distortion into Account

We have assumed so far that our camera was equipped with a perfect lens. As shown in Chapter 1, real lenses suffer from a number of aberrations. In this section we follow Tsai [1987a] and show how to account for *radial distortion*, a type of aberration that depends on the distance between the imaged point and the optical axis and can be modelled as

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

where λ is a polynomial function of $\hat{r}^2 \stackrel{\text{def}}{=} \hat{u}^2 + \hat{v}^2$, i.e., $\lambda = 1 + \kappa_1\hat{r}^2 + \kappa_2\hat{r}^4 + \ldots$.

Geometrically, radial distortion changes the distance between the image center and the image point $\boldsymbol{p}$ but it does not affect the direction of the vector joining these two points. This is called the *radial alignment constraint* by Tsai, and it can be expressed algebraically by writing

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

This is a linear constraint on the vectors $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$. Given n fiducial points we obtain n equations in the eight coefficients of the vectors $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$, namely

$$\mathcal{Q}\boldsymbol{n} = 0, \quad \text{where} \quad \mathcal{Q} \stackrel{\text{def}}{=} \begin{pmatrix} v_1\boldsymbol{P}_1^T & -u_1\boldsymbol{P}_1^T \\ \ldots & \ldots \\ v_n\boldsymbol{P}_n^T & -u_n\boldsymbol{P}_n^T \end{pmatrix} \quad \text{and} \quad \boldsymbol{n} = \begin{pmatrix} \boldsymbol{m}_1 \\ \boldsymbol{m}_2 \end{pmatrix}. \tag{6.3.9}$$

Note the similarity with the previous case. When $n \geq 8$, the system of equations (6.3.9) is in general overconstrained, and a solution with unit norm can be found using linear least squares.

We can as before determine the degenerate point configurations for which the vectors $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ will not be uniquely determined. The matrix $\mathcal{Q}$ has at most rank 7 since the vector $\boldsymbol{n}$ is in its nullspace. More generally, let us consider a vector $\boldsymbol{l}$ in the nullspace and the vectors $\boldsymbol{\lambda} = (l_1, l_2, l_3, l_4)^T$ and $\boldsymbol{\mu} = (l_5, l_6, l_7, l_8)^T$; we have

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

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

$$\boldsymbol{P}_i^T(\boldsymbol{m}_2\boldsymbol{\lambda}^T - \boldsymbol{m}_1\boldsymbol{\mu}^T)\boldsymbol{P}_i = 0 \quad \text{for} \quad i = 1, \ldots, n. \tag{6.3.10}$$

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. 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 (6.3.10). The nullspace of $\mathcal{P}$ contains the three-dimensional vector space spanned by these vectors and $\boldsymbol{l}$. Thus, as before, coplanar points should not be used in calibration.

More generally, for a given value of $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, the points P_i will form a degenerate configuration when they lie on the quadric surface defined by (6.3.10). 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 $\boldsymbol{m}_1 \cdot \boldsymbol{P} = \boldsymbol{m}_2 \cdot \boldsymbol{P} = 0$, and it must therefore be either two planes, a cone, a hyperboloid of one sheet or a hyperbolic paraboloid. In any case, for a large enough number of points in general position, there will be a unique solution to our least-squares problem.

Once $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ have been estimated, we can as before define the corresponding values of $\boldsymbol{a}_1$, $\boldsymbol{a}_2$, b_1 and b_2 and we obtain the constraints

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

In this setting, there are more unknowns (nine, i.e., the three coefficients of the rotation matrix, the two magnifications α and β, the coordinates u_0 and v_0 of the image center, the skew angle θ and the scale factor ρ) than equations (six scalar constraints corresponding to the two vector equations above).

As noted in [Tsai, 1987a], however, when the principal point is known we can take $u_0 = v_0 = 0$ and $\rho = 1$, and we obtain

$$\begin{cases} |\boldsymbol{a}_1| = \dfrac{|\alpha|}{\sin\theta}, \\ |\boldsymbol{a}_2| = \dfrac{|\beta|}{\sin\theta}. \end{cases}$$

In particular we have

$$\begin{cases} \cos\theta = -\varepsilon_u \varepsilon_v \dfrac{\boldsymbol{a}_1 \cdot \boldsymbol{a}_2}{|\boldsymbol{a}_1||\boldsymbol{a}_2|}, \\ \alpha = \varepsilon_u |\boldsymbol{a}_1| \sin\theta, \\ \beta = \varepsilon_v |\boldsymbol{a}_2| \sin\theta, \end{cases}$$

where as before $\varepsilon_u = \alpha/|\alpha|$ and $\varepsilon_v = \beta/|\beta|$. It is now easy to compute $\boldsymbol{r}_1$, $\boldsymbol{r}_2$ and $\boldsymbol{r}_3 = \boldsymbol{r}_1 \times \boldsymbol{r}_2$, then, as before, the translation components.

Once the intrinsic and extrinsic parameters have been estimated, the radial distortion coefficients can themselves be recovered using linear least squares and the radial alignment constraint.

6.3.3 Using Straight Lines for Calibration

Points are not the only geometric image features that constrain the camera parameters. We characterize in this section the projection of straight lines into an image, and show how they can be used, in principle at least, to perform camera calibration. We must first recall some elementary notions of line geometry. Let us introduce the operator "$\wedge$" that associates with two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ in $\mathbb{R}^4$ their *exterior product* defined as the 6-vector

$$\boldsymbol{a} \wedge \boldsymbol{b} \stackrel{\text{def}}{=} \begin{pmatrix} a_1 b_2 - a_2 b_1 \\ a_1 b_3 - a_3 b_1 \\ a_1 b_4 - a_4 b_1 \\ a_2 b_3 - a_3 b_2 \\ a_2 b_4 - a_4 b_2 \\ a_3 b_4 - a_4 b_3 \end{pmatrix}.$$

Note the similarity with the cross-product operator that also associates with two vectors (3-vectors of course, instead of 4-vectors) $\boldsymbol{a}$ and $\boldsymbol{b}$ the vector formed by all the 2×2 minors of the matrix $(\boldsymbol{a}, \boldsymbol{b})$.

Let us assume a fixed coordinate system. Geometrically, the exterior product associates with the homogeneous coordinate vectors of two points A and B in $\mathbb{E}^3$ the vector $\boldsymbol{\Delta} = (\Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5, \Delta_6)^T$ of *Plücker coordinates* of the line Δ joining them. To gain a better intuitive understanding of the situation, let us denote by O the origin of the coordinate system and by H its projection onto Δ (Figure 6.9), and let us identify the vectors $\overrightarrow{OA}$ and $\overrightarrow{OB}$ with their *non-homogeneous* coordinate vectors. It is easy to verify analytically (see exercises) that $\overrightarrow{AB} = -(\Delta_3, \Delta_5, \Delta_6)^T$ and $\overrightarrow{OA} \times \overrightarrow{OB} = \overrightarrow{OH} \times \overrightarrow{AB} = (\Delta_4, -\Delta_2, \Delta_1)^T$.

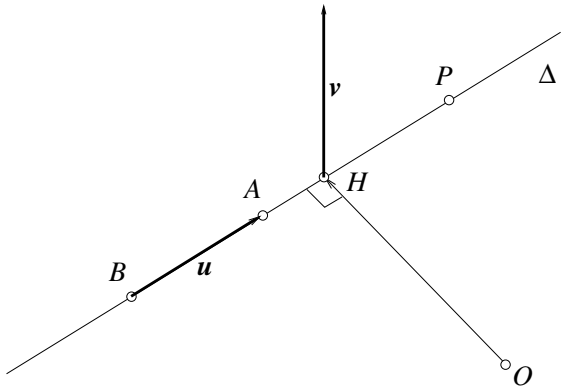

Figure 6.9. Geometric definition of line Plücker coordinates. In this figure $\boldsymbol{u} = (\Delta_3, \Delta_5, \Delta_6)^T$ and $\boldsymbol{v} = (\Delta_4, -\Delta_2, \Delta_1)^T$.

In turn, this implies that: (1) changing the position of A (or B) along Δ only changes the overall scale of $\boldsymbol{\Delta}$, so Plücker coordinates are homogeneous coordinates

only defined up to scale but otherwise independent from the choice of the points A and B along Δ; and (2) the Plücker coordinates of a line obey the quadratic constraint

$$\Delta_1\Delta_6 - \Delta_2\Delta_5 + \Delta_3\Delta_4 = 0. \tag{6.3.11}$$

It is also possible to define an inner product on the set of all lines by the formula

$$(\mathbf{\Delta}|\mathbf{\Delta}') \stackrel{\text{def}}{=} \Delta_1\Delta'_6 + \Delta_6\Delta'_1 - \Delta_2\Delta'_5 - \Delta_5\Delta'_2 + \Delta_3\Delta'_4 + \Delta_4\Delta'_3.$$

Clearly, a 6-vector $\mathbf{\Delta}$ represents a line if and only if $(\mathbf{\Delta}|\mathbf{\Delta}) = 0$, and it can also be shown that a necessary and sufficient condition for two lines to be coplanar is that $(\mathbf{\Delta}|\mathbf{\Delta}') = 0$.

Let us now follow Faugeras and Papadopoulo [1997] and show that the mapping between a line with Plücker coordinate vector $\mathbf{\Delta}$ and its image δ with homogeneous coordinates $\boldsymbol{\delta}$ can be represented by

$$\rho\boldsymbol{\delta} = \tilde{\mathcal{M}}\mathbf{\Delta}, \quad \text{where} \quad \tilde{\mathcal{M}} \stackrel{\text{def}}{=} \begin{pmatrix} (\boldsymbol{m}_2 \wedge \boldsymbol{m}_3)^T \\ (\boldsymbol{m}_3 \wedge \boldsymbol{m}_1)^T \\ (\boldsymbol{m}_1 \wedge \boldsymbol{m}_2)^T \end{pmatrix}, \tag{6.3.12}$$

$\boldsymbol{m}_1^T$, $\boldsymbol{m}_2^T$ and $\boldsymbol{m}_3^T$ denote as before the rows of $\mathcal{M}$ and ρ is an appropriate scale factor.

To prove this relation, let us consider a line Δ joining two points A and B, and denote by a and b the projections of these two points, with homogeneous coordinates $\boldsymbol{a} = \mathcal{M}\boldsymbol{a}$ and $\boldsymbol{b} = \mathcal{M}\boldsymbol{b}$. The points a and b lie on δ, thus $\boldsymbol{\delta} \cdot \boldsymbol{a} = \boldsymbol{\delta} \cdot \boldsymbol{b} = 0$. Hence, $\boldsymbol{\delta}$ (as an element of $\mathbb{R}^3$) is orthogonal to both $\mathcal{M}\boldsymbol{A}$ and $\mathcal{M}\boldsymbol{B}$ and must be parallel to their cross product. Thus we have

$$\rho\boldsymbol{\delta} = (\mathcal{M}\boldsymbol{A}) \times (\mathcal{M}\boldsymbol{B}) = \begin{pmatrix} (\boldsymbol{m}_2 \cdot \boldsymbol{A})(\boldsymbol{m}_3 \cdot \boldsymbol{B}) - (\boldsymbol{m}_3 \cdot \boldsymbol{A})(\boldsymbol{m}_2 \cdot \boldsymbol{B}) \\ (\boldsymbol{m}_3 \cdot \boldsymbol{A})(\boldsymbol{m}_1 \cdot \boldsymbol{B}) - (\boldsymbol{m}_1 \cdot \boldsymbol{A})(\boldsymbol{m}_3 \cdot \boldsymbol{B}) \\ (\boldsymbol{m}_1 \cdot \boldsymbol{A})(\boldsymbol{m}_2 \cdot \boldsymbol{B}) - (\boldsymbol{m}_2 \cdot \boldsymbol{A})(\boldsymbol{m}_1 \cdot \boldsymbol{B}) \end{pmatrix} \tag{6.3.13}$$

for some scale factor ρ.

Now, as shown in the exercises, the following identity holds for any 4-vectors $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$ and $\boldsymbol{d}$:

$$(\boldsymbol{a} \wedge \boldsymbol{b}) \cdot (\boldsymbol{c} \wedge \boldsymbol{d}) = (\boldsymbol{a} \cdot \boldsymbol{c})(\boldsymbol{b} \cdot \boldsymbol{d}) - (\boldsymbol{a} \cdot \boldsymbol{d})(\boldsymbol{b} \cdot \boldsymbol{c}).$$

Applying this identity to (6.3.13) yields

$$\rho\boldsymbol{\delta} = \begin{pmatrix} (\boldsymbol{m}_2 \wedge \boldsymbol{m}_3) \cdot (\boldsymbol{A} \wedge \boldsymbol{B}) \\ (\boldsymbol{m}_3 \wedge \boldsymbol{m}_1) \cdot (\boldsymbol{A} \wedge \boldsymbol{B}) \\ (\boldsymbol{m}_1 \wedge \boldsymbol{m}_2) \cdot (\boldsymbol{A} \wedge \boldsymbol{B}) \end{pmatrix},$$

and the result follows immediately.

The 3×6 matrix $\tilde{\mathcal{M}}$ is only defined up to scale with 17 coefficients. These parameters can be estimated as before via linear least squares (ignoring of course

the non-linear constraints imposed by the fact that the rows of $\tilde{\mathcal{M}}$ are Plücker coordinates) when $n \geq 9$ (eliminating ρ in (6.3.12) yields two independent constraints per line).

Once $\tilde{\mathcal{M}}$ is known, we can recover $\mathcal{M}$ as well through linear least squares. Indeed, it is easily shown (see exercises) that

$$\tilde{\mathcal{M}} = (\boldsymbol{c}_1 \times \boldsymbol{c}_2, \boldsymbol{c}_1 \times \boldsymbol{c}_3, \boldsymbol{c}_1 \times \boldsymbol{c}_4, \boldsymbol{c}_2 \times \boldsymbol{c}_3, \boldsymbol{c}_2 \times \boldsymbol{c}_4, \boldsymbol{c}_3 \times \boldsymbol{c}_4),$$

where the vectors $\boldsymbol{c}_i$ $(i = 1, \ldots, 4)$ are the columns of $\mathcal{M}$. We can thus estimate the vectors $\boldsymbol{c}_i$ $(i = 1, 2, 3, 4)$ using constraints such as

$$\begin{pmatrix} \tilde{\boldsymbol{c}}_{12}^T \\ \tilde{\boldsymbol{c}}_{13}^T \\ \tilde{\boldsymbol{c}}_{14}^T \end{pmatrix} \boldsymbol{c}_1 = 0, \quad \tilde{\boldsymbol{c}}_{12} \cdot \boldsymbol{c}_3 = \tilde{\boldsymbol{c}}_{23} \cdot \boldsymbol{c}_1,$$

where $\tilde{\boldsymbol{c}}_{ij}$ denotes the values of $\boldsymbol{c}_i \times \boldsymbol{c}_j$ $(1 \leq i < j \leq 4)$ stored in the columns of $\tilde{\mathcal{M}}$. Collecting the 20 different equations of this type obtained by permuting the appropriate subscripts yields a systems of linear equations in the coordinates of the vectors $\boldsymbol{c}_i$ that can be solved once again using linear least squares (at most 11 of these 20 equations are of course linearly independent in the noise-free case).

We leave it to the reader to characterize the degenerate line configurations for which this method fails.

6.3.4 Analytical Photogrammetry

We present in this section a non-linear approach to camera calibration that takes into account all the constraints associated with a camera. 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, etc. [Thompson *et al.*, 1966; Slama *et al.*, 1980]. 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 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*, while the extrinsinc parameters define its *exterior orientation*.

Let us write again the perspective projection equation as

$$\begin{cases} u = \dfrac{\boldsymbol{m}_1 \cdot \boldsymbol{P}}{\boldsymbol{m}_3 \cdot \boldsymbol{P}}, \\ v = \dfrac{\boldsymbol{m}_2 \cdot \boldsymbol{P}}{\boldsymbol{m}_3 \cdot \boldsymbol{P}}. \end{cases}$$

Let $\boldsymbol{\xi}$ denote the vector formed by all intrinsic and extrinsic parameters of a camera. We can explicitly parameterize the projection matrix $\mathcal{M}$ and its columns

$\boldsymbol{m}_i^T$ $(i = 1, 2, 3)$ by the vector $\boldsymbol{\xi}$ as in (6.2.7). In this setting, the problem of calibrating the camera reduces to minimizing the least-squares error

$$E(\boldsymbol{\xi}) = \sum_{i=1}^{n} [(u_i - \frac{\boldsymbol{m}_1(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i})^2 + (v_i - \frac{\boldsymbol{m}_2(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i}{\boldsymbol{m}_3(\boldsymbol{\xi}) \cdot \boldsymbol{P}_i})^2] \tag{6.3.14}$$

with respect to the coefficients $\boldsymbol{\xi}$.

Contrary to the cases studied so far, the dependency of each error term on the unknown parameters $\boldsymbol{\xi}$ is not linear. Instead, this dependency involves a combination of polynomial and trigonometric functions, and minimizing the overall error measure involves the use of *non-linear least squares* algorithms, as briefly discussed in the insert next page. These methods rely on the derivatives of the error function with respect to the unknown parameters to linearize the steps of the iterative minimization process. Non-linear least-squares techniques provide the means of computing a local minimum of E, and, when started with an appropriate initial guess, they will yield the global minimum as well.

Technique: Non-Linear Least Squares Methods

Let us consider a system of n non-linear equations in p unknowns:

$$\begin{cases} f_1(x_1, x_2, \ldots, x_p) = b_1, \\ f_2(x_1, x_2, \ldots, x_p) = b_2, \\ \ldots \\ f_n(x_1, x_2, \ldots, x_p) = b_n, \end{cases} \tag{6.3.15}$$

where f_i denotes, for $i = 1, \ldots, n$, a differentiable function from $\mathbb{R}^p$ to $\mathbb{R}$. When

$$f_i(x_1, x_2, \ldots, x_p) = a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ip}x_p,$$

we have of course exactly the same situation as in (6.3.2). In the general setting of *non-linear least squares*, the functions f_i can be arbitrarily non-linear. This time, we have (in general):

1. when $n < p$, there exists an $(p - n)$-dimensional *subspace of* $\mathbb{R}^p$ formed by the vectors $\boldsymbol{x}$ that are solutions of (6.3.15);
2. when $n = p$, there exists a *finite set* of solutions;
3. when $n > p$, there is no solution.

We have emphasized in this statement the main differences with the linear case: the dimension of the solution set will still be $p - n$ (in general) in the underconstrained case, but this set will not form a vector space anymore. Its structure will depend on the nature of the functions f_i. Likewise, there will be (in general) a finite number of solutions instead of a unique one in the case $n = p$. This time we will not go into the details of what it means for a family of functions f_i $(i = 1, \ldots, n)$ to satisfy the "general" conditions under which the above statement is true. Linear and polynomial functions, for example, do satisfy these conditions.

From now on we will consider the overconstrained case $n > p$ and minimize the error

$$E(\boldsymbol{x}) \stackrel{\text{def}}{=} \sum_{i=1}^{n} (f_i(\boldsymbol{x}) - b_i)^2.$$

The error function $E : \mathbb{R}^p \to \mathbb{R}^+$ may have many local minima, but it has (in general) a single global minimum. Except for the case of polynomial functions [Morgan, 1987], there is unfortunately no numerical method guaranteed to find this global minimum. Effective iterative methods (e.g., gradient descent) do exist for finding a local minimum of a non-linear function, and it can be hoped that these methods will find the global minimum when they start from a reasonable initial guess. We will give in the remaining of this note such a method specialized for non-linear least squares.

The idea is to linearize the process: indeed, any smooth function looks like its tangent in a (small) neighborhood of any of its points. Formally, this can be rewritten using a first-order Taylor expansion as

$$f_i(\boldsymbol{x} + \delta\boldsymbol{x}) = f_i(\boldsymbol{x}) + \delta x_1 \frac{\partial f_i}{\partial x_1}(\boldsymbol{x}) + \ldots + + \delta x_p \frac{\partial f_i}{\partial x_p}(\boldsymbol{x}) + O(|\delta\boldsymbol{x}|^2) \approx f_i(\boldsymbol{x}) + \nabla f_i(\boldsymbol{x}) \cdot \delta\boldsymbol{x}, \tag{6.3.16}$$

where $\nabla f_i(\boldsymbol{x}) \stackrel{\text{def}}{=} (\frac{\partial f_i}{\partial x_1}, \ldots, \frac{\partial f_i}{\partial x_p})^T$ is called the gradient of f_i at the point $\boldsymbol{x}$, and we have neglected the second-order term $O(|\delta\boldsymbol{x}|^2)$.

In the context of an iterative process, consider the problem of minimizing $E(\boldsymbol{x} + \delta\boldsymbol{x})$ with respect to $\delta\boldsymbol{x}$ for a given value of $\boldsymbol{x}$. Substituting (6.3.16) into (6.3.15) yields

$$E(\boldsymbol{x} + \delta\boldsymbol{x}) = \sum_{i=1}^{n} (f_i(\boldsymbol{x}) + \nabla f_i(\boldsymbol{x}) \cdot \delta\boldsymbol{x} - b_i)^2 = |\mathcal{J}\delta\boldsymbol{x} - \boldsymbol{c}|^2,$$

where

$$\mathcal{J} \stackrel{\text{def}}{=} \begin{pmatrix} \nabla f_1(\boldsymbol{x})^t \\ \ldots \\ \nabla f_n(\boldsymbol{x})^t \end{pmatrix} = \begin{pmatrix} \frac{\partial f_1}{x_1}(\boldsymbol{x}) & \ldots & \frac{\partial f_1}{x_p}(\boldsymbol{x}) \\ \ldots & \ldots & \ldots \\ \frac{\partial f_n}{x_1}(\boldsymbol{x}) & \ldots & \frac{\partial f_n}{x_p}(\boldsymbol{x}) \end{pmatrix} \text{ and } \boldsymbol{c} \stackrel{\text{def}}{=} \begin{pmatrix} b_1 \\ \ldots \\ b_n \end{pmatrix} - \begin{pmatrix} f_1(\boldsymbol{x}) \\ \ldots \\ f_n(\boldsymbol{x}) \end{pmatrix}.$$

At this point we are back in the linear least-squares setting, and the adjustment $\delta\boldsymbol{x}$ can be computed as $\delta\boldsymbol{x} = \mathcal{J}^\dagger \boldsymbol{c}$. In practice the process is started with some initial guess $\boldsymbol{x}_0$, and a few iterations of the form $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \delta\boldsymbol{x}$ are sufficient to find a local (and hopefully global) minimum of E. Variants of this method include the Levenberg-Marquardt algorithm, popular in computer vision circles.

In our setting, we must compute the derivatives of the image coordinates with respect to the camera parameters. If $\boldsymbol{p} = \mathcal{M}\boldsymbol{P}$ and $\boldsymbol{p} = (p, q, r)^T$, we have

$$\begin{cases} \dfrac{\partial u}{\partial \xi} = \dfrac{\partial}{\partial \xi}(\dfrac{p}{r}) = \dfrac{1}{r}\dfrac{\partial p}{\partial \xi} - \dfrac{p}{r^2}\dfrac{\partial r}{\partial \xi} = \dfrac{1}{r}(\dfrac{\partial}{\partial \xi}(\boldsymbol{m}_1 \cdot \boldsymbol{P}) - u\dfrac{\partial}{\partial \xi}(\boldsymbol{m}_3 \cdot \boldsymbol{P})), \\ \dfrac{\partial v}{\partial \xi} = \dfrac{\partial}{\partial \xi}(\dfrac{q}{r}) = \dfrac{1}{r}\dfrac{\partial q}{\partial \xi} - \dfrac{q}{r^2}\dfrac{\partial r}{\partial \xi} = \dfrac{1}{r}(\dfrac{\partial}{\partial \xi}(\boldsymbol{m}_2 \cdot \boldsymbol{P}) - v\dfrac{\partial}{\partial \xi}(\boldsymbol{m}_3 \cdot \boldsymbol{P})), \end{cases}$$

which is easily rewritten as

$$\begin{pmatrix} \dfrac{\partial u}{\partial \xi} \\ \dfrac{\partial v}{\partial \xi} \end{pmatrix} = \frac{1}{r}\begin{pmatrix} 1 & 0 & -u \\ 0 & 1 & -v \end{pmatrix}\frac{\partial \mathcal{M}}{\partial \xi}\boldsymbol{P}.$$

Note that u, v, r and $\boldsymbol{P}$ depend on the image considered, but that $\partial\mathcal{M}/\partial\xi$ only depends on the intrinsic and extrinsic parameters of the camera. Note also 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 earlier. Many other parameterizations are possible as well (Euler angles, matrix exponentials, quaternions, etc.), see [Craig, 1989; Faugeras, 1993; Haralick and Shapiro, 1992] for discussions.

6.4 Notes

The book by Craig [1989] offers a very good introduction to coordinate system representations and kinematics. A thorough presentation of camera models and the associated calibration methods can be found in the excellent book by Faugeras [1993]. The calibration technique for taking radial distortion into account is adapted from Tsai [1987b]. The line-based calibration technique is inspired by the characterization of line projections in terms of the exterior product introduced by Faugeras and Papadopoulo [1997]. The book of Haralick and Shapiro [1992] presents an excellent concise introduction to analytical photogrammetry. The *Manual of Photogrammetry* is of course the gold standard, and newcomers to this field (like the authors of this book) will probably find the ingenious mechanisms and rigorous methods described in the various editions of this book fascinating [Thompson *et al.*, 1966; Slama *et al.*, 1980]. We will come back to photogrammetry in the context of multiple images in Chapter 12.

6.5 Assignments

Exercises

1. Write formulas for the matrices ${}^A_B\mathcal{R}$ when (B) is deduced from (A) via a rotation of angle θ about the axes $\boldsymbol{i_A}$, $\boldsymbol{j}_A$ and $\boldsymbol{k}_A$ respectively.

2. Show that rotation matrices are characterized by the following properties: (1) the inverse of a rotation matrix is equal to its transpose, and (2) its determinant is equal to 1.

3. Show that the set of matrices associated with rigid transformations and equipped with the matrix product forms a group.

4. Let ${}^A\mathcal{T}$ denote the matrix associated with a rigid transformation $\mathcal{T}$ in the coordinate system (A), with

$$ {}^A\mathcal{T} = \begin{pmatrix} {}^A\mathcal{R} & {}^A\boldsymbol{T} \\ \mathbf{0} & 1 \end{pmatrix}. $$

   Construct the matrix ${}^B\mathcal{T}$ associated with $\mathcal{T}$ in the coordinate system (B) as a function of ${}^A\mathcal{T}$ and the rigid transformation separating (A) and (B).

5. Show that if the coordinate system (B) is obtained by applying to the coordinate system (A) the transformation associated with the 4×4 matrix $\mathcal{T}$, then ${}^BP = \mathcal{T}^{-1}{}^AP$.

6. Show that the rotation of angle θ about the $\boldsymbol{k}$ axis of the frame (F) can be represented by

$$ {}^FP' = \mathcal{R} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} {}^FP. $$

7. Show that the change of coordinates associated with a rigid transformation preserves distances and angles.

8. Show that when the camera coordinate system is skewed and the angle θ between the two image axes is not equal to 90 degrees, then (6.2.3) transforms into (6.2.4).

9. Let $\boldsymbol{C}$ denote the coordinate vector of the optical center of a camera in some reference frame, and let $\mathcal{M}$ denote the corresponding perspective projection matrix. Show that

$$ \mathcal{M} \begin{pmatrix} \boldsymbol{C} \\ 1 \end{pmatrix} = 0. $$

10. Show that the conditions of Theorem 2 are necessary.

11. Show that the conditions of Theorem 2 are sufficient. Note that the statement of this theorem is a bit different from the corresponding theorems in [Faugeras, 1993; Heyden, 1995], where the condition $\mathrm{Det}(\mathcal{A}) \neq 0$ is replaced by $\boldsymbol{a}_3 \neq 0$. But of course $\mathrm{Det}(\mathcal{A}) \neq 0$ implies $\boldsymbol{a}_3 \neq 0$.

12. Consider some coordinate system and the Plücker coordinate vector $\boldsymbol{\Delta}$ of the line Δ passing through the points A and B. Show that if O denotes the origin of the coordinate system and H denotes its projection onto Δ, then $\overrightarrow{AB} = -(\Delta_3, \Delta_5, \Delta_6)$ and $\overrightarrow{OA} \times \overrightarrow{OB} = \overrightarrow{OH} \times \overrightarrow{AB} = (\Delta_4, -\Delta_2, \Delta_1)^T$.

13. Show analytically that the following identity holds for any 4-vectors $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$ and $\boldsymbol{d}$:
$$(\boldsymbol{a} \wedge \boldsymbol{b}) \cdot (\boldsymbol{c} \wedge \boldsymbol{d}) = (\boldsymbol{a} \cdot \boldsymbol{c})(\boldsymbol{b} \cdot \boldsymbol{d}) - (\boldsymbol{a} \cdot \boldsymbol{d})(\boldsymbol{b} \cdot \boldsymbol{c}).$$

14. Show that
$$\tilde{\mathcal{M}} = (\boldsymbol{c}_1 \times \boldsymbol{c}_2, \boldsymbol{c}_1 \times \boldsymbol{c}_3, \boldsymbol{c}_1 \times \boldsymbol{c}_4, \boldsymbol{c}_2 \times \boldsymbol{c}_3, \boldsymbol{c}_2 \times \boldsymbol{c}_4, \boldsymbol{c}_3 \times \boldsymbol{c}_4) \quad \text{where} \quad \mathcal{M} = (\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3, \boldsymbol{c}_4).$$

Programming Assignments

Note: the assignments below require routines for solving square and overdetermined linear systems. An extensive set of such routines is available in MATLAB as well as in public-domain libraries such as LINPACK and LAPACK that can be downloaded from the Netlib repository (`http://www.netlib.org/`). Data for these assignments will be available in the CD companion to this book.

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

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