Multiview Geometry for Camera Networks

Richard J. Radke

Department of Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute
Troy, NY, USA

Abstract

Designing computer vision algorithms for camera networks requires an understanding of how images of the same scene from different viewpoints are related. This chapter introduces the basics of multiview geometry in computer vision, including image formation and camera matrices, epipolar geometry and the fundamental matrix, projective transformations, and N-camera geometry. We also discuss feature detection and matching, and describe basic estimation algorithms for the most common problems that arise in multiview geometry.

Key words: image formation, epipolar geometry, projective transformations, structure from motion, feature detection and matching, camera networks
PACS: 42.30.-d, 42.30.Sy, 42.30.Tz, 42.30.Va, 42.79.Pw, 02.40.Dr, 02.60.Pn, 07.05.Pj

1 Introduction

Multi-camera networks are emerging as valuable tools for safety and security applications in environments as diverse as nursing homes, subway stations, highways, natural disaster sites, and battlefields. While early multi-camera networks were contained in lab environments and were fundamentally under the control of a single processor (e.g., [1]), modern multi-camera networks are composed of many spatially-distributed cameras that may have their own processors or even power sources. To design computer vision algorithms that make the best use of these cameras' data, it is critical to thoroughly understand the imaging process of a single camera, and the geometric relationships involved among pairs or collections of cameras.

Our overall goal in this chapter is to introduce the basic terminology of multiview geometry, as well as to describe best practices for several of the most

common and important estimation problems. We begin in Section 2 by discussing the perspective projection model of image formation and the representation of image points, scene points, and camera matrices. In Section 3, we introduce the important concept of the epipolar geometry that relates a pair of perspective cameras, and its representation by the fundamental matrix. Section 4 describes projective transformations, which typically arise in camera networks that observe a common ground plane. In Section 5, we briefly discuss algorithms for detecting and matching feature points between images, a prerequisite for many of the estimation algorithms we consider. Section 6 discusses the general geometry of N cameras, and its estimation using factorization and structure-from-motion techniques. Finally, Section 7 concludes the chapter with pointers to further print and online resources that go into more detail on the problems introduced here.

2 Image Formation

In this section, we describe the basic perspective image formation model, which for the most part accurately reflects the phenomena observed in images taken by real cameras. Throughout the chapter, we denote scene points by $\mathbf{X} = (X, Y, Z)$, image points by $u = (x, y)$, and camera matrices by P.

2.1 Perspective Projection

An idealized "pinhole" camera $\mathcal{C}$ is described by:

(1) A center of projection $C \in \mathbb{R}^3$
(2) A focal length $f \in \mathbb{R}^+$
(3) An orientation matrix $R \in SO(3)$.

The camera's center and orientation are described with respect to a world coordinate system on $\mathbb{R}^3$. A point $\mathbf{X}$ expressed in the world coordinate system as $\mathbf{X} = (X_o, Y_o, Z_o)$ can be expressed in the camera coordinate system of $\mathcal{C}$ as

$$\mathbf{X} = \begin{bmatrix} X_{\mathcal{C}} \\ Y_{\mathcal{C}} \\ Z_{\mathcal{C}} \end{bmatrix} = R \left(\begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix} - C \right). \tag{1}$$

The purpose of a camera is to capture a two-dimensional image of a three-dimensional scene $\mathcal{S}$, i.e., a collection of points in $\mathbb{R}^3$. This image is produced by

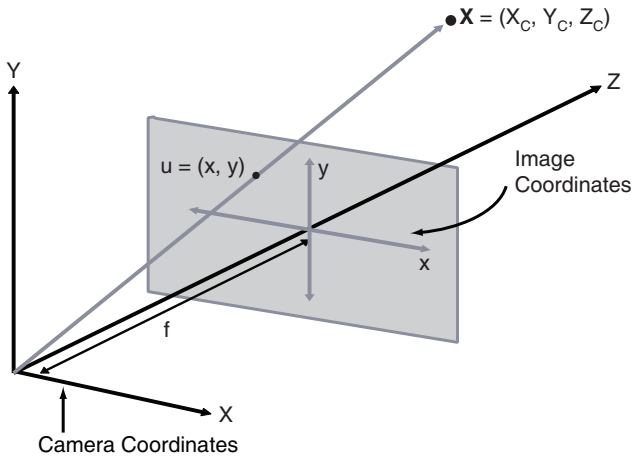


Fig. 1. A pinhole camera uses perspective projection to represent a scene point $\mathbf{X} \in \mathbb{R}^3$ as an image point $u \in \mathbb{R}^2$.

perspective projection as follows. Each camera $\mathcal{C}$ has an associated image plane $\mathcal{P}$, located in the camera coordinate system at $Z_\mathcal{C} = f$. As illustrated in Figure 1, the image plane inherits a natural orientation and two-dimensional coordinate system from the camera coordinate system's XY-plane. It is important to note that the three-dimensional coordinate systems in this derivation are left-handed. This is a notational convenience, implying that the image plane lies between the center of projection and the scene, and that scene points have positive $Z_\mathcal{C}$ coordinates.

A scene point $\mathbf{X} = (X_\mathcal{C}, Y_\mathcal{C}, Z_\mathcal{C})$ is projected onto the image plane $\mathcal{P}$ at the point $u = (x, y)$ by the perspective projection equations

$$x = f\frac{X_\mathcal{C}}{Z_\mathcal{C}} \qquad y = f\frac{Y_\mathcal{C}}{Z_\mathcal{C}}. \tag{2}$$

The image $\mathcal{I}$ that is produced is a map from $\mathcal{P}$ into some color space. The color of a point is typically a real-valued (grayscale) intensity or a triplet of RGB or YUV values. While the entire ray of scene points $\{\lambda(x, y, f) | \lambda > 0\}$ is projected to the image coordinate (x, y) by (2), the point on this ray that gives (x, y) its color in the image $\mathcal{I}$ is the one closest to the image plane (i.e., that point with minimal λ). This point is said to be visible; any scene point further along on the same ray is said to be occluded.

For real cameras, the relationship between the color of image points and the color of scene points is more complicated. To simplify matters, we often assume that scene points have the same color regardless of the viewing angle (this is called the Lambertian assumption), and that the color of an image point is the same as the color of a single corresponding scene point. In practice, the colors of corresponding image and scene points are different due to a host of factors in a real imaging system. These include the point spread function,

color space, and dynamic range of the camera, as well as non-Lambertian or semi-transparent objects in the scene. For more detail on the issues involved in image formation, see [2,3].

2.2 Camera Matrices

Frequently, an image coordinate (x, y) is represented by the *homogeneous coordinate* $\lambda(x, y, 1)$, where $\lambda \neq 0$. The image coordinate of a homogeneous coordinate (x, y, z) can be recovered as $(\frac{x}{z}, \frac{y}{z})$ when $z \neq 0$. Similarly, any scene point (X, Y, Z) can be represented in homogeneous coordinates as $\lambda(X, Y, Z, 1)$, where $\lambda \neq 0$. We use the symbol $\sim$ to denote the equivalence between a homogeneous coordinate and a non-homogeneous one.

A camera $\mathcal{C}$ with parameters (C, f, R) can be represented by a 3×4 matrix $P_{\mathcal{C}}$ that multiplies a scene point expressed as a homogeneous coordinate in $\mathbb{R}^4$ to produce an image point expressed as a homogeneous coordinate in $\mathbb{R}^3$. When the scene point is expressed in the world coordinate system, the matrix P is given by

$$P_{\mathcal{C}} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} [R \;\; | \;\; -RC] \,. \tag{3}$$

Here, the symbol | denotes the horizontal concatenation of two matrices. Then

$$\begin{aligned} P_{\mathcal{C}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} &= \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} [R \;\; | \;\; -RC] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} R \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - C \right) \qquad = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{\mathcal{C}} \\ Y_{\mathcal{C}} \\ Z_{\mathcal{C}} \end{bmatrix} \\ &= \begin{bmatrix} fX_{\mathcal{C}} \\ fY_{\mathcal{C}} \\ Z_{\mathcal{C}} \end{bmatrix} \qquad \sim \begin{bmatrix} f\frac{X_{\mathcal{C}}}{Z_{\mathcal{C}}} \\ f\frac{X_{\mathcal{C}}}{Z_{\mathcal{C}}} \end{bmatrix} . \end{aligned}$$

We often state this relationship succinctly as

$$u \sim P\mathbf{X}. \tag{4}$$

2.2.1 Intrinsic and Extrinsic Parameters

We note that the camera matrix can be factored as

$$P = K\ R\ [I\ \ |\ \ -C]. \tag{5}$$

The matrix K contains the *intrinsic parameters* of the camera, while the variables R and C comprise the *extrinsic parameters* of the camera, specifying its position and orientation in the world coordinate system. While the intrinsic parameter matrix K in (3) was just a diagonal matrix containing the focal length, a more general camera can be constructed using a K matrix of the form

$$K = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & s/m_x & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}. \tag{6}$$

In addition to the focal length of the camera f, this intrinsic parameter matrix includes m_x and m_y, the number of pixels per x and y unit of image coordinates, respectively; (p_x, p_y), the coordinates of the principal point of the image, and s, the skew (deviation from rectangularity) of the pixels. For high-quality cameras, the pixels are usually square, the skew is typically 0, and the principal point can often be approximated by the image origin.

A general camera matrix has 11 degrees of freedom (since it is only defined up to a scale factor). The camera center and the rotation matrix each account for 3 degrees of freedom, leaving 5 degrees of freedom for the intrinsic parameters.

2.2.2 Extracting Camera Parameters from P

Often, we begin with an estimate of a 3×4 camera matrix P and want to extract the intrinsic and extrinsic parameters from it. The homogeneous coordinates of the camera center C can be simply extracted as the right-hand null vector of P (i.e., a vector satisfying $PC = 0$). If we denote the left 3×3 block of P as M, then we can factor $M = KR$ where K is upper triangular and R is orthogonal using a modified version of the QR decomposition [4]. Enforcing

that K has positive values on its diagonal should remove any ambiguity about the factorization.

2.2.3 More General Cameras

While the perspective model of projection generally matches the image formation process of a real camera, an affine model of projection is sometimes more computationally appealing. While less mathematically accurate, such an approximation may be acceptable in cases where the depth of scene points is fairly uniform, or the field of view is fairly narrow. Common choices for linear models include *orthographic projection*, in which the camera matrix has the form

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}, \tag{7}$$

or *weak perspective projection*, in which the camera matrix has the form

$$P = \begin{bmatrix} \alpha_x & 0 & 0 & 0 \\ 0 & \alpha_y & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}. \tag{8}$$

Finally, we note that real cameras frequently exhibit *radial lens distortion*, which can be modeled by

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = L(\sqrt{x^2 + y^2}) \begin{bmatrix} x \\ y \end{bmatrix}, \tag{9}$$

where (x, y) is the result of applying the perspective model (4), and $L(\cdot)$ is a function of the radial distance from the image center, often modeled as a fourth-degree polynomial. The parameters of the distortion function are typically measured off-line using a calibration grid [5].

2.3 Estimating the Camera Matrix

The P matrix for a given camera is typically estimated based on a set of matched correspondences between image points $u_j \in \mathbb{R}^2$ and scene points

$\mathbf{X}_j$ with known coordinates in $\mathbb{R}^3$. That is, the goal is to select the matrix $P \in \mathbb{R}^{3\times 4}$ that best matches a given set of point mappings:

$$\{\mathbf{X}_j \mapsto u_j, j = 1, \ldots, N\}. \tag{10}$$

Each correspondence $(u_j, \mathbf{X}_j)$ produces two independent linear equations in the elements of P. Thus, the equations for all the data points can be collected into a linear system $Ap = 0$, where A is an $2N \times 12$ matrix involving the data, and $p = (p_{11}, \ldots, p_{34})^T$ is the vector of unknowns. Since the camera matrix is unique up to scale, we must fix some scaling (say, $||p||_2 = 1$) to ensure that Ap cannot become arbitrarily small.

The least-squares minimization problem is then

$$\min \|Ap\|_2 \quad \text{s.t. } p^T p = 1. \tag{11}$$

The solution to this problem is well known; the minimizer is the eigenvector $p \in \mathbb{R}^{12}$ of $A^T A$ corresponding to the minimal eigenvalue, which can be computed via the singular value decomposition. This eigenvector is then reassembled into a 3×4 matrix $\hat{P}$, which can be factored into intrinsic and extrinsic parameters as described above.

To maintain numerical stability, it is critical to normalize the data before solving the estimation problem. That is, each of the sets $\{u_j\}$ and $\{\mathbf{X}_j\}$ should be translated to have zero mean, and then isotropically scaled so that the average distance to the origin is $\sqrt{2}$ for the image points and $\sqrt{3}$ for the scene points. These translations and scalings can be represented by similarity transform matrices T and U that act on the homogeneous coordinates of the u_j and $\mathbf{X}_j$. After the camera matrix for the normalized points $\hat{P}$ has been estimated, the estimate of the camera matrix in the original coordinates is given by

$$P = T^{-1}\hat{P}U. \tag{12}$$

Several more advanced algorithms for camera matrix estimation (also called *resectioning*) are discussed in Hartley and Zisserman [6], typically requiring iterative, nonlinear optimization.

When the datasets contain outliers (i.e., incorrect point correspondences inconsistent with the underlying projection model), it is necessary to detect and reject them during the estimation process. Frequently, the robust estimation framework called RANSAC [7] is employed for this purpose. RANSAC is based on randomly selecting a large number of data subsets, each containing the minimal number of correspondences that make the estimation problem solvable. An estimate of the outlier probability is used to select a number

of subsets to guarantee that at least one of the minimal subsets has a high probability of containing all inliers.

The point correspondences required for camera matrix estimation are typically generated using images of a calibration grid resembling a high-contrast checkerboard. Bouguet authored a widely-disseminated camera calibration toolbox in Matlab that only requires the user to print and acquire several images of such a calibration grid [8].

3 Two-Camera Geometry

Next, we discuss the image relationships resulting from the same static scene being imaged by two cameras $\mathcal{C}$ and $\mathcal{C}'$, as illustrated in Figure 2. These could be two physically separate cameras, or a single moving camera at different points in time. Let the scene coordinates of a point $\mathbf{X}$ in the $\mathcal{C}$ coordinate system be (X, Y, Z), and in the $\mathcal{C}'$ coordinate system be (X', Y', Z'). We denote the corresponding image coordinates of $\mathbf{X}$ in $\mathcal{P}$ and $\mathcal{P}'$ by $u = (x, y)$ and $u' = (x', y')$, respectively. The points u and u' are said to be corresponding points, and the pair (u, u') is called a point correspondence.

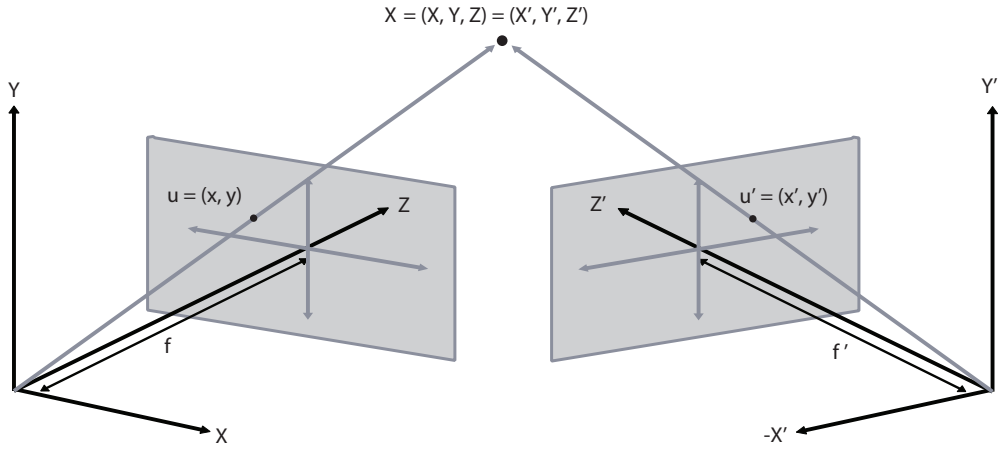


Fig. 2. The rigid motion of a camera introduces a change of coordinates, resulting in different image coordinates for the same scene point $\mathbf{X}$.

Assuming standard values for the intrinsic parameters, the scene point $\mathbf{X}$ is projected onto the image points u and u' via the perspective projection equations (2):

$$x = f\frac{X}{Z} \qquad y = f\frac{Y}{Z} \tag{13}$$

$$x' = f'\frac{X'}{Z'} \qquad y' = f'\frac{Y'}{Z'}. \tag{14}$$

Here f and f' are the focal lengths of $\mathcal{C}$ and $\mathcal{C}'$, respectively. We assume that the two cameras are related by a rigid motion, which means that the $\mathcal{C}'$ coordinate system can be expressed as a rotation R of the $\mathcal{C}$ coordinate system followed by a translation $[t_X\ t_Y\ t_Z]^T$. That is,

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_X \\ t_Y \\ t_Z \end{bmatrix}. \tag{15}$$

In terms of the parameters of the cameras, $R = R'R^{-1}$ and $t = R'(C - C')$. Alternately, we can write R as

$$R = \begin{pmatrix} \cos\alpha\cos\gamma + \sin\alpha\sin\beta\sin\gamma & \cos\beta\sin\gamma & -\sin\alpha\cos\gamma + \cos\alpha\sin\beta\sin\gamma \\ -\cos\alpha\sin\gamma + \sin\alpha\sin\beta\cos\gamma & \cos\beta\cos\gamma & \sin\alpha\sin\gamma + \cos\alpha\sin\beta\cos\gamma \\ \sin\alpha\cos\beta & -\sin\beta & \cos\alpha\cos\beta \end{pmatrix}, \tag{16}$$

where α, β and γ are rotation angles around the X, Y, and Z axes, respectively, of the $\mathcal{C}$ coordinate system.

By substituting equation (15) into the perspective projection equations (2), we obtain a relationship between the two sets of image coordinates:

$$x' = f'\frac{X'}{Z'} = \frac{r_{11}\frac{f'}{f}x + r_{12}\frac{f'}{f}y + r_{13}f' + \frac{t_X f'}{Z}}{\frac{r_{31}}{f}x + \frac{r_{32}}{f}y + r_{33} + \frac{t_Z}{Z}} \tag{17}$$

$$y' = f'\frac{Y'}{Z'} = \frac{r_{21}\frac{f'}{f}x + r_{22}\frac{f'}{f}y + r_{23}f' + \frac{t_Y f'}{Z}}{\frac{r_{31}}{f}x + \frac{r_{32}}{f}y + r_{33} + \frac{t_Z}{Z}}. \tag{18}$$

Here the r_{ij} are the elements of the rotation matrix given in (16). In Section 4 we will consider some special cases of (17)-(18).

3.1 The Epipolar Geometry and its Estimation

We now introduce the *fundamental matrix*, which encapsulates an important constraint on point correspondences between two images of the same scene. For each generic pair of cameras $(\mathcal{C}, \mathcal{C}')$, there exists a matrix F of rank two

such that for all correspondences $(u, u') = ((x, y), (x', y')) \in \mathcal{P} \times \mathcal{P}'$,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}^T F \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0. \tag{19}$$

The fundamental matrix is unique up to scale provided that there exists no quadric surface $\mathcal{Q}$ containing the line $\overline{C\ C'}$ and every point in the scene $\mathcal{S}$ [9].

Given the fundamental matrix F for a camera pair $(\mathcal{C}, \mathcal{C}')$, we obtain a constraint on the possible locations of point correspondences between the associated image pair $(\mathcal{I}, \mathcal{I}')$.

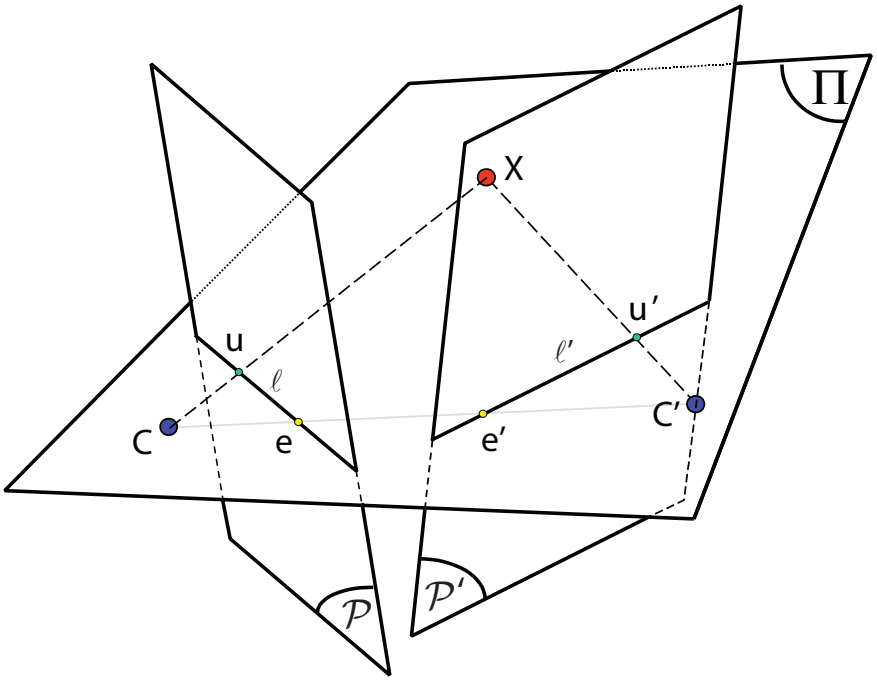


Fig. 3. The relationship between two images of the same scene is encapsulated by the epipolar geometry. For any scene point $\mathbf{X}$, we construct the plane Π containing $\mathbf{X}$ and the two camera centers. This plane intersects the image planes at the two epipolar lines ℓ and ℓ'. Any image point on ℓ in $\mathcal{I}$ must have its correspondence on ℓ' in $\mathcal{I}'$. The projections of the camera centers C' and C onto $\mathcal{I}$ and $\mathcal{I}'$ are the epipoles e and e', respectively.

The *epipolar line* corresponding to a point $u \in \mathcal{P}$ is the set of points:

$$\ell_u = \left\{ u' = (x', y')^T \in \mathcal{P}' \,\middle|\, \begin{bmatrix} u' \\ 1 \end{bmatrix}^T F \begin{bmatrix} u \\ 1 \end{bmatrix} = 0 \right\}. \tag{20}$$

If u is the image of scene point $\mathbf{X}$ in $\mathcal{P}$, the image u' of $\mathbf{X}$ in $\mathcal{P}'$ is constrained to lie on the epipolar line ℓ_u. Epipolar lines for points in $\mathcal{P}'$ can be defined accordingly. Hence, epipolar lines exist in conjugate pairs (ℓ, ℓ'), such that the match to a point $u \in \ell$ must lie on ℓ', and vice versa. Conjugate epipolar lines are generated by intersecting any plane Π containing the baseline $\overline{C\ C'}$ with the pair of image planes $(\mathcal{P}, \mathcal{P}')$ (see Figure 3). For a more thorough review of epipolar geometry, the reader is referred to [10].

The *epipoles* $e \in \mathcal{P}$ and $e' \in \mathcal{P}'$ are the projections of the camera centers C' and C onto $\mathcal{P}$ and $\mathcal{P}'$, respectively. It can be seen from Figure 3 that the epipolar lines in each image all intersect at the epipole. In fact, the homogeneous coordinates of the epipoles e and e' are the right and left eigenvectors of F, respectively, corresponding to the eigenvalue 0.

Since corresponding points must appear on conjugate epipolar lines, they form an important constraint that can be exploited while searching for feature matches, as illustrated in Figure 4.

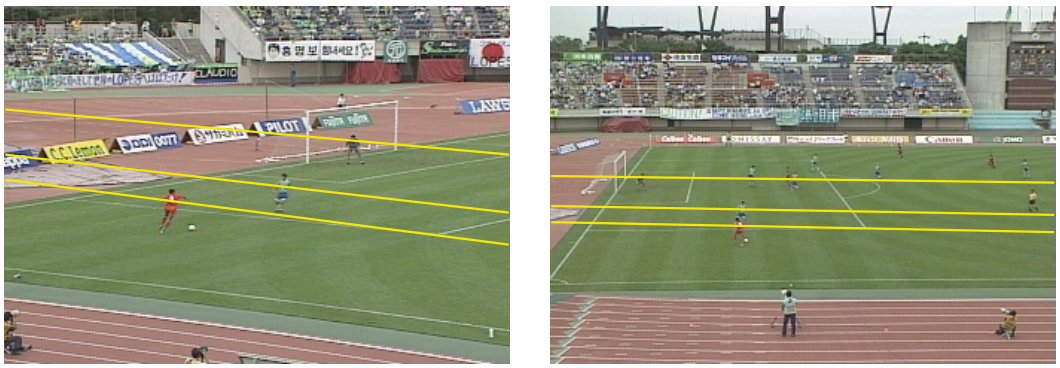


Fig. 4. Image pair, with sample epipolar lines. Corresponding points must occur along conjugate epipolar lines; for example, consider the goalie's head or the front corner of the goal area.

3.2 Relating the Fundamental Matrix to the Camera Matrices

The fundamental matrix can easily be constructed from the two camera matrices. If we assume that

$$P = K\ [I \ \mid \ 0] \qquad P' = K'\ [R \ \mid \ t], \tag{21}$$

then the fundamental matrix is given by

$$F = K'^{-T}[t]_\times R K^{-1} = K'^{-T} R [R^T t]_\times K^{-1}, \tag{22}$$

where $[t]_\times$ is the skew-symmetric matrix defined by

$$[t]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}. \tag{23}$$

In the form of (22), we can see that F is rank-2 since $[t]_\times$ is rank-2. Since F is only unique up to scale, the fundamental matrix has 7 degrees of freedom. The left and right epipoles can also be expressed in terms of the camera matrices as follows:

$$e = K R^T t \qquad e' = K' t. \tag{24}$$

From the above, the fundamental matrix for a pair of cameras is clearly unique up to scale. However, there are 4 degrees of freedom in extracting P and P' from a given F. This ambiguity arises since the camera matrix pairs (P, P') and $(PH, P'H)$ have the same fundamental matrix for any 4×4 nonsingular matrix H (e.g., the same rigid motion applied to both cameras). The family of (P, P') corresponding to a given F is described by:

$$P = [I \mid 0] \qquad P' = [[e']_\times F + e' v^T \mid \lambda e'], \tag{25}$$

where $v \in \mathbb{R}^3$ is an arbitrary vector and λ is a non-zero scalar.

In cases where the intrinsic parameter matrix K of the camera is known (e.g., estimated offline using a calibration grid), then the homogeneous image coordinates can be transformed using

$$\hat{u} = K^{-1} u \qquad \hat{u}' = K'^{-1} u'. \tag{26}$$

The camera matrices become

$$P = [I \mid 0] \qquad P' = [R \mid t], \tag{27}$$

and the fundamental matrix is now called the *essential matrix*. The advantage

of the essential matrix is that its factorization in terms of the extrinsic camera parameters

$$E = [t]_\times R = R[R^T t]_\times \tag{28}$$

is unique up to four possible solutions, the correct of which can be determined by requiring that all projected points lie in front of both cameras [6].

3.3 Estimating the Fundamental Matrix

The fundamental matrix is typically estimated based on a set of matched point correspondences (see Section 5). That is, the goal is to select the matrix $F \in \mathbb{R}^{3\times 3}$ that best matches a given set of point mappings:

$$\{u_j \mapsto u'_j \in \mathbb{R}^2, j = 1, \ldots, N\} \tag{29}$$

It is natural to try to minimize a least-squares cost functional such as

$$J(F) = \sum_{j=1}^{N} \begin{bmatrix} u'_j \\ 1 \end{bmatrix}^T F \begin{bmatrix} u_j \\ 1 \end{bmatrix} \tag{30}$$

over the class of admissible fundamental matrices. We recall that F must have rank two (see Section 3.2). Furthermore, the fundamental matrix is unique up to scale, so we must fix some scaling (say, $\|F\| = 1$ for some appropriate norm) to ensure that J cannot become arbitrarily small. Hence, the class of admissible estimates has only seven degrees of freedom. Constrained minimizations of this type are problematic due to the difficulty in parameterizing the class of admissible F. Faugeras and Luong [11–13] proposed some solutions in this regard and analyzed various cost functionals for the estimation problem.

A standard approach to estimating the fundamental matrix was proposed by Hartley [14]. Ignoring the rank-two constraint for the moment, we minimize (30) over the class $\{F \in \mathbb{R}^{3\times 3} \mid \|F\|_F = 1\}$, where $\|\cdot\|_F$ is the Frobenius norm.

Each correspondence (u_j, u'_j) produces a linear equation in the elements of F:

$$x_j x'_j f_{11} + x_j y'_j f_{21} + x_j f_{31} + y_j x'_j f_{21} + y_j y'_j f_{22} + y_j f_{23} + x'_j f_{31} + y'_j f_{32} + f_{33} = 0$$

The equations in all the data points can be collected into a linear system $Af =$

0, where A is an $N \times 9$ matrix involving the data, and $f = (f_{11}, f_{21}, f_{31}, f_{12}, f_{22}, f_{32}, f_{13}, f_{23}, f_{33})^T$ is the vector of unknowns. The least-squares minimization problem is then

$$\min \|Af\|_2 \quad \text{s.t. } f^T f = 1 \tag{31}$$

As in the resectioning estimation problem in Section 2.3, the minimizer is the eigenvector $f \in \mathbb{R}^9$ of $A^T A$ corresponding to the minimal eigenvalue, which can be computed via the singular value decomposition. This eigenvector is then reassembled into a 3×3 matrix $\hat{F}$.

To account for the rank-two constraint, we replace the full-rank estimate $\hat{F}$ by $\hat{F}^*$, the minimizer of

$$\min \|\hat{F} - \hat{F}^*\|_F \quad \text{s.t. } \mathit{rank}(\hat{F}^*) = 2. \tag{32}$$

Given the singular value decomposition $\hat{F} = UDV^T$, where $D = \mathrm{diag}(r, s, t)$ with $r > s > t$, the solution to (32) is

$$\hat{F}^* = U\hat{D}V^T, \tag{33}$$

where $\hat{D} = \mathrm{diag}(r, s, 0)$.

As in Section 2.3, to maintain numerical stability, it is critical to normalize the data before solving the estimation problem. Each of the sets $\{u_j\}$ and $\{u'_j\}$ should be translated to have zero mean, and then isotropically scaled so that the average distance to the origin is $\sqrt{2}$. These translations and scalings can be represented by 3×3 matrices T and T' that act on the homogeneous coordinates of the u_j and u'_j. After the fundamental matrix for the normalized points has been estimated, the estimate of the fundamental matrix in the original coordinates is given by

$$F = T'^T \hat{F}^* T. \tag{34}$$

The overall estimation process is called the *normalized eight-point algorithm*. Several more advanced algorithms for fundamental matrix estimation are discussed in Hartley and Zisserman [6], typically requiring iterative, nonlinear optimization. As before, outliers (i.e., incorrect point correspondences inconsistent with the underlying epipolar geometry) are often detected and rejected using RANSAC.

4 Projective Transformations

The fundamental matrix constrains the possible locations of corresponding points: they must occur along conjugate epipolar lines. However, there are two special situations in which the fundamental matrix for an image pair is undefined, and instead point correspondences between the images are related by an explicit one-to-one mapping.

Let us reconsider equations (17) and (18) that relate the image coordinates of a point seen by two cameras $\mathcal{C}$ and $\mathcal{C}'$. For this relationship to define a transformation that globally relates the image coordinates, for every scene point (X, Y, Z) we would require that the dependence on the world coordinate Z disappears, i.e. that

$$\frac{t_X}{Z} = a_{1X}x + a_{2X}y + b_X \tag{35}$$
$$\frac{t_Y}{Z} = a_{1Y}x + a_{2Y}y + b_Y \tag{36}$$
$$\frac{t_Z}{Z} = a_{1Z}x + a_{2Z}y + b_Z \tag{37}$$

for some set of a's and b's. These conditions are satisfied when either:

(1) $t_X = t_Y = t_Z = 0$ or
(2) $k_1X + k_2Y + k_3Z = 1$.

In the first case, corresponding to a camera whose optical center undergoes no translation, we obtain

$$x' = \frac{r_{11}\frac{f'}{f}x + r_{12}\frac{f'}{f}y + r_{13}f'}{\frac{r_{31}}{f}x + \frac{r_{32}}{f}y + r_{33}}$$
$$y' = \frac{r_{21}\frac{f'}{f}x + r_{22}\frac{f'}{f}y + r_{23}f'}{\frac{r_{31}}{f}x + \frac{r_{32}}{f}y + r_{33}}.$$

An example of three such images composed into the same frame of reference with appropriate projective transformations is illustrated in Figure 5.

In the second case, corresponding to a planar scene, (17) and (18) become:

Fig. 5. Images from a non-translating camera, composed into the same frame of reference by appropriate projective transformations.

$$x' = \frac{(r_{11}\frac{f'}{f} + t_X f' k_1)x + (r_{12}\frac{f'}{f} + t_X f' k_2)y + (r_{13} f' + t_X f' k_3)}{(\frac{r_{31}}{f} + t_Z k_1)x + (\frac{r_{32}}{f} + t_Z k_2)y + (r_{33} + t_Z k_3)}$$
$$y' = \frac{(r_{21}\frac{f'}{f} + t_Y f' k_1)x + (r_{22}\frac{f'}{f} + t_Y f' k_2)y + (r_{23} f' + t_Y f' k_3)}{(\frac{r_{31}}{f} + t_Z k_1)x + (\frac{r_{32}}{f} + t_Z k_2)y + (r_{33} + t_Z k_3)} \tag{38}$$

An example of a pair of images of a planar surface, registered by an appropriate projective transformation, is illustrated in Figure 6.

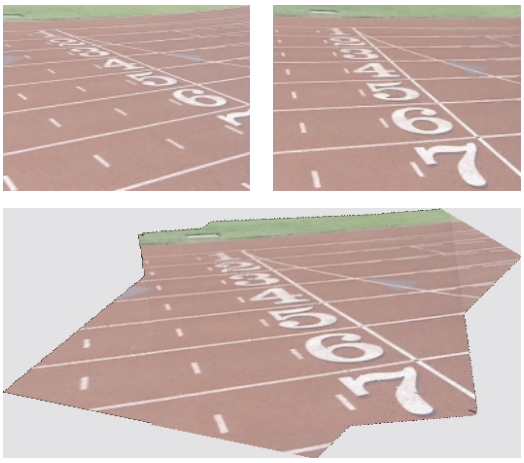


Fig. 6. Images of a planar scene, composed into the same frame of reference by appropriate projective transformations.

In either case, the transformation is of the form

$$x' = \frac{a_{11}x + a_{12}y + b_1}{c_1x + c_2y + d} \tag{39}$$

$$y' = \frac{a_{21}x + a_{22}y + b_2}{c_1x + c_2y + d}, \tag{40}$$

which can be written in homogeneous coordinates as

$$u' = Hu \tag{41}$$

for a nonsingular 3×3 matrix H defined up to scale. This relationship is called a *projective transformation* (sometimes also known as a collineation or a homography). When $c_1 = c_2 = 0$, the transformation is known as an *affine transformation*.

Projective transformations between images are induced by common camera configurations such as surveillance cameras that view a common ground plane or a panning and tilting camera mounted on a tripod. In the next section, we discuss how to estimate projective transformations relating an image pair.

We note that technically, affine transformations are induced by the motion of a perspective camera only under quite restrictive conditions. The image planes must both be parallel to the XY-plane. Furthermore, either the translation vector t must be identically 0, or Z must be constant for all points in the scene, i.e., the scene is a planar surface parallel to the image planes $\mathcal{P}$ and $\mathcal{P}'$. However, the affine assumption is often made when the scene is far from the camera (Z is large) and the rotation angles α and β are very small. This assumption has the advantage that the affine parameters can be easily estimated, usually in closed form.

4.1 Estimating Projective Transformations

The easiest approach to estimating a projective transformation from point correspondences is called the *direct linear transform (DLT)*. If the point correspondence $\{u_j \mapsto u'_j\}$ corresponds to a projective transformation, then the homogeneous coordinates u'_j and Hu_j are vectors in the same direction, i.e.,

$$u'_j \times Hu_j = 0. \tag{42}$$

Since (42) gives 2 independent linear equations in the 9 unknowns of H, N point correspondences give rise to a $2N \times 9$ linear system

$$Ah = 0, \tag{43}$$

where A is a $2N \times 9$ matrix involving the data, and $h = (a_{11}, a_{12}, a_{22}, a_{22}, b_1, b_2, c_1, c_2, d)^T$ is the vector of unknowns. We solve the problem in exactly the same way as (31) using the singular value decomposition (i.e., the solution h is the singular vector of A corresponding to the smallest singular value).

When the element d is expected to be far from 0, an alternate approach is to normalize $d = 1$, in which case the N point correspondences induce a system of $2N$ equations in the remaining 8 unknowns, which can be solved as a linear least-squares problem.

As with the fundamental matrix estimation problem, more accurate estimates of the projective transformation parameters can be obtained by the iterative minimization of a nonlinear cost function, such as the symmetric transfer error given by

$$\sum_{i=1}^{N} ||u_i' - Hu_i||_2^2 + ||u_i - H^{-1}u_i'||_2^2. \tag{44}$$

The reader is referred to Hartley and Zisserman [6] for more details. We note that an excellent turnkey approach to estimating projective transformations for real images is given by the Generalized Dual-Bootstrap ICP algorithm proposed by Yang et al. [15].

4.2 Rectifying Projective Transformations

We close this section by mentioning a special class of projective transformations called *rectifying projective transformations* that often simplify computer vision problems involving point matching along conjugate epipolar lines.

Since epipolar lines are generally not aligned with one of the coordinate axes of an image, or are even parallel, the implementation of algorithms that work with epipolar lines can be complicated. To this end, it is common to apply a technique called rectification to an image pair before processing, so that the epipolar lines are parallel and horizontal.

An associated image plane pair $(\mathcal{P}, \mathcal{P}')$ is said to be rectified when the funda-

mental matrix for $(\mathcal{P}, \mathcal{P}')$ is the skew-symmetric matrix

$$F_* = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}. \tag{45}$$

In homogeneous coordinates, the epipoles corresponding to F_* are $e_0 = e_1 = [1\ 0\ 0]^T$, which means the epipolar lines are horizontal and parallel. Furthermore, expanding the fundamental matrix equation for a correspondence $((x, y)^T, (x', y')^T) \in \mathcal{P} \times \mathcal{P}'$ gives

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}^T F_* \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0, \tag{46}$$

which is equivalent to $y' - y = 0$. This implies that not only are the epipolar lines in a rectified image pair horizontal, they are aligned, so that the lines $y = \lambda$ in $\mathcal{P}$ and $y' = \lambda$ in $\mathcal{P}'$ are conjugate epipolar lines.

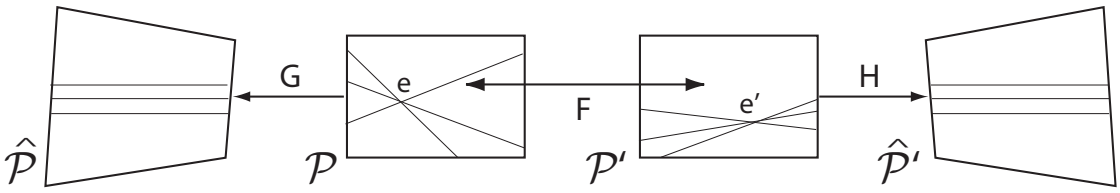


Fig. 7. Rectifying projective transformations G and H transform corresponding epipolar lines to be horizontal with the same y-value.

A pair of projective transformations (G, H) is called rectifying for an associated image plane pair $(\mathcal{P}, \mathcal{P}')$ with fundamental matrix F if

$$H^{-T} F G^{-1} = F_*. \tag{47}$$

By the above definition, if the projective transformations G and H are applied to $\mathcal{P}$ and $\mathcal{P}'$ to produce warped image planes $\hat{\mathcal{P}}_0$ and $\hat{\mathcal{P}}_1$, respectively, then $(\hat{\mathcal{P}}_0, \hat{\mathcal{P}}_1)$ is a rectified pair (Figure 7).

The rectifying condition (47) can be expressed as 9 equations in the 16 unknowns of the two projective transformations G and H, leading to 7 degrees of freedom in the choice of a rectifying pair. Seitz [16] and Hartley [17] described methods for deriving rectifying projective transformations from an estimate of the fundamental matrix relating an image pair. Isgrò and Trucco [18] observed

that rectifying transformations can be estimated without explicitly estimating the fundamental matrix as an intermediate step.

5 Feature Detection and Matching

As indicated in the previous sections, many algorithms for multiview geometry parameter estimation problems require a set of image correspondences as input, i.e., regions of pixels representing scene points that can be reliably, unambiguously matched in other images of the same scene.

Many indicators of pixel regions that constitute "good" features have been proposed, including line contours, corners, and junctions. A classical algorithm for detecting and matching "good" point features was proposed by Shi and Tomasi [19], described as follows:

(1) Compute the gradients $g_x(x, y)$ and $g_y(x, y)$ for $\mathcal{I}$. That is,

$$g_x(x, y) = \mathcal{I}(x, y) - \mathcal{I}(x - 1, y)$$
$$g_y(x, y) = \mathcal{I}(x, y) - \mathcal{I}(x, y - 1).$$

(2) For every $N \times N$ block of pixels Γ,
   (a) Compute the covariance matrix

$$B = \begin{bmatrix} \sum_{(x,y)\in\Gamma} g_x^2(x, y) & \sum_{(x,y)\in\Gamma} g_x g_y(x, y) \\ \sum_{(x,y)\in\Gamma} g_x g_y(x, y) & \sum_{(x,y)\in\Gamma} g_y^2(x, y) \end{bmatrix}.$$

   (b) Compute the eigenvalues of B, λ_1 and λ_2.
   (c) If λ_1 and λ_2 are both greater than some threshold τ, add Γ to the list of features.

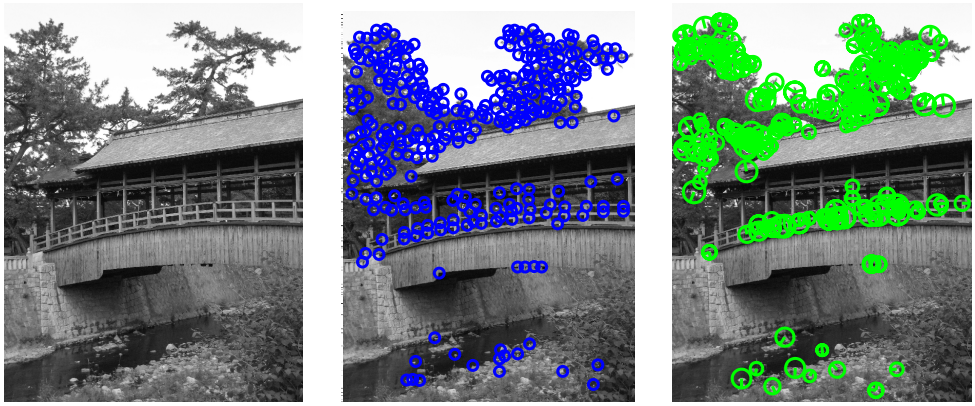
(3) For every block of pixels Γ in the list of features, find the $N \times N$ block of pixels in $\mathcal{I}'$ that has the highest normalized cross-correlation, and add the point correspondence to the feature list if the correlation is sufficiently high.

A recent focus in the computer vision community has been on different types of "invariant" detectors that select image regions that can be robustly matched even between images where the camera perspectives or zooms are quite different. An early approach was the Harris corner detector [20], which uses the same matrix B as the Shi-Tomasi algorithm, but instead of computing the eigenvalues of B, the quantity

$$\rho = \det(B) - \kappa \cdot \mathrm{trace}^2(Z) \tag{48}$$

is computed, with κ in a recommended range of 0.04-0.15. Blocks with local positive maxima of ρ are selected as features. Mikolajczyk and Schmid [21] later extended Harris corners to a multi-scale setting.

An alternate approach is to filter the image at multiple scales with a Laplacian-of-Gaussian (LOG) [22] or Difference-of-Gaussian (DOG) [23] filter; scale-space extrema of the filtered image give the locations of the interest points. The popular Scale Invariant Feature Transform, or SIFT, detector proposed by Lowe [23] is based on multiscale DOG filters. Feature points of this type typically resemble "blobs" at different scales, as opposed to "corners". Figure 8 illustrates example Harris corners and SIFT features detected for the same image. A broad survey of modern feature detectors was given by Mikolajczyk and Schmid [24].



(a) (b) (c)

Fig. 8. (a) Original image. (b) Example Harris corners. (c) Example SIFT features.

Block-matching correspondence approaches begin to fail when the motion of the camera or of scene objects induces too much of a change in the images. In this case, the assumption that a rectangular block of pixels in one image roughly matches a block of the same shape and size in the other breaks down. Normalized cross-correlation between blocks of pixels is likely to yield poor matches.

In such cases, it may be more appropriate to apply the SIFT feature descriptor [23], a histogram of gradient orientations designed to be invariant to scale and rotation of the feature. Typically, the algorithm takes a 16×16 grid of samples from the the gradient map at the feature's scale, and uses it to form a 4×4 aggregate gradient matrix. Each element of the matrix is quantized into 8 orientations, producing a descriptor of dimension 128. Mikolajczyk and Schmid [25] showed that the overall SIFT algorithm outperformed most other detector/descriptor combinations in their experiments, accounting for its widespread popularity in the computer vision community.

6 Multi-Camera Geometry

We now proceed to the relationships between $M > 2$ cameras that observe the same scene. While there is an entity called the trifocal tensor [26] relating three cameras' views that plays an analogous role to the fundamental matrix for two views, here we concentrate on the geometry of M general cameras.

We assume a camera network that contains M perspective cameras, each described by a 3×4 matrix P_i. Each camera images some subset of a set of N scene points $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\} \in \mathbb{R}^3$. We define an indicator function χ_{ij} where $\chi_{ij} = 1$ if camera i images scene point j. The projection of $\mathbf{X}_j$ onto P_i is given by $u_{ij} \in \mathbb{R}^2$, denoted

$$u_{ij} \sim P_i \mathbf{X}_j. \tag{49}$$

The main problem in multi-camera geometry is called *structure from motion (SFM)*. That is, given only the observed image projections $\{u_{ij}\}$, we want to estimate the corresponding camera matrices P_i and scene points $\mathbf{X}_j$. As we discuss in the next sections, this process typically proceeds in stages to find a good initial estimate, which is followed by a nonlinear optimization algorithm called *bundle adjustment*.

We note that SFM is closely related to a problem in the robotics community called *Simultaneous Localization and Mapping (SLAM)* [27], in which mobile robots must estimate their locations from sensor data as they move through a scene. SFM also forms the fundamental core of commercial software packages such as Boujou or SynthEyes for the problems of "matchmoving" or camera tracking, which are used to insert digital effects into Hollywood movies.

6.1 Affine Reconstruction

Many initialization methods for the SFM problem involve a *factorization* approach. The first such approach was proposed by Tomasi and Kanade [28] and applies only to affine cameras (we generalize it to perspective cameras in the next section).

For an affine camera, the projection equations take the form

$$u_{ij} = A_i \mathbf{X}_j + t_i \tag{50}$$

for $A_i \in \mathbb{R}^{2\times 3}$ and $t_i \in \mathbb{R}^2$. If we assume that each camera images all of the

scene points, then a natural formulation of the affine SFM problem is:

$$\min_{\{A_i,t_i,\mathbf{X}_j\}} \sum_{i=1}^{M}\sum_{j=1}^{N} \|u_{ij} - (A_i\mathbf{X}_j + t_i)\|^2 \tag{51}$$

If we assume that the $\mathbf{X}_j$ are centered at 0, taking the derivative with respect to t_i reveals that the minimizer

$$\hat{t}_i = \frac{1}{N}\sum_{i=1}^{N} u_{ij}. \tag{52}$$

Hence, we recenter the image measurements by

$$u_{ij} \longleftarrow u_{ij} - \frac{1}{N}\sum_{i=1}^{N} u_{ij} \tag{53}$$

and are faced with solving

$$\min_{\{A_i,\mathbf{X}_j\}} \sum_{i=1}^{M}\sum_{j=1}^{N} \|u_{ij} - A_i\mathbf{X}_j\|^2. \tag{54}$$

Tomasi and Kanade's key observation was that if all of the image coordinates are collected into a *measurement matrix* defined by

$$W = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1N} \\ u_{21} & u_{22} & \cdots & u_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ u_{M1} & u_{M2} & \cdots & u_{MN} \end{bmatrix}, \tag{55}$$

then in the ideal (noiseless) case, this matrix factors as

$$W = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_M \end{bmatrix} [\mathbf{X}_1\ \mathbf{X}_2\ \cdots\ \mathbf{X}_N], \tag{56}$$

revealing that the $2M \times N$ measurement matrix is ideally rank-3. They showed that solving (54) is equivalent to finding the best rank-3 approximation to W

in the Frobenius norm, which can easily be accomplished with the singular value decomposition. The solution is unique up to an affine transformation of the world coordinate system, and optimal in the sense of maximum likelihood estimation if the noise in the image projections is isotropic, zero-mean i.i.d. Gaussian.

6.2 Projective Reconstruction

Unfortunately, for the perspective projection model that more accurately reflects the way real cameras image the world, factorization is not as easy. We can form a similar measurement matrix W and factorize it as follows:

$$W = \begin{bmatrix} \lambda_{11}u_{11} & \lambda_{12}u_{12} & \cdots & \lambda_{1N}u_{1N} \\ \lambda_{21}u_{21} & \lambda_{22}u_{22} & \cdots & \lambda_{2N}u_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{M1}u_{M1} & \lambda_{m2}u_{m2} & \cdots & \lambda_{MN}u_{MN} \end{bmatrix} = \begin{pmatrix} P_1 \\ P_2 \\ \vdots \\ P_M \end{pmatrix} \begin{pmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_N \end{pmatrix}. \quad (57)$$

Here, both the u_{ij} and $\mathbf{X}_j$ are represented in homogeneous coordinates. Therefore, the measurement matrix is of dimension $3M \times N$ and is ideally rank-4. However, since the scale factors (also called projective depths) λ_{ij} multiplying each projection are different, these must also be estimated.

Sturm and Triggs [29,30] suggested a factorization method that recovers the projective depths as well as the structure and motion parameters from the measurement matrix. They used relationships between fundamental matrices and epipolar lines in order to obtain an initial estimate for the projective depths λ_{ij} (an alternate approach is to simply initialize all $\lambda_{ij} = 1$). Once the projective depths are fixed, the rows and columns of the measurement matrix are rescaled, and the camera matrices and scene point positions are recovered from the best rank-4 approximation to W obtained using the singular value decomposition. Given estimates of the camera matrices and scene points, the scene points can then be reprojected to obtain new estimates of the projective depths, and the process iterated until the parameter estimates converge.

There is substantial ambiguity in a projective reconstruction as obtained above, since

$$\begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_M \end{bmatrix} [\mathbf{X}_1 \ \mathbf{X}_2 \ \cdots \ \mathbf{X}_N] = \begin{bmatrix} P_1 H \\ P_2 H \\ \vdots \\ P_M H \end{bmatrix} [H^{-1}\mathbf{X}_1 \ H^{-1}\mathbf{X}_2 \ \cdots \ H^{-1}\mathbf{X}_N] \tag{58}$$

for any 4×4 nonsingular matrix H. This means that while some geometric properties of the reconstructed configuration will be correct compared to the truth (e.g., the order of 3D points lying along a straight line), others will not (e.g., the angles between lines/planes or the relative lengths of line segments). In order to make the reconstruction useful (i.e., to recover the correct configuration up to an unknown rotation, translation, and scale), we need to estimate the matrix H that turns the projective factorization into a metric factorization, so that

$$\hat{P}_i H = K_i R_i^T \left[I \ \mid \ -C_i \right]. \tag{59}$$

and K_i is in the correct form (e.g., we may force it to be diagonal). This process is also called *auto-calibration*.

The auto-calibration process depends fundamentally on several properties of projective geometry that are subtle and complex. We only give a brief overview here; see [31,32,6] for more details.

If we let $m_x = m_y = 1$ in (6), then the form of a camera's intrinsic parameter matrix is

$$K = \begin{bmatrix} \alpha_x & s & x \\ 0 & \alpha_y & y \\ 0 & 0 & 1 \end{bmatrix}. \tag{60}$$

We define a quantity called the *dual image of the absolute conic (DIAC)* as

$$\omega^* = K K^T = \begin{bmatrix} \alpha_x^2 + s^2 + x^2 & s\alpha_y + xy & x \\ s\alpha_y + xy & \alpha_y^2 + y^2 & y \\ x & y & 1 \end{bmatrix}. \tag{61}$$

If we put constraints on the camera's internal parameters, these correspond to constraints on the DIAC. For example, if we require that the pixels have zero skew and that the principal point is at the origin, then

$$\omega^* = \begin{bmatrix} \alpha_x^2 & 0 & 0 \\ 0 & \alpha_y^2 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{62}$$

and we have introduced 3 constraints on the entries of the DIAC:

$$\omega_{12}^* = \omega_{13}^* = \omega_{23}^* = 0. \tag{63}$$

If we further constrain that the pixels must be square, then we have a fourth constraint that $\omega_{11}^* = \omega_{22}^*$.

The DIAC is related to a second important quantity called the *absolute dual quadric*, denoted Q_∞^*. This is a special 4×4 symmetric rank-3 matrix that represents an imaginary surface in the scene that is fixed under similarity transformations. That is, points on Q_∞^* map to other points on Q_∞^* if a rotation, translation, and/or uniform scaling is applied to the scene coordinates.

The DIAC (which is different for each camera) and Q_∞^* (which is independent of the cameras) are connected by the important equation

$$\omega^{*i} = P^i Q_\infty^* P^{iT}. \tag{64}$$

That is, the two quantities are related through the camera matrices P^i. Therefore, each constraint we impose on each ω^{*i} (e.g., each of (63)) imposes a linear constraint on the 10 homogeneous parameters of Q_∞^*, in terms of the camera matrices P^i. Given enough camera matrices, we can thus solve a linear least-squares problem for the entries of Q_∞^*.

Once Q_∞^* has been estimated from the $\{P^i\}$ resulting from projective factorization, the correct H matrix in (59) that brings about a metric reconstruction is extracted using the relationship

$$Q_\infty^* = H \mathrm{diag}(1, 1, 1, 0) H^T. \tag{65}$$

The resulting reconstruction is related to the true camera/scene configuration by an unknown similarity transform that cannot be estimated without additional information about the scene.

We conclude this section by noting that there exist *critical motion sequences* for which auto-calibration is not fully possible [33], including pure translation or pure rotation of a camera, orbital motion around a fixed point, or motion confined to a plane.

6.4 Bundle Adjustment

After a good initial metric factorization is obtained, the final step is to perform bundle adjustment, i.e., the iterative optimization of the nonlinear cost function

$$\sum_{i=1}^{M}\sum_{j=1}^{N}\chi_{ij}(u_{ij}-P_i\mathbf{X}_j)^T\Sigma_{ij}^{-1}(u_{ij}-P_i\mathbf{X}_j). \tag{66}$$

Here, Σ_{ij} is the 2×2 covariance matrix associated with the noise in the image point u_{ij}. The quantity inside the sum is called the Mahalanobis distance between the observed image point and its projection based on the estimated camera/scene parameters.

The optimization is typically accomplished with the Levenberg-Marquardt algorithm [34], specifically an implementation that exploits the sparse block structure of the normal equations characteristic of SFM problems (since each scene point is typically observed by only a few cameras) [35].

Parameterization of the minimization problem, especially of the cameras, is a critical issue. If we assume that the skew, aspect ratio, and principal point of each camera are known prior to deployment, then each camera should be represented by 7 parameters: 1 for the focal length, 3 for the translation vector, and 3 for the rotation matrix parameters.

Rotation matrices are often minimally parameterized by 3 parameters (v_1, v_2, v_3) using the *axis-angle parameterization*. If we think of v as a vector in $\mathbb{R}^3$, then the rotation matrix R corresponding to v is a rotation through the angle $\|v\|$ about the axis v, computed by

$$R=\cos\|v\|I+\operatorname{sinc}\|v\|[v]_\times+\frac{1-\cos\|v\|}{\|v\|^2}vv^T. \tag{67}$$

Extracting the v corresponding to a given R is accomplished as follows. The direction of the rotation axis v is the eigenvector of R corresponding to the eigenvalue 1. The angle ϕ that defines $\|v\|$ is obtained from the two-argument arctangent function using

$$2\cos(\phi) = \text{trace}(R) - 1 \tag{68}$$
$$2\sin(\phi) = (R_{32} - R_{23}, R_{13} - R_{31}, R_{21} - R_{12})^T v. \tag{69}$$

Other parameterizations of rotation matrices and rigid motions were discussed by Chirikjian and Kyatkin [36].

Finally, it is important to remember that the set of cameras and scene points can only be estimated up to an unknown rotation, translation, and scale of the world coordinates, so it is common to fix the extrinsic parameters of the first camera as

$$P_1 = K_1 \; [I \;\; | \;\; 0]. \tag{70}$$

In general, any over-parameterization of the problem or ambiguity in the reconstructions creates a problem called *gauge freedom* [37], which can lead to slower convergence and computational problems. Therefore, it is important to strive for minimal parameterizations of SFM problems.

7 Further Resources

Multiple view geometry is a rich and complex area of study, and this chapter only gives an introduction to the main geometric relationships and estimation algorithms. The best and most comprehensive reference on epipolar geometry, structure from motion, and camera calibration is *Multiple View Geometry in Computer Vision*, by Richard Hartley and Andrew Zisserman [6], which is essential reading for further study of the material discussed here.

The collected volume *Vision Algorithms: Theory and Practice (Proceedings of the International Workshop on Vision Algorithms)*, edited by Bill Triggs, Andrew Zisserman, and Richard Szeliski [38] contains an excellent, detailed article on bundle adjustment in addition to important papers on gauges and parameter uncertainty.

Finally, *An Invitation to 3D Vision: From Images to Geometric Models*, by Yi Ma, Stefano Soatto, Jana Kosecka, and Shankar Sastry [39], is an excellent reference that includes a chapter giving a beginning-to-end recipe for structure-from-motion. The companion website includes Matlab code for most of the algorithms described in this chapter [40].

We conclude by noting that the concepts discussed in this chapter are now viewed as fairly classical in the computer vision community. However, there is still much interesting research to be done in extending and applying computer vision algorithms in the context of distributed camera networks; several such

cutting-edge algorithms are discussed in this book. The key challenges are that

(1) A very large number of widely-distributed cameras may be involved in a realistic camera network. The set-up is fundamentally different in terms of scale and spatial extent compared to typical multi-camera research undertaken in a research lab setting.
(2) The information from all cameras is unlikely to be available at a powerful, central processor, an underlying assumption of the multi-camera calibration algorithms discussed in Section 6. Instead, each camera may be attached to a power- and computation-constrained local processor that is unable to execute complex algorithms, and an antenna that is unable to transmit information across long distances.

These considerations argue for *distributed* algorithms that operate independently at each camera node, exchanging information between local neighbors to obtain solutions that approximate the best performance of a centralized algorithm. For example, we recently presented a distributed algorithm for the calibration of a multi-camera network that was designed with these considerations in mind [41]. We refer the reader to our recent survey on distributed computer vision algorithms [42] for more information.

References

[1] T. Kanade, P. Rander, P. Narayanan, Virtualized reality: Constructing virtual worlds from real scenes, IEEE Multimedia, Immersive Telepresence 4 (1) (1997) 34–47.

[2] D. Forsyth, J. Ponce, Computer Vision: A Modern Approach, Prentice Hall, 2003.

[3] P. Shirley, Fundamentals of Computer Graphics, A.K. Peters, 2002.

[4] G. Strang, Linear Algebra and its Applications, Harcourt Brace Jovanovich, 1988.

[5] B. Prescott, G. F. McLean, Line-based correction of radial lens distortion, Graphical Models and Image Processing 59 (1) (1997) 39–47.

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

[7] M. A. Fischler, R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Communications of the ACM 24 (1981) 381–395.

[8] J.-Y. Bouguet, Camera calibration toolbox for matlab, in: `http://www.vision.caltech.edu/bouguetj/calib_doc/`, 2008.

[9] S. Maybank, The angular velocity associated with the optical flowfield arising from motion through a rigid environment, Proc. Royal Soc. London A 401 (1985) 317–326.

[10] Z. Zhang, Determining the epipolar geometry and its uncertainty - a review, The International Journal of Computer Vision 27 (2) (1998) 161–195.

[11] O. Faugeras, Three-Dimensional Computer Vision: A Geometric Viewpoint, MIT Press, 1993.

[12] O. Faugeras, Q.-T. Luong, T. Papadopoulo, The Geometry of Multiple Images: The Laws That Govern the Formation of Multiple Images of a Scene and Some of Their Applications, MIT Press, 2001.

[13] Q.-T. Luong, O. Faugeras, The fundamental matrix: Theory, algorithms, and stability analysis, International Journal of Computer Vision 17 (1) (1996) 43–76.

[14] R. Hartley, In defence of the 8-point algorithm, in: Proc. ICCV '95, 1995, pp. 1064–1070.

[15] G. Yang, C. V. Stewart, M. Sofka, C.-L. Tsai, Registration of challenging image pairs: Initialization, estimation, and decision, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (11) (2007) 1973–1989.

[16] S. Seitz, Image-based transformation of viewpoint and scene appearance, Ph.D. thesis, University of Wisconsin at Madison (1997).

[17] R. Hartley, Theory and practice of projective rectification, International Journal of Computer Vision 35 (2) (1999) 115–127.

[18] F. Isgrò, E. Trucco, Projective rectification without epipolar geometry, in: Proc. CVPR '99, 1999.

[19] J. Shi, C. Tomasi, Good features to track, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994, pp. 593–600.

[20] C. Harris, M. Stephens, A combined corner and edge detector, in: Proceedings of the Fourth Alvey Vision Conference, Manchester, UK, 1988, pp. 147–151.

[21] K. Mikolajczyk, C. Schmid, Indexing based on scale invariant interest points, in: Procedings of the IEEE International Conference on Computer Vision (ICCV), Vancouver, Canada, 2001, pp. 525–531.

[22] T. Lindeberg, Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention, International Journal of Computer Vision 11 (3) (1994) 283–318.

[23] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.

[24] K. Mikolajczyk, C. Schmid, Scale and affine invariant interest point detectors, International Journal of Computer Vision 60 (1) (2004) 63–86.

[25] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (10) (2005) 1615–1630.

[26] A. Shashua, M. Werman, On the trilinear tensor of three perspective views and its underlying geometry, in: Proc. of the International Conference on Computer Vision (ICCV), 1995.

[27] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics: Intelligent Robotics and Autonomous Agents, MIT Press, 2005.

[28] C. Tomasi, T. Kanade, Shape and Motion from Image Streams: A Factorization Method Part 2. Detection and Trcking of Point Features, Tech. Rep. CMU-CS-91-132, Carnegie Mellon University (April 1991).

[29] P. Sturm, B. Triggs, A factorization based algorithm for multi-image projective structure and motion, in: Proceedings of the 4th European Conference on Computer Vision (ECCV '96), 1996, pp. 709–720.

[30] B. Triggs, Factorization methods for projective structure and motion, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR '96), IEEE Comput. Soc. Press, San Francisco, CA, USA, 1996, pp. 845–51.

[31] M. Pollefeys, R. Koch, L. J. Van Gool, Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 1998, pp. 90–95.

[32] M. Pollefeys, F. Verbiest, L. J. V. Gool, Surviving dominant planes in uncalibrated structure and motion recovery, in: Proceedings of the 7th European Conference on Computer Vision-Part II (ECCV '02), Springer-Verlag, London, UK, 2002, pp. 837–851.

[33] P. Sturm, Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 1997, pp. 1100–1105.

[34] J. Dennis, Jr., R. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, SIAM Press, 1996.

[35] B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon, Bundle adjustment – A modern synthesis, in: W. Triggs, A. Zisserman, R. Szeliski (Eds.), Vision Algorithms: Theory and Practice, LNCS, Springer Verlag, 2000, pp. 298–375.

[36] G. Chirikjian, A. Kyatkin, Engineering applications of noncommutative harmonic analysis with emphasis on rotation and motion groups, CRC Press, 2001.

[37] K. Kanatani, D. D. Morris, Gauges and gauge transformations for uncertainty description of geometric structure with indeterminacy, IEEE Transactions on Information Theory 47 (5) (2001) 2017–2028.

[38] B. Triggs, A. Zisserman, R. Szeliski (Eds.), Vision Algorithms: Theory and Practice (Proceedings of the International Workshop on Vision Algorithms Corfu, Greece, September 21-22, 1999), Springer, 2000.

[39] Y. Ma, S. Soatto, J. Kosecka, S. S. Sastry, An Invitation to 3-D Vision, Springer, 2004.

[40] Y. Ma, S. Soatto, J. Kosecka, S. S. Sastry, An invitation to 3-D vision, in: `http://vision.ucla.edu/MASKS/`.

[41] D. Devarajan, Z. Cheng, R. Radke, Calibrating distributed camera networks, Proceedings of the IEEE (Special Issue on Distributed Smart Cameras) 96 (10).

[42] R. Radke, A survey of distributed computer vision algorithms, in: H. Nakashima, J. Augusto, H. Aghajan (Eds.), Handbook of Ambient Intelligence and Smart Environments, Springer, 2008.