Chapter 3

MULTIPLE VIEW GEOMETRY

Anders Heyden

and Marc Pollefeys

3.1 Introduction

There exist intricate geometric relations between multiple views of a 3D scene. These relations are related to the camera motion and calibration as well as to the scene structure. In this chapter, we introduce these concepts and discuss how they can be applied to recover 3D models from images.

In Section 3.2, a rather thorough description of projective geometry is given. Section 3.3 gives a short introduction to tensor calculus, and Section 3.4 describes in detail the camera model used. In Section 3.5, a modern approach to multiple view geometry is presented and in Section 3.6 simple structure and motion algorithms are presented. In Section 3.7, more advanced algorithms are presented that are suited for automatic processing on real image data. Section 3.8 discusses the possibility of calibrating the camera from images. Section 3.9 describes how the depth can be computed for most image pixels, and Section 3.10 presents how the results of the previous sections can be combined to yield 3D models, render novel views, or combine real and virtual elements in video.

3.2 Projective Geometry

Projective geometry is a fundamental tool for dealing with structure from motion problems in computer vision, especially in multiple view geometry. The main reason is that the image formation process can be regarded as a projective transformation from a 3D to a 2D projective space. It is also a fundamental tool for dealing with autocalibration problems and examining critical configurations and critical motion sequences.

This section deals with the fundamentals of projective geometry, including the definitions of projective spaces, homogeneous coordinates, duality, projective transformations, and affine and Euclidean embeddings. For a traditional approach to projective geometry, see [9], and for more modern treatments, see [14, 15, 24].

3.2.1 The central perspective transformation

We start the introduction of projective geometry by looking at a central perspective transformation, which is very natural from an image formation point of view; see Figure 3.1.

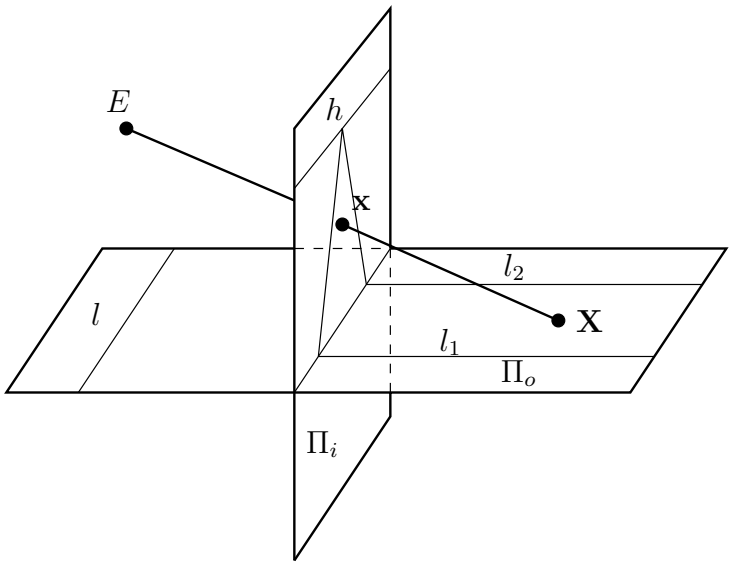

Figure 3.1. A central perspective transformation.

Definition 1. A **central perspective transformation** maps points, $\mathbf{X}$, on the object plane, Π_o, to points on the image plane Π_i, by intersecting the line through E, called the **center**, and $\mathbf{X}$ with Π_i. ■

We can immediately see the following properties of the planar perspective transformation:

- All points on Π_o map to points on Π_i except for points on l, where l is defined as the intersection of Π_o with the plane incident with E and parallel with Π_i.
- All points on Π_i are images of points on Π_o except for points on h, called the **horizon**, where h is defined as the intersection of Π_i with the plane incident with E and parallel with Π_o.
- Lines in Π_o are mapped to lines in Π_i.
- The images of parallel lines intersect in a point on the horizon, see l_1 and l_2 in Figure 3.1 for an example.
- In the limit where the point E moves infinitely far away, the planar perspective transformation turns into a parallel projection.

Identify the planes Π_o and Π_i with $\mathbb{R}^2$, with a standard cartesian coordinate system $O\mathbf{e}_1\mathbf{e}_2$ in Π_o and Π_i respectively. The central perspective transformation is *nearly* a bijective transformation between Π_o and Π_i, that is, from $\mathbb{R}^2$ to $\mathbb{R}^2$. The problem is the lines $l \in \Pi_o$ and $h \in \Pi_i$. If we remove these lines, we obtain a bijective transformation between $\mathbb{R}^2 \setminus l$ and $\mathbb{R}^2 \setminus h$, but this is not the path that we will follow. Instead, we extend each $\mathbb{R}^2$ with an extra line defined as the images of points on h and points that map to l in the natural way, i.e. maintaining continuity. Thus by adding one artificial line to each plane, it is possible to make the central perspective transformation bijective from ($\mathbb{R}^2$ + an extra line) to ($\mathbb{R}^2$ + an extra line). These extra lines correspond naturally to directions in the other plane, such as the images of the lines l_1 and l_2 intersects in a point on h corresponding to the direction of l_1 and l_2. The intersection point on h can be viewed as the limit of images of a point on l_1 moving infinitely far away. Inspired by this observation, we make the following definition:

Definition 2. Consider the set L of all lines parallel to a given line l in $\mathbb{R}^2$ and assign a point to each such set, $\mathbf{p}_{\text{ideal}}$, called an **ideal point** or **point at infinity**; see Figure 3.2. ■

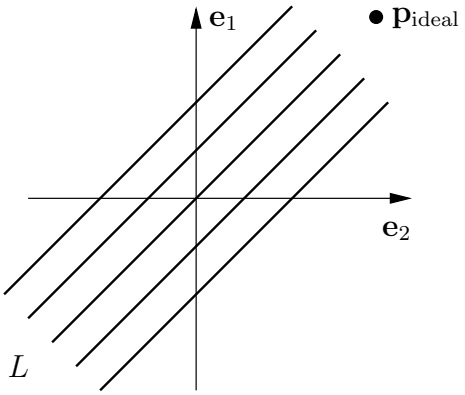

Figure 3.2. The point at infinity corresponding to the set of lines L.

3.2.2 Projective spaces

We are now ready to make a preliminary definition of the 2D projective space, namely, the projective plane.

Definition 3. The **projective plane**, $\mathbb{P}^2$, is defined according to

$$\mathbb{P}^2 = \mathbb{R}^2 \cup \{\text{ideal points}\}.$$

■

Definition 4. The **ideal line**, l_∞ or **line at infinity** in $\mathbb{P}^2$ is defined according to

$$l_\infty = \{\text{ideal points}\}.$$

■

The following constructions could easily be made in $\mathbb{P}^2$:

1. Two different points define a line (called the **join** of the points),
2. Two different lines intersect in a point,

with obvious interpretations for ideal points and the ideal line, for example, the line defined by an ordinary point and an ideal point is the line incident with the ordinary point with the direction given by the ideal point. Similarly we define

Definition 5. The **projective line**, $\mathbb{P}^1$, is defined according to

$$\mathbb{P}^1 = \mathbb{R}^1 \cup \{\text{ideal point}\}.$$

■

Observe that the projective line contains only one ideal point, which could be regarded as the point at infinity.

In order to define 3D projective space, $\mathbb{P}^3$, we start with $\mathbb{R}^3$ and assign an ideal point to each set of parallel lines; that is, to each direction.

Definition 6. The **projective space**, $\mathbb{P}^3$, is defined according to

$$\mathbb{P}^3 = \mathbb{R}^3 \cup \{\text{ideal points}\}.$$

■

Observe that the ideal points in $\mathbb{P}^3$ constitutes a 2D manifold, which motivates the following definition.

Definition 7. The ideal points in $\mathbb{P}^3$ build up a plane, called the **ideal plane** or **plane at infinity**, also containing ideal lines. ■

The plane at infinity contains lines, again called lines at infinity. Every set of parallel planes in $\mathbb{R}^3$ defines an ideal line and all ideal lines build up the ideal plane. A lot of geometrical constructions can be made in $\mathbb{P}^3$. For example,

1. Two different points define a line (called the **join** of the two points).
2. Three different points define a plane (called the **join** of the three points).
3. Two different planes intersect in a line.
4. Three different planes intersect in a point.

3.2.3 Homogeneous coordinates

It is often advantageous to introduce coordinates in the projective spaces, called analytic projective geometry. Introduce a cartesian coordinate system, $O\mathbf{e}_x\mathbf{e}_y$ in $\mathbb{R}^2$, and define the line $l : y = 1$; see Figure 3.3. We make the following simple observations:

Lemma 1. *The vectors* (p_1, p_2) *and* (q_1, q_2) *determine the same line through the origin iff*

$$(p_1, p_2) = \lambda(q_1, q_2), \quad \lambda \neq 0.$$

Proposition 1. *Every line,* $l_p = (p_1, p_2)t$, $t \in \mathbb{R}$, *through the origin, except for the x-axis, intersect the line* l *in one point,* $\mathbf{p}$.

We can now make the following definitions:

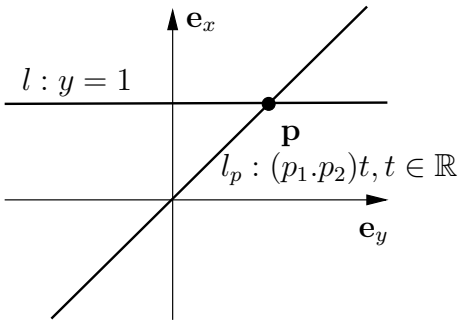

Figure 3.3. Definition of homogeneous coordinates in $\mathbb{P}^1$.

Definition 8. The pairs of numbers (p_1, p_2) and (q_1, q_2) are said to be **equivalent** if

$$(p_1, p_2) = \lambda(q_1, q_2), \quad \lambda \neq 0.$$

We write

$$(p_1, p_2) \sim (q_1, q_2).$$

■

There is a one-to-one correspondence between lines through the origin and points on the line l if we add an extra point on the line, corresponding to the line $x = 0$; that is, the direction $(1, 0)$. By identifying the line l augmented with this extra point, corresponding to the point at infinity, p_∞, with $\mathbb{P}^1$, we can make the following definitions:

Definition 9. The **1D projective space**, $\mathbb{P}^1$, consists of pairs of numbers (p_1, p_2) (under the equivalence above), where $(p_1, p_2) \neq (0, 0)$. The pair (p_1, p_2) is called **homogeneous coordinates** for the corresponding point in $\mathbb{P}^1$. ■

Theorem 1. *There is a natural division of* $\mathbb{P}^1$ *into two disjoint subsets*

$$\mathbb{P}^1 = \{(p_1, 1) \in \mathbb{P}^1\} \cup \{(p_1, 0) \in \mathbb{P}^1\},$$

corresponding to ordinary points and the ideal point.

The introduction of homogeneous coordinates can easily be generalized to $\mathbb{P}^2$ and $\mathbb{P}^3$ using three and four homogeneous coordinates respectively. In the case of $\mathbb{P}^2$ fix a cartesian coordinate system, $O\mathbf{e}_x\mathbf{e}_y\mathbf{e}_z$ in $\mathbb{R}^3$ and define

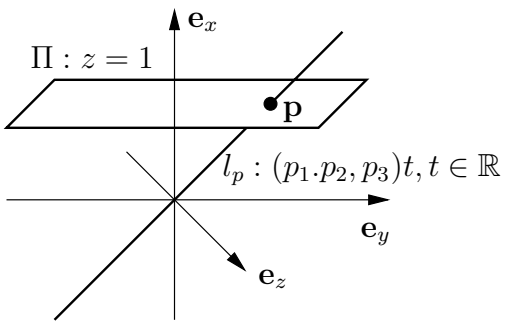

Figure 3.4. Definition of homogeneous coordinates in $\mathbb{P}^2$.

the plane $\Pi : z = 1$; see Figure 3.4. The vectors (p_1, p_2, p_3) and (q_1, q_2, q_3) determines the same line through the origin iff

$$(p_1, p_2, p_3) = \lambda(q_1, q_2, q_3), \quad \lambda \neq 0.$$

Every line through the origin, except for those in the x-y-plane, intersect the plane Π in one point. Again, there is a one-to-one correspondence between lines through the origin and points on the plane Π if we add an extra line corresponding to the lines in the plane $z = 0$, that is, the line at infinity, l_∞, built up by lines of the form $(p_1, p_2, 0)$. We can now identify the plane Π augmented with this extra line, corresponding to the points at infinity, l_∞, with $\mathbb{P}^2$.

Definition 10. The pairs of numbers (p_1, p_2, p_3) and (q_1, q_2, q_3) are said to be equivalent iff

$$(p_1, p_2, p_3) = \lambda(q_1, q_2, q_3), \quad \lambda \neq 0 \quad \text{written} \quad (p_1, p_2, p_3) \sim (q_1, q_2, q_3).$$

■

Definition 11. The 2D projective space $\mathbb{P}^2$ consists of all triplets of numbers $(p_1, p_2, p_3) \neq (0, 0, 0)$. The triplet (p_1, p_2, p_3) is called homogeneous coordinates for the corresponding point in $\mathbb{P}^2$. ■

Theorem 2. *There is a natural division of* $\mathbb{P}^2$ *into two disjoint subsets*

$$\mathbb{P}^2 = \{(p_1, p_2, 1) \in \mathbb{P}^2\} \cup \{(p_1, p_2, 0) \in \mathbb{P}^2\},$$

corresponding to ordinary points and ideal points (or points at infinity).

The same procedure can be carried out to construct $\mathbb{P}^3$ (and even $\mathbb{P}^n$ for any $n \in \mathbb{N}$), but it is harder to visualize, since we have to start with $\mathbb{R}^4$.

Definition 12. The **3D (nD) projective space** $\mathbb{P}^3$ ($\mathbb{P}^n$) is defined as the set of 1D linear subspaces in a vector space, $\mathbb{V}$ (usually $\mathbb{R}^4$ ($\mathbb{R}^{n+1}$)) of dimension 4 ($n+1$). Points in $\mathbb{P}^3$ ($\mathbb{P}^n$) are represented using **homogeneous coordinates** by vectors $(p_1, p_2, p_3, p_4) \neq (0, 0, 0, 0)$ ($(p_1, \ldots, p_{n+1}) \neq (0, \ldots, 0)$), where two vectors represent the same point iff they differ by a global scale factor. There is a natural division of $\mathbb{P}^3$ ($\mathbb{P}^n$) into two disjoint subsets

$$\mathbb{P}^3 = \{(p_1, p_2, p_3, 1) \in \mathbb{P}^3\} \cup \{(p_1, p_2, p_3, 0) \in \mathbb{P}^3\}$$
$$(\mathbb{P}^n = \{(p_1, \ldots, p_n, 1) \in \mathbb{P}^n\} \cup \{(p_1, \ldots, p_n, 0) \in \mathbb{P}^n\},$$

corresponding to ordinary points and ideal points (or points at infinity). ■

Finally, geometrical entities are defined similarly in $\mathbb{P}^3$.

3.2.4 Duality

Remember that a line in $\mathbb{P}^2$ is defined by two points $\mathbf{p}_1$ and $\mathbf{p}_2$ according to

$$l = \{\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{P}^2 \mid \mathbf{x} = t_1\mathbf{p}_1 + t_2\mathbf{p}_2, \quad (t_1, t_2) \in \mathbb{R}^2\}.$$

Observe that since (x_1, x_2, x_3) and $\lambda(x_1, x_2, x_3)$ represents the same point in $\mathbb{P}^2$, the parameters (t_1, t_2) and $\lambda(t_1, t_2)$ gives the same point. This gives the equivalent definition:

$$l = \{\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{P}^2 \mid \mathbf{x} = t_1\mathbf{p}_1 + t_2\mathbf{p}_2, \quad (t_1, t_2) \in \mathbb{P}^1\}.$$

By eliminating the parameters t_1 and t_2, we could also write the line in the form

$$l = \{\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{P}^2 \mid n_1x_1 + n_2x_2 + n_3x_3 = 0\}, \tag{3.1}$$

where the normal vector, $\mathbf{n} = (n_1, n_2, n_3)$, could be calculated as $\mathbf{n} = \mathbf{p}_1 \times \mathbf{p}_2$. Observe that if (x_1, x_2, x_3) fulfills (3.1), then $\lambda(x_1, x_2, x_3)$ also fulfills (3.1), and that if the line, l, is defined by (n_1, n_2, n_3), then the same line is defined by $\lambda(n_1, n_2, n_3)$, which means that $\mathbf{n}$ could be considered as an element in $\mathbb{P}^2$.

The line equation in (3.1) can be interpreted in two different ways; see Figure 3.5:

- Given $\mathbf{n} = (n_1, n_2, n_3)$, the points $\mathbf{x} = (x_1, x_2, x_3)$ that fulfill (3.1) constitute the line defined by $\mathbf{n}$.

– Given $\mathbf{x} = (x_1, x_2, x_3)$, the lines $\mathbf{n} = (n_1, n_2, n_3)$ that fulfill (3.1) constitute the lines coincident by $\mathbf{x}$.

Definition 13. The set of lines incident with a given point $\mathbf{x} = (x_1, x_2, x_3)$ is called a **pencil** of lines. ■

In this way, there is a one-to-one correspondence between points and lines in $\mathbb{P}^2$ given by

$$\mathbf{x} = (a, b, c) \quad \leftrightarrow \quad \mathbf{n} = (a, b, c),$$

as illustrated in Figure 3.5.

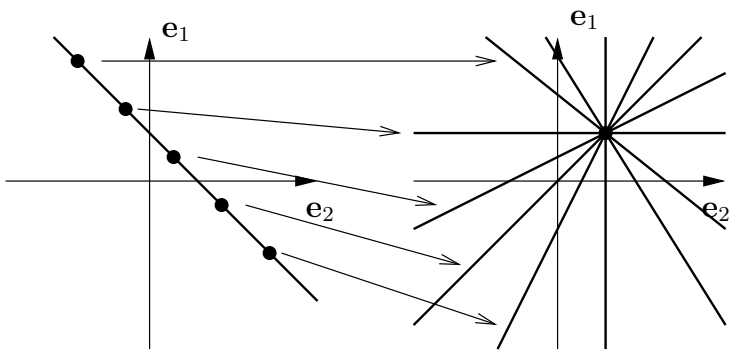

Figure 3.5. Duality of points and lines in $\mathbb{P}^2$.

Similarly, there exists a duality between points and planes in $\mathbb{P}^3$. A plane π in $\mathbb{P}^3$ consists of the points $\mathbf{x} = (x_1, x_2, x_3, x_4)$ that fulfill the equation

$$\pi = \{\mathbf{x} = (x_1, x_2, x_3, x_4) \in \mathbb{P}^3 \mid n_1x_1 + n_2x_2 + n_3x_3 + n_4x_4 = 0\}, \quad (3.2)$$

where $\mathbf{n} = (n_1, n_2, n_3, n_4)$ defines the plane. From (3.2) a similar argument leads to a duality between planes and points in $\mathbb{P}^3$. The following theorem is fundamental in projective geometry:

Theorem 3. *Given a statement valid in a projective space, the dual to that statement is also valid, where the dual is obtained by interchanging entities with their duals, intersection with join, and so on.*

For instance, a line in $\mathbb{P}^3$ could be defined as the join of two points. Thus, the dual to a line is the intersection of two planes, which again is a line; that is, the dual to a line in $\mathbb{P}^3$ is a line. We say that lines are **self-dual**. A line in $\mathbb{P}^3$ defined as the join of two points, $\mathbf{p}_1$ and $\mathbf{p}_2$, as in

$$l = \{\mathbf{x} = (x_1, x_2, x_3, x_4) \in \mathbb{P}^3 \mid \mathbf{x} = t_1\mathbf{p}_1 + t_2\mathbf{p}_2, \quad (t_1, t_2) \in \mathbb{P}^1\}$$

is said to be given in **parametric form** and (t_1, t_2) can be regarded as homogeneous coordinates on the line. A line in $\mathbb{P}^3$ defined as the intersection of two planes, π and μ, consisting of the common points to the pencil of planes in

$$l : \{s\pi + t\mu \mid (s,t) \in \mathbb{P}^1\}$$

is said to be given in **intersection form**.

Definition 14. A **conic**, c, in $\mathbb{P}^2$ is defined as

$$c = \{\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{P}^2 \mid \mathbf{x}^T C \mathbf{x} = 0\}, \tag{3.3}$$

where C denotes a 3×3 matrix. If C is nonsingular, the conic is said to be **proper**; otherwise, it is said to be **degenerate**. ■

The dual to a general curve in $\mathbb{P}^2$ ($\mathbb{P}^3$) is defined as the set of tangent lines (tangent planes) to the curve.

Theorem 4. *The dual, c^*, to a conic c: $\mathbf{x}^T C \mathbf{x}$ is the set of lines*

$$\{\mathbf{l} = (l_1, l_2, l_3) \in \mathbb{P}^2 \mid \mathbf{l}^T C' \mathbf{l} = 0\},$$

where $C' = C^{-1}$.

3.2.5 Projective transformations

The central perspective transformation in Figure 3.1 is an example of a projective transformation. The general definition is as follows:

Definition 15. A **projective transformation** from $\mathbf{p} \in \mathbb{P}^n$ to $\mathbf{p}' \in \mathbb{P}^m$ is defined as a linear transformation in homogeneous coordinates:

$$\mathbf{x}' \sim H\mathbf{x}, \tag{3.4}$$

where $\mathbf{x}$ and $\mathbf{x}'$ denote homogeneous coordinates for $\mathbf{p}$ and $\mathbf{p}'$ respectively, and H denote a $(m+1) \times (n+1)$ matrix of full rank. ■

All projective transformations form a group, denoted $\mathcal{G}_P$. For example, a projective transformation from $\mathbf{x} \in \mathbb{P}^2$ to $\mathbf{y} \in \mathbb{P}^2$ is given by

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \sim H \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix},$$

where H denote a nonsingular 3×3 matrix. Such a projective transformation from $\mathbb{P}^2$ to $\mathbb{P}^2$ is usually called a **homography**.

It is possible to embed an affine space in the projective space by a simple construction:

Definition 16. The subspaces

$$\mathcal{A}_i^n = \{ (x_1, x_2, \ldots, x_{n+1}) \in \mathbb{P}^n \mid x_i \neq 0 \}$$

of $\mathbb{P}^n$ are called **affine pieces** of $\mathbb{P}^n$. The plane $H_i : x_i = 0$ is called the **plane at infinity**, corresponding to the affine piece $\mathcal{A}_i^n$. Usually, $i = n+1$ is used and called the **standard affine piece**, and in this case the plane at infinity is denoted H_∞. ■

We can now identify points in $\mathbb{A}^n$ with points, $\mathbf{x}$, in $\mathcal{A}_i^n \subset \mathbb{P}^n$, by

$$\mathbb{P}^n \ni (x_1, x_2, \ldots, x_n, x_{n+1}) \sim (y_1, y_2, \ldots, y_n, 1) \equiv (y_1, y_2, \ldots, y_n) \in \mathbb{A}^n.$$

There is even an affine structure in this affine piece, given by the following proposition:

Proposition 2. *The subgroup, $\mathcal{H}$, of projective transformations, $\mathcal{G}_P$, that preserves the plane at infinity consists exactly of the projective transformations of the form (3.4), with*

$$H = \begin{bmatrix} A_{n\times n} & b_{n\times 1} \\ 0_{1\times n} & 1 \end{bmatrix},$$

where the indices denote the sizes of the matrices.

We can now identify the affine transformations in $\mathbb{A}$ with the subgroup $\mathcal{H}$:

$$\mathbb{A} \ni \mathbf{x} \mapsto A\mathbf{x} + b \in \mathbb{A},$$

which gives the affine structure in $\mathcal{A}_i^n \subset \mathbb{P}^n$.

Definition 17. When a plane at infinity has been chosen, two lines are said to be **parallel** if they intersect at a point at the plane at infinity. ■

We may even go one step further and define a Euclidean structure in $\mathbb{P}^n$.

Definition 18. The (singular, complex) conic, Ω, in $\mathbb{P}^n$ defined by

$$x_1^2 + x_1^2 + \ldots + x_n^2 = 0 \quad \text{and} \quad x_{n+1} = 0$$

is called the **absolute conic**. ■

Observe that the absolute conic is located at the plane at infinity, it contains only complex points, and it is singular.

Lemma 2. *The dual to the absolute conic, denoted Ω', is given by the set of planes*

$$\Omega' = \{\Pi = (\Pi_1, \Pi_2, \ldots \Pi_{n+1}) \mid \Pi_1^2 + \ldots + \Pi_n^2 = 0.$$

In matrix form, Ω' can be written as $\Pi^T C' \Pi = 0$ with

$$C' = \begin{bmatrix} I_{n\times n} & 0_{n\times 1} \\ 0_{1\times n} & 0 \end{bmatrix}.$$

Proposition 3. *The subgroup, $\mathcal{K}$, of projective transformations, $\mathcal{G}_P$, that preserves the absolute conic consists exactly of the projective transformations of the form (3.4), with*

$$H = \begin{bmatrix} cR_{n\times n} & t_{n\times 1} \\ 0_{1\times n} & 1 \end{bmatrix},$$

where $0 \neq c \in \mathbb{R}$ and R denote an orthogonal matrix; that is, $RR^T = R^T R = I$.

Observe that the corresponding transformation in the affine space $\mathbb{A} = \mathcal{A}_n^{n+1}$ can be written as

$$\mathbb{A} \ni \mathbf{x} \mapsto cR\mathbf{x} + t \in \mathbb{A},$$

which is a similarity transformation. This means that we have a Euclidean structure (to be precise, a similarity structure) in $\mathbb{P}^n$ given by the absolute conic.

3.3 Tensor Calculus

Tensor calculus is a natural tool to use when the objects at study are expressed in a specific coordinate system but have physical properties that are independent on the chosen coordinate system. Another advantage is that it gives a simple and compact notation and the rules for tensor algebra make it easy to remember even quite complex formulas. For a more detailed treatment, see [58], and for an engineering approach, see [42].

In this chapter, a simple definition of a tensor and the basic rules for manipulating tensors are given. We start with a straightforward definition:

Definition 19. An **affine tensor** is an object in a linear space, $\mathcal{V}$, that consists of a collection of numbers that are related to a specific choice of coordinate system in $\mathcal{V}$, indexed by one or several indices:

$$A^{i_1,i_2,\cdots,i_n}_{j_1,j_2,\cdots,j_m}.$$

Furthermore, this collection of numbers transforms in a predefined way when a change of coordinate system in $\mathcal{V}$ is made; see Definition 20. The number of indices $(n + m)$ is called the **degree** of the tensor. The indices may take any value from 1 to the dimension of $\mathcal{V}$. The upper indices are called **contravariant** indices and the lower indices are called **covariant** indices. ■

There are some simple conventions that have to be remembered:

- *The index rule* When an index appears in a formula, the formula is valid for every value of the index: $a_i = 0 \Rightarrow a_1 = 0, a_2 = 0, \ldots$.
- *The summation convention* When an index appears twice in a formula, it is implicitly assumed that a summation takes place over that index: $a_i b^i = \sum_{i=1,\dim \mathcal{V}} a_i b^i$.
- *The compatibility rule* A repeated index must appear once as a sub-index and once as a super-index.
- *The maximum rule* An index cannot be used more than twice in a term.

Definition 20. When the coordinate system in $\mathcal{V}$ is changed from $\mathbf{e}$ to $\widehat{\mathbf{e}}$ and the points with coordinates $\mathbf{x}$ are changed to $\widehat{\mathbf{x}}$, according to

$$\widehat{\mathbf{e}}_j = S^i_j \mathbf{e}_i \quad \Leftrightarrow \quad \mathbf{x}_i = S^i_j \widehat{\mathbf{x}}_j,$$

then the affine tensor components change according to

$$\widehat{\mathbf{u}}_k = S^j_k \mathbf{u}_j \quad \text{and} \quad \mathbf{v}^j = S^j_k \widehat{\mathbf{v}}^k,$$

for lower and upper indices respectively. ■

From this definition, the terminology for indices can be motivated, since the covariant indices covary with the basis vectors and the contravariant indices contravary with the basis vectors. It turns out that a vector (e.g., the coordinates of a point) is a contravariant tensor of degree one and that a one-form (e.g., the coordinate of a vector defining a line in $\mathbb{R}^2$ or a hyperplane in $\mathbb{R}^n$) is a covariant tensor of degree one.

Definition 21. The second-order tensor

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

is called the **Kronecker delta**. When $\dim \mathcal{V} = 3$, the third order tensor

$$\epsilon_{ijk} = \begin{cases} 1 & \text{(i,j,k) an even permutation} \\ -1 & \text{(i,j,k) an odd permutation} \\ 0 & \text{(i,j,k) has a repeated value} \end{cases}$$

is called the **Levi-Cevita epsilon**.

3.4 Modeling Cameras

This chapter deals with the task of building a mathematical model of a camera. We give a mathematical model of the standard pinhole camera and define intrinsic and extrinsic parameters. For more detailed treatment, see [24, 15], and for a different approach see [26].

3.4.1 The pinhole camera

The simplest optical system used for modeling cameras is the **pinhole camera**. The camera is modeled as a box with a small hole in one of the sides and a photographic plate at the opposite side; see Figure 3.6. Introduce a coordinate system as in Figure 3.6. Observe that the origin of the coordinate system is located at the center of projection, the **focal point**, and that the z-axis coincides with the optical axis. The distance from the focal point to the image, f, is called the **focal length**. Similar triangles give

$$\frac{x}{f} = \frac{X}{Z} \quad \text{and} \quad \frac{y}{f} = \frac{Y}{Z}. \tag{3.5}$$

This equation can be written in matrix form, using homogeneous coordinates, as

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{3.6}$$

where the **depth**, λ, is equal to Z.

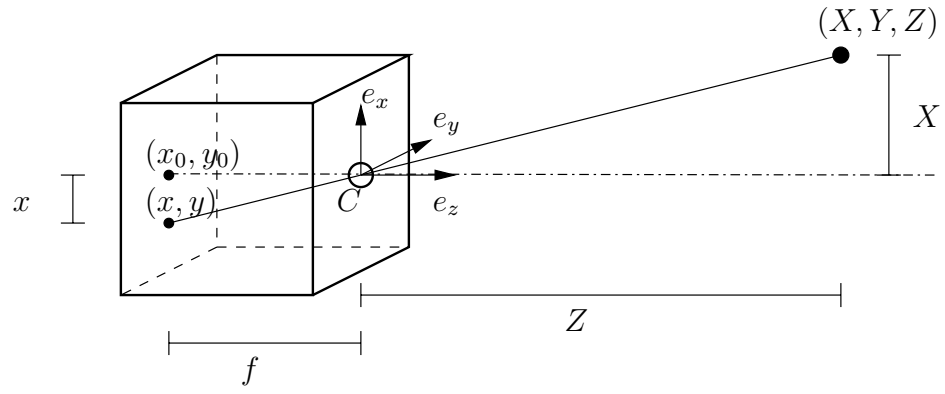

Figure 3.6. The pinhole camera with a coordinate system.

3.4.2 The camera matrix

Introducing the notation

$$K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{3.7}$$

in (3.6) gives

$$\lambda \mathbf{x} = K[\, I_{3\times 3} \,|\, \mathbf{0}_{3\times 1} \,]\mathbf{X} = P\mathbf{X}, \tag{3.8}$$

where $P = K\,[\, I_{3\times 3} \,|\, \mathbf{0}_{3\times 1} \,]$.

Definition 22. A 3×4 matrix P relating extended image coordinates $\mathbf{x} = (x, y, 1)$ to extended object coordinates $\mathbf{X} = (X, Y, Z, 1)$ via the equation

$$\lambda \mathbf{x} = P\mathbf{X}$$

is called a **camera matrix**, and the equation above is called the **camera equation**. ■

Observe that the focal point is given as the right nullspace to the camera matrix, since $P\mathbf{C} = 0$, where $\mathbf{C}$ denote homogeneous coordinates for the focal point, C.

3.4.3 The intrinsic parameters

In a refined camera model, the matrix K in (3.7) is replaced by

$$K = \begin{bmatrix} \gamma f & sf & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.9}$$

where the parameters have the following interpretations, see Figure 3.7:

- f : **focal length**, also called camera constant
- γ : **aspect ratio**, modeling nonquadratic light-sensitive elements
- s : **skew**, modeling nonrectangular light-sensitive elements
- (x_0, y_0) : **principal point**, orthogonal projection of the focal point onto the image plane; see Figure 3.6.

These parameters are called the **intrinsic parameters**, since they model intrinsic properties of the camera. For most cameras, $s = 0$ and $\gamma \approx 1$ and the principal point is located close to the center of the image.

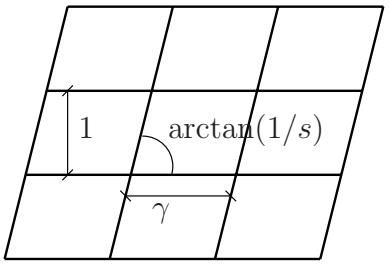

Figure 3.7. The intrinsic parameters.

Definition 23. A camera is said to be **calibrated** if K is known. Otherwise, it is said to be **uncalibrated**. ■

3.4.4 The extrinsic parameters

It is often advantageous to be able to express object coordinates in a different coordinate system than the camera coordinate system. This is especially the case when the relation between these coordinate systems is not known. For this purpose, it is necessary to model the relation between two different coordinate systems in 3D. The natural way to do this is to model the relation

as a Euclidean transformation. Denote the camera coordinate system with e_c and points expressed in this coordinate system with index c–for example, (X_c, Y_c, Z_c)–and similarly denote the object coordinate system with e_o and points expressed in this coordinate system with index o; see Figure 3.8. A

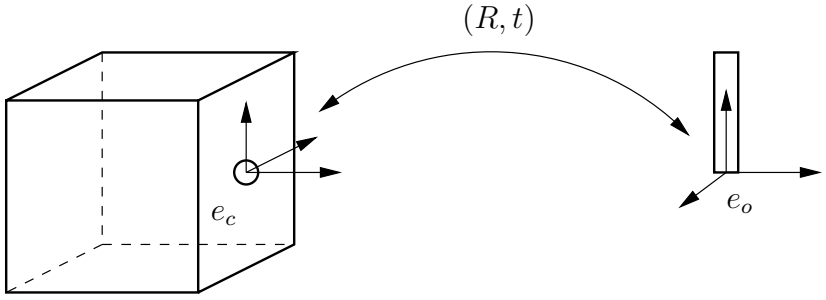

Figure 3.8. Using different coordinate systems for the camera and the object.

Euclidean transformation from the object coordinate system to the camera coordinate system can be written in homogeneous coordinates as

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R^T & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & -t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix} \quad \Longrightarrow \quad \mathbf{X}_c = \begin{bmatrix} R^T & -R^T t \\ 0 & 1 \end{bmatrix} \mathbf{X}_o, \quad (3.10)$$

where R denotes an orthogonal matrix and t a vector, encoding the rotation and translation in the rigid transformation. Observe that the focal point $(0, 0, 0)$ in the c-system corresponds to the point t in the o-system. Inserting (3.10) in (3.8), taking into account that $\mathbf{X}$ in (3.8) is the same as $\mathbf{X}_c$ in (3.10), we obtain

$$\lambda \mathbf{x} = K R^T [\, I \mid -t \,] \mathbf{X}_o = P \mathbf{X}, \quad (3.11)$$

with $P = K R^T [\, I \mid -t \,]$. Usually, it is assumed that object coordinates are expressed in the object coordinate system and the index o in $\mathbf{X}_o$ is omitted. Observe that the focal point, $C_f = t = (t_x, t_y, t_z)$, is given from the right nullspace to P according to

$$P \begin{bmatrix} t_x \\ t_y \\ t_z \\ 1 \end{bmatrix} = K R^T [\, I \mid -t \,] \begin{bmatrix} t_x \\ t_y \\ t_z \\ 1 \end{bmatrix} = \mathbf{0}.$$

Given a camera, described by the camera matrix P, this camera could also be described by the camera matrix μP, $0 \neq \mu \in \mathbb{R}$, since these give the same image point for each object point. This means that the camera matrix is only defined up to an unknown scale factor. Moreover, the camera matrix P can be regarded as a projective transformation from $\mathbb{P}^3$ to $\mathbb{P}^2$; see (3.8) and (3.4).

Observe also that replacing t with μt and (X, Y, Z) with $(\mu X, \mu Y, \mu Z)$, $0 \neq \mu \in \mathbb{R}$, gives the same image since

$$KR^T[\, I \mid -\mu t \,] \begin{bmatrix} \mu X \\ \mu Y \\ \mu Z \\ 1 \end{bmatrix} = \mu K R^T [\, I \mid -t \,] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}.$$

We refer to this ambiguity as the **scale ambiguity**.

We can now calculate the number of parameters in the camera matrix, P:

- K: 5 parameters (f, γ, s, x_0, y_0)
- R: 3 parameters
- t: 3 parameters

Summing up gives a total of 11 parameters, which is the same as in a general 3×4 matrix defined up to scale. This means that for an uncalibrated camera, the factorization $P = KR^T[\, I \mid -t \,]$ has no meaning, and we can instead deal with P as a general 3×4 matrix.

Given a calibrated camera with camera matrix $P = KR^T[\, I \mid -t \,]$ and corresponding camera equation

$$\lambda \mathbf{x} = KR^T[\, I \mid -t \,]\mathbf{X},$$

it is often advantageous to make a change of coordinates from $\mathbf{x}$ to $\widehat{\mathbf{x}}$ in the image according to $\mathbf{x} = K\widehat{\mathbf{x}}$, which gives

$$\lambda K\widehat{\mathbf{x}} = KR^T[\, I \mid -t \,]\mathbf{X} \quad \Rightarrow \quad \lambda \widehat{\mathbf{x}} = R^T[\, I \mid -t \,]\mathbf{X} = \widehat{P}\mathbf{X}.$$

Now the camera matrix becomes $\widehat{P} = R^T[\, I \mid -t]$.

Definition 24. A camera represented with a camera matrix of the form

$$P = R^T[\, I \mid -t]$$

is called a **normalized camera**. ■

Note that even when K is only approximatively known, the above normalization can be useful for numerical reasons. This is discussed more in detail in Section 3.7.

3.4.5 Properties of the pinhole camera

We conclude this section with some properties of the pinhole camera.

Proposition 4. *The set of 3D points that project to an image point,* $\mathbf{x}$*, is given by*

$$\mathbf{X} = \mathbf{C} + \mu P^{+}\mathbf{x}, \quad 0 \neq \mu \in \mathbb{R},$$

where $\mathbf{C}$ *denotes the focal point in homogeneous coordinates and* P^{+} *denotes the pseudo-inverse of* P*.*

Proposition 5. *The set of 3D-points that projects to a line,* $\mathbf{l}$*, is the points lying on the plane* $\Pi = P^T\mathbf{l}$*.*

Proof: It is obvious that the set of points lie on the plane defined by the focal point and the line $\mathbf{l}$. A point $\mathbf{x}$ on $\mathbf{l}$ fulfills $\mathbf{x}^T\mathbf{l} = 0$ and a point $\mathbf{X}$ on the plane Π fulfills $\Pi^T\mathbf{X} = 0$. Since $\mathbf{x} \sim P\mathbf{X}$, we have $(P\mathbf{X})^T\mathbf{l} = \mathbf{X}^T P^T\mathbf{l} = 0$ and identification with $\Pi^T\mathbf{X} = 0$ gives $\Pi = P^T\mathbf{l}$. ∎

Lemma 3. *The projection of a quadric,* $\mathbf{X}^T C\mathbf{X} = 0$ *(dually* $\Pi^T C'\Pi = 0$*,* $C' = C^{-1}$*), is an image conic,* $\mathbf{x}^T c\mathbf{x} = 0$ *(dually* $\mathbf{l}^T c'\mathbf{l} = 0$*,* $c' = c^{-1}$*), with* $c' = PC'P^T$*.*

Proof: Use the previous proposition. ∎

Proposition 6. *The image of the absolute conic is given by the conic* $\mathbf{x}^T\omega\mathbf{x} = 0$ *(dually* $\mathbf{l}^T\omega'\mathbf{l} = 0$*), where* $\omega' = KK^T$*.*

Proof: The result follows from the previous lemma:

$$\omega' \sim P\Omega'P^T \sim KR^T \begin{bmatrix} I & -t \end{bmatrix} \begin{bmatrix} I & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} \begin{bmatrix} I \\ -t^T \end{bmatrix} RK^T = KR^T RK^T = KK^T.$$

∎

3.5 Multiple View Geometry

Multiple view geometry is the subject in which relations between coordinates of feature points in different views are studied. It is an important tool for understanding the image formation process for several cameras and for designing reconstruction algorithms. For a more detailed treatment, see [27] or [24], and for a different approach, see [26]. For the algebraic properties of multilinear constraints, see [30].

3.5.1 The structure and motion problem

The following problem is central in computer vision:

Problem 1. *Structure and motion. Given a sequence of images with corresponding feature points* $\mathbf{x}_{ij}$*, taken by a perspective camera,*

$$\lambda_{ij}\mathbf{x}_{ij} = P_i\mathbf{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n,$$

determine the camera matrices, P_i *(***motion***) and the 3D points,* $\mathbf{X}_j$ *(***structure***), under different assumptions on the intrinsic and/or extrinsic parameters. This is called the* **structure and motion problem**.

It turns out that there is a fundamental limitation on the solutions to the structure and motion problem when the intrinsic parameters are unknown and possibly varying, namely with an **uncalibrated image sequence**.

Theorem 5. *Given an uncalibrated image sequence with corresponding points, it is only possible to reconstruct the object up to an unknown projective transformation.*

Proof: Assume that $\mathbf{X}_j$ is a reconstruction of n points in m images, with camera matrices P_i according to

$$\mathbf{x}_{ij} \sim P_i\,\mathbf{X}_j,\ i = 1, \ldots m,\ j = 1, \ldots n.$$

Then $H\,\mathbf{X}_j$ is also a reconstruction, with camera matrices $P_i\,H^{-1}$, for every nonsingular 4×4 matrix H, since

$$\mathbf{x}_{ij} \sim P_i\,\mathbf{X}_j \sim P_iH^{-1}H\mathbf{X}_j \sim (P_iH^{-1})\,(H\mathbf{X}_j).$$

The transformation

$$\mathbf{X} \mapsto H\mathbf{X}$$

corresponds to all projective transformations of the object. ∎

In the same way it can be shown that if the cameras are calibrated, then it is possible to reconstruct the scene up to an unknown similarity transformation.

3.5.2 The two-view case

The epipoles

Consider two images of the same point $\mathbf{X}$ as in Figure 3.9.

Definition 25. The **epipole**, $\mathbf{e}_{i,j}$, is the projection of the focal point of camera i in image j. ∎

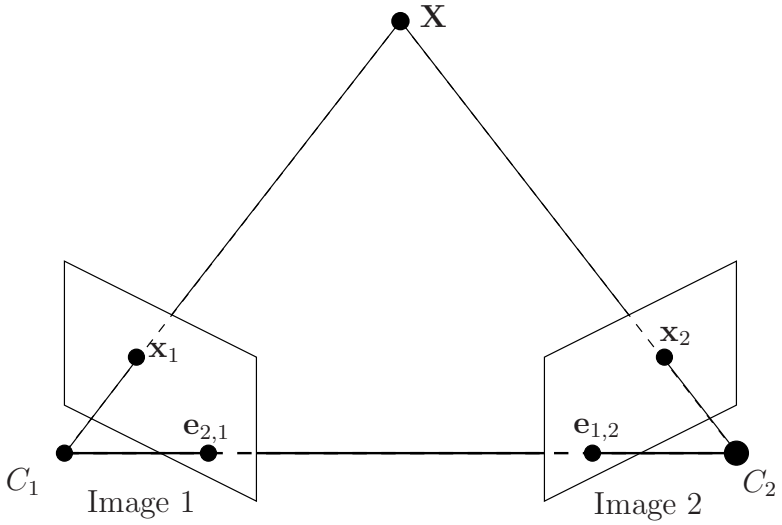

Figure 3.9. Two images of the same point and the epipoles.

Proposition 7. *Let*

$$P_1 = [\ A_1 \mid b_1\ ] \quad \textit{and} \quad P_2 = [\ A_2 \mid b_2\ ].$$

Then the epipole, $\mathbf{e}_{1,2}$ *is given by*

$$\mathbf{e}_{1,2} = -A_2 A_1^{-1} b_1 + b_2. \tag{3.12}$$

Proof: The focal point of camera 1, C_1, is given by

$$P_1 \begin{bmatrix} C_1 \\ 1 \end{bmatrix} = [\ A_1 \mid b_1\ ] \begin{bmatrix} C_1 \\ 1 \end{bmatrix} = A_1 C_1 + b_1 = 0;$$

that is, $C_1 = -A_1^{-1} b_1$, and then the epipole is obtained from

$$P_2 \begin{bmatrix} C_1 \\ 1 \end{bmatrix} = [\ A_2 \mid b_2\ ] \begin{bmatrix} C_1 \\ 1 \end{bmatrix} = A_2 C_1 + b_2 = -A_2 A_1^{-1} b_1 + b_2.$$

■

It is convenient to use the notation $A_{12} = A_2 A_1^{-1}$. Assume that we have calculated two camera matrices representing the two-view geometry,

$$P_1 = [\ A_1 \mid b_1\ ] \quad \text{and} \quad P_2 = [\ A_2 \mid b_2\ ].$$

According to Theorem 5, we can multiply these camera matrices with

$$H = \begin{bmatrix} A_1^{-1} & -A_1^{-1}b_1 \\ 0 & 1 \end{bmatrix}$$

from the right and obtain

$$\bar{P}_1 = P_1 H = [\ I \mid 0\ ] \quad \bar{P}_2 = P_2 H = [\ A_2 A_1^{-1} \mid b_2 - A_2 A_1^{-1} b_1\ ].$$

Thus, we may always assume that the first camera matrix is $[\ I \mid 0\ ]$. Observe that $\bar{P}_2 = [\ A_{12} \mid \mathbf{e}\ ]$, where $\mathbf{e}$ denotes the epipole in the second image. Observe also that we may multiply again with

$$\bar{H} = \begin{bmatrix} I & 0 \\ \mathbf{v}^T & 1 \end{bmatrix}$$

without changing $\bar{P}_1$, but

$$\bar{H}\bar{P}_2 = [\ A_{12} + \mathbf{e}\mathbf{v}^T \mid \mathbf{e}\ ];$$

that is, the last column of the second camera matrix still represents the epipole.

Definition 26. A pair of camera matrices is said to be in **canonical form** if

$$P_1 = [\ I \mid 0\ ] \quad \text{and} \quad P_2 = [\ A_{12} + \mathbf{e}\mathbf{v}^T \mid \mathbf{e}\ ], \tag{3.13}$$

where $\mathbf{v}$ denotes a three-parameter ambiguity. ■

The fundamental matrix

The fundamental matrix was originally discovered in the calibrated case in [38] and in the uncalibrated case in [13]. Consider a fixed point, $\mathbf{X}$, in two views:

$$\lambda_1 \mathbf{x}_1 = P_1 \mathbf{X} = [\ A_1 \mid b_1\ ]\mathbf{X}, \quad \lambda_2 \mathbf{x}_2 = P_2 \mathbf{X} = [\ A_2 \mid b_2\ ]\mathbf{X}.$$

Use the first camera equation to solve for X, Y, Z:

$$\lambda_1 \mathbf{x}_1 = P_1 \mathbf{X} = [\ A_1 \mid b_1\ ]\mathbf{X} = A_1 \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + b_1 \quad \Rightarrow \quad \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = A_1^{-1}(\lambda_1 \mathbf{x}_1 - b_1)$$

and insert into the second one

$$\lambda_2 \mathbf{x}_2 = A_2 A_1^{-1}(\lambda_1 \mathbf{x}_1 - b_1) + b_2 = \lambda_1 A_{12}\mathbf{x}_1 + (-A_{12}b_1 - b_2).$$

That is, $\mathbf{x}_2$, $A_{12}\mathbf{x}_1$ and $\mathbf{t} = -A_{12}b_1 + b_2 = \mathbf{e}_{1,2}$ are linearly dependent. Observe that $\mathbf{t} = \mathbf{e}_{1,2}$, the epipole in the second image. This condition can be written as $\mathbf{x}_1^T A_{12}^T T_{\mathbf{e}} \mathbf{x}_2 = \mathbf{x}_1^T F \mathbf{x}_2 = 0$, with $F = A_{12}^T T_{\mathbf{e}}$, where $T_{\mathbf{x}}$ denotes the skew-symmetric matrix corresponding to the vector $\mathbf{x}$; that is, $T_{\mathbf{x}}(y) = x \times y$.

Definition 27. The bilinear constraint

$$\mathbf{x}_1^T F \mathbf{x}_2 = 0 \tag{3.14}$$

is called the **epipolar constraint** and

$$F = A_{12}^T T_{\mathbf{e}}$$

is called the **fundamental matrix**. ■

Theorem 6. *The epipole in the second image is obtained as the right nullspace to the fundamental matrix and the epipole in the left image is obtained as the left nullspace to the fundamental matrix.*

Proof: Follows from $F\mathbf{e} = A_{12}^T T_{\mathbf{e}} \mathbf{e} = A_{12}^T(\mathbf{e} \times \mathbf{e}) = 0$. The statement about the epipole in the left image follows from symmetry. ■

Corollary 1. *The fundamental matrix is singular:* $\det F = 0$.

Given a point, $\mathbf{x}_1$, in the first image, the coordinates of the corresponding point in the second image fulfill

$$0 = \mathbf{x}_1^T F \mathbf{x}_2 = (\mathbf{x}_1^T F)\mathbf{x}_2 = \mathbf{l}(\mathbf{x}_1)^T \mathbf{x}_2 = 0,$$

where $\mathbf{l}(\mathbf{x}_1)$ denotes the line represented by $\mathbf{x}_1^T F$.

Definition 28. The line $\mathbf{l} = F^T \mathbf{x}_1$ is called the **epipolar line** corresponding to $\mathbf{x}_1$. ■

The geometrical interpretation of the epipolar line is the geometric construction in Figure 3.10. The points $\mathbf{x}_1$, C_1, and C_2 define a plane, Π, intersecting the second image plane in the line $\mathbf{l}$, containing the corresponding point.

From the previous considerations, we have the following pair:

$$F = A_{12}^T T_{\mathbf{e}} \quad \Leftrightarrow \quad P_1 = [\, I \mid 0 \,], \quad P_2 = [\, A_{12} \mid \mathbf{e} \,]. \tag{3.15}$$

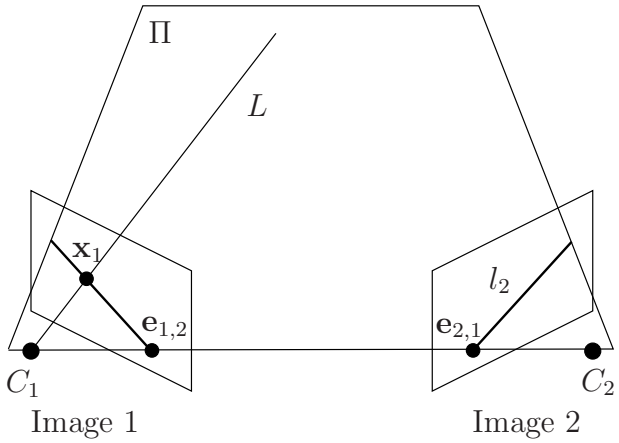

Figure 3.10. The epipolar line.

Observe that

$$F = A_{12}^T T_e = (A_{12} + \mathbf{e}\mathbf{v}^T)^T T_{\mathbf{e}}$$

for every vector $\mathbf{v}$, since

$$(A_{12} + \mathbf{e}\mathbf{v})^T T_{\mathbf{e}}(\mathbf{x}) = A_{12}^T(\mathbf{e} \times \mathbf{x}) + \mathbf{v}\mathbf{e}^T(\mathbf{e} \times \mathbf{x}) = A_{12}^T T_{\mathbf{e}}\mathbf{x},$$

since $\mathbf{e}^T(\mathbf{e} \times \mathbf{x}) = \mathbf{e} \cdot (\mathbf{e} \times \mathbf{x}) = 0$. This ambiguity corresponds to the transformation

$$\bar{H}\bar{P}_2 = [\ A_{12} + \mathbf{e}\mathbf{v}^T \mid \mathbf{e}\ ].$$

We conclude that there are three free parameters in the choice of the second camera matrix when the first is fixed to $P_1 = [\ I \mid 0\ ]$.

The infinity homography

Consider a plane in the 3D object space, Π, defined by a vector $\mathbf{V}$: $\mathbf{V}^T\mathbf{X} = 0$ and the following construction; see Figure 3.11. Given a point in the first image, construct the intersection with the optical ray and the plane Π and project to the second image. This procedure gives a homography between points in the first and second images that depends on the chosen plane Π.

Proposition 8. *The homography corresponding to the plane* $\Pi : \mathbf{V}^T\mathbf{X} = 0$ *is given by the matrix*

$$H_\Pi = A_{12} - \mathbf{e}\mathbf{v}^T,$$

where $\mathbf{e}$ *denotes the epipole and* $\mathbf{V} = [\ \mathbf{v}\, 1\ ]^T$.

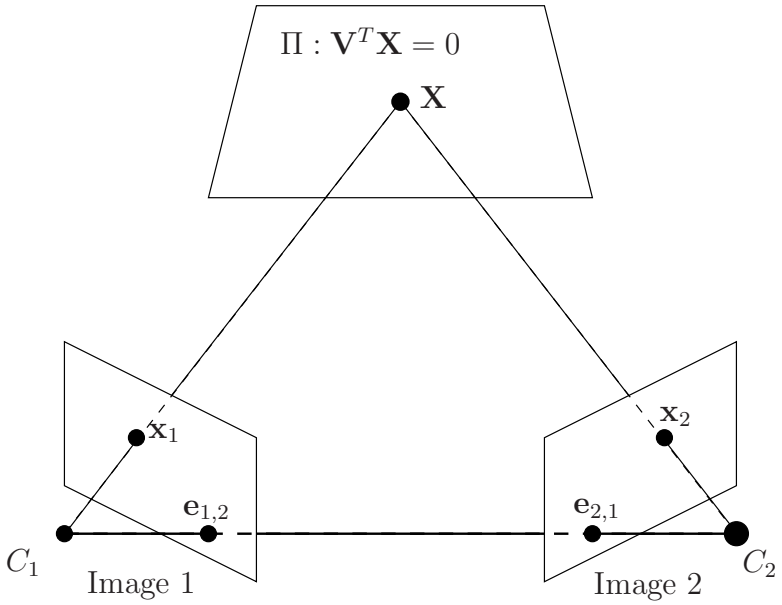

Figure 3.11. The homography corresponding to the plane Π.

Proof: Assume that

$$P_1 = [\, I \mid 0 \,], \quad P_2 = [\, A_{12} \mid \mathbf{e} \,].$$

Write $\mathbf{V} = [\, v_1\, v_2\, v_3\, 1 \,]^T = [\, \mathbf{v}\, 1 \,]^T$ (assuming $v_4 \neq 0$; that is, the plane is not incident with the origin, or the focal point of the first camera) and $\mathbf{X} = [\, X\, Y\, Z\, W \,]^T = [\, \mathbf{w}\, W \,]^T$, which gives

$$\mathbf{V}^T\mathbf{X} = \mathbf{v}^T\mathbf{w} + W, \tag{3.16}$$

which implies that $\mathbf{v}^T\mathbf{w} = -W$ for points in the plane Π. The first camera equation gives

$$\mathbf{x}_1 \sim [\, I \mid 0 \,]\mathbf{X} = \mathbf{w},$$

and using (3.16) gives $\mathbf{v}^T\mathbf{x}_1 = -W$. Finally, the second camera matrix gives

$$\mathbf{x}_2 \sim [\, A_{12} \mid \mathbf{e} \,] \begin{bmatrix} \mathbf{x}_1 \\ -\mathbf{v}^T\mathbf{x}_1 \end{bmatrix} = A_{12}\mathbf{x}_1 - \mathbf{e}\mathbf{v}^T\mathbf{x}_1 = (A_{12} - \mathbf{e}\mathbf{v}^T)\mathbf{x}_1.$$

■

Observe that when $\mathbf{V} = (0, 0, 0, 1)$, that is, $\mathbf{v} = (0, 0, 0)$, the plane Π is the plane at infinity.

Definition 29. The homography

$$H_\infty = H_{\Pi_\infty} = A_{12}$$

is called the **homography corresponding to the plane at infinity** or **infinity homography**. ■

Note that the epipolar line through the point $\mathbf{x}_2$ in the second image can be written as $\mathbf{x}_2 \times \mathbf{e}$, implying

$$(\mathbf{x}_2 \times \mathbf{e})^T H \mathbf{x}_1 = \mathbf{x}_1^T H^T T_\mathbf{e} \mathbf{x}_2 = 0,$$

that is, the epipolar constraint, and we get

$$F = H^T T_e.$$

Proposition 9. *There is a one-to-one correspondence between planes in 3D, homographies between two views, and factorization of the fundamental matrix as* $F = H^T T_e$.

Finally, we note that the matrix $H_\Pi^T T_e H_\Pi$ is skew symmetric, implying that

$$F H_\Pi + H_\Pi^T F^T = 0. \tag{3.17}$$

3.5.3 Multiview constraints and tensors

Consider one object point, $\mathbf{X}$, and its m images, $\mathbf{x}_i$, according to the camera equations $\lambda_i \mathbf{x}_i = P_i \mathbf{X}$, $i = 1 \ldots m$. These equations can be written as

$$\underbrace{\begin{bmatrix} P_1 & \mathbf{x}_1 & 0 & 0 & \ldots & 0 \\ P_2 & 0 & \mathbf{x}_2 & 0 & \ldots & 0 \\ P_3 & 0 & 0 & \mathbf{x}_3 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ P_m & 0 & 0 & 0 & \ldots & \mathbf{x}_m \end{bmatrix}}_{M} \begin{bmatrix} \mathbf{X} \\ -\lambda_1 \\ -\lambda_2 \\ -\lambda_3 \\ \vdots \\ -\lambda_m \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{3.18}$$

We immediately get the following proposition:

Proposition 10. *The matrix, M, in (3.18) is rank deficient; that is,*

$$\operatorname{rank} M < m + 4,$$

which is referred to as the **rank condition**.

The rank condition implies that all $(m+4) \times (m+4)$ minors of M are equal to 0. These can be written using Laplace expansions as sums of products of determinants of *four rows* taken from the first four columns of M and of image coordinates. There are three different categories of such minors depending on the number of rows taken from each image, since one row has to be taken from each image, and then the remaining four rows can be distributed freely. The three different types are:

1. Taking the two remaining rows from one camera matrix and the two remaining rows from another camera matrix, gives 2-view constraints.
2. Taking the two remaining rows from one camera matrix, one row from another and one row from a third camera matrix, gives three-view constraints.
3. Taking one row from each of four different camera matrices, gives four-view constraints.

Observe that the minors of M can be factorized as products of the two-, three-, or four-view constraint and image coordinates in the other images. In order to get a notation that connects to the tensor notation, we use (x^1, x^2, x^3) instead of (x, y, z) for homogeneous image coordinates. We also denote row number i of a camera matrix P by P^i.

The monofocal tensor

Before we proceed to the multiview tensors, we make the following observation:

Proposition 11. *The epipole in image* 2 *from camera* 1, $\mathbf{e} = (e^1, e^2, e^3)$ *in homogeneous coordinates, can be written as*

$$e^j = \det \begin{bmatrix} P_1^1 \\ P_1^2 \\ P_1^3 \\ P_2^j \end{bmatrix} . \tag{3.19}$$

Proposition 12. *The numbers e^j constitute a first-order contravariant tensor, where the transformations of the tensor components are related to projective transformations of the image coordinates.*

Definition 30. The first-order contravariant tensor, e^j, is called the **monofocal tensor**.

The bifocal tensor

Considering minors obtained by taking three rows from one image, and three rows from another image:

$$\det \begin{bmatrix} P_1 & \mathbf{x}_1 & 0 \\ P_2 & 0 & \mathbf{x}_2 \end{bmatrix} = \det \begin{bmatrix} P_1^1 & x_1^1 & 0 \\ P_1^2 & x_1^2 & 0 \\ P_1^3 & x_1^3 & 0 \\ P_2^1 & 0 & x_2^1 \\ P_2^2 & 0 & x_2^2 \\ P_2^3 & 0 & x_2^3 \end{bmatrix} = 0,$$

which gives a *bilinear constraint*:

$$\sum_{i,j=1}^{3} F_{ij}\mathbf{x}_1^i\mathbf{x}_2^j = F_{ij}\mathbf{x}_1^i\mathbf{x}_2^j = 0, \tag{3.20}$$

where

$$F_{ij} = \sum_{i',i'',j',j''=1}^{3} \epsilon_{ii'i''}\epsilon_{jj'j''} \det \begin{bmatrix} P_1^{i'} \\ P_1^{i''} \\ P_2^{j'} \\ P_2^{j''} \end{bmatrix}.$$

The following proposition follows from (3.20).

Proposition 13. *The numbers F_{ij} constitute a second-order covariant tensor.*

Here the transformations of the tensor components are related to projective transformations of the image coordinates.

Definition 31. The second-order covariant tensor, F_{ij}, is called the **bifocal tensor**, and the bilinear constraint in (3.20) is called the **bifocal constraint**. ■

Observe that the indices tell us which row to *exclude* from the corresponding camera matrix when forming the determinant. The geometric interpretation of the bifocal constraint is that corresponding view-lines in two images intersect in 3D; see Figure 3.9. The bifocal tensor can also be used to transfer a point to the corresponding epipolar line (see Figure 3.10), according to $\mathbf{l}_j^2 = F_{ij}\mathbf{x}_1^i$. This transfer can be extended to a homography between epipolar lines in the first view and epipolar lines in the second view according to

$$\mathbf{l}_i^1 = F_{ij}\epsilon_{j''}^{jj'}\mathbf{l}_{j'}^2\mathbf{e}^{j''},$$

since $\epsilon_{j''}^{jj'}\mathbf{l}_{j'}^2\mathbf{e}^{j''}$ gives the cross product between the epipole $\mathbf{e}$ and the line $\mathbf{l}^2$, which gives a point on the epipolar line.

The trifocal tensor

The trifocal tensor was originally discovered in the calibrated case in [60] and in the uncalibrated case in [55]. Considering minors obtained by taking three rows from one image, two rows from another image, and two rows from a third image, for example,

$$\det\begin{bmatrix} P_1^1 & x_1^1 & 0 & 0 \\ P_1^2 & x_1^2 & 0 & 0 \\ P_1^3 & x_1^3 & 0 & 0 \\ P_2^1 & 0 & x_2^1 & 0 \\ P_2^2 & 0 & x_2^2 & 0 \\ P_3^1 & 0 & 0 & x_3^1 \\ P_3^3 & 0 & 0 & x_3^3 \end{bmatrix} = 0,$$

gives a *trilinear constraints*:

$$\sum_{i,j,j',k,k'=1}^{3} T_i^{jk}\mathbf{x}_1^i\epsilon_{jj'j''}\mathbf{x}_2^{j'}\epsilon_{kk'k''}\mathbf{x}_3^{k'} = 0, \tag{3.21}$$

where

$$T_i^{jk} = \sum_{i',i''=1}^{3}\epsilon_{ii'i''}\det\begin{bmatrix} P_1^{i'} \\ P_1^{i''} \\ P_2^{j} \\ P_3^{k} \end{bmatrix}. \tag{3.22}$$

Note that there are in total nine constraints indexed by j'' and k'' in (3.21).

Proposition 14. *The numbers T_i^{jk} constitute a third-order mixed tensor that is covariant in i and contravariant in j and k.*

Definition 32. The third order mixed tensor, T_i^{jk} is called the **trifocal tensor**, and the trilinear constraint in (3.21) is called the **trifocal constraint**. ■

Again, the lower index tells us which row to *exclude* from the first camera matrix, and the upper indices tell us which rows to *include* from the second and third camera matrices respectively, and these indices becomes covariant and contravariant respectively. Observe that the order of the images is

important, since the first image is treated differently. If the images are permuted, another set of coefficients is obtained. The geometric interpretation of the trifocal constraint is that the view-line in the first image and the planes corresponding to arbitrary lines coincident with the corresponding points in the second and third images (together with the focal points) respectively intersect in 3D; see Figure 3.12. The following theorem is straightforward to prove.

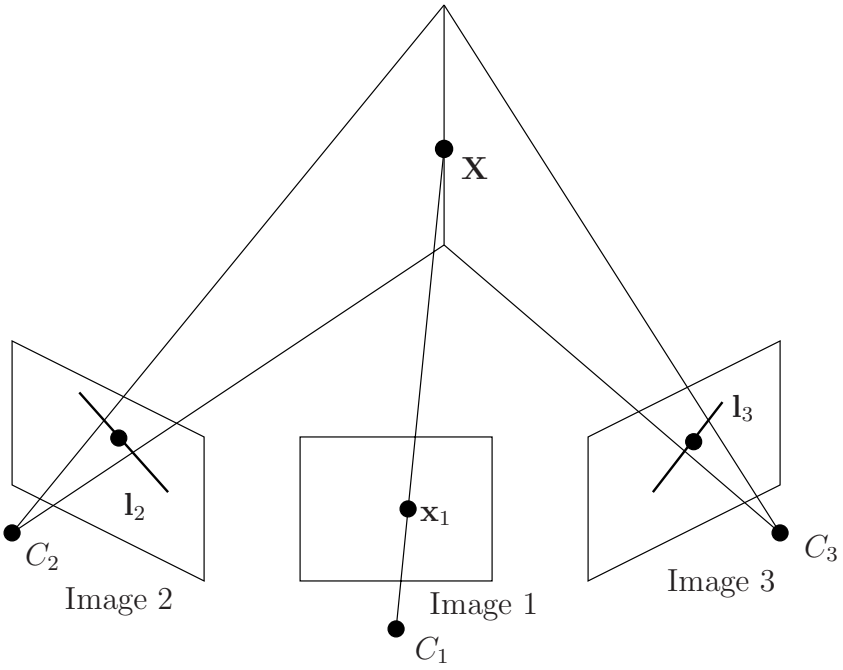

Figure 3.12. Geometrical interpretation of the trifocal constraint.

Theorem 7. *Given three corresponding lines,* $\mathbf{l}^1$, $\mathbf{l}^2$, *and* $\mathbf{l}^3$, *in three images, represented by the vectors* $(\mathbf{l}^1_1, \mathbf{l}^1_2, \mathbf{l}^1_3)$ *and so on, then*

$$\mathbf{l}^3_k = T^{ij}_k \mathbf{l}^1_i \mathbf{l}^2_j. \tag{3.23}$$

From this theorem it is possible to transfer the images of a line seen in two images to a third image, called **tensorial transfer**. The geometrical interpretation is that two corresponding lines define two planes in 3D that

intersect in a line, that can be projected onto the third image. There are also other transfer equations, such as

$$\mathbf{x}_2^j = T_i^{jk}\mathbf{x}_1^i\mathbf{l}_k^3 \quad \text{and} \quad \mathbf{x}_3^k = T_i^{jk}\mathbf{x}_2^j\mathbf{l}_k^3,$$

with obvious geometrical interpretations.

The quadrifocal tensor

The quadrifocal tensor was independently discovered in several papers, including [64, 26]. Considering minors obtained by taking two rows from each one of four different images gives a *quadrilinear constraint*,

$$\sum_{i,i',j,j',k,k',l,l'=1}^{3} Q^{ijkl}\epsilon_{ii'i''}\mathbf{x}_{i'}^1\epsilon_{jj'j''}\mathbf{x}_{j'}^2\epsilon_{kk'k''}\mathbf{x}_{k'}^3\epsilon_{ll'l''}\mathbf{x}_{l'}^4 = 0, \tag{3.24}$$

where

$$Q^{ijkl} = \det\begin{bmatrix} P_1^i \\ P_2^j \\ P_3^k \\ P_4^l \end{bmatrix}.$$

Note that there are in total 81 constraints indexed by i'', j'', k'', and l'' in (3.24).

Proposition 15. *The numbers Q^{ijkl} constitute a fourth-order contravariant tensor.*

Definition 33. The fourth-order contravariant tensor, Q^{ijkl} is called the **quadrifocal tensor** and the quadrilinear constraint in (3.24), is called the **quadrifocal constraint**. ■

Note that there are in total 81 constraints indexed by i'', j'', k'', and l''. Again, the upper indices tell us which rows to *include* from each camera matrix respectively. They become contravariant indices. The geometric interpretation of the quadrifocal constraint is that the four planes corresponding to arbitrary lines coincident with the corresponding points in the images intersect in 3D.

3.6 Structure and Motion I

We now study the structure and motion problem in detail. First, we solve the problem when the structure in known, called *resection*, then when the

motion is known, called *intersection*. Then we present a linear algorithm to solve for both structure and motion using the multifocal tensors, and finally a factorization algorithm is presented. Again, refer to [24] for a more detailed treatment.

3.6.1 Resection

Problem 2 (Resection). *Assume that the structure is given; that is, the object points, $\mathbf{X}_j$, $j = 1, \ldots n$, are given in some coordinate system. Calculate the camera matrices P_i, $i = 1, \ldots m$ from the images, that is, from $\mathbf{x}_{i,j}$.*

The simplest solution to this problem is the classical DLT algorithm based on the fact that the camera equations

$$\lambda_j \mathbf{x}_j = P\mathbf{X}_j, \quad j = 1 \ldots n$$

are linear in the unknown parameters, λ_j and P.

3.6.2 Intersection

Problem 3 (Intersection). *Assume that the motion is given; that is, the camera matrices, P_i, $i = 1, \ldots m$, are given in some coordinate system. Calculate the structure $\mathbf{X}_j$, $j = 1, \ldots n$ from the images, that is, from $\mathbf{x}_{i,j}$.*

Consider the image of $\mathbf{X}$ in camera 1 and 2

$$\begin{cases} \lambda_1 \mathbf{x}_1 = P_1 \mathbf{X}, \\ \lambda_2 \mathbf{x}_2 = P_2 \mathbf{X}, \end{cases} \tag{3.25}$$

which can be written in matrix form as (compared to (3.18)):

$$\begin{bmatrix} P_1 & \mathbf{x}_1 & \mathbf{0} \\ P_2 & \mathbf{0} & \mathbf{x}_2 \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ -\lambda_1 \\ -\lambda_2 \end{bmatrix} = \mathbf{0}, \tag{3.26}$$

which again is linear in the unknowns, λ_i and $\mathbf{X}$. This linear method can of course be extended to an arbitrary number of images.

3.6.3 Linear estimation of tensors

We are now able to solve the structure and motion problem given in Problem 1. The general scheme is as follows:

1. Estimate the components of a multiview tensor linearly from image correspondences.
2. Extract the camera matrices from the tensor components.
3. Reconstruct the object using intersection, that is, (3.26).

The eight-point algorithm

Each point correspondence gives one linear constraint on the components of the bifocal tensor according to the bifocal constraint:

$$F_{ij}\mathbf{x}_1^i\mathbf{x}_2^j = 0.$$

Each pair of corresponding points gives a linear homogeneous constraint on the nine tensor components F_{ij}. Thus, given at least eight corresponding points, we can solve linearly (e.g., by SVD) for the tensor components. After the bifocal tensor (fundamental matrix) has been calculated, it has to be factorized as $F = A_{12}T_{\mathbf{e}}^T$, which can be done by first solving for $\mathbf{e}$ using $F\mathbf{e} = 0$ (i.e., finding the right nullspace to F) and then for A_{12} by solving a linear system of equations. One solution is

$$A_{12} = \begin{bmatrix} 0 & 0 & 0 \\ F_{13} & F_{23} & F_{33} \\ -F_{12} & -F_{22} & -F_{32} \end{bmatrix},$$

which can be seen from the definition of the tensor components. In the case of noisy data, it might happen that $\det F \neq 0$, and the right nullspace does not exist. One solution is to solve $F\mathbf{e} = 0$ in least-squares sense using SVD. Another possibility is to project F to the closest rank-2 matrix, again using SVD. Then the camera matrices can be calculated from (3.26) and finally using intersection, (3.25), to calculate the structure.

The seven-point algorithm

A similar algorithm can be constructed for the case of corresponding points in three images.

Proposition 16. *The trifocal constraint in (3.21) contains four linearly independent constraints in the tensor components T_i^{jk}.*

Corollary 2. *At least seven corresponding points in three views are needed in order to estimate the* 27 *homogeneous components of the trifocal tensor.*

The main difference from the eight-point algorithm is that it is not obvious how to extract the camera matrices from the trifocal tensor components. Start with the transfer equation

$$\mathbf{x}_2^j = T_i^{jk}\mathbf{x}_1^i\mathbf{l}_k^3,$$

which can be seen as a homography between the first two images by fixing a line in the third image. The homography is the one corresponding to the plane Π defined by the focal point of the third camera and the fixed line in the third camera. Thus, we know from (3.17) that the fundamental matrix between image 1 and image 2 obeys

$$FT_{\cdot}^{\cdot J} + (T_{\cdot}^{\cdot J})^T F^T = 0,$$

where $T_{\cdot}^{\cdot J}$ denotes the matrix obtained by fixing the index J. Since this is a linear constraint on the components of the fundamental matrix, it can easily be extracted from the trifocal tensor. Then the camera matrices P_1 and P_2 could be calculated, and finally, the entries in the third camera matrix P_3 can be recovered linearly from the definition of the tensor components in (3.22); see [27].

An advantage of using three views is that lines could be used to constrain the geometry, using (3.23), giving two linearly independent constraints for each corresponding line.

The six-point algorithm

Again, a similar algorithm can be constructed for the case of corresponding points in four images.

Proposition 17. *The quadrifocal constraint in (3.24) contains* 16 *linearly independent constraints in the tensor components* Q^{ijkl}.

From this proposition, it seems that five corresponding points would be sufficient to calculate the 81 homogeneous components of the quadrifocal tensor. However, the following proposition says that this is not possible.

Proposition 18. *The quadrifocal constraint in (3.24) for two corresponding points contains 31 linearly independent constraints in the tensor components* Q^{ijkl}.

Corollary 3. *At least six corresponding points in three views are needed in order to estimate the 81 homogeneous components of the quadrifocal tensor.*

Since one independent constraint is lost for each pair of corresponding points in four images, we get $6 \cdot 16 - \binom{6}{2} = 81$ linearly independent constraints.

Again, it is not obvious how to extract the camera matrices from the trifocal tensor components. First, a trifocal tensor has to be extracted, and then a fundamental matrix and the camera matrices must be extracted. It is outside the scope of this work to give the details for this; see [27]. Also, in this case, corresponding lines can be used by looking at transfer equations for the quadrifocal tensor.

3.6.4 Factorization

A disadvantage with using multiview tensors to solve the structure and motion problem is that when many images ($\gg 4$) are available, the information in all images cannot be used with equal weight. An alternative is to use a factorization method, see [59].

Write the camera equations

$$\lambda_{i,j}\mathbf{x}_{i,j} = P_i\mathbf{X}_j, \quad i = 1,\dots,m, \quad j = 1,\dots,n$$

for a fixed image i in matrix form as

$$\mathfrak{x}_i\Lambda_i = P_i\mathbb{X}, \tag{3.27}$$

where

$$\mathfrak{x}_i = \begin{bmatrix}\mathbf{x}_{i,1}^T & \mathbf{x}_{i,2}^T & \dots & \mathbf{x}_{i,n}^T\end{bmatrix}, \quad \mathbb{X} = \begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T & \dots & \mathbf{X}_n^T\end{bmatrix},$$
$$\Lambda_i = \operatorname{diag}(\lambda_{i,1}, \lambda_{i,2}, \dots, \lambda_{i,n}).$$

The camera matrix equations for all images can now be written as

$$\widehat{\mathfrak{x}} = \mathbb{P}\mathbb{X}, \tag{3.28}$$

where

$$\widehat{\mathfrak{x}} = \begin{bmatrix}\mathfrak{x}_1\Lambda_1 \\ \mathfrak{x}_2\Lambda_2 \\ \vdots \\ \mathfrak{x}_m\Lambda_m\end{bmatrix}, \quad \mathbb{P} = \begin{bmatrix}P_1 \\ P_2 \\ \vdots \\ P_3\end{bmatrix}.$$

Observe that $\widehat{\mathfrak{x}}$ only contains image measurements apart from the unknown depths.

Proposition 19.

$$\operatorname{rank}\widehat{\mathfrak{x}} \leq 4$$

This follows from (3.28), since $\widehat{\mathfrak{x}}$ is a product of a $3m \times 4$ and a $4 \times n$ matrix. Assume that the depths, Λ_i, are known, corresponding to affine cameras, we may use the following simple **factorization** algorithm:

1. Build up the matrix $\widehat{\mathfrak{x}}$ from image measurements.
2. Factorize $\widehat{\mathfrak{x}} = U\Sigma V^T$ using SVD.
3. Extract $\mathbb{P}$ = the first four columns of $U\Sigma$ and $\mathbb{X}$ = the first four rows of V^T.

In the perspective case, this algorithm can be extended to the **iterative factorization** algorithm:

1. Set $\lambda_{i,j} = 1$.
2. Build up the matrix $\widehat{\mathfrak{X}}$ from image measurements and the current estimate of $\lambda_{i,j}$.
3. Factorize $\widehat{\mathfrak{X}} = U\Sigma V^T$ using SVD.
4. Extract $\mathbb{P}$ = the first four columns of $U\Sigma$ and $\mathbb{X}$ = the first four rows of V^T.
5. Use the current estimate of $\mathbb{P}$ and $\mathbb{X}$ to improve the estimate of the depths from the camera equations $\mathfrak{X}_i\Lambda_i = P_i\mathbb{X}$.
6. If the error (reprojection errors or σ_5) is too large, go to step 2.

Definition 34. The fifth singular value, σ_5, in the SVD above is called the **proximity measure** and is a measure of the accuracy of the reconstruction. ■

Theorem 8. *The algorithm above minimizes the proximity measure.*

Figure 3.13 shows an example of a reconstruction using the iterative factorization method applied on four images of a toy block scene. Observe that the proximity measure decreases quite fast and the algorithm converges in about 20 steps.

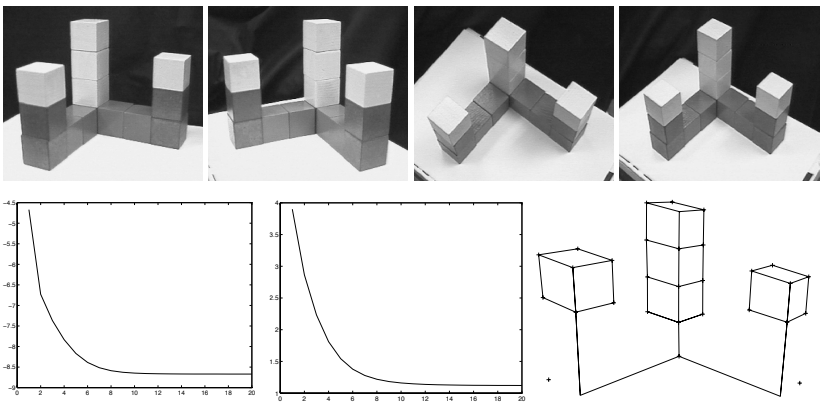

Figure 3.13. Top: Four images. Bottom: The proximity measure for each iteration, the standard deviation of reprojection errors, and the reconstruction.

3.7 Structure and Motion II

In this section, we discuss the problem of structure and motion recovery from a more practical point of view. We present an approach to automatically build up the projective structure and motion from a sequence of images. The presented approach is sequential, which offers the advantage that corresponding features are not required to be visible in all views.

We assume that for each pair of consecutive views we are given a set of potentially corresponding feature points. The feature points are typically obtained by using a corner detector [19]. If the images are not too widely separated, corresponding features can be identified by comparing the local intensity neighborhoods of the feature points on a pixel-by-pixel basis. In this case, typically only feature points that have similar coordinates are compared. In case of video, it might be more appropriate to use a feature tracker [56] that follows features from frame to frame. For more widely separated views, more complex features would have to be used [39, 53, 69, 40].

3.7.1 Two-view geometry computation

The first step of our sequential structure and motion computation approach consists of computing the geometric relation between two consecutive views. As seen in Section 3.5.2, this consists of recovering the fundamental matrix. In principle, the linear method presented in Section 3.6.3 could be used. In this section, a practical algorithm is presented to compute the fundamental matrix from a set of corresponding points perturbed with noise and containing a significant proportion of outliers.

Linear algorithms

Before presenting the complete robust algorithm, we revisit the linear algorithm. Given a number of corresponding points, (3.14) can be used to compute F. This equation can be rewritten in the following form:

$$\begin{bmatrix} x_1x_2 & y_1x_2 & x_2 & x_1y_2 & y_1y_2 & y_2 & x_1 & y_1 & 1 \end{bmatrix} \mathbf{f} = 0, \tag{3.29}$$

with $\mathbf{x}_1 = [x_1\, y_1\, 1]^\top, \mathbf{x}_2 = [x_2\, y_2\, 1]^\top$ and $\mathbf{f}$ a vector containing the elements of the fundamental matrix. As discussed before, stacking eight or more of these equations allows for a linear solution. Even for seven corresponding points, the one parameter family of solutions obtained by solving the linear equations can be restricted to one or three solutions by enforcing the cubic rank-2 constraint $\det(F_1 + \lambda F_2) = 0$. Note also that, as pointed out by

Hartley [22], it is important to normalize the image coordinates before solving the linear equations. Otherwise, the columns of (3.29) would differ by several orders of magnitude and the error would be concentrated on the coefficients corresponding to the smaller columns. This normalization can be achieved by transforming the image center to the origin and scaling the images so that the coordinates have a standard deviation of unity.

Nonlinear algorithms

The result of the linear equations can be refined by minimizing the following criterion [70]:

$$\mathcal{C}(F) = \sum \left(d(\mathbf{x}_2, F\mathbf{x}_1)^2 + d(\mathbf{x}_1, F^\top \mathbf{x}_2)^2 \right), \tag{3.30}$$

with $d(.,.)$ representing the Euclidean distance in the image. This criterion can be minimized through a Levenberg-Marquardt algorithm [50]. An even better approach consists of computing the maximum likelihood estimation (for Gaussian noise) by minimizing the following criterion:

$$\mathcal{C}(F, \widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2) = \sum \left(d(\widehat{\mathbf{x}}_1, \mathbf{x}_1)^2 + d(\widehat{\mathbf{x}}_2, \mathbf{x}_2)^2 \right) \text{ with } \widehat{\mathbf{x}}_2^\top F \widehat{\mathbf{x}}_1 = 0. \tag{3.31}$$

Although in this case the minimization has to be carried out over a much larger set of variables, this can be achieved efficiently by taking advantage of the sparsity of the problem.

Robust algorithm

To compute the fundamental matrix from a set of matches that were automatically obtained from a pair of real images, it is important to explicitly deal with outliers. If the set of matches is contaminated with even a small set of outliers, the result of the above methods can become unusable. This is typical for all types of least-squares approaches (even nonlinear ones). The problem is that the quadratic penalty allows for a single outlier that is very far away from the true solution to completely bias the final result.

An approach that can be used to cope with this problem is the RANSAC algorithm that was proposed by Fischler and Bolles [17]. A minimal subset of the data, in this case seven point correspondences, is randomly selected from the set of potential correspondences, and the solution obtained from it is used to segment the remainder of the dataset in *inliers* and *outliers*. If the initial subset contains no outliers, most of the correct correspondences will support the solution. However, if one or more outliers are contained in

the initial subset, it is highly improbable that the computed solution will find a lot of support among the remainder of the potential correspondences, yielding a low "inlier" ratio. This procedure is repeated until a satisfying solution is obtained. This is typically defined as a probability in excess of 95% that a good subsample was selected. The expression for this probability is $\Gamma = 1-(1-\rho^p)^m$ with ρ the fraction of inliers, p the number of features in each sample, seven in this case, and m the number of trials (see Rousseeuw [51]).

Two-view geometry computation

The different algorithms described above can be combined to yield a practical algorithm to compute the two-view geometry from real images:

1. Compute initial set of potential correspondences (and set $\rho_{max} = 0, m = 0$).
2. While $(1 - (1 - \rho_{max}^7)^m) < 95\%$,
   (a) Randomly select a minimal sample (seven pairs of corresponding points),
   (b) Compute the solution(s) for F (yielding one or three solutions),
   (c) Determine percentage of inliers ρ (for all solutions),
   (d) Increment m, update $\rho_{\max}$ if $\rho_{\max} < \rho$.
3. Refine F based on all inliers.
4. Look for additional matches along epipolar lines.
5. Refine F based on all correct matches (preferably using (3.31)).

3.7.2 Structure and motion recovery

Once the epipolar geometry has been computed between all consecutive views, the next step consists of reconstructing the structure and motion for the whole sequence. Contrary to the factorization approach of Section 3.6.4, here a sequential approach is presented. First, the structure and motion is initialized for two views and then gradually extended toward the whole sequence. Finally, the solution is refined through a global minimization over all the unknown parameters.

Initializing the structure and motion

Initial motion computation Two images of the sequence are used to determine a reference frame. The world frame is aligned with the first camera.

The second camera is chosen so that the epipolar geometry corresponds to the retrieved fundamental matrix F:

$$\begin{array}{rcl} P_1 & = & [\quad I_{3\times 3} \quad | \quad \mathbf{0}_3 \quad] \\ P_2 & = & [\quad T_e F + \mathbf{e}\mathbf{v}^\top \quad | \quad \sigma \mathbf{e} \quad] \end{array} . \tag{3.32}$$

Eq. (3.32) is not completely determined by the epipolar geometry (i.e., F and $\mathbf{e}$), but has 4 more degrees of freedom (i.e., $\mathbf{v}$ and σ). The vector $\mathbf{v}$ determines the position of the reference plane (i.e., the plane at infinity in an affine or metric frame), and σ determines the global scale of the reconstruction. The location of the reference plane should not make any difference if the algorithm is projectively invariant. To achieve this, it is important to use homogeneous representations for all 3D entities and to only use image measurements for minimizations. The value for the parameter σ has no importance and can be fixed to one.

Initial structure computation Once two projection matrices have been fully determined, the matches can be reconstructed through triangulation. Due to noise, the lines of sight will not intersect perfectly. In the projective case, the minimizations should be carried out in the images and not in projective 3D space. Therefore, the distance between the reprojected 3D point and the image points should be minimized:

$$d(\mathbf{x}_1, P_1\mathbf{X})^2 + d(\mathbf{x}_2, P_2\mathbf{X})^2. \tag{3.33}$$

It was noted by Hartley and Sturm [23] that the only important choice is to select in which epipolar plane the point is reconstructed. Once this choice is made, it is trivial to select the optimal point from the plane. Since a bundle of epipolar planes has only one parameter, the dimension of the problem is reduced from three to one. Minimizing the following equation is thus equivalent to minimizing equation (3.33).

$$d(\mathbf{x}_1, \mathbf{l}_1(\lambda))^2 + d(\mathbf{x}_2, \mathbf{l}_2(\lambda))^2, \tag{3.34}$$

with $\mathbf{l}_1(\lambda)$ and $\mathbf{l}_2(\lambda)$ the epipolar lines obtained in function of the parameter λ describing the bundle of epipolar planes. It turns out (see [23]) that this equation is a polynomial of degree 6 in λ. The global minimum of equation (3.34) can thus easily be computed directly. In both images, the points on the epipolar line $\mathbf{l}_1(\lambda)$ and $\mathbf{l}_2(\lambda)$ closest to the points $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively are selected. Since these points are in epipolar correspondence, their lines of sight intersect exactly in a 3D point. In the case where (3.31) is minimized to obtain the fundamental matrix F, the procedure described here is unnecessary and the pairs $(\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2)$ can be reconstructed directly.

Updating the structure and motion

The previous section dealt with obtaining an initial reconstruction from two views. This section discusses how to add a view to an existing reconstruction. First, the pose of the camera is determined, then the structure is updated based on the added view, and finally new points are initialized.

Projective pose estimation For every additional view, the pose toward the preexisting reconstruction is determined. This is illustrated in Figure 3.14. It is assumed that the epipolar geometry has been computed between view $i-1$ and i. The matches that correspond to already reconstructed points are used to infer correspondences between 2D and 3D. Based on these, the projection matrix P_i is computed using a robust procedure similar to the

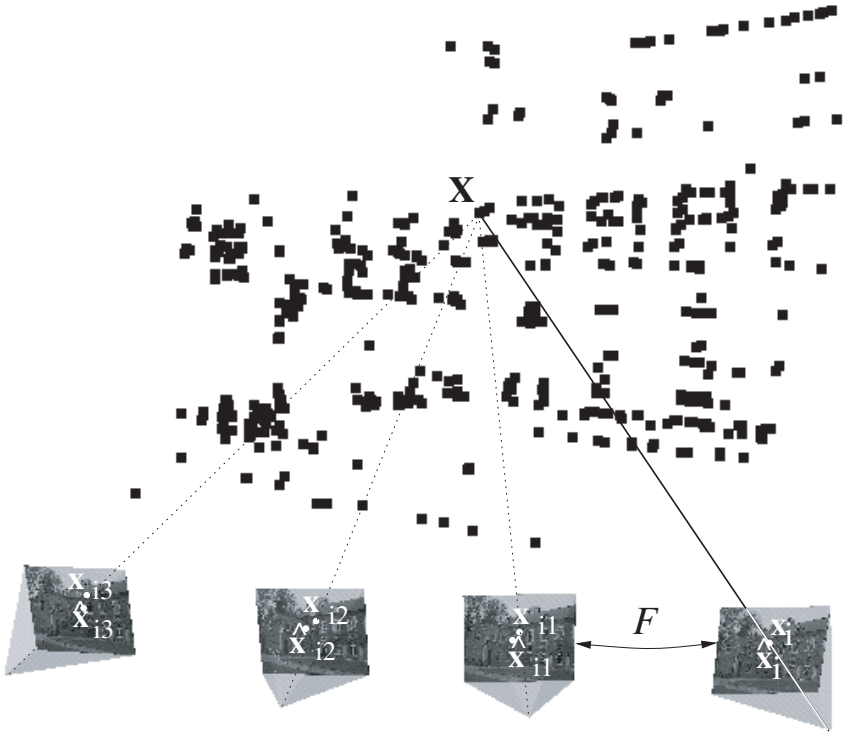

Figure 3.14. Image matches $(\mathbf{x}_{i-1}, \mathbf{x}_i)$ are found as described before. Since some image points, $\mathbf{x}_{i-1}$, relate to object points, $\mathbf{X}$, the pose for view i can be computed from the inferred matches $(\mathbf{X}, \mathbf{x}_i)$. A point is accepted as an inlier if a solution for $\widehat{\mathbf{X}}$ exists for which $d(P\widehat{\mathbf{X}}, \mathbf{x}_i) < 1$ for each view k in which $\mathbf{X}$ has been observed.

one laid out for computing the fundamental matrix. In this case, a minimal sample of six matches is needed to compute P_i. A point is considered an inlier if there exists a 3D point that projects sufficiently close to all associated image points. This requires refining the initial solution of $\mathbf{X}$ based on all observations, including the last. Because this is computationally expensive (remember that this has to be done for each generated hypothesis), it is advised to use a modified version of RANSAC that cancels the verification of unpromising hypothesis [5]. Once P_i has been determined, the projection of already reconstructed points can be predicted so that some additional matches can be obtained. This means that the search space is gradually reduced from the full image to the epipolar line to the predicted projection of the point.

This procedure relates only the image to the previous image. In fact, it is implicitly assumed that once a point gets out of sight, it will not come back. Although this is true for many sequences, this assumption does not always hold. Assume that a specific 3D point got out of sight, but that it becomes visible again in the two most recent views. This type of point could be interesting to avoid error accumulation. However, the naive approach would just reinstantiate a new independent 3D point. A possible solution to this problem was proposed in [35].

Refining and extending structure The structure is refined using an iterated linear reconstruction algorithm on each point. The scale factors can also be eliminated from (3.25) so that homogeneous equations in $\mathbf{X}$ are obtained:

$$\begin{array}{rcl} \mathtt{P}_3\mathbf{X}x - \mathtt{P}_1\mathbf{X} & = & 0 \\ \mathtt{P}_3\mathbf{X}y - \mathtt{P}_2\mathbf{X} & = & 0 \end{array} \tag{3.35}$$

with $\mathtt{P}_i$ the i-th row of P and (x, y) being the image coordinates of the point. An estimate of $\mathbf{X}$ is computed by solving the system of linear equations obtained from all views where a corresponding image point is available. To obtain a better solution, the criterion $\sum d(P\mathbf{X}, \mathbf{x})^2$ should be minimized. This can be approximately obtained by iteratively solving the following weighted linear least-squares problem:

$$\frac{1}{\mathtt{P}_3\widetilde{\mathbf{X}}} \begin{bmatrix} \mathtt{P}_3x - \mathtt{P}_1 \\ \mathtt{P}_3y - \mathtt{P}_2 \end{bmatrix} \mathbf{X} = 0, \tag{3.36}$$

where $\widetilde{\mathbf{X}}$ is the previous solution for $\mathbf{X}$. This procedure can be repeated a few times. By solving this system of equations through SVD, a normalized homogeneous point is automatically obtained. If a 3D point is not observed,

the position is not updated. In this case one can check if the point was seen in a sufficient number of views to be kept in the final reconstruction. This minimum number of views can, for example, be put to three. This avoids having an important number of outliers due to spurious matches.

Of course, in an image sequence some new features will appear in every new image. If point matches are available that were not related to an existing point in the structure, then a new point can be initialized, as described in Section 3.7.2.

Refining structure and motion

Once the structure and motion has been obtained for the whole sequence, it is recommended to refine it through a global minimization step so that a bias toward the initial views is avoided. A maximum likelihood estimation can be obtained through *bundle adjustment* [67, 57]. The goal is to find the parameters of the camera view P_k and the 3D points $\mathbf{X}_i$ for which the sum of squared distances between the observed image points $\mathtt{m}_{ki}$ and the reprojected image points $P_k(\mathbf{X}_i)$ is minimized. It is advised to extend the camera projection model to also take radial distortion into account. For m views and n points, the following criterion should be minimized:

$$\min_{P_k, \mathbf{X}_i} \sum_{k=1}^{m} \sum_{i=1}^{n} d(\mathbf{x}_{ki}, P_k(\mathbf{X}_i))^2. \tag{3.37}$$

If the errors on the localization of image features are independent and satisfy a zero-mean Gaussian distribution, then it can be shown that bundle adjustment corresponds to a maximum likelihood estimator. This minimization problem is huge; for example, for a sequence of 20 views and 100 points/view, a minimization problem in more than 6000 variables has to be solved (most of them related to the structure). A straightforward computation is obviously not feasible. However, the special structure of the problem can be exploited to solve the problem much more efficiently [67, 57]. The key reason for this is that a specific residual is only dependent on one point and one camera, which results in a very sparse structure for the normal equations.

Structure and motion recovery algorithm

To conclude this section, an overview of the structure and motion recovery algorithm is given. The whole procedure consists of the following steps:

1. Match or track points over the whole image sequence (see Section 3.5.2).

2. Initialize the structure and motion recovery:
   (a) Select two views that are suited for initialization,
   (b) Set up the initial frame,
   (c) Reconstruct the initial structure.
3. For every additional view,
   (a) Infer matches to the existing 3D structure,
   (b) Compute the camera pose using a robust algorithm,
   (c) Refine the existing structure,
   (d) Initialize new structure points.
4. Refine the structure and motion through bundle adjustment.

The results of this algorithm are the camera poses for all the views and the reconstruction of the interest points. For most applications, such as MatchMoving (aligning a virtual camera with the motion of a real camera; see Section 3.10.3), the camera poses are the most useful.

3.8 Autocalibration

As shown by Theorem 5, for a completely uncalibrated image sequence, the reconstruction is only determined up to a projective transformation. While it is true that the full calibration is often not available, some knowledge of the camera intrinsics is usually available. This knowledge can be used to recover the structure and motion up to a similarity transformation. This type of approach is called *autocalibration* or self-calibration in the literature. A first class of algorithms assumes constant, but unknown, intrinsic camera parameters [16, 21, 44, 28, 65]. Another class of algorithms assumes some intrinsic camera parameters to be known, while others can vary [47, 29]. Specific algorithms have also been proposed for restricted camera motion, such as pure rotation [20, 11], or restricted scene structure, such as planar scenes [66].

The absolute conic and its image

The central concept for autocalibration is the absolute conic. As stated in Proposition 3, the absolute conic allows us to identify the similarity structure in a projective space. In other words, if, given a projective reconstruction,

we could locate the conic corresponding to the absolute conic in the real world, this would be equivalent to recovering the structure of the observed scene up to a similarity. In this case, a transformation that transforms the absolute conic to its canonical representation in Euclidean space–that is, $\Omega' = diag(1, 1, 1, 0)$–would yield a reconstruction similar to the original (i.e., identical up to orientation, position, and scale).

As was seen in Proposition 6, the image of the absolute conic is directly related to the intrinsic camera parameters, and this independently of the choice of projective basis:

$$P\Omega' P^\top \sim KK^\top. \tag{3.38}$$

Therefore, constraints on the intrinsics can be used to constrain the location of the conic corresponding to the absolute conic. Most autocalibration algorithms are based on (3.38).

Critical motion sequences

Autocalibration is not always guaranteed to yield a unique solution. Depending on the available constraints on the intrinsics and on the camera motion, the remaining ambiguity on the reconstruction might be larger than a similarity. This problem was identified as the problem of *critical motion sequences*. The first complete analysis of the problem for constant intrinsic camera parameters was made by Sturm [61]. Analysis for some other cases can be found in [62, 43, 32]. It was also shown that in some cases, the ambiguity notwithstanding, correct novel views could be generated [45].

Linear autocalibration

In this section, we present a simple linear algorithm for autocalibration of cameras. The approach, published in [49], is related to the initial approach published in [46], but avoids most of the problems due to critical motion sequences by incorporating more a priori knowledge. As input, it requires a projective representation of the camera projection matrices.

As discussed in Section 3.4.3, for most cameras it is reasonable to assume that the pixels are close to square and that the principal point is close to the center of the image. The focal length (measured in pixel units) is typically

of the same order of magnitude as the image size. It is therefore a good idea to perform the following normalization:

$$P_N = K_N^{-1} P \text{ with } K_N = \begin{bmatrix} w+h & 0 & \frac{w}{2} \\ & w+h & \frac{h}{2} \\ & & 1 \end{bmatrix}, \tag{3.39}$$

where w and h are the width and height respectively of the image. After normalization, the focal length should be of the order of unity and the principal point should be close to the origin. The above normalization would scale a focal length of a 60mm lens to 1, and thus focal lengths in the range of 20mm to 180mm would end up in the range $[1/3, 3]$, and the principal point should now be close to the origin. The aspect ratio is typically around 1, and the skew can be assumed 0 for all practical purposes. Making this a priori knowledge more explicit and estimating reasonable standard deviations yields $f \approx rf \approx 1 \pm 3$, $u \approx v \approx 0 \pm 0.1$, $r \approx 1 \pm 0.1$, and $s = 0$, which approximately translates to the following expectations for ω':

$$\omega' \sim KK^\top = \begin{bmatrix} \gamma^2 f^2 + x_0^2 & x_0 y_0 & x_0 \\ x_0 y_0 & f^2 + y_0^2 & y_0 \\ x_0 & y_0 & 1 \end{bmatrix} \approx \begin{bmatrix} 1 \pm 9 & \pm 0.01 & \pm 0.1 \\ \pm 0.01 & 1 \pm 9 & \pm 0.1 \\ \pm 0.1 & \pm 0.1 & 1 \end{bmatrix}, \tag{3.40}$$

and $\omega'_{22}/\omega'_{11} \approx 1 \pm 0.2$. These constraints can also be used to constrain the left-hand side of (3.38). The uncertainty can be accounted for by weighting the equations, yielding the following set of constraints:

$$\begin{aligned} \tfrac{\nu}{9}\left(\mathtt{P}_1 \Omega' {\mathtt{P}_1}^\top - \mathtt{P}_3 \Omega' {\mathtt{P}_3}^\top\right) &= 0 \\ \tfrac{\nu}{9}\left(\mathtt{P}_2 \Omega' {\mathtt{P}_2}^\top - \mathtt{P}_3 \Omega' {\mathtt{P}_3}^\top\right) &= 0 \\ \tfrac{\nu}{0.2}\left(\mathtt{P}_1 \Omega' {\mathtt{P}_1}^\top - \mathtt{P}_2 \Omega' {\mathtt{P}_2}^\top\right) &= 0 \\ \tfrac{\nu}{0.1}\left(\mathtt{P}_1 \Omega' {\mathtt{P}_2}^\top\right) &= 0 \\ \tfrac{\nu}{0.1}\left(\mathtt{P}_1 \Omega' {\mathtt{P}_3}^\top\right) &= 0 \\ \tfrac{\nu}{0.01}\left(\mathtt{P}_2 \Omega' {\mathtt{P}_3}^\top\right) &= 0 \end{aligned} \tag{3.41}$$

with $\mathtt{P}_i$ the i-th row of P and ν a scale factor that can be set to 1. If for the solution $\mathtt{P}_3 \Omega' {\mathtt{P}_3}^\top$ varies widely for the different views, we might want to iterate with $\nu = (\mathtt{P}_3 \widetilde{\Omega}' {\mathtt{P}_3}^\top)^{-1}$, with $\widetilde{\Omega}'$ the result of the previous iteration. Since Ω' is a symmetric 4×4 matrix, it is linearly parametrized by 10 coefficients. An estimate of the dual absolute quadric Ω' can be obtained by solving the above set of equations for all views through homogeneous linear least-squares. The rank-3 constraint can be imposed by forcing the smallest singular value to zero. The upgrading transformation T can be obtained from $\text{diag}\,(1, 1, 1, 0) = T \Omega' T^\top$ by decomposition of Ω'.

Autocalibration refinement

This result can be further refined through bundle adjustment (see Section 3.7.2). In this case, the constraints on the intrinsics should be enforced during the minimization process. Constraints on the intrinsics can be enforced either exactly through parametrization or approximately by adding a residual for the deviation from the expected value in the global minimization process.

3.9 Dense Depth Estimation

With the camera calibration given for all viewpoints of the sequence, we can proceed with methods developed for calibrated structure from motion algorithms. The feature tracking algorithm already delivers a sparse surface model based on distinct feature points. This, however, is not sufficient to reconstruct geometrically correct and visually pleasing surface models. This task is accomplished by a dense disparity matching that estimates correspondences from the gray level images directly by exploiting additional geometrical constraints. The dense surface estimation is done in a number of steps. First, image pairs are rectified to the standard stereo configuration. Then, disparity maps are computed through a stereo matching algorithm. Finally, a multiview approach integrates the results obtained from several view pairs.

3.9.1 Rectification

Since the calibration between successive image pairs was computed, the epipolar constraint that restricts the correspondence search to a 1D search range can be exploited. Image pairs can be warped so that epipolar lines coincide with image scan lines. The correspondence search is then reduced to a matching of the image points along each image scanline. This results in a dramatic increase of the computational efficiency of the algorithms by enabling several optimizations in the computations.

For some motions (i.e., when the epipole is located in the image), standard rectification based on planar homographies [2] is not possible, and a more advanced procedure should be used. The approach proposed in [48] avoids this problem. The method works for all possible camera motions. The key idea is to use polar coordinates with the epipole as origin. Corresponding lines are given through the epipolar geometry. By taking the orientation [36] into account, the matching ambiguity is reduced to half epipolar lines. A minimal image size is achieved by computing the angle between two consec-

utive epipolar lines that correspond to rows in the rectified images to have the worst case pixel on the line preserve its area.

Some examples A first example comes from the *castle* sequence. In Figure 3.15, an image pair and the associated rectified image pair are shown. A second example was filmed with a handheld digital video camera in the Béguinage in Leuven. Due to the narrow streets, only forward motion is feasible. In this case, the full advantage of the polar rectification scheme becomes clear, since this sequence could not have been handled through traditional planar rectification. An example of a rectified image pair is given in Figure 3.16. Note that the whole left part of the rectified images corresponds to the epipole. On the right side of this figure, a model that was obtained by combining the results from several image pairs is shown.

Figure 3.15. Original image pair (left) and rectified image pair (right).

3.9.2 Stereo matching

The goal of a dense stereo algorithm is to compute the corresponding pixel for every pixel of an image pair. After rectification, the correspondence search is limited to corresponding scanlines. As illustrated in Figure 3.17, finding

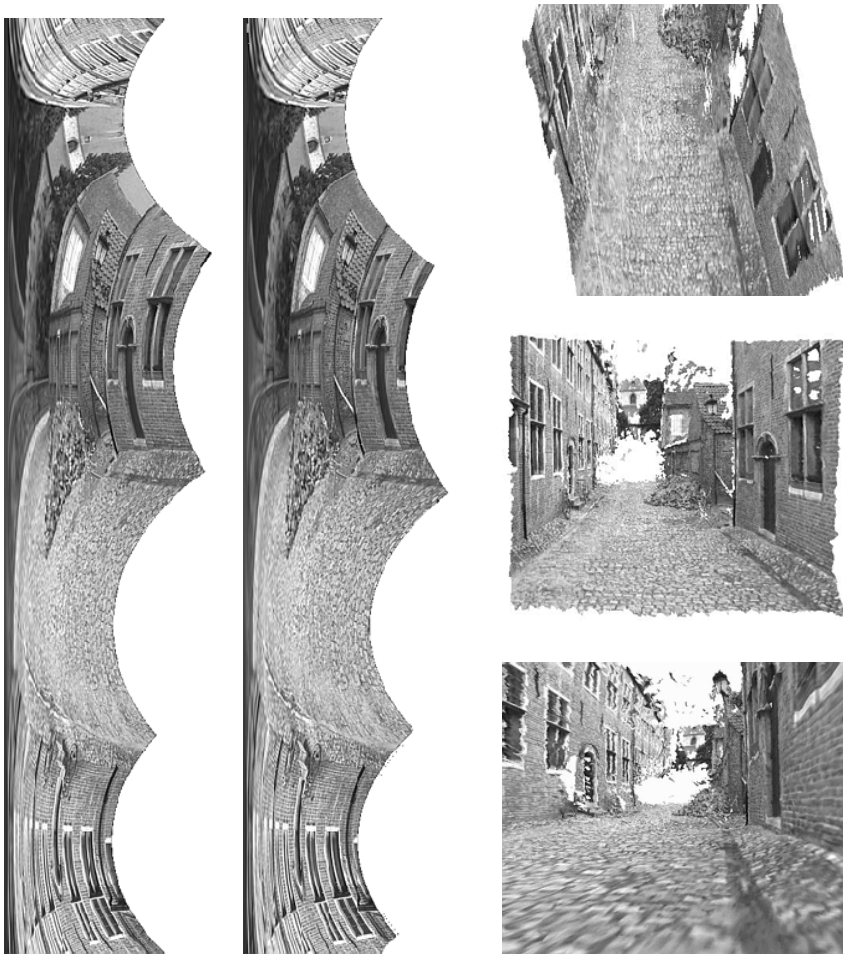

Figure 3.16. Rectified image pair (left) and some views of the reconstructed scene (right).

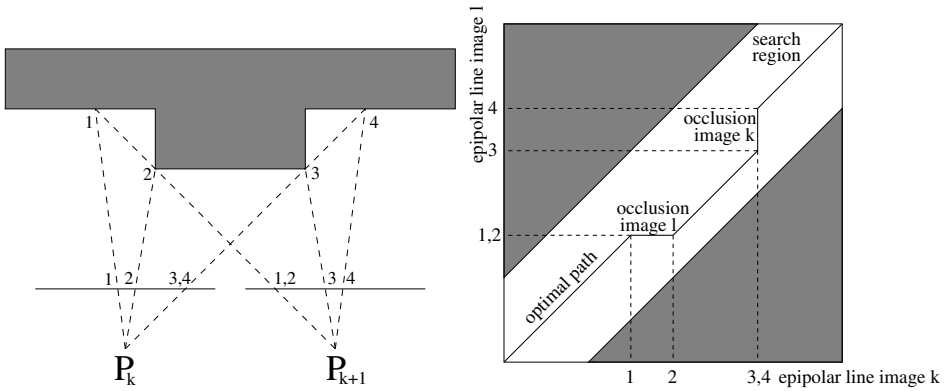

Figure 3.17. The ordering constraint (left) and dense matching as a path search problem (right).

the correspondences for a pair of scanlines can be seen as a path search problem. Besides the epipolar geometry, other constraints, like preserving the order of neighboring pixels, bidirectional uniqueness of the match, and detection of occlusions, can be exploited. In most cases, it is also possible to limit the search to a certain disparity range (an estimate of this range can be obtained from the reconstructed 3D feature points). Besides these constraints, a stereo algorithm should also take into account the similarity between corresponding points and the continuity of the surface. It is possible to compute the optimal path, taking all the constraints into account, using dynamic programming [8, 12, 41]. Other computationally more expensive approaches also take continuity across scanlines into account. Real-time approaches, on the other hand, estimate the best match independently for every pixel. A complete taxonomy of stereo algorithms can be found in [52].

3.9.3 Multiview linking

The pairwise disparity estimation allows us to compute image-to-image correspondences between adjacent rectified image pairs and independent depth estimates for each camera viewpoint. An optimal joint estimate is achieved by fusing all independent estimates into a common 3D model. The fusion can be performed in an economical way through controlled correspondence linking (see Figure 3.18). A point is transferred from one image to the next as follows:

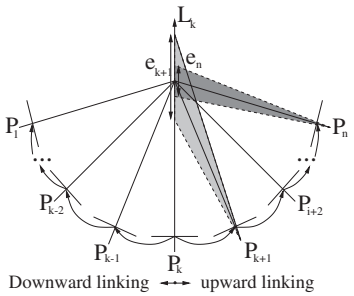

Figure 3.18. Depth fusion and uncertainty reduction from correspondence linking.

$$\mathbf{x}_2 = R'^{-1}(R(\mathbf{x}_1) + D(R(\mathbf{x}_1)), \tag{3.42}$$

with $R(.)$ and $R'(.)$ functions that map points from the original image into the rectified image and $D(.)$ a function that corresponds to the disparity map. When the depth obtained from the new image point $\mathbf{x}_2$ is outside the confidence interval, the linking is stopped; otherwise, the result is fused with the previous values through a Kalman filter. This approach is discussed in more detail in [34]. This approach combines the advantages of small baseline and wide baseline stereo. The depth resolution is increased through the combination of multiple viewpoints and large global baseline, while the matching is simplified through the small local baselines. It can provide a very dense depth map by avoiding most occlusions. Due to multiple observations of a single surface point, the texture can be enhanced and noise and highlights can be removed.

3.10 Visual Modeling

In the previous sections, we explained how the camera motion and calibration and the depth estimates for (almost) every pixel could be obtained. This yields all the necessary information to build different types of visual models. In this section, several types of models are considered. First, the construction of texture-mapped 3D surface models is discussed. Then, a combined image- and geometry-based approach is presented that can render models ranging from pure plenoptic to view-dependent texture and geometry models. Finally, the possibility of combining real and virtual scenes is treated.

3.10.1 3D surface reconstruction

The 3D surface is approximated by a triangular mesh to reduce geometric complexity and to tailor the model to the requirements of computer graphics visualization systems. A simple approach consists of overlaying a 2D triangular mesh on top of one of the images and then building a corresponding 3D mesh by placing the vertices of the triangles in 3D space according to the values found in the corresponding depth map. To reduce noise, it is recommended to first smooth the depth image (the kernel can be chosen of the same size as the mesh triangles). The image itself can be used as a texture map. While projective texture mapping would normally be required, the small size of the triangles allows us to use standard (affine) texture mapping (the texture coordinates are trivially obtained as the 2D coordinates of the vertices).

It can happen that for some vertices no depth value is available or that the confidence is too low. In these cases, the corresponding triangles are not reconstructed. The same happens when triangles are placed over discontinuities. This is achieved by selecting a maximum angle between the normal of a triangle and the line-of-sight through its center (e.g., 85 degrees). This simple approach works very well on the dense depth maps as obtained through multiview linking. The surface reconstruction approach is illustrated in Figure 3.19.

A further example is shown in Figure 3.20. The video sequence was recorded with a handheld camcorder on an archaeological site in Sagalasso, Turkey (courtesy of Marc Waelkens). It shows a decorative medusa head that was part of a monumental fountain. The video sequence was processed fully automatically by using the algorithms discussed in sections 3.7, 3.8, 3.9, and 3.10. From the bundle adjustment and the multiview linking, the accuracy

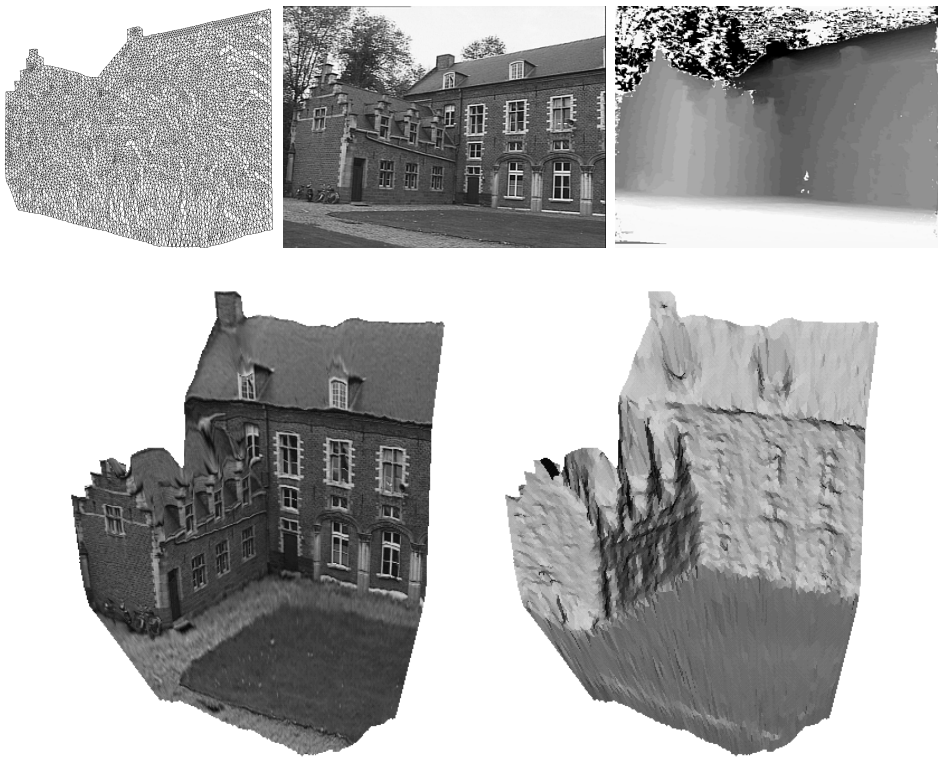

Figure 3.19. Surface reconstruction approach (top): Triangular mesh is overlaid on top of the image. The vertices are back-projected in space according to the depth values. From this, a 3D surface model is obtained (bottom).

was estimated to be of $\frac{1}{500}$ (compared to the size of the reconstructed object). This has to be compared with the image resolution of 720×576. Note that the camera was uncalibrated and, besides the unknown focal length and principal point, has significant radial distortion and an aspect ratio different from one (i.e., 1.09), which were all automatically recovered from the video sequence.

To reconstruct more complex shapes, it is necessary to combine results from multiple depth maps. The simplest approach consists of generating separate models independently and then loading them together in the graphics system. Since all depth maps are located in a single coordinate frame, registration is not an issue. Often it is interesting to integrate the different meshes into a single mesh. A possible approach is given in [10].

Figure 3.20. 3D-from-video: one of the video frames (upper-left), recovered structure and motion (upper-right), textured and shaded 3D model (middle), and more views of textured 3D model (bottom).

3.10.2 Image-based rendering

In the previous section, we presented an approach to construct 3D models. If the goal is to generate novel views, other approaches are available. In recent years, a multitude of image-based approaches have been proposed that render images from images without the need for an explicit intermediate 3D model. The best known approaches are lightfield and lumigraph rendering [37, 18] and image warping [4, 54, 1].

Here we briefly introduce an approach to render novel views directly from images recorded with a handheld camera. If available, some depth information can also be used to refine the underlying geometric assumption. A more extensive discussion of this work can be found in [25, 33]. A related approach was presented in [3]. A lightfield is the collection of the lightrays corresponding to all the pixels in all the recorded images. Therefore, rendering from a lightfield consists of looking up the "closest" ray(s) passing through every pixel of the novel view. Determining the closest ray consists of two steps: (1) determining in which views the closest rays are located, and (2) within that view selecting the ray that intersects the implicit geometric assumption in the same point. For example, if the assumption is that the scene is far away, the corresponding implicit geometric assumption might be Π_∞ so that parallel rays would be selected.

In our case, the view selection works as follows. All the camera projection centers are projected in the novel view and Delaunay triangulated. For every pixel within a triangle, the recorded views corresponding to the three vertices are selected as "closest" views. If the implicit geometric assumption is planar, a homography relates the pixels in the novel view with those in a recorded view. Therefore, a complete triangle in the novel view can be efficiently drawn using texture mapping. The contributions of the three cameras can be combined using alpha blending. The geometry can be approximated by one plane for the whole scene, one plane per camera triple, or by several planes for one camera triple. The geometric construction is illustrated in Figure 3.21.

This approach is illustrated in Figure 3.22 with an image sequence of 187 images recorded by waving a camera over a cluttered desk. In the lower part of Figure 3.22, a detail of a view is shown for the different methods. In the case of one global plane (left image), the reconstruction is sharp where the approximating plane intersects the actual scene geometry. The reconstruction is blurred where the scene geometry diverges from this plane. In the case of local planes (middle image), at the corners of the triangles, the reconstruction is almost sharp, because there the scene geometry is considered

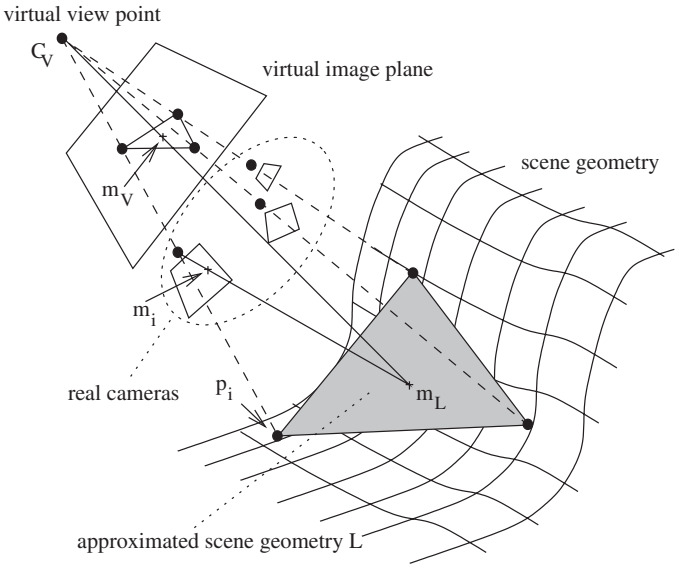

Figure 3.21. Drawing triangles of neighboring projected camera centers and approximated scene geometry.

directly. Within a triangle, ghosting artifacts occur where the scene geometry diverges from the particular local plane. If these triangles are subdivided (right image), these artifacts are reduced further.

3.10.3 Match-moving

Another interesting application of the presented algorithms consists of adding virtual elements to real video. This has important applications in the entertainment industry and several products, such as 2d3's *boujou* and RealViz's *MatchMover*, exist that are based on the techniques described in Section 3.7 and Section 3.8. The key issue consists of registering the motion of the real and the virtual camera. The presented techniques can be used to compute the motion of the camera in the real world. This allows us to restrict the problem of introducing virtual objects in video to determine the desired position, orientation, and scale with respect to the reconstructed camera motion. More details on this approach can be found in [7]. Note that to achieve a seamless integration, other effects such as occlusion and lighting must also be taken care of.

An example is shown in Figure 3.23. The video shows the remains of one of the ancient monumental fountains of Sagalassos. A virtual reconstruction

Figure 3.22. Top: image of the *desk* sequence and sparse structure-and-motion result (left), artificial view rendered using one plane per image triple (right). Details of rendered images showing the differences between the approaches (bottom): one global plane of geometry (left), one local plane for each image triple (middle), and refinement of local planes (right).

of the monument was overlaid on the original frame. The virtual camera was setup to mimic exactly the computed motion and calibration of the original camera.

3.11 Conclusion

In this chapter, the relations between multiple views of a 3D scene were discussed. The discussion started by introducing the basic concepts of projective geometry and tensor calculus. Then, the pinhole camera model, the epipolar geometry, and the multiple view tensors were discussed. Next, approaches

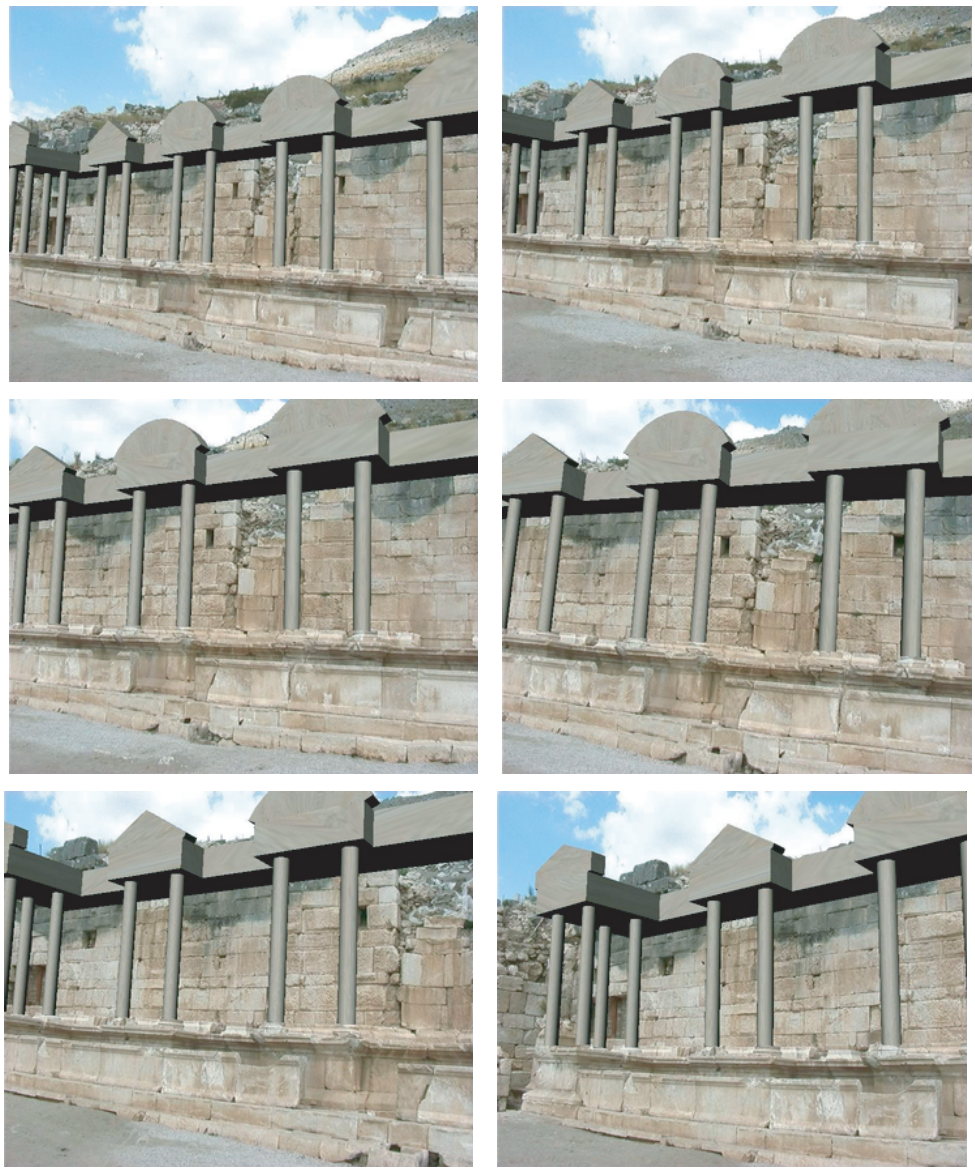

Figure 3.23. Augmented video: six frames (out of 250) from a video where a virtual reconstruction of an ancient monument has been added.

that rely on those concepts to recover both the structure and motion from a sequence of images were presented and illustrated with some real-world examples and applications.

Acknowledgment

The authors wish to acknowledge the financial support of the EU projects InViews, Attest, Vibes, and Murale, as well as the contributions of their (former) colleagues in Leuven and Lund, where most of the presented work was carried out. Luc Van Gool, Maarten Vergauwen, Frank Verbiest, Kurt Cornelis, Jan Tops, Reinhard Koch, and Benno Heigl have contributed to generate the results that illustrate this chapter.

Bibliography

[1] S. Avidan and A. Shashua. Novel view synthesis in tensor space. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1034–1040, 1997.

[2] N. Ayache. *Artificial Vision for Mobile Robots: Stereo Vision and Multi-sensory Perception*. MIT Press, 1991.

[3] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured lumigraph rendering. In *Proc. SIGGRAPH*, 2001.

[4] S. Chen and L. Williams. View interpolation for image synthesis. *Computer Graphics*, 27(Annual Conference Series):279–288, 1993.

[5] O. Chum and J. Matas. Randomized ransac and t(d,d) test. In *Proc. British Machine Vision Conference*, 2002.

[6] K. Cornelis, M. Pollefeys, M. Vergauwen, and L. Van Gool. Augmented reality from uncalibrated video sequences. In A. Zisserman, M. Pollefeys, L. Van Gool, and A. Fitzgibbon (Eds.), *3D Structure from Images–SMILE 2000*, volume 2018, pages 150–167. Springer-Verlag, 2001.

[7] K. Cornelis, M. Pollefeys, M. Vergauwen, F. Verbiest, and L. Van Gool. Tracking based structure and motion recovery for augmented video productions. In *Proceedings ACM Symposium on Virtual Reality Software and Technology–VRST 2001*, pages 17–24, November 2001.

[8] I. Cox, S. Hingoraini, and S. Rao. A maximum likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63(3):542–567, May 1996.

[9] H. S. M. Coxeter. *Projective Geometry*. Blaisdell Publishing Company, 1964.

[10] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. SIGGRAPH '96*, pages 303–312, 1996.

[11] L. de Agapito, R. Hartley, and E. Hayman. Linear selfcalibration of a rotating and zooming camera. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, volume I, pages 15–21, June 1999.

[12] L. Falkenhagen. Depth estimation from stereoscopic image pairs assuming piecewise continuous surfaces. In *Proc. of European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, November 1994.

[13] O. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *Proc. European Conf. on Computer Vision*, pages 563–578, 1992.

[14] O. Faugeras. *Three-Dimensional Computer Vision*. MIT Press, 1993.

[15] O. Faugeras and Q.-T. Luong. *The Geometry of Multiple Images*. MIT Press, 2001.

[16] O. Faugeras, Q.-T. Luong, and S. Maybank. Camera self-calibration: Theory and experiments. In *Computer Vision–ECCV'92*, volume 588 of *Lecture Notes in Computer Science*, pages 321–334. Springer-Verlag, 1992.

[17] M. Fischler and R. Bolles. Random sampling consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Commun. Assoc. Comp. Mach.*, 24:381–395, 1981.

[18] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The Lumigraph. In *Proc. SIGGRAPH '96*, pages 43–54. ACM Press, New York, 1996.

[19] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, pages 147–151, 1988.

[20] R. Hartley. An algorithm for self calibration from several views. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, pages 908–912, 1994.

[21] R. Hartley. Euclidean reconstruction from uncalibrated views. In J. L. Mundy, A. Zisserman, and D. Forsyth (Eds.), *Applications of Invariance in Computer Vision*, volume 825 of *Lecture Notes in Computer Science*, pages 237–256. Springer-Verlag, 1994.

[22] R. Hartley. In defense of the eight-point algorithm. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(6):580–593, June 1997.

[23] R. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997.

[24] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[25] B. Heigl, R. Koch, M. Pollefeys, J. Denzler, and L. Van Gool. Plenoptic modeling and rendering from image sequences taken by hand-held camera. In *Proc. DAGM*, pages 94–101, 1999.

[26] A. Heyden. *Geometry and Algebra of Multiple Projective Transformations*. PhD thesis, Lund University, Sweden, 1995.

[27] A. Heyden. Tensorial properties of multilinear constraints. *Mathematical Methods in the Applied Sciences*, 23:169–202, 2000.

[28] A. Heyden and K. Åström. Euclidean reconstruction from constant intrinsic parameters. In *Proc. 13th International Conference on Pattern Recognition*, pages 339–343. IEEE Computer Soc. Press, 1996.

[29] A. Heyden and K. Åström. Euclidean reconstruction from image sequences with varying and unknown focal length and principal point. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 438–443. IEEE Computer Soc. Press, 1997.

[30] A. Heyden and K. Åström. Algebraic properties of multilinear constraints. *Mathematical Methods in the Applied Sciences*, 20:1135–1162, 1997.

[31] F. Kahl. Critical motions and ambiguous euclidean reconstructions in autocalibration. In *Proc. International Conference on Computer Vision*, pages 469–475, 1999.

[32] F. Kahl, B. Triggs, and K. Åström. Critical motions for autocalibration when some intrinsic parameters can vary. *Journal of Mathematical Imaging and Vision*, 13(2):131–146, 2000.

[33] R. Koch, B. Heigl, and M. Pollefeys. Image-based rendering from uncalibrated lightfields with scalable geometry. In G. Gimel'farb, R. Klette, and T. Huang (Eds.), *Multi-Image Analysis*, volume 2032 of *Lecture Notes in Computer Science*, pages 51–66. Springer-Verlag, 2001.

[34] R. Koch, M. Pollefeys, and L. Van Gool. Multi viewpoint stereo from uncalibrated video sequences. In *Proc. European Conference on Computer Vision*, pages 55–71, 1998.

[35] R. Koch, M. Pollefeys, B. Heigl, L. Van Gool, and H. Niemann. Calibration of hand-held camera sequences for plenoptic modeling. In *Proc. International Conference on Computer Vision*, pages 585–591, 1999.

[36] S. Laveau and O. Faugeras. Oriented projective geometry for computer vision. In B. Buxton and R. Cipolla (Eds.), *Proc. European Conference on Computer Vision, Lecture Notes in Computer Science*, Vol. 1064, pages 147–156. Springer-Verlag, 1996.

[37] M. Levoy and P. Hanrahan. Lightfield rendering. In *Proc. SIGGRAPH '96*, pages 31–42. ACM Press, 1996.

[38] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[39] D. Lowe. Object recognition from local scale-invariant features. In *Proc. International Conference on Computer Vision*, pages 1150–1157, 1999.

[40] J. Matas, S. Obdrzalek, and O. Chum. Local affine frames for wide-baseline stereo. In *Proc. 16th International Conference on Pattern Recognition*, volume 4, pages 363–366, 2002.

[41] G. Van Meerbergen, M. Vergauwen, M. Pollefeys, and L. Van Gool. A hierarchical symmetric stereo algorithm using dynamic programming. *International Journal on Computer Vision*, 47(1/2/3):275–285, 2002.

[42] N. Myklestad. *Cartesian Tensors–the Mathematical Language of Engineering*. Van Nostrand, Princeton, 1967.

[43] M. Pollefeys. *Self-calibration and metric 3D reconstruction from uncalibrated image sequences*. PhD thesis, Katholieke Universiteit Leuven, 1999.

[44] M. Pollefeys and L. Van Gool. Stratified self-calibration with the modulus constraint. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):707–724, 1999.

[45] M. Pollefeys and L. Van Gool. Do ambiguous reconstructions always give ambiguous images? In *Proc. International Conference on Computer Vision*, pages 187–192, 2001.

[46] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proc. International Conference on Computer Vision*, pages 90–95. Narosa Publishing House, 1998.

[47] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. *International Journal of Computer Vision*, 32(1):7–25, 1999.

[48] M. Pollefeys, R. Koch, and L. Van Gool. A simple and efficient rectification method for general motion. In *Proc. International Conference on Computer Vision*, pages 496–501, 1999.

[49] M. Pollefeys, F. Verbiest, and L. Van Gool. Surviving dominant planes in uncalibrated structure and motion recovery. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen (Eds.), *7th European Conference on Computer Vision*, volume 2351 of *Lecture Notes in Computer Science*, pages 837–851, 2002.

[50] W. Press, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[51] P. Rousseeuw. *Robust Regression and Outlier Detection*. Wiley, 1987.

[52] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1/2/3):7–42, April–June 2002.

[53] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE PAMI*, 19(5):530–534, 1997.

[54] S. Seitz and C. Dyer. View morphing. *Computer Graphics*, Annual Conference Series, 30:21–30, 1996.

[55] A. Shashua. Trilinearity in visual recognition by alignment. In *Proc. European Conf. on Computer Vision*, 1994.

[56] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[57] C. Slama. *Manual of Photogrammetry*. 4th ed., American Society of Photogrammetry, USA, 1980.

[58] B. Spain. *Tensor Calculus*. University Mathematical Texts, Oliver and Boyd, Edingburgh, 1953.

[59] G. Sparr. Simultaneous reconstruction of scene structure and camera locations from uncalibrated image sequences. In *Proc. International Conf. on Pattern Recognition*, 1996.

[60] M. E. Spetsakis and J. Aloimonos. A unified theory of structure from motion. In *Proc. DARPA IU Workshop*, 1990.

[61] P. Sturm. Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction. In *Proc. Conference on Computer Vision and Pattern Recognition*, pages 1100–1105. IEEE Computer Society Press, 1997.

[62] P. Sturm. Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length. In *Proc. 10th British Machine Vision Conference*, pages 63–72, 1999.

[63] P. Sturm. Critical motion sequences for the self-calibration of cameras and stereo systems with variable focal length. In *Proc. BMVC*, 1999.

[64] B. Triggs. Matching constraints and the joint image. In *Proc. Int. Conf. on Computer Vision*, 1995.

[65] B. Triggs. The absolute quadric. In *Proc. 1997 Conference on Computer Vision and Pattern Recognition*, pages 609–614. IEEE Computer Society Press, 1997.

[66] B. Triggs. Autocalibration from planar scenes. In *ECCV*, volume I, pages 89–105, June 1998.

[67] B. Triggs, P. McLauchlan, R. Hartley, and A. Fiztgibbon. Bundle adjustment– A modern synthesis. In R. Szeliski, B. Triggs, A. Zisserman (Eds.), *Vision Algorithms: Theory and Practice*, LNCS Vol. 1883, pages 298–372. Springer-Verlag, 2000.

[68] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *Proceedings of SIGGRAPH '94*, pages 311–318, 1994.

[69] T. Tuytelaars and L. Van Gool. Wide baseline stereo based on local, affinely invariant regions. In *British Machine Vision Conference*, pages 412–422, 2000.

[70] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence Journal*, 78:87–119, October 1995.