Chapter 13

Structure from motion

13.1 Introduction

In this chapter and the next, we will describe tools and techniques for obtaining information about the geometry of 3D scenes from 2D images. This task is challenging because the image formation process is not generally invertible: from its projected position in a camera image plane, a scene point can only be recovered up to a one-parameter ambiguity corresponding to its distance from the camera. Hence, additional information is needed to solve the reconstruction problem.

One possibility is to exploit prior knowledge about the scene to reduce the number of degrees of freedom. For example parallelism and coplanarity constraints can be used to reconstruct simple geometric shapes such as line segments and planar polygons from their projected positions in individual views. This is the approach considered in the next chapter, which describes a practical interactive system for recovering geometric models from photographs of architectural scenes.

Another possibility is to use corresponding image points in *multiple* views. Given its image in two or more views, a 3D point can be reconstructed by *triangulation*. An important prerequisite is the determination of camera calibration and pose, which may be expressed by a *projection matrix*. The geometrical theory of *structure from motion* allows projection matrices and 3D points to be computed simultaneously using only corresponding points in each view. More formally, given n projected points $\mathbf{u}_{ij}$, $i \in \{1 \ldots m\}$, $j \in \{1 \ldots n\}$ in m images, the goal is to find both projection matrices $\mathbf{P}_1 \ldots \mathbf{P}_m$ and a consistent structure $\mathbf{X}_1 \ldots \mathbf{X}_n$. This chapter reviews briefly some associated theory and surveys existing work in this area.

Structure from motion techniques are used in a wide range of applications including photogrammetric survey [23], the automatic reconstruction of virtual reality models from video sequences [55], and for the determination of camera motion (*e.g.* so that computer-generated objects can be inserted into video footage of real-world scenes [25]).

13.1.1 Chapter outline

This chapter begins by reviewing the familiar pinhole projection model (Section 13.2) and explaining how a projection matrix can be determined in the laboratory by photographing a suitable calibration target (Section 13.3). Having identified corresponding points in two views (Section 13.4), it is possible to compute their *epipolar geometry* (Section 13.5). This relationship is expressed algebraically by a *fundamental matrix*, which can decomposed to recover a pair of compatible projection matrices (Section 13.6). Then associated 3D points can obtained by triangulation (Section 13.7). Methods for obtaining structure and motion parameters for more than two views are described in Section 13.8. The final stage of most structure from motion algorithms is *bundle adjustment*, which is used to obtain a maximum likelihood parameter values by non-linear optimisation (Section 13.9).

13.2 Image projection

13.2.1 Notation

In what follows, it will be convenient to work with *homogenous*[1] as well as Euclidean coordinates. In homogenous coordinates, a point in N-dimensional space is expressed by a vector with $N+1$ elements that is defined only up to scale, *i.e.* multiplying the vector by an arbitrary non-zero scale factor will not change its meaning. Provided the $N+1$'th element is non-zero, a homogenous coordinate may be related to its Euclidean equivalent by dividing the first N elements by the $N+1$'th. Otherwise, the coordinate describes a point at infinity.

For example, a homogenous 3D point has the form $\tilde{\mathbf{X}} \sim [\,\tilde{X} \quad \tilde{Y} \quad \tilde{Z} \quad \tilde{W}\,]^\top$ (where $\sim$ means equality up to scale). Provided $\tilde{W}$ is non-zero, $\tilde{\mathbf{X}}$ is related to its Euclidean equivalent $\mathbf{X} = [\,X \quad Y \quad Z\,]^\top$ by the following equations:

$$\mathbf{X} = [\,\tilde{X}/\tilde{W} \quad \tilde{Y}/\tilde{W} \quad \tilde{Z}/\tilde{W}\,]^\top \qquad \tilde{\mathbf{X}} \sim [\,X \quad Y \quad Z \quad 1\,]^\top \tag{13.1}$$

Similarly, a homogenous 2D point $\tilde{\mathbf{x}} \sim [\,\tilde{x} \quad \tilde{y} \quad \tilde{w}\,]^\top$ is related to its Euclidean equivalent $\mathbf{x} = [\,x \quad y\,]^\top$:

$$\mathbf{x} = [\,\tilde{x}/\tilde{w} \quad \tilde{y}/\tilde{w}\,]^\top \qquad \tilde{\mathbf{x}} \sim [\,x \quad y \quad 1\,]^\top \tag{13.2}$$

13.2.2 Pinhole camera

The most common camera model is *pinhole projection*. This model is a good approximation to the behaviour of most real cameras, although in some cases it can be improved by taking non-linear effects (such as radial distortion) into account. According to the pinhole projection model, the relationship between

[1] Or *homogeneous*.

a 3D point and its corresponding 2D image point has three components, which are described below:

1. The first component is the rigid body transformation that relates points $\tilde{\mathbf{X}} \sim [\,X \quad Y \quad Z \quad 1\,]^{\top}$ in the world coordinate system to points $\tilde{\mathbf{X}}_{\mathrm{c}} \sim [\,X_{\mathrm{c}} \quad Y_{\mathrm{c}} \quad Z_{\mathrm{c}} \quad 1\,]$ in the camera coordinate system (see Figure 13.1). This transformation can be written as:

$$\begin{bmatrix} X_{\mathrm{c}} \\ Y_{\mathrm{c}} \\ Z_{\mathrm{c}} \\ 1 \end{bmatrix} \sim \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{13.3}$$

   where $\mathbf{R}$ is a 3×3 rotation matrix that represents camera orientation and $\mathbf{T}$ is a three vector that represents camera translation, *i.e.* the position of the world origin O in the camera coordinate system. Together, these are known as camera *extrinsic* parameters and describe camera pose.

2. The second component is the 3D to 2D transformation that relates 3D points $\tilde{\mathbf{X}}_{\mathrm{c}} \sim [\,X_{\mathrm{c}} \quad Y_{\mathrm{c}} \quad Z_{\mathrm{c}} \quad 1\,]$ (in the camera coordinate system) to 2D points $\tilde{\mathbf{x}} \sim [\,x \quad y \quad 1\,]^{\top}$ on the camera image plane. By using similar triangles (Figure 13.1), we obtain the following relationship:

$$x = f\frac{X_{\mathrm{c}}}{Z_{\mathrm{c}}} \qquad\qquad y = f\frac{Y_{\mathrm{c}}}{Z_{\mathrm{c}}} \tag{13.4}$$

   where f is the focal length. Since changing the value of f corresponds simply to scaling the image, we can set $f = 1$ and account for the missing scale factor within the camera calibration matrix (below). Then, using homogenous coordinates, the relationship can be expressed by the following matrix equation:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_{\mathrm{c}} \\ Y_{\mathrm{c}} \\ Z_{\mathrm{c}} \\ 1 \end{bmatrix}. \tag{13.5}$$

   Because $\tilde{\mathbf{x}}$ is defined only up to scale, it is independent of the magnitude of $\mathbf{X}_c$, *i.e.* it depends only on the direction of the 3D point relative to the camera, and not how far away it is.

3. The final component is the 2D to 2D transformation that relates points $\tilde{\mathbf{x}}$ on the camera image plane to pixel coordinates $\tilde{\mathbf{u}} \sim [\,u \quad v \quad 1\,]^{\top}$. This is written as follows:

$$\tilde{\mathbf{u}} \sim \mathbf{K}\tilde{\mathbf{x}} \tag{13.6}$$

   where $\mathbf{K}$ is an upper triangular camera calibration matrix of the form:

$$\mathbf{K} = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{13.7}$$

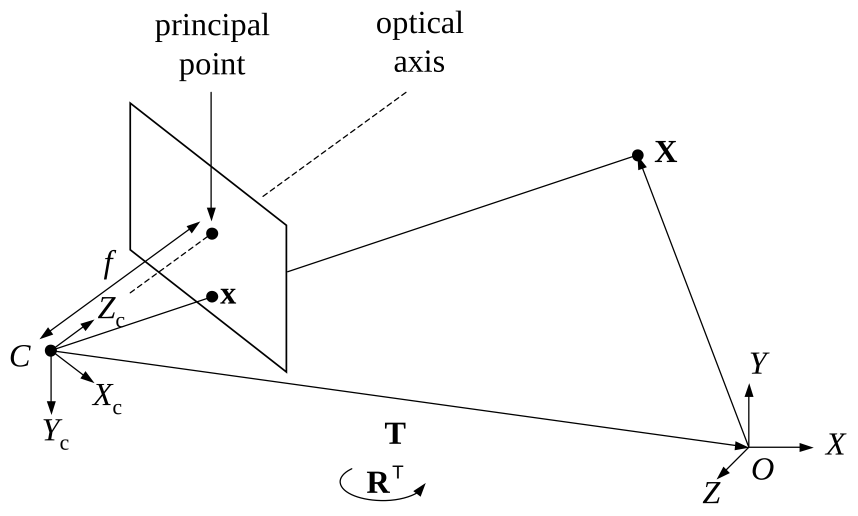


Figure 13.1: Pinhole projection of a 3D point $\mathbf{X}$ onto a camera image plane. The extrinsic parameters of the camera $\mathbf{R}$, $\mathbf{T}$ represent the rigid body transformation between the world XYZ coordinate system (origin O) and the camera $X_cY_cZ_c$ coordinate system (origin C). Note that the image plane is shown here in front of the optical centre C. In a real camera, the image plane would be behind the optical centre, and the image would be inverted.

and α_u and α_v are scale factors, s is skew, and $\mathbf{u}_0 = [\,u_0 \quad v_0\,]^\top$ is the principal point. These are camera *intrinsic* parameters. Usually, pixels are assumed to be square in which case $\alpha_u = \alpha_v = \alpha$ and $s = 0$. Hence, α can be considered to be the *focal length* of the lens expressed in units of the pixel dimension. The principal point is where the optical axis intersects that camera's image plane.

Finally, it is convenient to combine equations 13.3, 13.5, and 13.6 into a single linear equation. Using homogenous coordinates, a 3D point $\tilde{\mathbf{X}}$ is related to its pixel position $\tilde{\mathbf{u}}$ in a 2D image array by the following relationship:

$$\tilde{\mathbf{u}} \sim \mathbf{P}\tilde{\mathbf{X}} \tag{13.8}$$

where $\mathbf{P} \sim \mathbf{K}[\,\mathbf{R} \quad \mathbf{T}\,]$ is a 3×4 *projection matrix*.

13.2.3 Radial distortion

Lens distortion means that image points are displaced from the position predicted by the ideal pinhole projection model. The most common form of distortion is *radial distortion*, which is inherent in all single-element lenses. Under radial distortion, image points are displaced in a radial direction from the centre

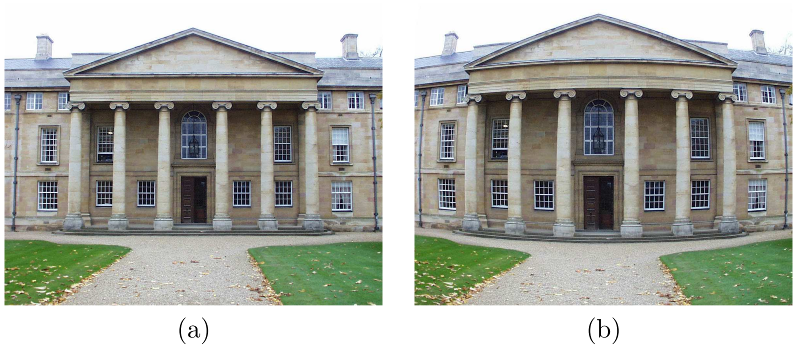


Figure 13.2: Radial distortion illustration. (a) Distortion-free image. (b) Under radial distortion, image points are displaced radially from the image centre.

of distortion (see Figure 13.2). In some cases, the linear projection model in equation 13.8 can be significantly improved by taking this effect into account.

Let $\tilde{\mathbf{x}} \sim [\,x \quad y \quad 1\,]^\top$ denote the image point associated with pixel position $\tilde{\mathbf{u}} \sim [\,u \quad v \quad 1\,]^\top$, *i.e.* $\tilde{\mathbf{x}} \sim \mathbf{K}^{-1}\tilde{\mathbf{u}}$. Under the assumption that the centre of distortion is the same as the principal point, radial distortion can be corrected using the following equation (Tsai [50], Weng et al. [53]):

$$\hat{x} = x + L(r)x \tag{13.9}$$

$$\hat{y} = y + L(r)y \tag{13.10}$$

where $[\,\hat{x} \quad \hat{y}\,]^\top$ is the corrected point for $[\,x \quad y\,]^\top$ and $r^2 = x^2 + y^2$. $L(r)$ is the distortion function and can be approximated by $L(r) \approx k_1 r^2 + k_2 r^4$. k_1 and k_2, the coefficients of radial distortion, are considered to be camera intrinsic parameters. In what follows, radial distortion will be assumed to be negligible unless otherwise specified, *i.e.* $k_1 = k_2 = 0$.

Sometimes it is necessary to find an inverse of the relationship in equations 13.9 and 13.10, *e.g.* when correcting an image for radial distortion. However, the true inverse is not simple to express mathematically. Hence, under the assumption that radial distortion is small, the following approximation is often used instead:

$$x \approx \hat{x} - L(r)\hat{x} \tag{13.11}$$

$$y \approx \hat{y} - L(r)\hat{y} \tag{13.12}$$

13.3 Camera calibration

Camera intrinsic and extrinsic parameters can be determined for a particular camera and lens combination by photographing a controlled scene. For example, we might position the camera to view the calibration object shown in Figure 13.3a and automatically extract the image positions of known 3D points (Figure 13.3b). Let $[\,u_i \quad v_i\,]^\top$ be the measured image position of 3D point

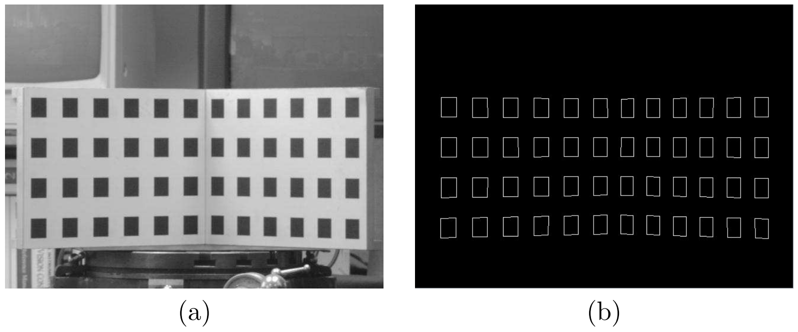


Figure 13.3: A photograph of a camera calibration object (a) from which the image positions of known 3D points can be extracted automatically (b).

$[X_i \quad Y_i \quad Z_i]^\top$. From 13.8, each such correspondence generates two equations that the elements of the projection matrix $\mathbf{P}$ must satisfy:

$$\begin{aligned} u_i &= \frac{p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}} \\ v_i &= \frac{p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}. \end{aligned}$$

These equations can be rearranged to give two linear equations in the 12 unknown elements of $\mathbf{P}$. For n calibration points we have $2n$ equations:

$$\begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -u_1X_1 & -u_1Y_1 & -u_1Z_1 & -u_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1X_1 & -v_1Y_1 & -v_1Z_1 & -v_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -u_nX_n & -u_nY_n & -u_nZ_n & -u_n \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_nX_n & -v_nY_n & -v_nZ_n & -v_n \end{bmatrix} \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ p_{34} \end{bmatrix} = \mathbf{0}.$$

Since there are 11 unknowns (scale is arbitrary), we need to observe at least 6 3D points to recover the projection matrix and calibrate the camera.

13.3.1 Numerical considerations

The equations can be solved using orthogonal least squares. First, we write them in matrix form:

$$\mathbf{Ap} = \mathbf{0} \tag{13.13}$$

where $\mathbf{p}$ is the 12×1 vector of unknowns (the 12 elements of the projection matrix, p_{ij}), $\mathbf{A}$ is the $2n \times 12$ matrix of measurements, and n is the number of 3D points. The linear least squares solution minimizes $||\mathbf{Ap}||$ subject to $||\mathbf{p}|| = 1$

and is given by the unit eigenvector corresponding to the smallest eigenvalue of $\mathbf{A}^\top\mathbf{A}$. Numerically, this computation is performed via the *singular value decomposition* [41] of the matrix:

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^\top$$

where $\boldsymbol{\Lambda} = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_{12})$ is the diagonal matrix of singular values and the matrices $\mathbf{U}$ and $\mathbf{V}$ are orthonormal. The columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{A}^\top\mathbf{A}$ and the required solution is the column of $\mathbf{V}$ corresponding the smallest singular value σ_{12}. However, the least squares solution is only approximate and should be used as the starting point for non-linear optimisation: *i.e.* finding the elements of the projection matrix $\mathbf{P}$ that minimize the sum of squared errors between the measured and predicted pixel positions $\mathbf{u}_i$ and $\hat{\mathbf{u}}_i(\mathbf{P}, \mathbf{X}_i)$:

$$\min_{\mathbf{P}} \sum_i ||\mathbf{u}_i, \hat{\mathbf{u}}_i(\mathbf{P}, \mathbf{X}_i)||^2.$$

At this stage, it is also possible to extend the projection model $\hat{\mathbf{u}}_i(\mathbf{P}, \mathbf{X}_i)$ to take account of radial distortion if desired.

Once the projection matrix has been estimated, the first 3×3 sub-matrix can be decomposed (by *QR decomposition* [41]) into an upper triangular camera calibration matrix $\mathbf{K}$ and an orthonormal rotation matrix $\mathbf{R}$.

13.4 The correspondence problem

The geometrical theory of structure from motion assumes that one is able to solve the *correspondence problem*, which is to identify points in two or more views that are the projections of the same point in space.

One solution is to identify corresponding points *interactively* in each view. An important advantage is that surfaces can be defined simultaneously with correspondences, *e.g.* by having the user identify geometric primitives such as cuboids and prisms [6]. A disadvantage is that the interactive approach is time consuming; also, the accuracy of the resulting reconstruction will depend critically on how carefully the user positions the image points.

Years of research [1, 12, 13, 21, 31, 34, 38] have shown that, in general, the correspondence problem is difficult to solve automatically. Automatic algorithms work by computing some measure of agreement between image pixels. Usually, it is impossible to compare every pixel of one image with every pixel of the next because of combinatorial complexity. In any case, not all points are equally well suited for matching. Hence, local-scale image features are used instead.

Feature matching works by detecting *interest points* in the images, *e.g.* Harris corner points [14] are located at the maxima of the local image auto-correlation function. The local neighbourhoods of such points contain a lot of intensity variation and hence they are comparatively easier to differentiate.

Having detected interest points, image appearance in their local vicinity is characterised by an appropriate descriptor. Features with more similar descriptors are considered to be more likely matches.

Feature matching techniques may be divided into two categories: *narrow-* and *wide-baseline*.

Narrow-baseline matching. Under the assumption that the change in camera position and orientation is small, the local vicinity of interest points will look similar in two nearby views. Hence, image features can be characterised simply by a set of pixel intensity values sampled from a rectangular window centred on the interest point. Pixel intensity values are compared by *normalised cross-correlation* or the *sum of squared differences* [22].

A disadvantage of narrow-baseline matching is that depth computation is quite sensitive to image coordinate measurement noise for closely spaced viewpoints. However, by tracking feature correspondences throughout video sequences, it is possible to recover structure and motion parameters accurately (see Section 13.8, below). Impressive results have been demonstrated by Beardsley et al. [3], Pollefeys et al. [39], Fitzgibbon and Zisserman [11], and Zisserman et al. [55], amongst others.

Wide-baseline matching. Whilst narrow-baseline matching algorithms perform well for the closely spaced viewpoints in video sequences, they are not applicable to images obtained from viewpoints in more general configuration. When the baseline is large, surfaces in the two images may exhibit substantial change of scale, different degrees of foreshortening, different patterns of occlusion, and large disparities in their locations. All of these factors make it much more difficult to determine correct correspondences automatically.

In recent years, considerable progress has been made in the development of wide-baseline matching algorithms that are invariant to various classes of image transformation. For example, Schmid et al. [45] characterise image features using rotationally invariant Gaussian derivatives so that matching can be achieved in a way that is immune to image rotation (see also [2, 32]). Dufournaud et al. [7] use a multi-scale framework to match images at different scales. Here, features are detected at several scales and a robust matching algorithm is used to select the correct scale relationship. Also, Lowe [28] achieves scale invariance by detecting features at local maxima in image *scale space*[2] (also [32, 33]). Finally, Pritchett and Zisserman [42] have described an affine-invariant feature matching approach that works by finding homographies relating dominant planes between views. Other affine-invariant matching schemes work by iteratively refining the shape of an interest region until some appropriate convergence criteria is met [2, 33, 51].

[2] *Scale-space* features are located at image positions where the response to a Laplacian filter is at a local maximum in the three-dimensional parameter space defined by the image u, v coordinate and the scale of the Laplacian.

13.5 Two-view geometry

In 1913, Kruppa [24] proved the fundamental result that given two views of five distinct 3D points, one could recover the relative position and orientation of the cameras as well as the positions of the points (up to an unknown global scale factor). Then the work of Longuet-Higgins [27] in the early 1980s showed how an *essential matrix* relating a pair of calibrated views can be estimated from eight or more point correspondences by solving a linear equation, and also how the essential matrix can be decomposed to give relative camera orientation and position.

13.5.1 The essential matrix

Given the projection of a 3D point in one image, its projection in a second image is restricted to the corresponding *epipolar line*. This is illustrated in Figure 13.4. The epipolar line $\tilde{\mathbf{l}}'$ can be seen to be the projection of the ray going from the optical centre C through the image point $\tilde{\mathbf{x}}$ on the first image plane. This is equivalent to intersecting the plane generated by the optical centres C, C' and the image point $\tilde{\mathbf{x}}$ (epipolar plane) with the second image plane. Note that all epipolar lines in an image have a common point: the projection of the second optical centre. This point is called the *epipole* and is denoted $\tilde{\mathbf{e}}$ and $\tilde{\mathbf{e}}'$ respectively for first and second cameras in Figure 13.4.

The epipolar constraint can be formulated algebraically using the essential matrix $\mathbf{E}$, which relates corresponding image points in two views [8]. Consider two pinhole cameras with projection matrices $\mathbf{P}$ and $\mathbf{P}'$. Given a Euclidean point $\mathbf{X}'$ in the coordinate system of camera C', its position $\mathbf{X}$ in the coordinate system of C is given by:

$$\mathbf{X} = \mathbf{R}\mathbf{X}' + \mathbf{T} \tag{13.14}$$

where $\mathbf{R}$ is a 3×3 rotation matrix and $\mathbf{T}$ is a 3-vector. Pre-multiplying both sides by $\mathbf{X}^\top [\mathbf{T}]_\times$ gives:

$$\mathbf{X}^\top [\mathbf{T}]_\times \mathbf{R}\mathbf{X}' = \mathbf{X}^\top \mathbf{E}\mathbf{X}' = 0 \tag{13.15}$$

where 3×3 essential matrix $\mathbf{E} \sim [\mathbf{T}]_\times \mathbf{R}$ and $[\mathbf{T}]_\times$ is the cross product matrix[3]. Equation 13.15 also holds for image points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$, which gives the *epipolar constraint*:

$$\tilde{\mathbf{x}}^\top \mathbf{E}\tilde{\mathbf{x}}' = 0 \tag{13.16}$$

Note that $\mathbf{E}$ depends only on $\mathbf{R}$ and $\mathbf{T}$ and is defined only up to an arbitrary scale factor. Thus, it has 5 parameters.

[3]For $\mathbf{T} = [\,t_x \quad t_y \quad t_z\,]^\top$, $[\mathbf{T}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$.

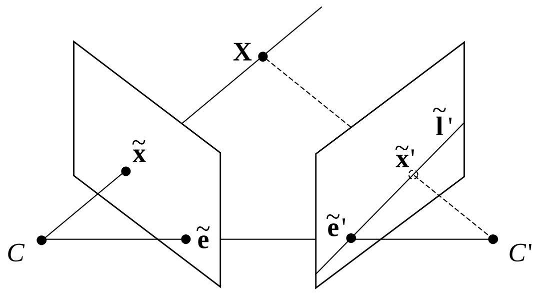


Figure 13.4: Epipolar geometry for two cameras. Given the projection $\tilde{\mathbf{x}}$ of a 3D point $\mathbf{X}$ in one image, its projection $\tilde{\mathbf{x}}'$ in a second image is restricted to the corresponding *epipolar line* $\tilde{\mathbf{l}}'$.

13.5.2 The fundamental matrix

From equation 13.6, image points $\tilde{\mathbf{x}}$ may be related to pixel positions $\tilde{\mathbf{u}}$ by the inverse camera calibration matrix $\mathbf{K}^{-1}$:

$$\tilde{\mathbf{x}} \sim \mathbf{K}^{-1}\tilde{\mathbf{u}} \tag{13.17}$$

This means the epipolar constraint (equation 13.16) may be rewritten in terms of pixel positions:

$$\begin{aligned} (\mathbf{K}^{-1}\tilde{\mathbf{u}})^\top \mathbf{E}(\mathbf{K}'^{-1}\tilde{\mathbf{u}}') &= 0 \\ \tilde{\mathbf{u}}^\top (\mathbf{K}^{-1\top}\mathbf{E}\mathbf{K}'^{-1})\tilde{\mathbf{u}}' &= 0 \\ \tilde{\mathbf{u}}^\top \mathbf{F}\tilde{\mathbf{u}}' &= 0 \end{aligned} \tag{13.18}$$

where $\mathbf{F} \sim \mathbf{K}^{-1\top}\mathbf{E}\mathbf{K}'^{-1}$ is the *fundamental matrix* [8]. $\mathbf{F}$ is a 3×3 matrix that has rank 2 (the epipole $\tilde{\mathbf{e}}$ is the null space of $\mathbf{F}$). It can be estimated linearly given 8 or more corresponding points. Considerable attention has been given to the problem of estimating the fundamental matrix accurately from noisy image data [16], and robustly in the presence of outliers [48, 54].

13.5.3 Estimating the fundamental matrix

From equation 13.18, we see that each point correspondence, $\tilde{\mathbf{u}}_i \sim [\, u_i \quad v_i \quad 1 \,]^\top$ and $\tilde{\mathbf{u}'}_i \sim [\, u'_i \quad v'_i \quad 1 \,]^\top$, generates one constraint on the elements of the fundamental matrix $\mathbf{F}$:

$$\begin{bmatrix} u'_i & v'_i & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = 0.$$

For n pairs of correspondences, the constraints can be rearranged as linear equations in the 9 unknown elements of the fundamental matrix:

$$\begin{bmatrix} u'_1u_1 & u'_1v_1 & u'_1 & v'_1u_1 & v'_1v_1 & v'_1 & u_1 & v_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u'_nu_n & u'_nv_n & u'_1 & v'_nu_n & v'_nv_n & v'_n & u_n & v_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = \mathbf{0}$$

or in matrix form:

$$\mathbf{Af} = \mathbf{0}$$

where $\mathbf{A}$ is an $n \times 9$ measurement matrix, and $\mathbf{f}$ represents the elements of the fundamental matrix f_{ij} as a 9-vector. Given 8 or more correspondences a least squares solution[4] can be found as the unit eigenvector ($\mathbf{f}$ is defined up to an arbitrary scale) corresponding to the minimum eigenvalue of $\mathbf{A}^\top\mathbf{A}$. A unique solution is obtained unless the points and the camera centres lie on a ruled quadric or all the points lie on a plane (Faugeras and Maybank [9]).

The computation can be poorly conditioned and it is important to pre-condition the image points by normalizing them to improve the condition number of $\mathbf{A}^\top\mathbf{A}$ before estimating the elements of the fundamental matrix by singular value decomposition (Hartley [16]).

Two steps can be taken to improve the solution if desired (Luong and Faugeras [29]). The most important requires enforcing the rank 2 property of the fundamental matrix. This can be achieved by a suitable parameterisation of $\mathbf{F}$. Another improvement requires finding the 7 independent parameters of the fundamental matrix that minimize the distances between the image points and their epipolar lines.

13.6 Recovering projection matrices

As shown above, the fundamental matrix depends on the relative position and orientation of a pair of views, and can be estimated using point correspondences. In this section, we explain how a fundamental matrix can be decomposed to recover the camera motion, and, thereby, camera projection matrices. As we shall see, provided that the camera calibration matrices are known, it is possible to recover a pair of compatible projection matrices up to a 1 parameter ambiguity, corresponding to an unknown scale for the camera translation.

[4]Note that the fundamental matrix has only 7 degrees of freedom since its determinant must be zero. A non-unique solution can be obtained from only 7 point correspondences and is described by Huang and Netravali [19].

If the camera calibration matrices $\mathbf{K}$ and $\mathbf{K}'$ are known, we can transform the recovered fundamental matrix into an essential matrix (13.18)

$$\mathbf{E} \sim \mathbf{K}'^{\top}\mathbf{F}\mathbf{K} \tag{13.19}$$

and decompose this matrix into a skew-symmetric matrix corresponding to translation and an orthonormal matrix corresponding to the rotation between the views:

$$\mathbf{E} \sim [\mathbf{T}]_{\times}\mathbf{R}. \tag{13.20}$$

The latter is in fact only possible if the essential matrix has rank 2 and two equal singular values.

Numerical considerations

This decomposition can be achieved by computing the singular value decomposition [41] of the essential matrix (Hartley [15]):

$$\mathbf{E} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^{\top} \tag{13.21}$$

where $\mathbf{\Lambda} = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)$ and the matrices $\mathbf{U}$ and $\mathbf{V}$ are orthogonal. The decomposition into a translation vector and the rotation between the two views requires that $\sigma_1 = \sigma_2 \neq 0$ and $\sigma_3 = 0$. The nearest essential matrix (in the sense of minimizing the Frobenius norm between the two matrices) with the correct properties can be obtained by setting the two largest singular values to be equal to their average and the smallest one to zero (Hartley [15]). The translation and axis and angle of rotation can then be obtained directly up to arbitrary signs and unknown scale for the translation:

$$[\mathbf{T}]_{\times} = \mathbf{U}\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\mathbf{U}^{\top} \tag{13.22}$$

$$\mathbf{R} = \mathbf{U}\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\mathbf{V}^{\top} \tag{13.23}$$

The projection matrices follow directly from the recovered translation and rotation by aligning the reference coordinate system with the first camera to give:

$$\begin{aligned} \mathbf{P} &= \mathbf{K}[\mathbf{I} \mid \mathbf{0}] \\ \mathbf{P}' &= \mathbf{K}'[\mathbf{R} \mid \mathbf{T}] \end{aligned}$$

where $\mathbf{T}$ is typically scaled such that $|\mathbf{T}| = 1$. Four solutions are still possible due to the arbitrary choice of signs for the translation $\mathbf{T}$ and rotation $\mathbf{R}$, however the correct one is easily disambiguated by ensuring that the reconstructed points lie in front of the cameras.

13.7 Triangulation

Given projection matrices, 3D points can be computed from their measured image positions in two or more views. This is *triangulation* [17]. Ideally, 3D points should lie at the point of intersection of the back-projected rays. However, because of measurement noise, back-projected rays will not generally intersect. Thus 3D points must be chosen in such a way as to minimize an appropriate error metric.

The 'gold standard' reconstruction algorithm minimizes the sum of squared errors between the measured and predicted image positions of the 3D point in all views in which it is visible, *i.e.*

$$\mathbf{X} = \arg\min_{\mathbf{X}} \sum_i ||\mathbf{u}_i - \hat{\mathbf{u}}_i(\mathbf{P}_i, \mathbf{X})||^2.$$

where $\mathbf{u}_i$ and $\hat{\mathbf{u}}_i(\mathbf{P}_i, \mathbf{X})$ are the measured and predicted image positions in view i (see Figure 13.5). Under the assumption that image coordinate measurement noise is Gaussian-distributed, this approach gives the *maximum likelihood* solution for $\mathbf{X}$. Hartley and Sturm [17] describe a non-iterative solution for two views. For more than two views, the minimization can be achieved iteratively by non-linear optimisation. However, this approach necessitates a sufficiently good initialisation, otherwise it might fail by finding a local minimum cost solution. A popular strategy works by exploiting equation 13.8. Because the homogenous 3-vectors $\tilde{\mathbf{u}}_i$ and $\mathbf{P}_i\tilde{\mathbf{X}}$ are parallel, it is possible to write:

$$[\tilde{\mathbf{u}}_i]_\times \mathbf{P}_i \tilde{\mathbf{X}} = 0.$$

This equation has three rows but provides only two constraints on $\tilde{\mathbf{X}}$ since each row can be expressed as a linear combination of the other two. All such constraints can be arranged into a matrix equation of the form

$$\mathbf{A}\tilde{\mathbf{X}} = 0$$

where $\mathbf{A}$ is a $3n \times 4$ matrix and n is the number of views in which the reconstructed point is visible. The required solution for the homogenous 3D point $\tilde{\mathbf{X}}$ minimizes $||\mathbf{A}\tilde{\mathbf{X}}||$ subject to $||\tilde{\mathbf{X}}|| = 1$ and is given by the eigenvector of $\mathbf{A}^\top\mathbf{A}$ corresponding to the smallest eigenvalue. It can be found by the singular value decomposition of the symmetric matrix $\mathbf{A}^\top\mathbf{A}$.

13.8 Multiple-view structure from motion

We have seen that the essential or fundamental matrix encapsulates the geometric constraint relating pairs of views. Next, we will turn our attention to solving the structure and motion problem for an arbitrary number of views. The final stage is usually *bundle adjustment*, which is used iteratively to refine structure and motion parameters by the minimisation of an appropriate cost

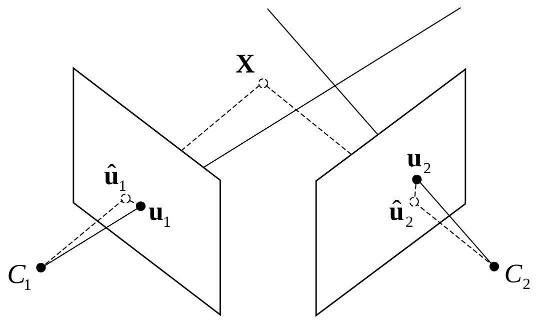


Figure 13.5: Triangulation illustration. Given projection matrices, a 3D point $\mathbf{X}$ can be computed from its measured pixel positions $(\mathbf{u}_1, \mathbf{u}_2, \ldots)$ in two or more views $(C_1, C_2, \ldots)$. Ideally, $\mathbf{X}$ should lie at the intersection of the back-projected rays (solid lines). However, because of measurement noise, these rays will not generally intersect. Hence $\mathbf{X}$ should be chosen so as to minimise the sum of squared errors between measured and predicted pixel positions ($\mathbf{u}_i$ and $\hat{\mathbf{u}}_i$).

function (see Section 13.9, below). However, bundle adjustment depends critically on a suitable initialisation – otherwise the algorithm may fail by converging to a local minimum cost solution. The following sections discuss *sequential* and *factorisation* algorithms for m-view structure from motion.

13.8.1 Sequential methods

Sequential algorithms are the most popular. They work by incorporating successive views one at a time (*e.g.* Figure 13.6). As each view is registered, a partial reconstruction is extended by computing the positions of all 3D points that are visible in two or more views using *triangulation* (see Section 13.7). A suitable initialisation is typically obtained by decomposing the fundamental matrix relating the first two views of the sequence (see Section 13.6).

There exist several strategies for registering successive views:

- **Epipolar constraints**. One possibility is to exploit the two-view epipolar geoemtry that relates each view to its predecessor. For example, where camera intrinsic parameters are known, essential matrices can be used (Figure 13.6). Essential matrices are estimated linearly using 8 or more point correspondences and decomposed to give relative camera orientation and the direction of camera translation (Sections 13.5 and 13.6). The magnitude of the translation can be fixed using the image in the new view of a single known 3D point, *i.e.* a point that has already been reconstructed from its image in earlier views.

- **Resection.** An alternative is to determine the pose of each additional

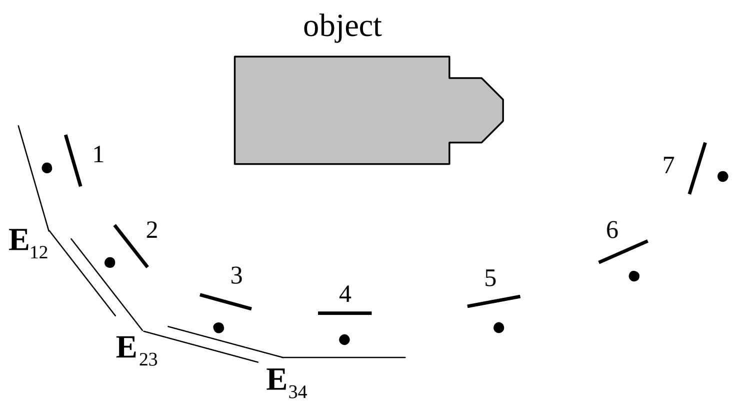


Figure 13.6: Sequential registration illustration. Views 1 to 7 are registered one at a time by computing the essential matrices $\mathbf{E}_{12}$, $\mathbf{E}_{23}$, *etc.* relating each one to its predecessor. The essential matrix can be decomposed to give relative orientation and the direction of translation and 3D to 2D correspondences are used to determine the magnitude of the translation. As each new view is incorporated, the partial reconstruction is extended by reconstructing all 3D points that are visible in two or more views.

view using already-reconstructed 3D points [3, 4, 15, 40]. As we have seen, 6 or more 3D to 2D correspondences allow linear solution for the 12 elements of a projection matrix (Section 13.3).

- **Merging partial reconstructions.** Another alternative is to merge partial reconstructions using corresponding 3D points [10, 11]. Typically, two- or three-view reconstructions are obtained using adjacent image pairs or triplets; then they are merged using corresponding 3D points. In [11], longer reconstructions are built up hierarchically by merging progressively longer subsequences.

These sequential registration schemes have some important limitations. In the context of interactive modelling systems, one disadvantage is that a large number of corresponding points must be defined in each view. For uncalibrated reconstruction, commercial photogrammetry software (such as *ImageModeler*[5]) usually requires a minimum of 7 correspondences per view (and more are recommended for better accuracy). Since corresponding points must usually be visible in three or more views, this means substantial overlap is required. For long sequences of views (*e.g.* along a city street), this requirement can be prohibitive. Another complication is that there exist various kinds of degenerate structure and motion configuration for which the standard algorithms will fail [43]. For example: (i) camera rotation in the absence of translation, (ii) planar scenes, (iii) a 3D point lying on a line passing through the optical centres of the

[5]See http://www.realviz.com/

cameras in which it is visible. In practice, it may be hard to avoid these kinds of degeneracy, especially if views are obtained without careful (or even expert) planning.

13.8.2 Factorisation methods

Unlike sequential methods, *batch* methods work by computing camera pose and scene geometry using all image measurements simultaneously. One advantage is that reconstruction errors can be distributed meaningfully across all measurements; thus, gross errors associated with sequence closure can be avoided.

One family of batch structure from motion algorithms are called *factorisation* methods (after Tomasi and Kanade [47]). Fast and robust linear methods based on direct SVD factorisation of the image point measurements have been developed for a variety of simplified linear (*affine*) camera models, *e.g.* *orthographic* (Tomasi and Kanade [47]), *weak perspective* (Weinshall and Tomasi [52]), and *para-perspective* (Poelman and Kanade [37]). Unfortunately, none of these methods are generally applicable to real-world scenes because real camera lenses are too wide-angle to be approximated as linear.

More recently, a number of researchers have described 'factorisation-like' algorithms for *perspective* cameras too (Sturm and Triggs [46], Heyden [18], Schaffalitzky et al. [44]). However, these methods are iterative and there is no guarantee that they will converge to the optimal solution.

Again, one limitation of all of these algorithms [18, 37, 44, 46, 47, 52] is that there exist degenerate structure and motion configurations for which they will fail. Another is that they cannot cope with missing data, *i.e.* every 3D points must be visible in every view. Hence, they are not applicable to sparse modelling problems (except perhaps as a means of initialising sequential algorithms like in [44]). Jacobs [20] overcomes this limitation, but only for affine cameras (and at the expense of the optimality of the solution).

13.9 Bundle adjustment

From image features $\mathbf{u}_{ij}$, structure from motion gives an initial estimate of projection matrices $\mathbf{P}_i$ and 3D points $\mathbf{X}_j$. Usually it will be necessary to refine this estimate using iterative non-linear optimisation to minimize an appropriate cost function. This is *bundle adjustment* [5]. Bundle adjustment works by minimising a cost function that is related to a weighted sum of squared reprojection errors. Usually Gauss-Newton iteration is used (with an appropriate step control policy) for rapid convergence.

This section provides a brief review of established bundle adjustment theory. For a more comprehensive treatment of this subject, the reader is referred to the excellent article by Triggs et al. [49].

13.9.1 Problem definition

The goal of bundle adjustment is to determine an optimal estimate of a set of parameters $\boldsymbol{\theta}$, given a set of noisy observations. Most bundle parameters cannot be observed directly, *e.g.* projection matrices, 3D point coordinates. Instead, they allow us to make predictions of quantities that can, *e.g.* the measured pixel coordinates of imaged 3D points.

Let the set of predictions be $\mathbf{z}(\boldsymbol{\theta})$ and the set of corresponding observations be $\bar{\mathbf{z}}$. Then *residual prediction error* $\Delta\mathbf{z}$ is given by:

$$\Delta\mathbf{z} = \bar{\mathbf{z}} - \mathbf{z}(\boldsymbol{\theta}) \tag{13.24}$$

In general, the observation vector $\bar{\mathbf{z}}$ may be partitioned into a set of statistically independent measurements $\bar{\mathbf{z}}_1 \ldots \bar{\mathbf{z}}_N$ with associated predictions $\mathbf{z}_1(\boldsymbol{\theta}) \ldots \mathbf{z}_N(\boldsymbol{\theta})$.

Bundle adjustment proceeds by minimizing an appropriate cost function. For a maximum likelihood parameter estimate, the cost function should reflect the likelihood of the residual $\Delta\mathbf{z}$. Under the assumption of Gaussian-distributed measurement noise, the appropriate cost function is a sum of squared errors, which is the negative sum of log likelihoods:

$$f(\boldsymbol{\theta}) = \frac{1}{2}\sum_i \Delta\mathbf{z}_i(\boldsymbol{\theta})^\top \mathbf{W}_i \Delta\mathbf{z}_i(\boldsymbol{\theta}) \qquad \Delta\mathbf{z}_i(\boldsymbol{\theta}) = \bar{\mathbf{z}}_i - \mathbf{z}_i(\boldsymbol{\theta}) \tag{13.25}$$

Here $\Delta\mathbf{z}_i(\boldsymbol{\theta})$ is the feature prediction error and $\mathbf{W}_i$ is a symmetric positive definite weight matrix that is chosen to approximate the inverse covariance of the measurement noise associated with measurement $\bar{\mathbf{z}}_i$.

13.9.2 Numerical optimisation

Newton's method

The minimization of the cost function is an iterative process. At each iteration, we seek a parameter displacement $\boldsymbol{\theta} \to \boldsymbol{\theta} + \delta\boldsymbol{\theta}$ that minimizes $f(\boldsymbol{\theta})$. By fitting a Taylor expansion-based local model to the cost function, it is possible to solve approximately for $\delta\boldsymbol{\theta}$ using linear algebra [35].

The quadratic Taylor series is:

$$f(\boldsymbol{\theta} + \delta\boldsymbol{\theta}) \approx f(\boldsymbol{\theta}) + \mathbf{g}^T\delta\boldsymbol{\theta} + \frac{1}{2}\delta\boldsymbol{\theta}^T\mathbf{H}\delta\boldsymbol{\theta} \tag{13.26}$$

where $\mathbf{g}$ is the gradient vector, and $\mathbf{H}$ is the *Hessian* matrix. Under the assumption that the Hessian is positive definite, the local model is then a simple quadratic with a unique global minimum. Setting $\frac{df}{d\boldsymbol{\theta}}(\boldsymbol{\theta} + \delta\boldsymbol{\theta}) \approx \mathbf{H}\delta\boldsymbol{\theta} + \mathbf{g}$ to zero for the stationary point gives the *Newton step*:

$$\delta\boldsymbol{\theta} = -\mathbf{H}^{-1}\mathbf{g} \tag{13.27}$$

Iterating the Newton step gives *Newton's method*.

The Gauss-Newton approximation

Differentiating the weighted sum of squared errors cost function in equation (13.25) gives the Hessian and gradient in terms of the *Jacobian* $\mathbf{J} = \frac{d\mathbf{z}}{d\boldsymbol{\theta}}$:

$$\mathbf{g} \equiv \frac{df}{d\boldsymbol{\theta}} = \mathbf{J}^\top \mathbf{W} \Delta\mathbf{z} \qquad \mathbf{H} \equiv \frac{d^2 f}{d\boldsymbol{\theta}^2} = \mathbf{J}^\top \mathbf{W} \mathbf{J} + \sum_i (\Delta\mathbf{z}^\top \mathbf{W})_i \frac{d^2(\Delta\mathbf{z})_i}{d\boldsymbol{\theta}^2} \tag{13.28}$$

where $\mathbf{W} = \operatorname{diag}(\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_N)$ and the notation $(\ldots)_i$ means the i'th element of a vector. In practice, the $\frac{d^2(\Delta\mathbf{z})_i}{d\boldsymbol{\theta}^2}$ term in $\mathbf{H}$ is likely to be small in comparison to $\mathbf{J}^\top\mathbf{W}\mathbf{J}$ if either: (i) the prediction error $\Delta\mathbf{z}(\boldsymbol{\theta})$ is small or (ii) the model is nearly linear, $\frac{d^2 z_i}{d\boldsymbol{\theta}^2} = 0$ [49]. Dropping this term gives the *Gauss-Newton* approximation to the least squares Hessian, $\mathbf{H} \approx \mathbf{J}^\top\mathbf{W}\mathbf{J}$. Thus, the Newton step prediction equations become the *Gauss-Newton* or *normal* equations [35]:

$$(\mathbf{J}^\top\mathbf{W}\mathbf{J})\delta\boldsymbol{\theta} = -\mathbf{J}^\top\mathbf{W}\Delta\mathbf{z} \tag{13.29}$$

For well-parameterised bundle problems under an outlier-free least squares cost model evaluated near the cost minimum, the Gauss-Newton approximation is usually very accurate [49].

Parameterisation

We have seen that bundle adjustment works by fitting a local quadratic approximation to the cost function at each iteration. In practice, the rate of convergence depends critically on the quality of this approximation. Thus, it is important to choose a parameter representation such that the effect of parameter variation will be locally as near linear as possible on the chosen cost model [49]. An interesting problem is to parameterise camera orientations in such a way as to avoid the familiar singularities associated with Euler angle representations. A good solution is to leave camera orientations $\mathbf{R}_i$ as 3×3 rotation matrices but to compute incremental updates $\mathbf{R}_i \rightarrow \boldsymbol{\Delta}(\boldsymbol{\Omega}_i)\mathbf{R}_i$ at each iteration as function of a parameter vector $\boldsymbol{\Omega}_i = [\,\omega_x \quad \omega_y \quad \omega_z\,]^\top$. The 3×3 rotation matrix $\boldsymbol{\Delta}(\boldsymbol{\Omega}_i)$ is expressed using Rodriguez formula:

$$\boldsymbol{\Delta}(\boldsymbol{\Omega}_i) = \mathbf{I} + \sin\phi[\boldsymbol{\Omega}_i]_\times + (1 - \cos\phi)[\boldsymbol{\Omega}_i]_\times^2 \tag{13.30}$$

where $\phi = |\boldsymbol{\Omega}_i|$, and

$$[\boldsymbol{\Omega}_i]_\times = \frac{1}{\phi}\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{13.31}$$

For small ω_i, this method gives approximately the same result as applying a sequence of three Euler angle rotations ω_x, ω_y, and ω_z about the world X, Y, and Z axes. The same result could be obtained by representing incremental rotations using *quaternions* [36] or *geometric algebra* [26].

Step control

In practice, Newton's method may fail to converge to a minimum cost solution. It can converge to a saddle point instead of a minimum, and, for large steps, the quadratic cost model may be so inaccurate that the cost will actually increase. To guarantee convergence, a suitable *step control* policy must be added. Each step must follow a descent direction, *i.e.* a direction with a non-negligible downhill cost component, or, if the gradient is near a saddle point, down a negative curvature direction of the Hessian [35].

An elegant solution is the Levenberg-Marquardt method [30], which varies smoothly between the two extremes of Newton and gradient descent. Where the local quadratic cost model is accurate, the Newton step in equation 13.27 should give a significant cost reduction. Otherwise, simple gradient descent is about the best that can be achieved. Levenberg-Marquardt works by replacing the Hessian $\mathbf{H}$ in the Newton step equation (13.27) with a modified version $\mathbf{H}'$ that has been altered according to the following prescription:

$$h'_{jj} = h_{jj}(1 + \epsilon) \tag{13.32}$$

$$h'_{jk} = h_{jk} \qquad (j \neq k) \tag{13.33}$$

where h_{ij} means the i, j'th element of the matrix. When ϵ is large, the modified Hessian is forced to be diagonally dominant so that the update is a simple gradient descent. But as ϵ tends to zero, its effect becomes negligible and we have Newton descent. ϵ is adjusted dynamically so that it tends to decrease as the quadratic approximation to the cost function improves.

The complete procedure is as follows [41]:

1. Compute the cost function $f(\boldsymbol{\theta})$
2. Set $\epsilon = 0.001$.
3. Solve the normal equations (13.29) for $\delta\boldsymbol{\theta}$ using the *modified* Hessian $\mathbf{H}'$ and evaluate $f(\boldsymbol{\theta} + \delta\boldsymbol{\theta})$
4. If $f(\boldsymbol{\theta} + \delta\boldsymbol{\theta}) \geq f(\boldsymbol{\theta})$, increase ϵ by a factor of 10 and go back to 3.
5. If $f(\boldsymbol{\theta} + \delta\boldsymbol{\theta}) \leq f(\boldsymbol{\theta})$, decrease ϵ by a factor of 10, update the trial solution $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \delta\boldsymbol{\theta}$, and go back to 3.

Convergence criteria

By iterating the Gauss-Newton step (possibly modified according to the above step control strategy), we hope to obtain a progressively better parameter estimate. The remaining difficulty is to decide when to stop iterating. Since the minimum cost solution is at best only a statistical estimate of the true solution, it will usually be wasteful and unnecessary to iterate until convergence to machine accuracy or round off limit. In any case, it is not uncommon to find that the update equation becomes near-degenerate in the vicinity of the minimum because the cost surface has a valley-like topography. This means there is no well-defined solution for the update step; instead, a range of solutions is about equally good. In consequence, the Levenberg-Marquardt method is prone to wandering about in parameter space without achieving any significant reduction in cost. A simple but effective solution is described in [41]. Iteration is stopped on the second successive occasion where the cost function $f(\boldsymbol{\theta})$ decreases[6] by less than a small fractional amount (10^{-3}).

13.10 Summary

In this chapter, we have described methods for the simultaneous recovery of 3D points and camera projection matrices using corresponding image points in multiple views. This is structure from motion. Given sufficiently many corresponding points in two calibrated views, it will be possible in general to compute a fundamental matrix, which can be decomposed to give the camera's motion (at least up to an unknown scale factor for the translation). Then 3D point coordinates can be computed by triangulation. There also exist sequential and factorisation-based structure from motion algorithms for more than two views. The final stage of most algorithms is bundle adjustment, which is used iteratively to refine structure and motion parameters to achieve a maximum likelihood estimate by the minimisation of an appropriate cost function.

Bibliography

[1] H. H. Baker and T. O. Binford. Depth from edge and intensity based stereo. In *International Joint Conference on Artificial Intelligence*, pages 631–636, 1981.

[2] A. Baumberg. Reliable feature matching across widely separated views. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'00)*, pages 774–781, 2000.

[3] P. A. Beardsley, P. Torr, and A. Zisserman. 3D model acquisition from extended image sequences. In *European Conference on Computer Vision (ECCV'96)*, pages 683–695, 1996.

[4] P. A. Beardsley, A. Zisserman, and D. Murray. Sequential updating of projective and affine structure from motion. *International Journal of Computer Vision*, 23(3):235–259, 1997.

[5] D. C. Brown. The bundle adjustment - progress and prospects. *International Archives of Photogrammetry*, 21(3), 1976.

[6] P. E. Debevec, C. J. Taylor, and J. Malik. Modelling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Computer Graphics (SIGGRAPH'96)*, pages 11–20, 1996.

[6] Note that iteration does not stop after a step where $f(\boldsymbol{\theta})$ *increases*. This simply means that ϵ has not yet adjusted itself optimally.

[7] Y. Dufournaud, C. Schmid, and R. Horaud. Matching images with different resolutions. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'00)*, pages 612–618, 2000.

[8] O. D. Faugeras. *Three Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, Boston, 1993.

[9] O. D. Faugeras and S. J. Maybank. Motion from point matches: multiplicity of solutions. *International Journal of Computer Vision*, 4(3):255–246, 1990.

[10] O. D. Faugeras, L. Robert, S. Laveau, G. Csurka, C. Zeller, C. Gauclin, and I. Zoghlami. 3-D reconstruction of urban scenes from image sequences. *Computer Vision and Image Understanding*, 69(3):292–309, 1998.

[11] A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *European Conference on Computer Vision (ECCV'98)*, pages 311–326, 1998.

[12] D. J. Fleet, A. D. Jepson, and M. R. M. Jenkin. Phase-based disparity measurement. *Computer Vision, Graphics and Image Processing: Image Understanding*, 53(2):198–210, 1991.

[13] W. E. L. Grimson. *From Images to Surface*. MIT Press, 1981.

[14] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 189–192, 1988.

[15] R. I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In *European Conference on Computer Vision (ECCV'92)*, pages 579–587, 1992.

[16] R. I. Hartley. In defence of the 8-point algorithm. In *International Conference on Computer Vision (ICCV'95)*, pages 1064–1070, 1995.

[17] R. I. Hartley and P. Sturm. Triangulation. In *American Image Understanding Workshop*, pages 957–966, 1994.

[18] A. Heyden. Projective structure and motion from image sequences using subspace methods. In *Scandinavian Conference on Image Analysis*, pages 963–968, 1997.

[19] T. S. Huang and A. N. Netravali. Motion and structure from feature correspondences: A review. *Proceedings of IEEE*, 82(2):252–267, 1994.

[20] D. Jacobs. Linear fitting with missing data. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97)*, pages 206–212, 1997.

[21] D. Jones and J. Malik. Computational framework for determining stereo correspondence from a set of linear spatial filters. *Image and Vision Computing*, 10(10):699–708, 1992.

[22] T. Kanade and M. Okutomi. A stereo matching algorithm with an adapative window: Theory and experiment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):920–932, 1994.

[23] K. Kraus. *Photogrammetry, Volumes I and II.* Dümmler, 1997.

[24] E. Kruppa. Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa.*, 122:1939–1948, 1913.

[25] K. N. Kutulakos and J. R. Vallino. Calibration-free augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):1–20, 1998.

[26] J. Lasenby, A. N. Lasenby, C. J. L. Doran, and W. J. Fitzgerald. New geometric methods for computer vision: An application to structure and motion estimation. *International Journal of Computer Vision*, 26(3):191–213, 1997.

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

[28] D. G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV'99)*, pages 1150–1157, 1999.

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

[30] D. W. Marquardt. *Journal of the Society for Industrial and Applied Mathematics*, 11:431–441, 1963.

[31] D. Marr and T. Poggio. A computational theory of human stereo vision. In *Royal Society of London*, volume 204, pages 301–328, 1979.

[32] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *International Conference on Computer Vision (ICCV'01)*, pages 525–531, 2001.

[33] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *European Conference on Computer Vision (ECCV'02)*, pages 128–142, 2002.

[34] H. K. Nishihara. Practical real-time imaging stereo matcher. *Optical Engineering*, 23(5):536–545, 1984.

[35] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer-Verlag, 1999.

[36] E. Pervin and J. A. Webb. Quaternions for computer vision and robotics. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'83)*, pages 382–383, 1983.

[37] C. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. In *European Conference on Computer Vision (ECCV'94)*, pages 97–108, 1994.

[38] S. B. Pollard, J. E. W. Mayhew, and J. P. Frisby. A stereo correspondence algorithm using a disparity gradient limit. *Perception*, 14:449–470, 1985.

[39] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *International Conference on Computer Vision (ICCV'98)*, pages 90–95, 1998.

[40] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool. Hand-held acquisition of 3D models with a video camera. In *International Conference on 3D Digital Imaging and Modeling*, pages 14–23, 1999.

[41] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 1992.

[42] P. Pritchett and A. Zisserman. Wide baseline stereo matching. In *International Conference on Computer Vision (ICCV'98)*, pages 754–760, 1998.

[43] P.Torr, A. Fitzgibbon, and A. Zisserman. The problem of degeneracy in structure and motion recovery from uncalibrated image sequences. *International Journal of Computer Vision*, 32(1):27–44, 1999.

[44] F. Schaffalitzky, A. Zisserman, R. I. Hartley, and P. H. S. Torr. A six point solution for structure and motion. In *European Conference on Computer Vision (ECCV'00)*, pages 632–648, 2000.

[45] C. Schmid, R. Mohr, and C. Bauckhage. Comparing and evaluating interest points. In *International Conference on Computer Vision (ICCV'98)*, pages 230–235, 1998.

[46] P. F. Sturm and W. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *European Conference on Computer Vision (ECCV'96)*, pages 709–720, 1996.

[47] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[48] P. H. S. Torr and A. Zisserman. Robust computation and parameterization of multiple view relations. In *International Conference on Computer Vision (ICCV'98)*, pages 727–732, 1998.

[49] W. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

[50] R. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, 1987.

[51] T. Tuytelaars and L. J. Van Gool. Content-based image retrieval based on local affinely invariant regions. In *International Conference on Visual Information Systems*, pages 493–500, 1999.

[52] D. Weinshall and C. Tomasi. Linear and incremental acquisition of invariant shape models from image sequences. In *International Conference on Computer Vision (ICCV'93)*, pages 675–682, 1993.

[53] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965–980, 1992.

[54] 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, 1995.

[55] A. Zisserman, A. Fitzgibbon, and G. Cross. VHS to VRML: 3D graphical models from video sequences. In *ICMCS*, pages 51–57, 1999.

Chapter 14

Architectural modelling

14.1 Introduction

In this chapter, we describe a practical method for recovering computer models of the geometry and appearance of architectural scenes from photographs. This method has been implemented in the form of a software system called *PhotoBuilder* for creating large-scale architectural models from photographs and maps[1]. Such models are used in a variety of applications including town planning [6], remote measurement [4], and for the creation of photorealistic virtual reality environments [8].

In what follows, we assume that architectural scenes can be decomposed into geometric building blocks called *primitives*. The simplest primitives are points, line segments, and planar polygons. Combined with appropriate geometric constraints, these primitives can be used to describe a range of more complex shapes such as cuboids and prisms.

The starting point will be a set of uncalibrated photographs, typically obtained using an ordinary digital camera. It is also possible to use maps or plans as additional (orthographic) views [12]. We will assume here that corresponding geometric primitives can be identified manually in each view [2, 5, 9, 12, 13, 14, 15]. The main benefit is the possibility of exploiting the user's higher-level knowledge to solve the difficult correspondence problem and to obtain a parsimonious description of the scene's geometry.

Having identified corresponding primitives in the input views, the aim is to recover projection matrices $\mathbf{P}_i \sim \mathbf{K}[\,\mathbf{R}_i \quad \mathbf{T}_i\,]$ and a set of 3D points $\mathbf{X}_j$ (which are the vertices of the line segments and planar polygons). The final stage of the reconstruction algorithm is bundle adjustment, which is used iteratively to refine structure and motion parameters to obtain a maximum likelihood estimate. However, bundle adjustment will only succeed given a sufficiently good starting point; otherwise, the algorithm may fail by converging to a local minimum cost solution. Hence, the key problem is to obtain a suitable initialisation.

[1]See `http://mi.eng.cam.ac.uk/photobuilder/`.

The solution described here works by decoupling the problem of obtaining camera orientations from that of obtaining camera positions and scene geometry (after [2, 5, 13, 14]). First, camera orientations $\mathbf{R}_i$ are recovered independently for each view. Then all camera translations $\mathbf{T}_i$ and 3D points $\mathbf{X}_j$ can be recovered simultaneously by solving a *linear* equation. Using scene constraints such as parallelism and coplanarity, it is also possible to reconstruct points that are visible in fewer than two views. Compared to the standard structure from motion algorithms (Section 13.8), a significant advantage is that fewer corresponding points are required and non-coplanar points are unnecessary. Hence, viewpoints can be selected without such careful planning, and models can be recovered using fewer views and less user intervention.

14.1.1 Assumptions

In what follows, it will be helpful to assume that the camera's (possibly unknown) intrinsic parameters are constant for all views, *i.e.* that all photographs have been taken by the same camera and lens combination. The main benefit is that the number of degrees of freedom associated with the reconstruction problem is significantly reduced so that certain kinds of mathematical degeneracy can be more easily avoided.

Initially, radial distortion will be assumed to be negligible, *i.e.* k_1 and k_2, the coefficients of radial distortion in equations 13.10 and 13.10, will be assumed to be 0. In practice, this assumption is at least sufficiently good to allow the approach to succeed for most real cameras and lenses. In any case, it may be relaxed during the final bundle adjustment stage of the reconstruction process.

14.1.2 Method outline

The complete method is illustrated in Figure 14.1. Starting with an unordered set of one or more uncalibrated views, a 3D model is computed in six stages:

1. Firstly, the user identifies corresponding geometric primitives in each photograph, and, optionally, on a map or plan. The important primitives are vertices (3D points), line segments (which have two vertices), and planar polygons (which have three or more). The user also defines simple scene constraints. Sets of line segments and polygon normals can be defined to be parallel to a common direction, and pairs of directions can be defined to be perpendicular.

2. The next step is to estimate the unknown camera calibration matrix $\mathbf{K}$. An approximate solution is obtained using *vanishing points* in one of the input views. Under the assumption that pixels are square, three vanishing points belonging to a set of orthogonal directions in the scene can be used to determine the focal length and principal point.

3. Given the calibration matrix, camera orientations $\mathbf{R}_i$ are recovered independently for each view using vanishing points belonging to known directions.

4. Given camera orientations, all camera translations $\mathbf{T}_i$ and scene structure $\mathbf{X}_j$ can be recovered simultaneously by solving a *linear* equation. By exploiting scene constraints, it is possible to reconstruct points that are visible in fewer than two views.

5. Next, a *maximum likelihood* estimate of camera intrinsic and extrinsic parameters and 3D point positions is obtained by bundle adjustment.

6. Finally, the recovered projection matrices can be used to *texture map* the model's polygons using the input images.

14.1.3 Chapter outline

The remainder of this chapter is organised as follows. First, Section 14.2 describes a method for computing the camera calibration matrix for a single perspective view using vanishing points belonging to orthogonal directions. Given the camera calibration matrix, Section 14.3 explains how camera orientations can be determined using image plane vanishing points associated with known directions. Next, Section 14.4 explains how all camera positions and scene structure are computed by solving a linear equation and Section 14.5 explains how a maximum likelihood parameter estimate is obtained by bundle adjustment. Finally, Section 14.6 presents some results obtained using our interactive modelling system.

14.2 Obtaining the camera calibration matrix

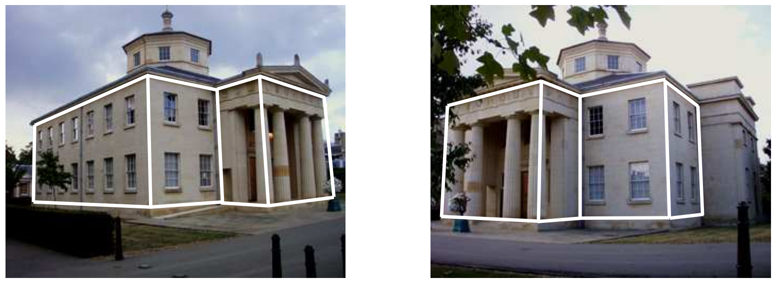
This section describes a simple method for computing the camera calibration matrix $\mathbf{K}$ for a single perspective view of an architectural scene using vanishing points belonging to orthogonal directions.

14.2.1 Vanishing points

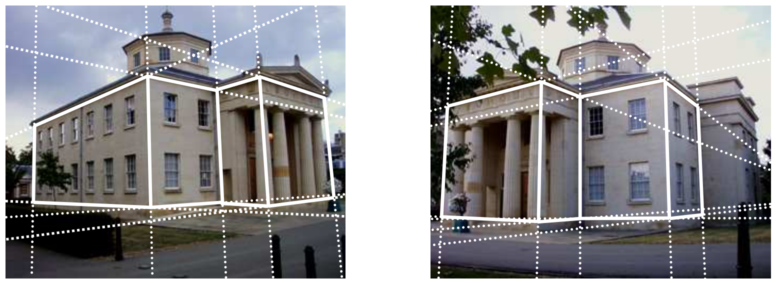
In a perspective view, the images of parallel lines in the world intersect at a *vanishing point* (see Figure 14.2). This can be thought of as the projection of a homogenous point at infinity $[\,\mathbf{d} \quad 0\,]^\top$ where $\mathbf{d}$ is the Euclidean line direction, *i.e.*

$$\tilde{\mathbf{v}} \sim \mathbf{P}[\,\mathbf{d} \quad 0\,]^\top. \tag{14.1}$$

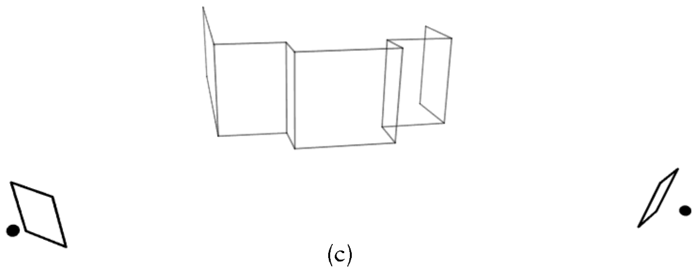
Vanishing points depend only on the first 3×3 sub-matrix of the projection matrix and are therefore independent of camera position. Here, vanishing points will be expressed using normalised homogenous coordinates, *i.e.* $\tilde{\mathbf{v}} \sim [\,\tilde{u} \quad \tilde{v} \quad \tilde{w}\,]^\top$ with $\tilde{\mathbf{v}}$ scaled such that $|\tilde{\mathbf{v}}| = 1$. Methods for computing vanishing points from imaged line segments are described in Appendix A.



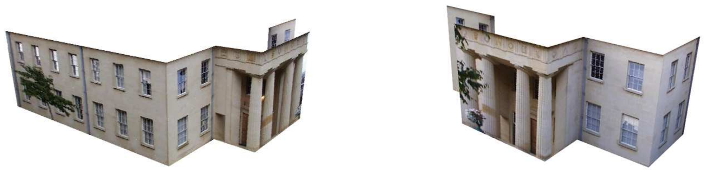
(a)



(b)



(c)



(d)

Figure 14.1: Method outline. (a) The user defines corresponding geometric primitives (points, line segments, and planar polygons) in each view. (b) Camera orientations are obtained using vanishing points belonging to known directions (vanishing points are indicated by dotted lines). (c) Camera positions and scene geometry are recovered as the solution of a linear equation. (d) Recovered projection matrices can be used to texture map the model, which can then be used to synthesise novel views.

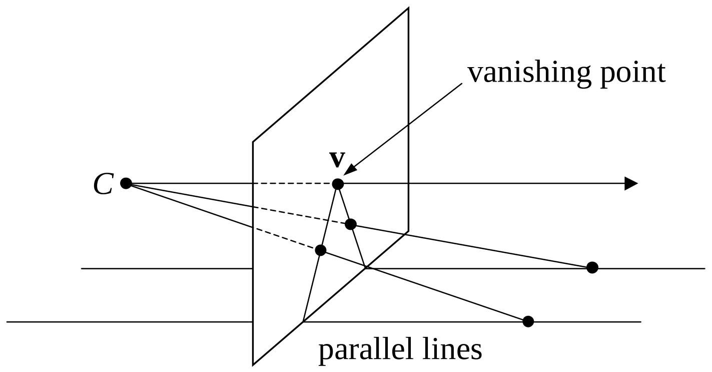


Figure 14.2: Vanishing point illustration for a perspective camera: parallel lines in the world appear to meet at a vanishing point **v** in the image.

14.2.2 Camera intrinsic constraints

In what follows, the world XYZ coordinate system is considered to be have been aligned with a set of orthogonal directions in the scene. Then, using equation 14.1 and by considering the projection of points at infinity corresponding to the three orthogonal directions, it is simple to derive the following constraints on the elements of the projection matrix **P**:

$$\begin{bmatrix} \lambda_1\tilde{u}_1 & \lambda_2\tilde{u}_2 & \lambda_3\tilde{u}_3 \\ \lambda_1\tilde{v}_1 & \lambda_2\tilde{v}_2 & \lambda_3\tilde{v}_3 \\ \lambda_1\tilde{w}_1 & \lambda_2\tilde{w}_2 & \lambda_3\tilde{w}_3 \end{bmatrix} = \mathbf{P} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{14.2}$$

where the λ_i are unknown scale factors that are introduced because we have replaced the equality-up-to-scale relationship in equation 14.1 with simple equality. Then, using (13.8), this equation can be rearranged and expressed in terms of the camera calibration matrix **K** and the camera orientation **R**:

$$\begin{bmatrix} \tilde{u}_1 & \tilde{u}_2 & \tilde{u}_3 \\ \tilde{v}_1 & \tilde{v}_2 & \tilde{v}_3 \\ \tilde{w}_1 & \tilde{w}_2 & \tilde{w}_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} = \mathbf{KR} \tag{14.3}$$

Since **R** is a rotation matrix, $\mathbf{R}^\top\mathbf{R}$ is just the 3×3 identity matrix. Hence, equation 14.3 gives the following constraint on the camera calibration matrix and the unknown scale factors λ_i:

$$\begin{bmatrix} \tilde{u}_1 & \tilde{u}_2 & \tilde{u}_3 \\ \tilde{v}_1 & \tilde{v}_2 & \tilde{v}_3 \\ \tilde{w}_1 & \tilde{w}_2 & \tilde{w}_3 \end{bmatrix} \begin{bmatrix} \lambda_1^2 & 0 & 0 \\ 0 & \lambda_2^2 & 0 \\ 0 & 0 & \lambda_3^2 \end{bmatrix} \begin{bmatrix} \tilde{u}_1 & \tilde{u}_2 & \tilde{u}_3 \\ \tilde{v}_1 & \tilde{v}_2 & \tilde{v}_3 \\ \tilde{w}_1 & \tilde{w}_2 & \tilde{w}_3 \end{bmatrix}^\top = \mathbf{K}\mathbf{K}^\top \tag{14.4}$$

Equation 14.4 shows that orthogonal vanishing points provide a simple constraint on the 3×3 matrix $\mathbf{KK}^\top$ (which is sometimes called the *dual image of the absolute conic*) [10].

14.2.3 Geometric interpretation

Under the assumption of known aspect ratio and zero skew, equation 14.3 can be rewritten as:

$$\begin{bmatrix} \lambda_1\tilde{u}_1 & \lambda_2\tilde{u}_2 & \lambda_3\tilde{u}_3 \\ \lambda_1\tilde{v}_1 & \lambda_2\tilde{v}_2 & \lambda_3\tilde{v}_3 \\ \lambda_1\tilde{w}_1 & \lambda_2\tilde{w}_2 & \lambda_3\tilde{w}_3 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{R} \tag{14.5}$$

Hence,

$$\mathbf{R} = \begin{bmatrix} \lambda_1(\tilde{u}_1 - \tilde{w}_1 u_0)/\alpha & \lambda_2(\tilde{u}_2 - \tilde{w}_2 u_0)/\alpha & \lambda_3(\tilde{u}_3 - \tilde{w}_3 u_0)/\alpha \\ \lambda_1(\tilde{v}_1 - \tilde{w}_1 v_0)/\alpha & \lambda_2(\tilde{v}_2 - \tilde{w}_2 v_0)/\alpha & \lambda_3(\tilde{v}_3 - \tilde{w}_3 v_0)/\alpha \\ \lambda_1\tilde{w}_1 & \lambda_2\tilde{w}_2 & \lambda_3\tilde{w}_3 \end{bmatrix} \tag{14.6}$$

The orthonormality of $\mathbf{R}$ can be used to recover the principal point $\mathbf{u}_0 = [\, u_0 \quad v_0 \,]^\top$ and focal length α. Writing the shorthand $\hat{\mathbf{v}}_i = [\, \tilde{u}_i \quad \tilde{v}_i \,]^\top$ and using the orthogonality of the first two columns of $\mathbf{R}$ gives:

$$\lambda_1\lambda_2 \left[(\hat{\mathbf{v}}_1 - \tilde{w}_1\mathbf{u}_0)^\top (\hat{\mathbf{v}}_2 - \tilde{w}_2\mathbf{u}_0)/\alpha^2 + \tilde{w}_1\tilde{w}_2 \right] = 0 \tag{14.7}$$

Since $\lambda_i \neq 0$, equation 14.7 can be rewritten:

$$(\hat{\mathbf{v}}_1/\tilde{w}_1 - \mathbf{u}_0)^\top (\hat{\mathbf{v}}_2/\tilde{w}_2 - \mathbf{u}_0) + \alpha^2 = 0 \tag{14.8}$$

and considering the other column pairs of $\mathbf{R}$ similarly gives:

$$(\hat{\mathbf{v}}_2/\tilde{w}_2 - \mathbf{u}_0)^\top (\hat{\mathbf{v}}_3/\tilde{w}_3 - \mathbf{u}_0) + \alpha^2 = 0 \tag{14.9}$$

$$(\hat{\mathbf{v}}_1/\tilde{w}_1 - \mathbf{u}_0)^\top (\hat{\mathbf{v}}_3/\tilde{w}_3 - \mathbf{u}_0) + \alpha^2 = 0 \tag{14.10}$$

Subtracting 14.10 from 14.8 gives:

$$(\hat{\mathbf{v}}_1/\tilde{w}_1 - \mathbf{u}_0)^\top (\hat{\mathbf{v}}_2/\tilde{w}_2 - \mathbf{u}_0) - (\hat{\mathbf{v}}_1/\tilde{w}_1 - \mathbf{u}_0)^\top (\hat{\mathbf{v}}_3/\tilde{w}_3 - \mathbf{u}_0) = 0 \tag{14.11}$$

Hence,

$$(\hat{\mathbf{v}}_1/\tilde{w}_1 - \mathbf{u}_0)^\top (\hat{\mathbf{v}}_2/\tilde{w}_2 - \hat{\mathbf{v}}_3/\tilde{w}_3) = 0 \tag{14.12}$$

Under the assumption that the vanishing points $\tilde{\mathbf{v}}_i$ are not at infinity, $\tilde{w}_i \neq 0$ and $\hat{\mathbf{v}}_i/\tilde{w}_i = [\, \tilde{u}_i/\tilde{w}_i \quad \tilde{v}_i/\tilde{w}_i \,]^\top$, which is just the Euclidean coordinate $\mathbf{v}_i$. Thus, equation 14.12 expresses the condition that $\mathbf{v}_1 - \mathbf{u}_0$ is orthogonal to $\mathbf{v}_2 - \mathbf{v}_3$ (Figure 14.3). The other two orthogonality conditions can be derived similarly and imply that the principal point $\mathbf{u}_0$ is the orthocentre of the triangle $\triangle\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3$. Using this value for $\mathbf{u}_0$, equation 14.8, 14.9, or 14.10 could be used to compute α^2 (all give the same result). This is the result derived by Caprile and Torre [3].

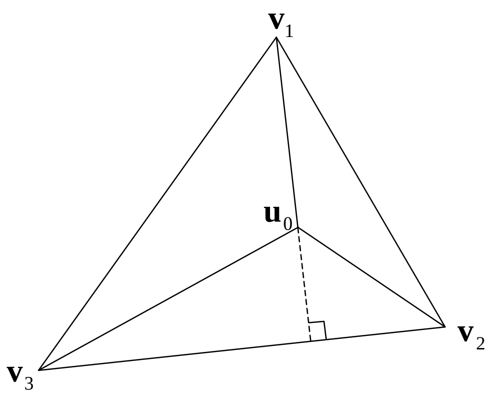


Figure 14.3: Interpretation of $\mathbf{u}_0$ as the orthocentre of the triangle $\triangle \mathbf{v}_1\mathbf{v}_2\mathbf{v}_3$ defined by the three vanishing points.

14.2.4 Degenerate configurations

Considering vanishing points belonging to three orthogonal directions, it is possible that one or two vanishing points will be at infinity, *i.e.* with $w_i = 0$. Then the method described above is degenerate:

- If a single vanishing point is at infinity, it will be impossible to solve the orthocentre constraints (equation 14.12 *etc.*) for $\mathbf{u}_0$ because of the terms in $1/\tilde{w}_i$. However, if $\mathbf{u}_0$ is known, then it will still be possible to solve one of equations 14.8, 14.9, and 14.10 for α^2. *E.g.* if $\tilde{w}_1 = 0$ then equation 14.9 (which is independent of $\tilde{w}_1$) can still be solved for α^2.
- If two vanishing points are at infinity, it will be impossible to solve equations 14.8, 14.9, and 14.10 for α. However, it is simple to show that the remaining vanishing point corresponds to the principal point.

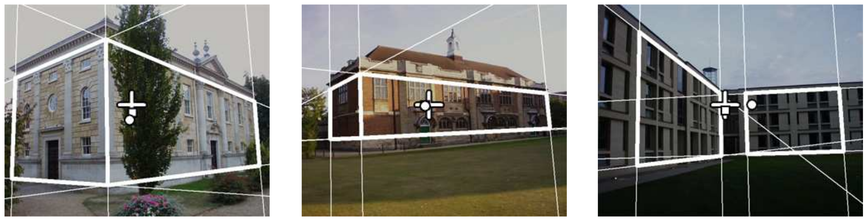
Since the method described in Section 14.2.3 is degenerate for vanishing points at infinity, a practical concern is that it will be sensitive to noise for vanishing points that are near infinity, *i.e.* with $\tilde{w}_i \approx 0$. For this reason, it will usually be preferable to use an approximate rather than a calculated value for the principal point (typically obtained by setting it to the image centre[2]). Then equations 14.8, 14.9, and 14.10 can be solved for α^2 even if a single vanishing point is at infinity. They are first rearranged as follows:

$$(\hat{\mathbf{v}}_1 - \tilde{w}_1\mathbf{u}_0)^\top(\hat{\mathbf{v}}_2 - \tilde{w}_2\mathbf{u}_0) + \tilde{w}_1\tilde{w}_2\alpha^2 = 0 \tag{14.13}$$

$$(\hat{\mathbf{v}}_2 - \tilde{w}_2\mathbf{u}_0)^\top(\hat{\mathbf{v}}_3 - \tilde{w}_3\mathbf{u}_0) + \tilde{w}_2\tilde{w}_3\alpha^2 = 0 \tag{14.14}$$

$$(\hat{\mathbf{v}}_1 - \tilde{w}_1\mathbf{u}_0)^\top(\hat{\mathbf{v}}_3 - \tilde{w}_3\mathbf{u}_0) + \tilde{w}_1\tilde{w}_3\alpha^2 = 0 \tag{14.15}$$

[2]Since real lenses give the best optical performance near the optical axis, the imaging arrays of digital camera are usually centred on the optical axis.



$\alpha = 1324(1382)$ (a) $\alpha = 1276(1369)$ (b) $\alpha = 1482(1373)$ (c)

Figure 14.4: Computing the camera intrinsic parameters using vanishing points. A few parallel line segments were defined manually in each view and camera intrinsic parameters α and $\mathbf{u}_0$ (shown as a circle) were recovered using vanishing points belonging to orthogonal directions. The experiment was repeated after correcting the image for radial distortion (α in brackets, $\mathbf{u}_0$ shown as a square). Ground truth $\alpha \approx 1400$, and ground truth $\mathbf{u}_0$ is shown as a cross. The estimated values are in good agreement.

Given $\mathbf{u}_0$, these equations can be solved for α^2 using standard linear least squares[3].

14.2.5 Examples

Figure 14.4 shows three photographs obtained using an ordinary digital camera. A few parallel line segments were identified manually in each image and vanishing points were computed using a maximum likelihood method, which is described in Appendix A, below. Ground truth camera intrinsic parameters were obtained separately using the chessboard calibration method of Zhang [17].

Camera intrinsic parameters α and $\mathbf{u}_0$ were estimated using vanishing points belonging to the three orthogonal directions in the scene. Figure 14.4 shows the results. Recovered focal lengths agree with ground truth to within $\pm 10\%$ and the principal point to within 150 pixels (image dimensions were 1280×1024). These results are representative of those obtained for a wide range of images.

An interesting question concerns the extent to which ignoring the effects of radial distortion reduces accuracy. Figure 14.4 also shows the results of repeating the experiment using images that had been corrected in advance for radial distortion using ground truth distortion parameters. Now recovered focal lengths agree with ground truth to within about $\pm$ 2% and the principal points to within 30 pixels, approximately a five-fold improvement in each case.

[3]It is easy to show that these equations might give a negative value for α^2 if the supplied principal point lies outside the triangle defined by the three orthogonal vanishing points. In this case, the method will fail. However, this is unlikely in practice unless two of the three vanishing points are near infinity – and this condition is easy to detect.

14.3 Finding camera orientations

Given the camera calibration matrix, camera orientations can be determined independently for each view using vanishing points belonging to known directions in the world coordinate system. Let $\mathbf{v}_i$ be the vanishing point associated with a known direction $\mathbf{d}_i$ (which is a unit vector). Then, using equations 14.1 and 13.8, it is simple to derive the following constraint on the camera orientation matrix $\mathbf{R}$:

$$\mathbf{K}^{-1}\tilde{\mathbf{v}}_i \sim \mathbf{R}\mathbf{d}_i. \tag{14.16}$$

Given two or more such constraints, it is possible to recover $\mathbf{R}$. Let $\mathbf{a}_i \sim \mathbf{K}^{-1}\tilde{\mathbf{v}}_i$, where $\mathbf{a}_i$ is scaled such that $|\mathbf{a}_i| = 1$. Then the required solution is obtained by minimising the following sum of squares cost function:

$$\mathbf{R} = \arg\min_{\mathbf{R}} \sum_i ||\mathbf{a}_i - \mathbf{R}\mathbf{d}_i||^2. \tag{14.17}$$

This is the standard *absolute orientation* problem and there are several well-known solutions. An efficient and stable algorithm (due to Arun et al. [1]) uses the singular value decomposition. Writing $\mathbf{A} = [\,\mathbf{a}_1 \quad \mathbf{a}_2 \quad \ldots\,]$ and $\mathbf{B} = [\,\mathbf{b}_1 \quad \mathbf{b}_2 \quad \ldots\,]$, it can be shown that the least squares rotation matrix is

$$\mathbf{R} = \mathbf{V}\mathbf{U}^\top \tag{14.18}$$

where $\mathbf{U}$ and $\mathbf{V}$ are the matrices obtained from the singular value decomposition [11] of $\mathbf{B}\mathbf{A}^\top$:

$$\mathbf{B}\mathbf{A}^\top = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\top. \tag{14.19}$$

Here $\mathbf{\Lambda}$ is the diagonal matrix whose elements are the singular values, and $\mathbf{U}$ and $\mathbf{V}$ are 3×3 unitary matrices. However, some precautions must be taken when the points are nearly coplanar, since in these cases, the above algorithm can lead to a reflection transformation instead of a rotation. In such situations, the least square rotation matrix is simply obtained by changing the sign of the column of $\mathbf{V}$ that corresponds to the singular value that is close to zero (if there is no such singular value, which is very unlikely, the algorithm fails).

Often [2, 5, 10, 13, 15], directions $\mathbf{d}_i$ are fixed by assuming that the world coordinate system is aligned with a dominant set of orthogonal directions in the scene. However, this method is not applicable to scenes containing buildings with differing orientations. More generally, the vertical direction can be set to $[\,0 \quad 0 \quad 1\,]^\top$ and important horizontal directions (of the form $[\,X \quad Y \quad 0\,]^\top$) can be found using a map or plan (Robertson and Cipolla [12]). Horizontal directions can be estimated from one or more line segments marked on a map (Figure 14.5). This approach allows a large number of views to be registered in a meaningful *global* coordinate system rather than a local one associated with a particular building, and considerably facilitates building larger-scale models.

Note that there is a sign ambiguity associated with each $\mathbf{a}_i$, which arises because the associated vanishing point might belong to the forwards or backwards direction $+\mathbf{d}_i$ or $-\mathbf{d}_i$. In an interactive system, this ambiguity could be resolved by asking the user to specify a sign for each vanishing point or the map positions of two or more vertical lines in each view.

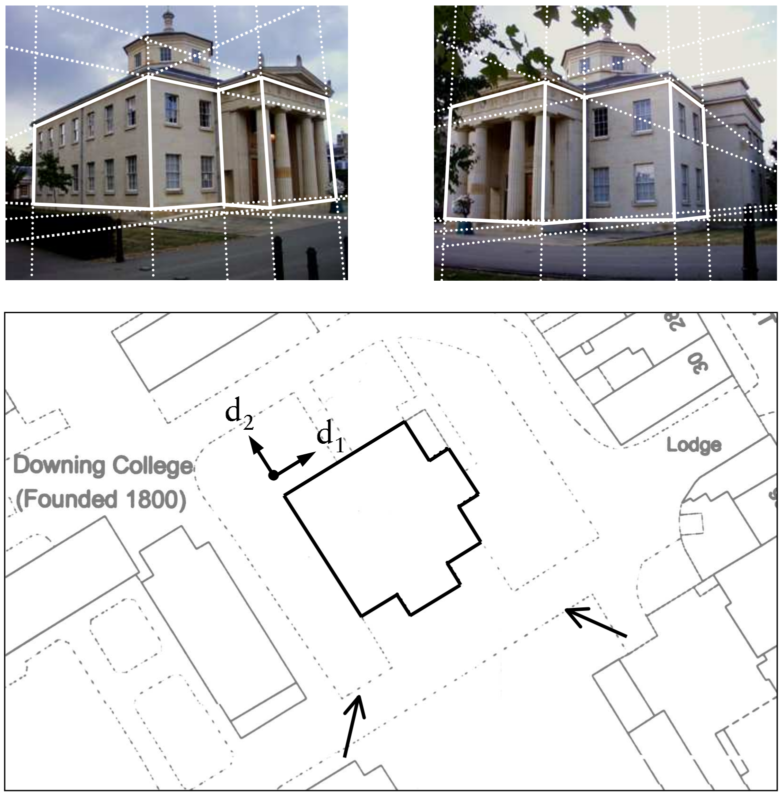


Figure 14.5: Computing camera orientations using a map. Vanishing points in the photographs (indicated using dotted lines) correspond to vertical and horizontal directions in the scene. The vertical direction is set to $[\,0 \quad 0 \quad 1\,]^\top$ and important horizontal directions $\mathbf{d}_1$ and $\mathbf{d}_2$ (of the form $[\,X \quad Y \quad 0\,]^\top$) are estimated using one or more parallel line segments on the map. Hence camera orientations (indicated by arrows) can be determined in a meaningful global coordinate system (camera positions are found later).

14.4 Computing 3D points and camera positions

The previous sections explain how vanishing points belonging to known directions can be used to recover camera orientations independently for each view, either in a local coordinate system aligned with the dominant orthogonal directions in the scene or in a global coordinate system defined by a map. The next stage is to recover a complete geometric model using corresponding primitives in *all* views. Given the camera calibration matrix $\mathbf{K}$, camera orientations $\mathbf{R}_i$, and directions $\mathbf{d}_i$, unknown camera translations $\mathbf{T}_i$ and 3D points $\mathbf{X}_j$ can be obtained by solving a linear system comprising all viewing and scene constraints (Shum et al. [13]).

14.4.1 Viewing and scene constraints

Image measurements provide a constraint on 3D points $\mathbf{X}_j$ and camera translations $\mathbf{T}_i$. Let $\tilde{\mathbf{x}}$ be the image point associated with pixel position $\tilde{\mathbf{u}}$ (*i.e.* $\tilde{\mathbf{x}} \sim \mathbf{K}^{-1}\tilde{\mathbf{u}}$) with $\tilde{\mathbf{x}}$ scaled such that $|\tilde{\mathbf{x}}| = 1$. Then, from (13.8) it is simple to derive the following linear constraint on the Euclidean point $\mathbf{X} = [\, X \quad Y \quad Z \,]^\top$ and camera translation $\mathbf{T}$:

$$[\tilde{\mathbf{x}}]_\times (\mathbf{R}\mathbf{X} + \mathbf{T}) = \mathbf{0} \tag{14.20}$$

where the subscript $\times$ denotes the cross product matrix. Equation 14.20 expresses the constraint that the vector $\mathbf{R}\mathbf{X} + \mathbf{T}$ is parallel to the direction $\tilde{\mathbf{x}}$. Its 3 rows provide only a rank 2 constraint (since any row can be expressed as a linear combination of the other two).

Other sources of information also provide constraints on 3D points. For example, heights (Z coordinates) or 3D positions (XYZ coordinates) can sometimes be fixed or measured directly. Another useful idea is to incorporate measurements made using maps or plans (XY coordinates). Large-scale map data showing individual building footprints is readily available for most urban areas.

Finally, it is also possible to exploit various kinds of scene constraint, such as parallelism and coplanarity. An important benefit is the possibility of reconstructing points that are visible in fewer than two views. For example, the constraint that a line segment has a known direction $\mathbf{d}$ can be written as:

$$[\mathbf{d}]_\times (\mathbf{X}_1 - \mathbf{X}_2) = \mathbf{0} \tag{14.21}$$

where $\mathbf{X}_1$ and $\mathbf{X}_2$ are the endpoints of the line segment. Similarly, coplanarity can be expressed using the plane equation:

$$\mathbf{d}^\top \mathbf{X} - d = 0 \tag{14.22}$$

where X_j is a set of coplanar points belonging to the plane with normal $\mathbf{d}$ and perpendicular distance from the origin d.

Table 14.1 lists the viewing and scene constraints used in our system. Note that these are *linear* in unknown $\mathbf{X}_j$, camera translations $\mathbf{T}_i$, and plane distances

Constraint	**Equation**
Image point $\tilde{\mathbf{x}} \sim \mathbf{K}^{-1}[\,\bar{\mathbf{u}} \quad 1\,]^\top$	$[\tilde{\mathbf{x}}]_\times(\mathbf{R}\mathbf{X} + \mathbf{T}) = \mathbf{0}$
Known 3D point $\bar{\mathbf{X}}$	$\mathbf{X} = \bar{\mathbf{X}}$
Point $[\,\bar{X} \quad \bar{Y}\,]^\top$ on map	$[\,X \quad Y\,]^\top = [\,\bar{X} \quad \bar{Y}\,]^\top$
Line length	$(\mathbf{X}_1 - \mathbf{X}_2) = \pm l\mathbf{d}$
Parallelism	$[\mathbf{d}]_\times(\mathbf{X}_1 - \mathbf{X}_2) = \mathbf{0}$
Coplanarity	$\mathbf{d}^\top\mathbf{X} - d = 0$

Table 14.1: Linear constraints on Euclidean 3D points $\mathbf{X}_j$, camera translations $\mathbf{T}_i$, and plane distances d_l. Camera calibration matrices $\mathbf{K}_i$, orientations $\mathbf{R}_i$, and directions $\mathbf{d}_k$ are known.

d_l. Hence, all such constraints may be assembled into a matrix equation of the form:

$$\mathbf{L}\boldsymbol{\phi} = \mathbf{b} \tag{14.23}$$

where $\boldsymbol{\phi}$ comprises all unknown camera translations $\mathbf{T}_i$, Euclidean 3D points $\mathbf{X}_j$, and plane distances d_l.

14.4.2 Solving the linear system

In general, the linear system in equation 14.23 will consist of one or more *connected components*, *i.e.* independent subsystems that could be solved independently [13]. For a connected component to be solvable:

- the number of constraints should be no fewer than the number of unknowns; and
- sufficient ground truth data should be be provided to fix the origin and scale (via the equations in Table 14.1 that have non-zero right hand sides).

Ground truth data is provided in the form or known lengths, heights (Z coordinates), map measurements (X, Y coordinates), or 3D points (X, Y, Z coordinates). A benefit of using map measurements is that far apart buildings can be reconstructed with the correct geometric relationship even in the absence of intermediate views. The only extra information required is the (Z-axis) height of a single point on each building.

Equation 14.23 can be solved using the singular value decomposition. Let $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^\top$ where $\boldsymbol{\Lambda} = \mathrm{diag}(\sigma_1, \sigma_2, \ldots)$ is the matrix of singular values and $\mathbf{U}$ and $\mathbf{V}$ are orthonormal. Then the solution for $\boldsymbol{\phi}$ is given by:

$$\boldsymbol{\phi} = \mathbf{V}\bar{\boldsymbol{\Lambda}}\mathbf{U}^\top\mathbf{b} \tag{14.24}$$

where $\bar{\boldsymbol{\Lambda}} = \mathrm{diag}(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \ldots)$ [11]. In the event that one or more of the singular values σ_i is at or near 0, then the solution is poorly constrained by the available data and the corresponding elements of $\bar{\boldsymbol{\Lambda}}$ should be replaced by 0. Also, the elements of the corresponding columns of $\mathbf{V}$ that are not at or near zero indicate which elements of $\boldsymbol{\phi}$ are underdetermined.

14.4.3 Iterative solution

By exploiting parallelism and coplanarity constraints (Table 14.1), it is possible to reconstruct points that are visible in fewer than two views. However, line directions and plane normals ($\mathbf{d}_l$) must be obtained in advance. Where the world coordinate system has been aligned with a dominant set of orthogonal directions in the scene, it is likely that many line directions and plane normals will be aligned with the coordinate axes. More generally, important horizontal directions can be estimated using lines on a map. But some directions might still be unknown, *e.g.* the normals to pitched roofs. Consequently, some 3D points might be underdetermined by the available constraints.

This problem can be solved using a simple iterative reconstruction scheme. Given a partial reconstruction (comprising some known and some unknown directions and 3D points), extra directions can be estimated using already-reconstructed points (line directions can be estimated linearly from one or more pairs of 3D points, and plane normals from three or more coplanar 3D points). Then a more complete reconstruction is obtained using all known directions. This process is repeated until no more directions or 3D points can be estimated.

14.5 Bundle adjustment

Camera and structure parameters computed by the method described in the preceding sections will not be optimal in the maximum likelihood sense. Hence, the final stage of the reconstruction process is bundle adjustment – using the Gauss-Newton framework described in the previous chapter. Having obtained an initial estimate for all camera parameters ($\mathbf{K}$, $\mathbf{R}_i$, $\mathbf{T}_i$), structure ($\mathbf{X}_j$), directions ($\mathbf{d}_k$), and plane distances (d_l), all parameters are refined in such a way as to minimize the sum of squared measurement errors. This gives the maximum likelihood solution under the assumption of Gaussian-distributed measurement noise. At this stage it is also possible to refine radial distortion coefficients k_{1i} and k_{2i} if desired.

Here, the set of observations includes the measured image (and possibly map) positions of the 3D points that form the vertices of the model. In addition, line direction constraints and planarity constraints can also be modelled as observations. For example, the distance $\mathbf{d}^\top\mathbf{X} - d$ of a point ($\mathbf{X}$) from an associated plane (with normal $\mathbf{d}$ and distance d) can be considered a measured quantity with a predicted value of 0. Alternatively, constraints can be enforced precisely using the method of *sequential quadratic programming* (Triggs et al. [16]).

An interesting question concerns how to choose prediction weights for the various measurements. To reiterate, prediction weights reflect the associated uncertainties. For maximum likelihood parameter estimation, the weight matrices $\mathbf{W}_i$ belonging to each observation $\bar{\mathbf{z}}_i$ should be set equal to the inverse covariance matrix of the associated measurement noise. For image measurements, the main source of error is likely to be the inability of the user to position corresponding points precisely in images with finite resolution and so standard deviations are usually expressed in pixels. A useful idea is to allow the user to choose from a range of possible certainty levels: suggested standard deviations are 1, 2, 4, 8, and 16 pixels. For map measurements, the most significant source of error will usually be the limited accuracy of the map. An important reason for inaccuracy is that mapmakers sometimes approximate complex geometry with simple geometry. Thus, standard deviations are approximated using the quoted accuracy of the map. Parallelism and coplanarity constraints must also be given appropriate prediction weights. These constraints will be enforced increasingly strongly as the weights are increased – and it is useful to allow the user to adjust the weights to suit the type of scene. To choose a suitable range of variation it will be helpful to give the reconstruction a scale, *e.g.* by using a map coordinate system, or specifying a known length. Then measurement uncertainties can be expressed in physically meaningful units, *e.g.* the standard deviation associated with the prediction ($\mathbf{d}^\top \mathbf{X} - d = 0$) that a 3D point $\mathbf{X}$ lies on a plane ($\mathbf{d}$, d) can be expressed in metres.

After bundle adjustment, two kinds of feedback can be given to the user. Firstly, inconsistent or outlying observations can be identified because they have comparatively higher cost, *e.g.* by plotting exaggerated reprojection errors for image and map measurements. Secondly, the uncertainty associated with estimated parameter values can be determined by computing the covariance matrix for the parameter vector (which is just the inverse of the Hessian $\mathbf{H}^{-1}$ [16]).

14.6 Implementation and testing

The method described in this chapter forms the basis of a working system called *PhotoBuilder* for creating large-scale architectural models from photographs and maps[4]. This software has been implemented using Visual C++ in the form of a Microsoft Windows application, which provides the facility to export models using the *Virtual Reality Modeling* [sic] *Language* (*VRML*)[5], which is designed to allow users to explore virtual reality environments via the Internet. This section presents some illustrative results obtained using this software.

14.6.1 Downing College library

Figure 14.6 shows nine photographs of Downing College Library, which were obtained using an ordinary digital camera attached to a tripod. The camera had a medium-wide angle lens and a pixel resolution of 1280×1024. Approximate ground truth camera positions were obtained by taking measurements between the tripod head and the corners of nearby buildings with a tape measure. These measurements were transferred to a 1:1000 scale map using a compass (see Figure 14.7a).

[4]See `http://mi.eng.cam.ac.uk/photobuilder/`.

[5]See `http://www.web3d.org/x3d/spec/vrml/ISO_IEC_14772-All/` for the current ISO specifications.

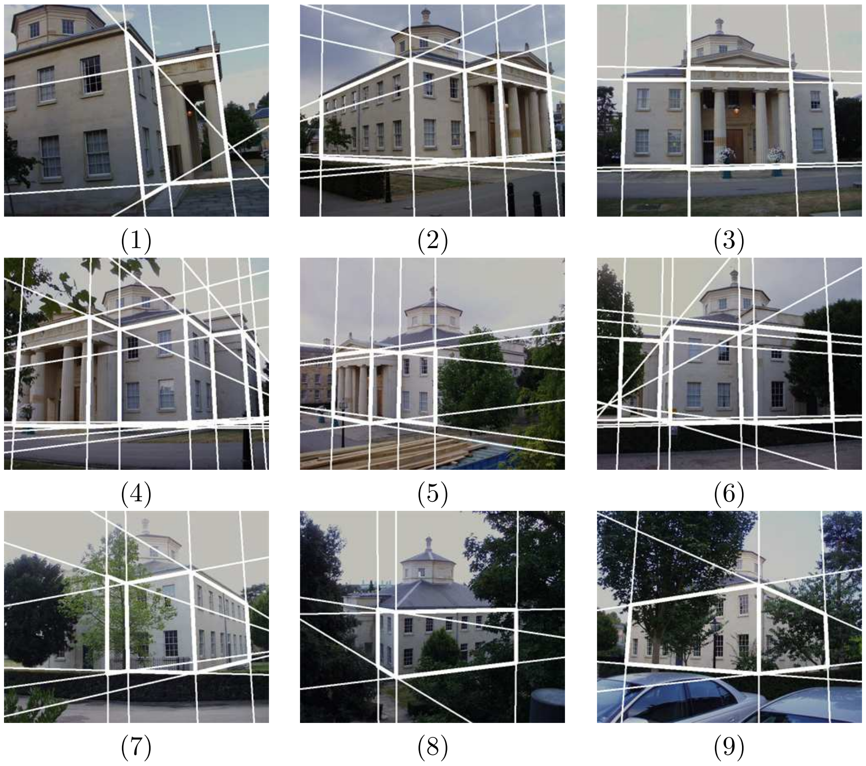
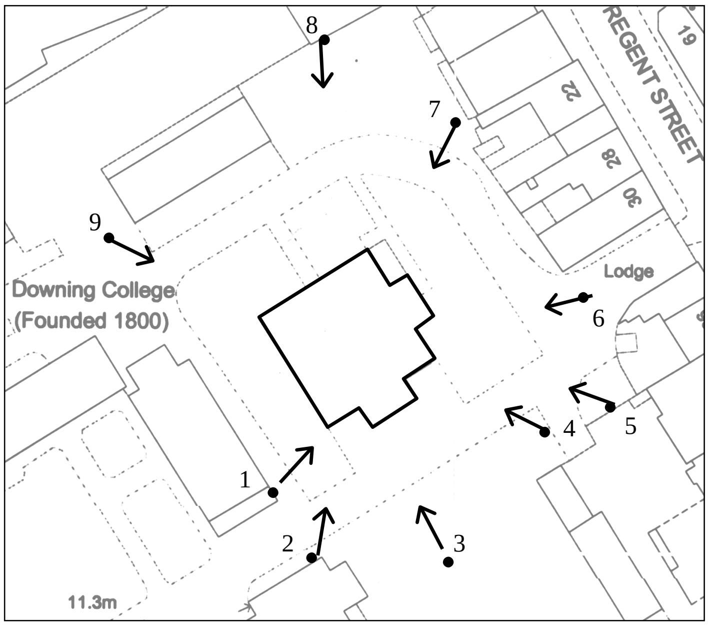


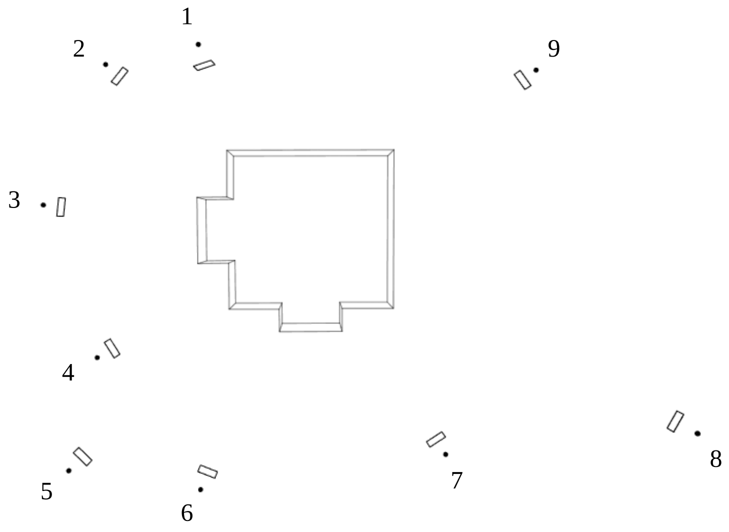
Figure 14.6: Nine photographs of the library at Downing College, Cambridge. Corresponding geometric primitives were defined interactively (thick lines) and vanishing points were computed for parallel line segments (thin lines).

A small number of corresponding geometric primitives (line segments and rectangular plane facets) were identified interactively in each photograph and on the map. Approximate camera intrinsic values were obtained using the vanishing point calibration method described in Section 14.2 for the first view (the principal point was set to the image centre). Then camera orientations were determined in the map coordinate system using vanishing points belonging to the vertical direction and two horizontal directions, which were estimated using line segments on the map. Finally, 3D points were computed using the linear method described in Section 14.4. So that equation 14.23 would be non-singular, it was necessary to fix the coordinate system for the reconstruction by specifying arbitrarily the height of a single 3D point.

Figure 14.7 shows reconstruction results before bundle adjustment. The recovered geometry is visibly accurate (the aspect ratio of the square building is 1.0). The quality of the reconstructions can also be assessed in terms of reprojection error, *i.e.* how well the model agrees with the image data. RMS reprojection error was 5 pixels (an error in viewing direction of 0.2°).



(a)



(b)

Figure 14.7: Linear reconstruction results for the Downing College library photographs. (a) Recovered camera positions and orientations are represented by arrows, ground truth camera positions are shown as dots. (b) Recovered geometry is visually accurate (the real building is also square). The map is reproduced at approximately 1:1000 scale.

Next, a more complete model was obtained by identifying more geometric primitives. Again, the linear reconstruction algorithm was used to obtain an initial estimate of camera positions and structure. Then all structure and motion parameters (including camera intrinsic parameters) were refined using bundle adjustment. The standard deviations associated with image and map measurements were set to 2 pixels and 0.25 m respectively; the standard deviations associated with parallelism and coplanarity constraints (which were treated as measurements) were set to 0.05 m.

Figure 14.8 show some novel views of the resulting model. The wire frame views show that recovered geometry is visually accurate: nominally parallel lines are close to parallel and nominally coplanar points are close to coplanar. After bundle adjustment, RMS reprojection error was 1 pixel. Given the wide baseline relating the images and the fact that a number of vertices were occluded by trees in this data set, this error is no greater than that likely to have been introduced by defining point correspondences manually, *i.e.* a credible optimum has been obtained. The focal length and principal point estimated by bundle adjustment were accurate to within 1

14.6.2 Building large models using map information

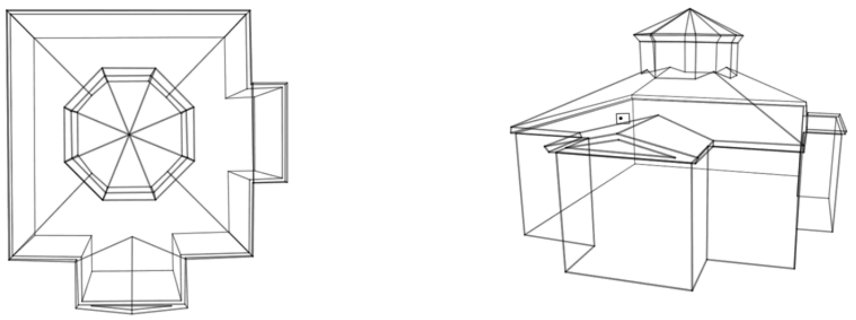
By combining map and image data, it is simple to model scenes containing a large number of buildings, possibly with differing orientations and non-rectangular plan. This section presents illustrative results.

Trumpington Street model

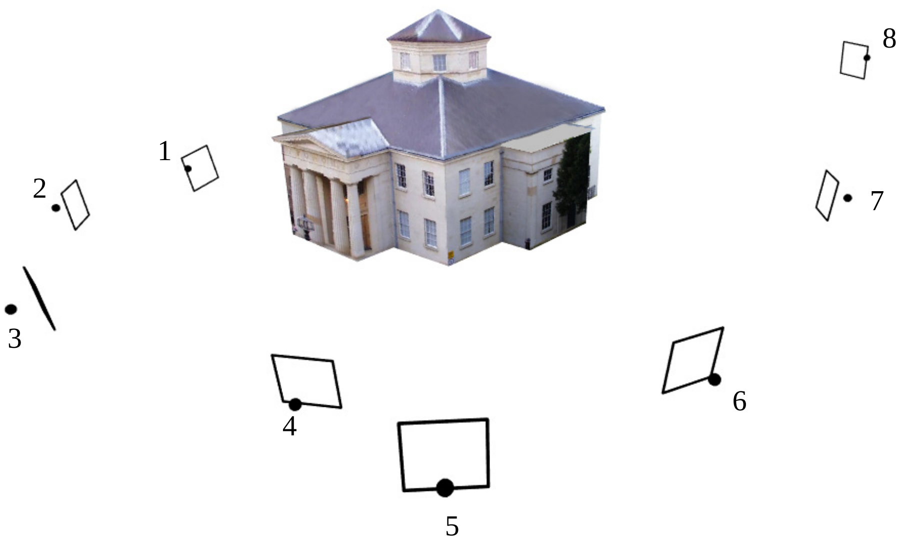
Figure 14.9 shows the first three views out of a sequence of 22 obtained using an ordinary digital camera in a section of Trumpington Street in Cambridge. This street is curved, and most of the buildings have different orientations. Viewpoints were chosen somewhat arbitrarily: sometimes the camera was simply rotated from one viewpoint to the next and some adjacent views exhibit very little overlap.

The reconstruction shown in Figure 14.10 was obtained using the method described in this chapter. First an approximate value for focal length was estimated using vanishing points in the first view belonging to orthogonal directions (the principal point was set to the image centre and radial distortion was assumed initially to be negligible). Then camera orientations were estimated using vanishing points belonging to vertical and horizontal directions (horizontal directions were estimated using parallel line segments on the map). Finally, bundle adjustment was used to refine all structure and motion parameters. The recovered focal length and principal point were accurate to within 1.7% and 10 pixels of ground-truth values obtained by laboratory calibration.

Since only approximate values were used for the camera intrinsic parameters, an interesting question concerns how sensitive the reconstruction algorithm is to inaccuracy. Since bundle adjustment is used as the final stage, recovered structure and motion parameters should be unaffected by small variations in



(a)



(b)

Figure 14.8: Reconstruction results. By defining corresponding primitives interactively in each view, a more complete model has been obtained. (a) The recovered geometry is visually accurate: the wire frame views show that nominally parallel lines are parallel. (b) Recovered camera pose and a texture-mapped model. Recovered camera positions may be related to the images in Figure 14.6.

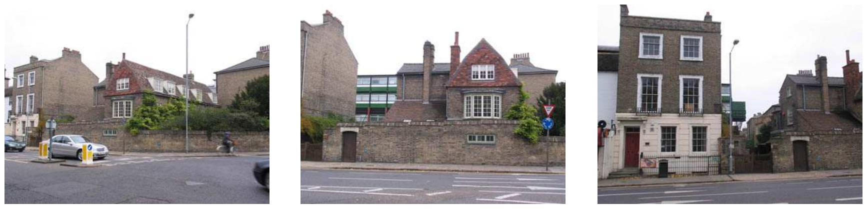


Figure 14.9: The first three views from a sequence of 22 that was obtained in Trumpington Street, Cambridge.

the starting conditions. For this dataset, the same results were obtained (after bundle adjustment) despite focal length errors in the approximate range -50% to +200%.

Jesus College virtual tour

The second example is a model of Jesus College in Cambridge. This model was created as part of a 'virtual tour' of the college, which is available online[6]. The tour allows the user to navigate a large 3D model of the college via the Internet.

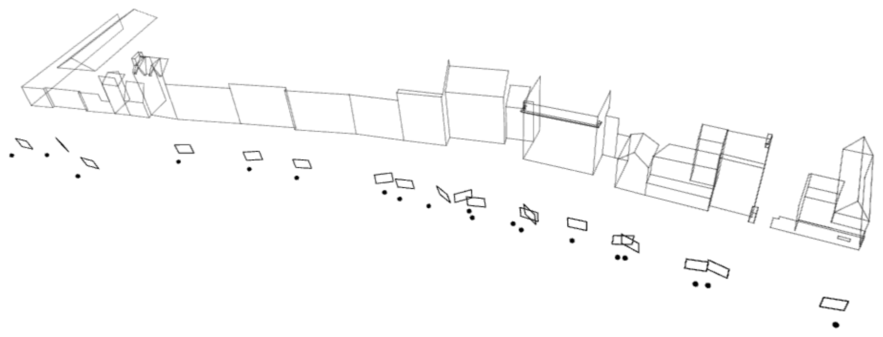
Figure 14.11 shows some screen shots of the model, which was reconstructed using around 50 photographs. The buildings in Jesus have differing orientations and so a plan of the college was used for to determine camera orientations. The trees in the model were segmented manually in the images and are represented by 'cardboard cut-outs', *i.e.* vertically oriented planes with transparency.

One practical difficulty here was the requirement that the model should be compact enough for efficient network transfer. In part, this has been achieved by using repeating synthetic textures for relatively uninteresting areas, such as the grass and pathway. It has also been necessary to subdivide the large model into several smaller parts (individual courtyards) that can be downloaded separately.

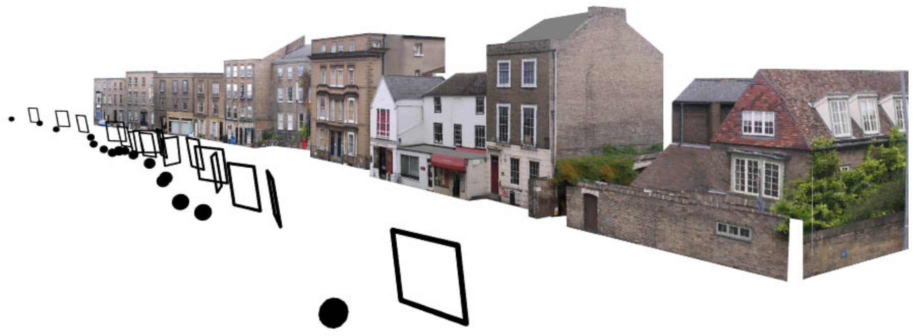
14.7 Summary

This chapter has described an interactive system (*PhotoBuilder*) for recovering models of the geometry and appearance of architectural scenes from photographs. The central problem is the recovery of camera projection matrices and scene structure. A key insight is that it is possible to solve this problem in two stages. First camera calibration matrices and orientations are obtained using single-view constraints. Then all camera positions and scene structure can be determined simultaneously by solving a linear equation. Compared to systems that use standard sequential structure from motion algorithms, an advantage is that fewer corresponding points are required and non-coplanar points are unnecessary. Hence, viewpoints can be selected without such careful plan-

[6]See `http://www.jesus.cam.ac.uk/virtualtour/entrancepage/entrance.htm`



(a)



(b)

Figure 14.10: Trumpington Street model. Two novel views showing (a) recovered camera pose and wire frame geometry and (b) the texture-mapped model. Note that some of the buildings have different orientations.

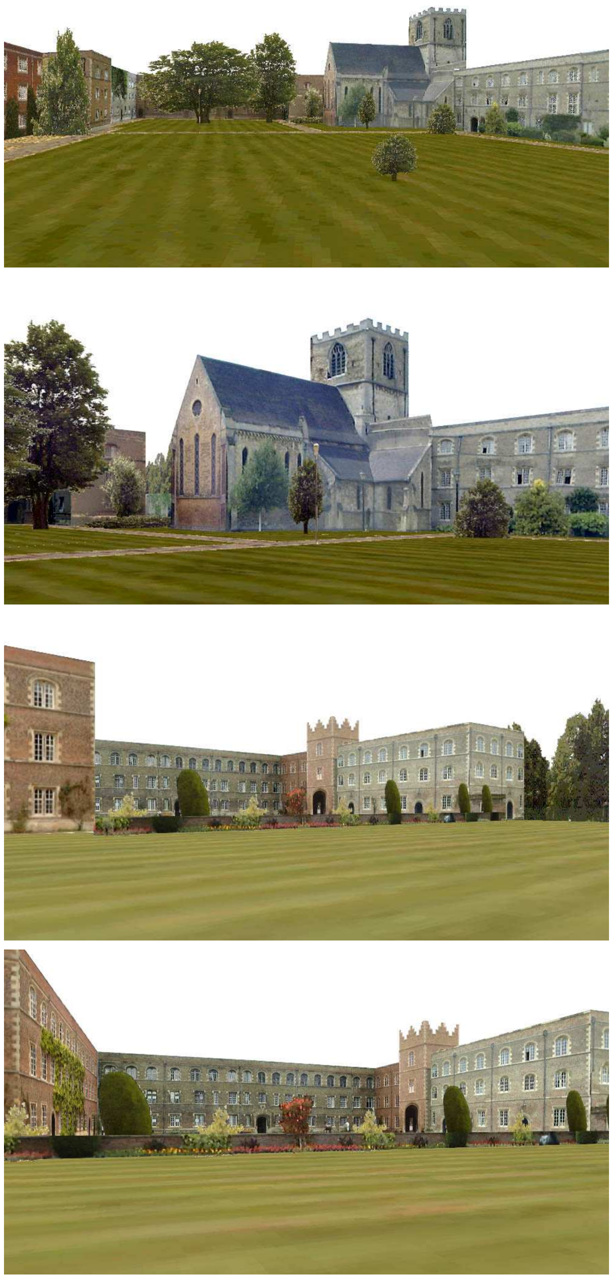


Figure 14.11: Some views of a large model of Jesus College, Cambridge. This model was reconstructed using around 50 photographs obtained using a camera with fixed but unknown focal length. The buildings in Jesus have differing orientations and so a plan of the college was used for camera calibration.

ning, models can be obtained using fewer views, and less user intervention is required.

Two simple, geometrically intuitive methods have been proposed for obtaining the camera calibration matrix and orientation for a *single* perspective view. One strategy (after Caprile and Torre [3]) is to use vanishing points corresponding to orthogonal directions in the scene. However, it is difficult to extend this approach to scenes that include buildings with differing orientations or non-rectangular plan. An interesting alternative is to use readily available map data. This approach allows views to be oriented in a *global* map coordinate system rather than a local one associated with a particular building. By combining the constraints provided by a map and *multiple* perspective views, it is simple to model scenes containing a large number of buildings, possibly with differing orientations or non-rectangular plan.

Appendix A: Finding vanishing points

In general, the images of more than two parallel lines will not intersect at a single point because of measurement noise. Hence, vanishing points must be chosen so as to minimize an appropriate error metric.

Maximum likelihood estimate. A good strategy is to fit a set of lines that do intersect at a point, and minimize the sum of squared orthogonal errors from the endpoints of the measured line segments (Hartley and Zisserman [7]). This approach is illustrated in Figure 14.12. The vanishing point estimate $\tilde{\mathbf{v}}$ is refined by iterative non-linear optimisation. To avoid the singularity associated with vanishing points at infinity, it is convenient to represent $\tilde{\mathbf{v}}$ by a homogenous 3-vector, subject to the constraint that $|\tilde{\mathbf{v}}| = 1$.

Linear least squares solution. An initial value for $\tilde{\mathbf{v}}$ is obtained by solving a linear least squares problem. The method is given a simple 3D interpretation in Figure 14.13.

An imaged line segment may be represented by a vector $\tilde{\mathbf{l}}$ that is normal to the plane defined by the line and the centre of projection. Let $\tilde{\mathbf{x}}_1$ and $\tilde{\mathbf{x}}_2$ be the viewing directions associated with the endpoints of the imaged line segment, *i.e.* $\tilde{\mathbf{x}}_i \sim \mathbf{K}^{-1}\tilde{\mathbf{u}}_i$. Then $\tilde{\mathbf{l}} = \tilde{\mathbf{x}}_1 \times \tilde{\mathbf{x}}_2$ where $\times$ denotes the cross product. Here, each $\tilde{\mathbf{x}}_i$ is normalised such that $|\tilde{\mathbf{x}}_i| = 1$.

The vectors $\tilde{\mathbf{l}}_i$ corresponding to parallel lines in the world all lie in a plane (see Figure 14.13). Hence, the common vanishing direction $\tilde{\mathbf{y}} \sim \mathbf{K}^{-1}\tilde{\mathbf{v}}$ may be obtained by solving the following linear least squares estimation problem:

$$\tilde{\mathbf{y}} = \arg\min_{\tilde{\mathbf{y}}} \sum_i (\tilde{\mathbf{l}}_i^\top \tilde{\mathbf{y}}) \tag{14.25}$$

which may be written as:

$$\tilde{\mathbf{y}} = \arg\min_{\tilde{\mathbf{y}}} |\mathbf{L}\tilde{\mathbf{y}}|^2 \tag{14.26}$$

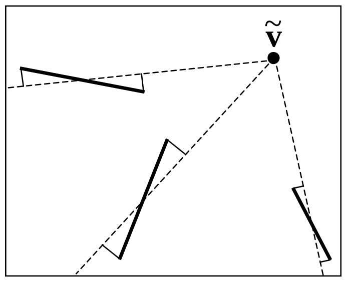


Figure 14.12: Maximum likelihood vanishing point estimation. The vanishing point $\tilde{\mathbf{v}}$ is the intersection of a set of fitted lines that minimize sum of squared orthogonal errors from the endpoints of the measured line segments.

where $\mathbf{L} = [\tilde{\mathbf{l}}_1 \quad \tilde{\mathbf{l}}_2 \quad \ldots]^\top$. The solution is the eigenvector of $\mathbf{L}^\top\mathbf{L}$ associated with the smallest eigenvalue and can be obtained by the singular value decomposition [11].

In practice, the camera calibration matrix $\mathbf{K}$ might be unknown. But since this linear estimation step is used only to provide an initialisation for non-linear optimisation, an approximate value for $\mathbf{K}$ will be perfectly adequate. Typically, pixels are assumed to be square and the principal point is set to the image centre. The focal length is set to the right approximate order of magnitude.

Bibliography

[1] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d points sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987.

[2] S. Becker and J. V. M. Bove. Semi-automatic 3-D model extraction from uncalibrated 2-D camera views. *SPIE Visual Data Exploration and Analysis II*, 2410:447–461, 1995.

[3] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International Journal of Computer Vision*, 4(2):127–140, 1990.

[4] A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. In *International Conference on Computer Vision (ICCV'99)*, pages 941–947, 1999.

[5] P. E. Debevec, C. J. Taylor, and J. Malik. Modelling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Computer Graphics (SIGGRAPH'96)*, pages 11–20, 1996.

[6] P. M. Grant. Urban GIS: The application of information technologies to urban management. In *Informing Technologies for Construction, Civil Engineering and Transport*, pages 195–199, 1993.

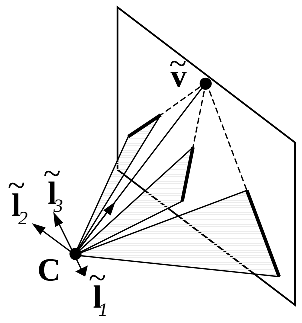


Figure 14.13: A 3D interpretation of the linear least squares method of vanishing point estimation. A projected line segment may be represented by the normal $\tilde{\mathbf{l}}_i$ to a plane (shaded) defined by its end points and the optical centre $\mathbf{C}$. In the absence of noise, the vanishing direction $\mathbf{K}^{-1}\tilde{\mathbf{v}}$ is perpendicular to all such plane normals $\tilde{\mathbf{l}}_i$.

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

[8] T. Kanade, P. W. Rander, and P. J. Narayanan. Virtualized reality: constructing virtual worlds from real scenes. *IEEE MultiMedia Magazine*, 1 (1):34–47, 1997.

[9] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. In *Eurographics*, volume 18, pages 39–50, 1999.

[10] D. Liebowitz and A. Zisserman. Combining scene and auto-calibration constraints. In *International Conference on Computer Vision (ICCV'99)*, pages 293–300, 1999.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 1992.

[12] D. P. Robertson and R. Cipolla. Building architectural models from many views using map constraints. In *European Conference on Computer Vision (ECCV'02)*, pages 155–169, 2002.

[13] H-Y. Shum, M. Han, and R. Szeliski. Interactive construction of 3D models from panoramic mosaics. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'98)*, pages 427–433, 1998.

[14] A. Streilein and U. Hirschberg. Integration of digital photogrammetry and CAAD: constraint-based modelling and semi-automatic measurement. In *CAAD Futures '95 International Conference*, 1995.

[15] P. F. Sturm and S. J. Maybank. A method for interactive 3D reconstruction of piecewise planar objects from single images. In *British Machine Vision Conference (BMVC'99)*, pages 265–274, 1999.

[16] W. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

[17] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.