Chapter 2
Passive 3D Imaging

Stephen Se and Nick Pears

Abstract We describe passive 3D imaging systems that recover 3D information from scenes that are illuminated only with ambient lighting. Although we briefly overview monocular reconstruction, much of the material is concerned with using the geometry of stereo 3D imaging to formulate estimation problems. Firstly, we present an overview of the common techniques used to recover 3D information from camera images. Secondly, we discuss camera modeling and camera calibration as an essential introduction to the geometry of the imaging process and the estimation of geometric parameters. Thirdly, we focus on 3D recovery from multiple views, which can be obtained using multiple cameras at the same time (stereo), or a single moving camera at different times (structure from motion). Epipolar geometry and finding image correspondences associated with the same 3D scene point are two key aspects for such systems, since epipolar geometry establishes the relationship between two camera views, and depth information can be inferred from the correspondences. The details of both stereo and structure from motion, the two essential forms of multiple-view 3D reconstruction technique, are presented. We include a brief overview of the recent trend of applying deep learning to passive 3D imaging. Finally, we present several real-world applications.

2.1 Introduction

Passive 3D imaging has been studied extensively for several decades, and it is a core topic in many of the major computer vision conferences and journals. Essentially, a *passive* 3D imaging system, also known as a passive 3D vision system, is one in which we can recover 3D scene information, without that system having to project

S. Se (✉)
FLIR Systems Inc., 12051 Riverside Way, Richmond, BC V6W 1K7, Canada
e-mail: stephen.se@flir.com

N. Pears
Department of Computer Science, University of York,
Deramore Lane, York YO10 5GH, UK
e-mail: nick.pears@york.ac.uk

© Springer Nature Switzerland AG 2020

Y. Liu et al. (eds.), *3D Imaging, Analysis and Applications*,
https://doi.org/10.1007/978-3-030-44070-1_2

its own source of light or other source of electromagnetic radiation (EMR) onto that scene. By contrast, an *active* 3D imaging system has an EMR projection subsystem, which is commonly in the infrared or visible wavelength region.

Several passive 3D information sources (cues) relate closely to human vision and other animal vision. For example, in stereo vision, fusing the images recorded by our two eyes and exploiting the difference between them gives us a sense of depth. The aim of this chapter is to present the fundamental principles of passive 3D imaging systems so that readers can understand their strengths and limitations, as well as how to implement a subset of such systems, namely, those that exploit multiple views of the scene.

Passive, multiple-view 3D imaging originates from the mature field of photogrammetry and, more recently, from the younger field of computer vision. In contrast to photogrammetry, computer vision applications rely on fast, automatic techniques, sometimes at the expense of precision. Our focus is on the computer vision perspective.

A recurring theme of this chapter is that we consider some aspect of the geometry of 3D imaging and formulate a linear least squares estimation problem to estimate the associated geometric parameters. These estimates can then optionally be improved, depending on the speed and accuracy requirements of the application, using the linear estimate as an initialization for a non-linear least squares refinement. In contrast to the linear stage, this non-linear stage usually optimizes a cost function that has a well-defined geometric meaning.[1] Multi-view 3D reconstruction is now a mature technology. Figure 2.1 shows a 3D reconstruction of a building from a 2D image sequence.

2.1.1 Chapter Outline

We will start with an overview of various techniques for passive 3D imaging systems, including single-view and multiple-view approaches. However, the main body of this chapter is focused on 3D recovery from multiple views, which can be obtained using multiple cameras simultaneously (stereo) or a single moving camera (structure from motion). A good starting point to understand this subject matter is knowledge of the image formation process in a single camera and how to capture this process in a camera model. This modeling is presented in Sect. 2.3, and the following section describes camera calibration: the estimation of the parameters in the developed camera model. In order to understand how to search efficiently for left–right feature pairs that correspond to the same scene point in a stereo image pair (the *correspondence* problem), a good understanding of two-view geometry is required, which establishes the relationship between two camera views. Hence, Sect. 2.5 details this geometry, known as *epipolar geometry*, and shows how it can be captured and used in linear (vector–matrix) form. Following this, we can begin to consider the correspondence

[1] Parameters obtained from a linear estimate are often a complex function of geometric parameters.

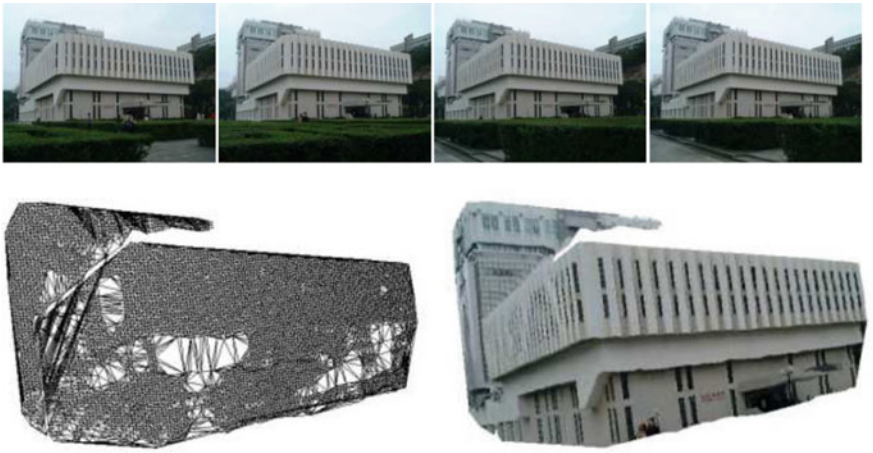

Fig. 2.1 Top: partial image sequence of the library building at the Chinese University of Hong Kong. Bottom: 3D reconstruction of the library building, obtained automatically from the uncalibrated image sequence (left: Delaunay triangulation; right: with texture). Figure courtesy of [16]

problem and the first step is to simplify the search to be across the same horizontal scan lines in each image, by warping the stereo image pair in a process known as *rectification*. This is described in Sect. 2.6. The following section then focuses on the correspondence search itself, and then Sect. 2.8 details the process of generating a 3D point cloud from a set of image correspondences. A brief overview is included on how deep learning techniques, such as convolutional neural networks and recurrent neural networks, have been applied to passive 3D imaging recently and had had a big impact. Despite this, we believe that it is important that this chapter covers the underlying mathematical modeling explicitly for a good understanding of the subject area.

With increasing computer processing power and decreasing camera prices, many real-world applications of passive 3D imaging systems have been emerging in recent years. Thus, later in the chapter (Sect. 2.10), some recent applications involving such systems are discussed. Several commercially available stereo vision systems will first be presented, as well as stereo cameras for people counting in retail analytics. We then describe 3D modeling systems that generate photo-realistic 3D models from image sequences, which have a wide range of applications. Later in this section, passive 3D imaging systems for mobile robot pose estimation and visual SLAM are described. In the following section, multiple-view passive 3D imaging systems are compared to their counterpart within active 3D imaging systems.

The final sections in the chapter cover concluding remarks, further reading, software resources, questions, and exercises for the reader.

2.2 An Overview of Passive 3D Imaging Systems

Most cameras today use either a *Charge Coupled Device* (CCD) image sensor or a *Complementary Metal Oxide Semiconductor* (CMOS) sensor, both of which capture light and convert it into electrical signals. Historically, CCD sensors have provided higher quality, lower noise images, whereas CMOS sensors have been cheaper, more compact, and consume less power. Presently, CMOS sensors are more prevalent perhaps due to their now highly favorable performance-to-cost ratio. The cameras employing such image sensors can be hand-held or mounted on different platforms such as *Unmanned Ground Vehicles* (UGVs), *Unmanned Aerial Vehicles* (UAVs), *Unmanned Underwater Vehicles* (UUVs), and optical satellites.

Passive 3D vision techniques can be categorized as follows: (i) Multiple-view approaches and (ii) single-view approaches. We outline each of these in the following two subsections.

2.2.1 Multiple-View Approaches

In multiple-view approaches, the scene is observed from two or more viewpoints, by either multiple cameras at the same time (stereo) or a single moving camera at different times (structure from motion). From the gathered images, the system is to infer information on the 3D structure of the scene.

Stereo refers to multiple images taken simultaneously using two or more cameras, which are collectively called a stereo camera. For example, binocular stereo uses two viewpoints, trinocular stereo uses three viewpoints, or alternatively there may be many cameras distributed around the viewing sphere of an object. *Stereo* derives from the Greek word *stereos* meaning *solid*, thus implying a 3D form of visual information. In this chapter, we will use the term *stereo vision* to imply a binocular stereo system. At the top of Fig. 2.2, we show an outline of such a system.

If we can determine that imaged points in the left and right cameras correspond to the same scene point, then we can determine two directions (3D rays) along which the 3D point must lie. (The camera parameters required to convert the 2D image positions to 3D rays come from a camera calibration procedure.) Then, we can intersect the 3D rays to determine the 3D position of the scene point, in a process known as *triangulation*. A scene point, **X**, is shown in Fig. 2.2 as the intersection of two rays (colored black) and a nearer point is shown by the intersection of two different rays (colored blue). Note that the difference between left and right image positions, the *disparity*, is greater for the nearer scene point and inversely proportional to the range. In fact, at some maximum range, the disparity becomes so small and corruptable by noise, which usefully accurate range measurements can no longer be made. Compared to some other 3D imaging modalities, stereo is relatively short range. Also, regions close to the stereo camera rig are outside of the field of view of either one or both cameras, and so stereo cameras have some defined minimum

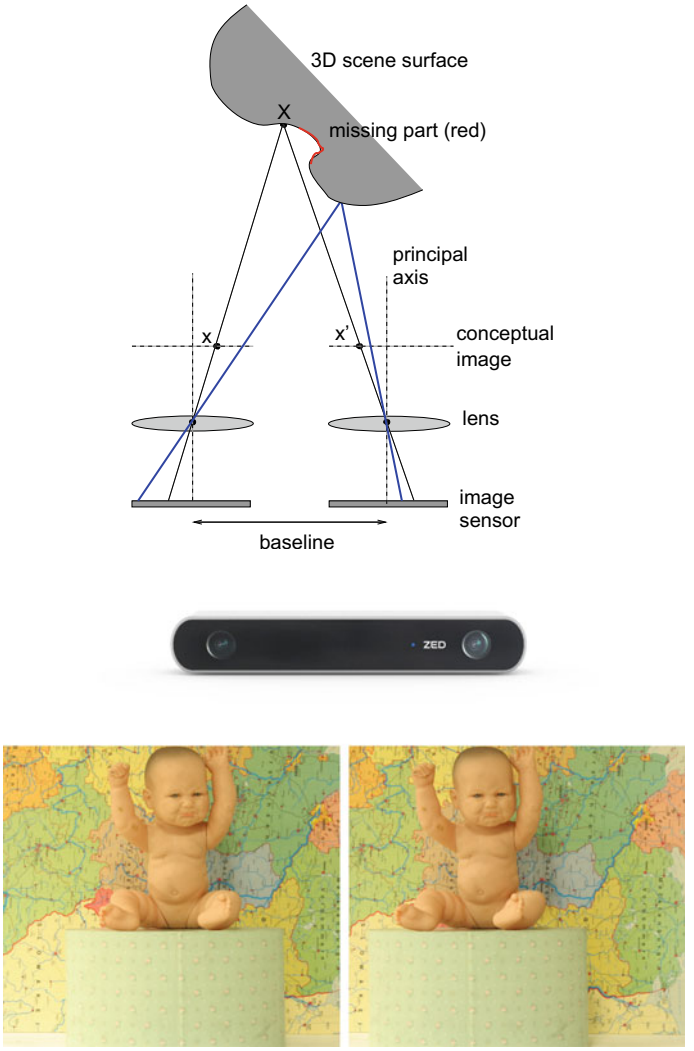

Fig. 2.2 Top: Plan view of the operation of a simple stereo rig. Here the optical axes of the two cameras are parallel to form a rectilinear rig. However, often the cameras are rotated toward each other (verged) to increase the overlap in their fields of view. Center: The commercial ZED stereo camera, supplied by Stereolabs [82]. Bottom: Left and right views of a stereo pair (images courtesy of [74])

range of operation. Note also that the scene surface colored red cannot be observed by the right camera, in which case no 3D shape measurement can be made. This scene portion is sometimes referred to as a *missing part* and is the result of self-occlusion. The baseline length of a stereo camera is a trade-off between accuracy and missing parts.

A modeling point to note is that, although the real image sensor is behind the lens, it is common practice to envisage and use a conceptual image position in front of the lens so that the image is the same orientation as the scene (i.e., not inverted top to bottom and left to right), and this is shown by the dashed horizontal line in the figure.

Despite the apparent simplicity of Fig. 2.2(top), a large part of this chapter is required to present the various aspects of stereo 3D imaging in detail, such as calibration, determining left-to-right image correspondences and dense 3D shape reconstruction. A typical commercial stereo camera, called the ZED and supplied by Stereolabs [82], is shown in the center of Fig. 2.2, although many computer vision researchers build their own stereo rigs, using off-the-shelf digital cameras and a slotted steel bar mounted on a tripod. Finally, at the bottom of Fig. 2.2, we show the left and right views of a typical stereo pair taken from the Middlebury webpage [74].

In contrast to stereo vision, *structure from motion* (SfM) refers to a single moving camera scenario, where image sequences are captured over a period of time. While stereo refers to fixed relative viewpoints with synchronized image capture, SfM refers to variable viewpoints with sequential image capture. For image sequences captured at a high frame rate, optical flow can be computed, which estimates the motion field from the image sequences, based on the spatial and temporal variations of the image brightness. Using the local brightness constancy alone, the problem is under-constrained as the number of variables is twice the number of measurements. Therefore, it is augmented with additional global smoothness constraints, so that the motion field can be estimated by minimizing an energy function [41, 49]. 3D motion of the camera and the scene structure can then be recovered from the motion field.

2.2.2 Single-View Approaches

In contrast to these two multiple-view approaches, 3D shape can be inferred from a single viewpoint using information sources (cues) such as shading, texture, and focus. The collection of such approaches is sometimes termed *Shape from X*. Not surprisingly, the mentioned techniques are called *Shape from Shading*, *Shape from Texture*, and *Shape from Focus*, respectively.

Shading on a surface can provide information about local surface orientations and overall surface shape, as illustrated in Fig. 2.3, where the technique in [42] has been used. Shape from shading [40] uses the shades in a grayscale image to infer the shape of the surfaces, based on the reflectance map which links image intensity with surface orientation. After the surface normals have been recovered at each pixel, they can be integrated into a depth map using regularized surface fitting. The

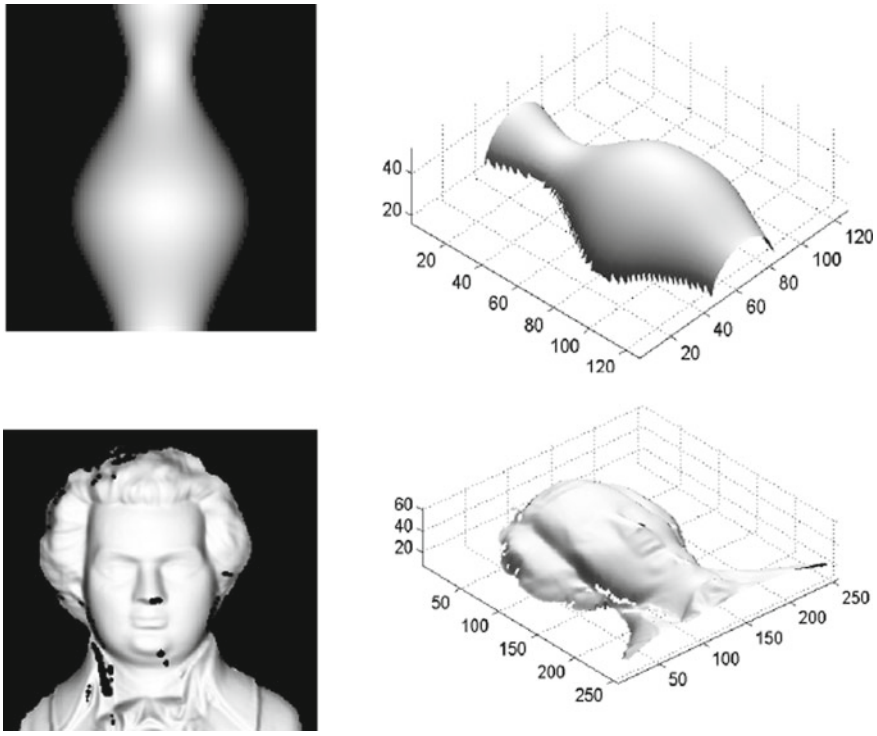

Fig. 2.3 Examples of synthetic shape from shading images (left column) and corresponding shape from shading reconstruction (right column)

computations involved are considerably more complicated than for multiple-view approaches. Moreover, various assumptions, such as uniform albedo, reflectance, and known light source directions, need to be made and there are open issues with convergence to a solution. The survey in [97] reviews various techniques and provides some comparative results. The approach can be enhanced when lights shining from different directions can be turned on and off separately. This technique is known as *photometric stereo* [91], and it takes two or more images of the scene from the same viewpoint but under different illuminations in order to estimate the surface normals.

The foreshortening of regular patterns depends on how the surface slants away from the camera viewing direction and provides another cue on the local surface orientation. Shape from texture [34] estimates the shape of the observed surface from the distortion of the texture created by the imaging process, as illustrated in Fig. 2.4. Therefore, this approach works only for images with texture surfaces and assumes the presence of a regular pattern. Shape from shading is combined with shape from texture in [90] where the two techniques can complement each other. While the texture components provide information in textured region, shading helps in the uniform region to provide detailed information on the surface shape.

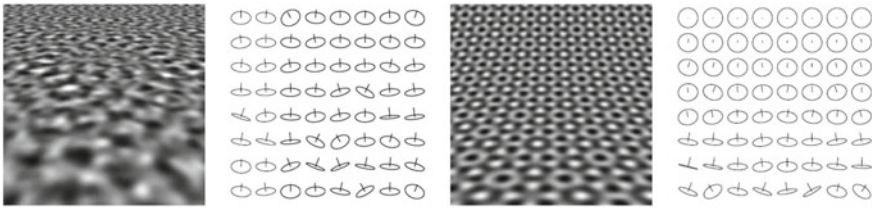

Fig. 2.4 Examples of synthetic shape from texture images (**a**, **c**) and corresponding surface normal estimates (**b**, **d**). Figure courtesy of [34]

Shape from focus [58, 65] estimates depth using two input images captured from the same viewpoint but at different camera depths of field. The degree of blur is a strong cue for object depth as it increases as the object moves away from the camera's focusing distance. The relative depth of the scene can be constructed from the image blur where the amount of defocus can be estimated by averaging the squared gradient values in a region.

Another approach that is specialized for a particular object class is the *Analysis by Synthesis* approach. Here, the assumption is that we have a 3D model of the object class of interest, such as the human face [10] or human head [22]. If we also model all the physical components that generate an image, such as surface reflectance, scene lighting, and the projective imaging processes, and if we parameterize these model components accurately by model fitting, we can synthesize an image that is very similar to the actual image. The 3D structure obtained from the parametrization of the 3D model is then the 3D reconstruction of the single-viewpoint image [9].

Single-view metrology [21] allows shape recovery from a single perspective view of a scene given some geometric information determined from the image. By exploiting scene constraints such as orthogonality and parallelism, a vanishing line and a vanishing point in a single image can be determined. Relative measurements of shape can then be computed, which can be upgraded to absolute metric measurements if the dimensions of a reference object in the scene are known.

While 3D recovery from a single view is possible, such methods are often not practical in terms of either robustness or speed or both. Therefore, the most commonly used approaches are based on multiple views, which is the focus of this chapter. The first step to understanding such approaches is to understand how to model the image formation process in the cameras of a stereo rig. Then we need to know how to estimate the parameters of this model. Thus camera modeling and camera calibration are discussed in the following two main sections.

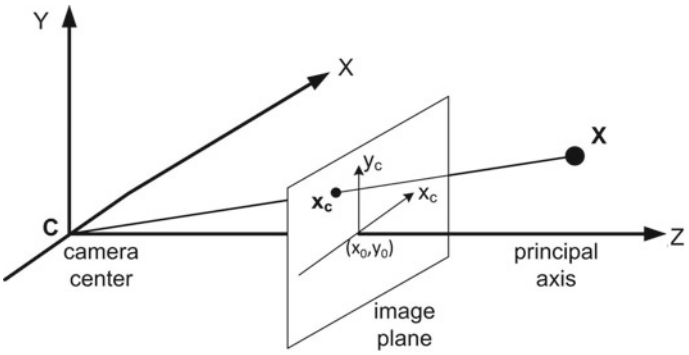

Fig. 2.5 Projection based on a pinhole camera model where a 3D object is projected onto the image plane. Note that, although the real image plane is behind the camera center, it is common practice to employ a virtual image plane in front of the camera, so that the image is conveniently at the same orientation as the scene

2.3 Camera Modeling

A camera is a device in which the 3D scene is projected down onto a 2D image. The most commonly used projection in computer vision is 3D perspective projection. Figure 2.5 illustrates perspective projection based on the pinhole camera model, where **C** is the position of the pinhole, termed the camera center or the *center of projection*. Note that, although the real image plane is behind the camera center, it is common practice to employ a virtual image plane in front of the camera, so that the image is conveniently at the same orientation as the scene.

Clearly, from this figure, the path of imaged light is modeled by a ray that passes from a 3D world point **X** through the camera center. The intersection of this ray with the image plane defines where the image, $\mathbf{x_c}$, of the 3D scene point, **X**, lies. We can reverse this process and say that, for some point on the image plane, its corresponding scene point must lie somewhere along the ray connecting the center of projection, **C**, and that imaged point, $\mathbf{x_c}$. We refer to this as *back-projecting* an image point to an infinite ray that extends out into the scene. Since we do not know how far along the ray the 3D scene point lies, explicit depth information is lost in the imaging process. This is the main source of geometric ambiguity in a single image and is the reason why we refer to the recovery of the depth information from stereo and other cues as *3D reconstruction*.

Before we embark on our development of a mathematical camera model, we need to digress briefly and introduce the concept of homogeneous coordinates (also called projective coordinates), which is the natural coordinate system of analytic projective geometry and hence has wide utility in geometric computer vision.

2.3.1 Homogeneous Coordinates

We are all familiar with expressing the position of some point in a plane using a pair of coordinates as $[x, y]^T$. In general for such systems, n coordinates are used to describe points in an n-dimensional space, $\mathbb{R}^n$. In analytic projective geometry, which deals with algebraic theories of points and lines, such points and lines are typically described by *homogeneous coordinates*, where $n + 1$ coordinates are used to describe points in an n-dimensional space. For example, a general point in a plane is described as $\mathbf{x} = [x_1, x_2, x_3]^T$, and the general equation of a line is given by $\mathbf{l}^T\mathbf{x} = 0$ where $\mathbf{l} = [l_1, l_2, l_3]^T$ are the homogeneous coordinates of the line.[2] Since the right-hand side of this equation for a line is zero, it is a homogeneous equation, and any non-zero multiple of the point $\lambda[x_1, x_2, x_3]^T$ is the same point; similarly, any non-zero multiple of the line's coordinates is the same line. The symmetry in this equation is indicative of the fact that points and lines can be exchanged in many theories of projective geometry; such theories are termed *dual* theories. For example, the cross-product of two lines, expressed in homogeneous coordinates, yields their intersecting point, and the cross-product of a pair of points gives the line between them. This is very useful for manipulating points and lines in the image plane. For example, if we had four image points, then to determine the point of intersection, $\mathbf{x}$, of any two lines defined by any two points we simply compute:

$$\mathbf{x} = (\mathbf{x}_i \times \mathbf{x}_j) \times (\mathbf{x}_k \times \mathbf{x}_l).$$

Note that we can easily convert from homogeneous to inhomogeneous coordinates, simply by dividing through by the third element, thus $[x_1, x_2, x_3]^T$ maps to $[\frac{x_1}{x_3}, \frac{x_2}{x_3}]^T$. A key point about homogeneous coordinates is that they allow the relevant transformations in the imaging process to be represented as linear mappings, which of course are expressed as matrix–vector equations. However, although the mapping between homogeneous world coordinates of a point and homogeneous image coordinates is linear, the mapping from homogeneous to inhomogeneous coordinates is non-linear, due to the required division.

The use of homogeneous coordinates fits well with the relationship between image points and their associated back-projected rays into the scene space. Imagine a mathematical (virtual) image plane at a distance of one metric unit in front of the center of projection, as shown in Fig. 2.5. With the camera center, $\mathbf{C}$, the homogeneous coordinates $[x, y, 1]^T$ define a 3D scene ray as $[\lambda x, \lambda y, \lambda]^T$, where λ is the unknown distance ($\lambda > 0$) along the ray. Thus there is an intuitive link between the depth ambiguity associated with the 3D scene point and the equivalence of homogeneous coordinates up to an arbitrary non-zero scale factor.

Extending the idea of thinking of homogeneous image points as 3D rays, consider the cross-product of two homogeneous points. This gives a direction that is the normal

[2] You may wish to compare $\mathbf{l}^T\mathbf{x} = 0$ to two well-known parameterizations of a line in the (x,y) plane, namely, $ax + by + c = 0$ and $y = mx + c$ and, in each case, write down homogeneous coordinates for the point $\mathbf{x}$ and the line $\mathbf{l}$.

of the plane that contains the two rays. The line between the two image points is the intersection of this plane with the image plane. The dual of this is that the cross-product of two lines in the image plane gives a point on the image plane whose associated 3D ray represents the intersection of their associated planes. The 3D ray has a direction orthogonal to the normals of both of these planes and is the direction of the ray that defines the point of intersection of the two lines in the image plane. Note that any point with its third homogeneous element zero defines a ray parallel to the image plane and hence meets it at infinity. Such a point is termed a *point at infinity*, and there is an infinite set of these points $[x_1, x_2, 0]^T$ that lie on the *line at infinity* $[0, 0, 1]^T$; Finally, note that the 3-tuple $[0, 0, 0]^T$ has no meaning and is undefined. For further reading on homogeneous coordinates and projective geometry, please see [20, 38].

2.3.2 Perspective Projection Camera Model

We now return to the perspective projection (central projection) camera model, and we note that it maps 3D world points in standard metric units into the pixel coordinates of an image sensor. It is convenient to think of this mapping as the concatenation of three successive stages:

1. A six degree-of-freedom (DOF) rigid transformation consisting of a rotation, R (3 DOF), and translation, $\mathbf{t}$ (3 DOF), that maps points expressed in world coordinates to the same points expressed in camera-centered coordinates.
2. A perspective projection from the 3D world to the 2D image plane.
3. A mapping from metric image coordinates to pixel coordinates.

We now discuss each of these projective mappings in turn.

2.3.2.1 Camera Modeling: The Coordinate Transformation

As shown in Fig. 2.5, the camera frame has its (X,Y) plane parallel to the image plane and Z is in the direction of the principal axis of the lens and encodes depth from the camera. Suppose that the camera center has *inhomogeneous* position $\tilde{\mathbf{C}}$ in the world frame[3] and the rotation of the camera frame is R_c relative to the world frame orientation. This means that we can express any *inhomogeneous* camera frame points as:

$$\tilde{\mathbf{X}}_c = \mathsf{R}_c^T(\tilde{\mathbf{X}} - \tilde{\mathbf{C}}) = \mathsf{R}\tilde{\mathbf{X}} + \mathbf{t}. \tag{2.1}$$

Here $\mathsf{R} = \mathsf{R}_c^T$ represents the rigid rotation and $\mathbf{t} = -\mathsf{R}_c^T\tilde{\mathbf{C}}$ represents the rigid translation that maps a scene point expressed in the world coordinate frame into a camera-

[3] We use a tilde to differentiate n-tuple inhomogeneous coordinates from (n+1)-tuple homogeneous coordinates.

centered coordinate frame. Equation 2.1 can be expressed as a projective mapping, namely, one that is linear in homogeneous coordinates, to give

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathsf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}.$$

We denote P_r as the 4×4 homogeneous matrix representing the rigid coordinate transformation in the above equation.

2.3.2.2 Camera Modeling: Perspective Projection

Observing the similar triangles in the geometry of perspective imaging, we have

$$\frac{x_c}{f} = \frac{X_c}{Z_c}, \quad \frac{y_c}{f} = \frac{Y_c}{Z_c}, \tag{2.2}$$

where (x_c, y_c) is the position (metric units) of a point in the camera's image plane and f is the distance (metric units) of the image plane to the camera center. (This is usually set to the focal length of the camera lens.) The two equations above can be written in linear form as

$$Z_c \begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}.$$

We denote P_p as the 3×4 perspective projection matrix, defined by the value of f, in the above equation. If we consider an abstract image plane at $f = 1$, then points on this plane are termed *normalized image coordinates*[4] and from Eq. 2.2, these are given by

$$x_n = \frac{X_c}{Z_c}, \quad y_n = \frac{Y_c}{Z_c}.$$

2.3.2.3 Camera Modeling: Image Sampling

Typically, the image on the image plane is sampled by an image sensor, such as a CCD or CMOS device, at the locations defined by an array of pixels. The final part of camera modeling defines how that array is positioned on the $[x_c, y_c]^T$ image plane,

[4] We need to use a variety of image coordinate normalizations in this chapter. For simplicity, we will use the same subscript n, but it will be clear about how the normalization is achieved.

so that pixel coordinates can be generated. In general, pixels in an image sensor are not square and the number of pixels per unit distance varies between the x_c and y_c directions; we will call these scalings m_x and m_y. Note that pixel positions have their origin at the corner of the sensor and so the position of the principal point (where the principal axis intersects the image plane) is modeled with pixel coordinates $[x_0, y_0]^T$. Finally, many camera models also cater for any skew,[5] s, so that the mapping into pixels is given by

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m_x & s & x_0 \\ 0 & m_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix}.$$

We denote P_c as the 3×3 projective matrix defined by the five parameters m_x, m_y, s and x_0, y_0 in the above equation.

2.3.2.4 Camera Modeling: Concatenating the Projective Mappings

We can concatenate the three stages described in the three previous subsections to give

$$\lambda \mathbf{x} = \mathsf{P}_c \mathsf{P}_p \mathsf{P}_r \mathbf{X}$$

or simply

$$\lambda \mathbf{x} = \mathsf{P} \mathbf{X}, \tag{2.3}$$

where λ is non-zero and positive. We note the following points concerning the above equation:

1. For any homogeneous image point scaled to $\lambda[x, \ y, \ 1]^T$, the scale λ is equal to the imaged point's depth in the camera-centered frame ($\lambda = Z_c$).
2. Any non-zero scaling of the projection matrix $\lambda_P \mathsf{P}$ performs the same projection since, in Eq. 2.3, any non-zero scaling of homogeneous image coordinates is equivalent.
3. A camera with projection matrix P, or some non-zero scalar multiple of that, is informally referred to as *camera P* in the computer vision literature and, because of point 2 above, it is referred to as being defined *up to scale*.

The matrix P is a 3×4 projective camera matrix with the following structure:

$$\mathsf{P} = \mathsf{K}[\mathsf{R}|\mathbf{t}]. \tag{2.4}$$

The parameters within K are the camera's *intrinsic parameters*. These parameters are those combined from Sects. 2.3.2.2 and 2.3.2.3 above, so that

[5] Skew models a lack of orthogonality between the two image sensor sampling directions. For most imaging situations, it is zero.

$$\mathsf{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix},$$

where $\alpha_x = fm_x$ and $\alpha_y = fm_y$ represent the focal length in pixels in the x and y directions, respectively. Together, the rotation and translation in Eq. 2.4 are termed the camera's *extrinsic parameters*. Since there are 5 DOF from intrinsic parameters and 6 DOF from extrinsic parameters, a camera projection matrix has only 11 DOF, not the full 12 of a general 3×4 matrix. This is also evident from the fact that we are dealing with homogeneous coordinates and so the overall scale of P does not matter.

By expanding Eq. 2.3, we have

$$\lambda \overbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}^{\substack{homogeneous \\ image \\ coordinates}} = \overbrace{\begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}}^{\substack{intrinsic \\ camera \\ parameters}} \overbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}}^{\substack{extrinsic \\ camera \\ parameters}} \overbrace{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}^{\substack{homogeneous \\ world \\ coordinates}}, \tag{2.5}$$

which indicates that both the intrinsic and extrinsic camera parameters are necessary to fully define a ray (metrically, not just in pixel units) in 3D space and hence make absolute measurements in multiple-view 3D reconstruction. Finally, we note that any non-zero scaling of scene homogeneous coordinates $[X, Y, Z, 1]^T$ in Eq. 2.5 gives the same image coordinates[6] which, for a single image, can be interpreted as ambiguity between the scene scale and the translation vector $\mathbf{t}$.

2.3.3 Radial Distortion

Typical cameras have a lens distortion which disrupts the assumed linear projective model. Thus a camera may not be accurately represented by the pinhole camera model that we have described, particularly if a low-cost lens or a wide field-of-view (short focal length) lens such as a fish-eye lens is employed. Some examples of lens distortion effects are shown in Fig. 2.6. Note that the effect is non-linear and, if significant, it must be corrected so that the camera can again be modeled as a linear device. The estimation of the required distortion parameters to do this is often encompassed within a camera calibration procedure, which is described in Sect. 2.4. With reference to our previous three-stage development of a projective camera in Sect. 2.3.2, lens distortion occurs at the second stage which is the 3D to 2D projection, and this distortion is sampled by the image sensor.

[6] The same homogeneous image coordinates *up to scale* or the same inhomogeneous image coordinates.

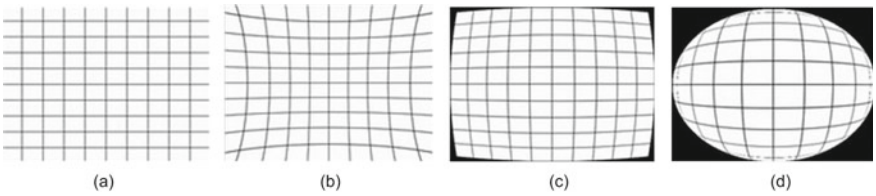

Fig. 2.6 Examples of radial distortion effects in lenses: (**a**) No distortion, (**b**) Pincushion distortion, (**c**) Barrel distortion, (**d**) Fish-eye distortion

Detailed distortion models contain a large number of parameters that model both radial and tangential distortions [13]. However, radial distortion is the dominant factor and usually it is considered sufficiently accurate to model this distortion only, using a low-order polynomial such as

$$\begin{bmatrix} x_{nd} \\ y_{nd} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} + \begin{bmatrix} x_n \\ y_n \end{bmatrix} (k_1 r^2 + k_2 r^4),$$

where $[x_n, y_n]^T$ is the undistorted image position (i.e., that obeys our linear projection model) in normalized coordinates, $[x_{nd}, y_{nd}]^T$ is the distorted image position in normalized coordinates, k_1 and k_2 are the unknown radial distortion parameters, and $r = \sqrt{x_n^2 + y_n^2}$. Assuming zero skew, we also have

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} (x - x_0) \\ (y - y_0) \end{bmatrix} (k_1 r^2 + k_2 r^4), \tag{2.6}$$

where the distorted position $[x_d, y_d]^T$ is now expressed in pixel coordinates and $[x, y]^T$ are the usual pixel coordinates predicted by the linear pinhole model. Note that r is still defined in normalized image coordinates and so a non-unity aspect ratio ($m_x \neq m_y$) in the image sensor does not invalidate this equation. Also note that both Eq. 2.6 and Fig. 2.6 indicate that distortion increases away from the center of the image. In the barrel distortion, shown in Fig. 2.6c, distortion correction requires that image points are moved slightly toward the center of the image, more so if they are near the edges of the image. Correction could be applied to the whole image, as in dense stereo, or just a set of relevant features, such as extracted corner points. Clearly, the latter process is computationally cheaper.

Now that we have discussed the modeling of a camera's image formation process in detail; we now need to understand how to estimate the parameters within this model. This is the focus of the next section, which details camera calibration.

2.4 Camera Calibration

Camera calibration [14] is the process of finding the parameters of the camera that produced a given image of a scene. This includes both extrinsic parameters $\mathsf{R}, \mathbf{t}$ and intrinsic parameters, comprising those within the matrix K and radial distortion parameters, k_1, k_2. Once the intrinsic and extrinsic camera parameters are known, we know the camera projection matrix P and, taking into account any radial distortion present, we can back-project any image pixel to a 3D ray in space. Clearly, as the intrinsic camera calibration parameters are tied to the focal length, changing the zoom on the lens would make the calibration invalid. It is also worth noting that calibration is not always required. For example, we may be more interested in approximate shape, where we need to know what objects in a scene are co-planar, rather than their absolute 3D position measurements. However, for stereo systems at least, camera calibration is commonplace.

Generally, it is not possible for an end user to get the required calibration information to the required accuracy from camera manufacturer's specifications and external measurement of the position of cameras in some frames. Hence, some sort of camera calibration procedure is required, of which there are several different categories. The longest established of these is *photogrammetric calibration*, where calibration is performed using a scene object of precisely known physical dimensions. Typically, several images of a special 3D target, such as three orthogonal planes with calibration grids (chessboard patterns of black and white squares), are captured and precise known translations may be used [89]. Although this gives accurate calibration results, it lacks flexibility due to the need for precise scene knowledge.

At the other end of the spectrum is *self-calibration* (auto-calibration) [38, 55], where no calibration target is used. The correspondences across three images of the same rigid scene provide enough constraints to recover a set of camera parameters which allow 3D reconstruction up to a similarity transform. Although this approach is flexible, there are many parameters to estimate and reliable calibrations cannot always be obtained.

Between these two extremes are "desktop" camera calibration approaches that use images of planar calibration grids, captured at several unknown positions and orientations (i.e., a single planar chessboard pattern is manually held at several random poses, and calibration images are captured and stored). This gives a good compromise between the accuracy of photogrammetric calibration and the ease of use of self-calibration. A seminal example is given by Zhang [98].

Although there are a number of publicly available camera calibration packages on the web, such as the Caltech camera calibration toolbox for MATLAB [11] and in the OpenCV computer vision library [62], a detailed study of at least one approach is essential to understand calibration in detail. We will use Zhang's work [98] as a seminal example, and this approach consists of two main parts:

1. A *basic* calibration that is based on linear least squares and hence has a closed-form solution. In the formulation of the linear problem, a set of nine parameters need to be estimated. These are rather complicated combinations of the camera's intrinsic

parameters, and the algebraic least squares minimization to determine them has no obvious geometric meaning. Once intrinsic parameters have been extracted from these estimated parameters, extrinsic parameters can be determined using the homography associated with each calibration grid image.

2. A *refined* calibration is based on non-linear least squares and hence has an iterative solution. Here, it is possible to formulate a least squares error between the observed (inhomogeneous) image positions of the calibration grid corners and the positions predicted by the current estimate of intrinsic and extrinsic camera parameters. This has a clear geometric interpretation, but the sum of squares function that we wish to minimize is non-linear in terms of the camera parameters. A standard approach to solving this kind of problem is the *Levenberg–Marquardt* (LM) algorithm, which employs gradient descent when it is far from a minimum and Gauss–Newton minimization when it gets close to a minimum. Since the LM algorithm is a very general procedure, it is straightforward to employ more complex camera models, such as those that include parameters for the radial distortion associated with the camera lens.

The iterative optimization in (2) above needs to be within the basin of convergence of the global minimum and so the linear method in (1) is used to determine an initial estimation of camera parameters. The raw data used as inputs to the process consists of the image corner positions, as detected by an automatic corner detector [37] [81], of all corners in all calibration images and the corresponding 2D world positions, $[X,\ Y]^T$, of the corners on the calibration grid. Typically, correspondences are established by manually clicking one or more detected image corners, and making a quick visual check that the imaged corners are matched correctly using overlaying graphics or text. A typical set of targets is shown in Fig. 2.7.

In the following four subsections, we outline the theory and practice of camera calibration. The first subsection details the estimation of the planar projective map-

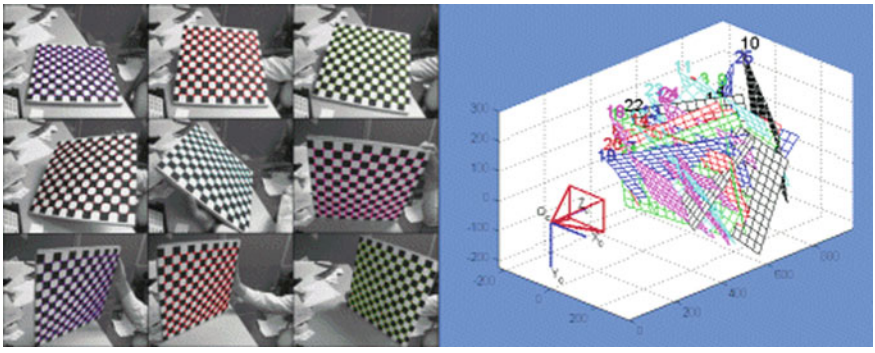

Fig. 2.7 Left: calibration targets used in a camera calibration process. Right: after calibration, it is possible to determine the positions of the calibration planes using the estimated extrinsic parameters (Figure courtesy of the *Camera Calibration Toolbox for MATLAB* webpage at Caltech by Jean-Yves Bouguet [11])

ping between a scene plane (calibration grid) and its image. The next two subsections closely follow Zhang [98] and detail the basic calibration and then the refined calibration, as outlined above. These subsections refer to the case of a single camera, and so a final fourth subsection is used to describe the additional issues associated with the calibration of a stereo rig.

2.4.1 Estimation of a Scene-to-Image Planar Homography

A homography is a projective transformation (*projectivity*) that maps points to points and lines to lines. It is a highly useful imaging model when we view planar scenes, which is common in many computer vision processes, including the process of camera calibration.

Suppose that we view a planar scene, then we can define the (X, Y) axes of the world coordinate system to be within the plane of the scene, and hence $Z = 0$ everywhere. Equation 2.5 indicates that, as far as a planar scene is concerned, the imaging process can be reduced to

$$\lambda \mathbf{x} = \mathsf{K}[\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t}][X, Y, 1]^T,$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are the first and second columns of the rotation matrix R; hence,

$$\lambda \mathbf{x} = \mathsf{H}[X, Y, 1]^T, \quad \mathsf{H} = \mathsf{K}[\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t}]. \tag{2.7}$$

The 3×3 matrix H is termed a planar homography, which is defined up to a scale factor,[7] and hence has eight degrees of freedom instead of nine.

By expanding the above equation, we have

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}. \tag{2.8}$$

If we map homogeneous coordinates to inhomogeneous coordinates, by dividing through by λ, this gives

$$x = \frac{h_{11}X + h_{12}Y + h_{13}}{h_{31}X + h_{32}Y + h_{33}} \tag{2.9}$$

$$y = \frac{h_{21}X + h_{22}Y + h_{23}}{h_{31}X + h_{32}Y + h_{33}}. \tag{2.10}$$

[7] Due to the scale equivalence of homogeneous coordinates.

From a set of four correspondences in a general position,[8] we can formulate a set of eight linear equations in the eight unknowns of a homography matrix. This is because each correspondence provides a pair of constraints of the form given in Eqs. 2.9 and 2.10.

Rearranging terms in four pairs of those equations allows us to formulate the homography estimation problem in the form:

$$\mathsf{A}\mathbf{h} = \mathbf{0}, \tag{2.11}$$

where A is an 8×9 data matrix derived from image and world coordinates of corresponding points and $\mathbf{h}$ is the 9-vector containing the elements of the homography matrix. Since A has rank 8, it has a one-dimensional null space, which provides a non-trivial (non-zero vector) solution for Eq. 2.11. This can be determined from a Singular Value Decomposition (SVD) of the data matrix, which generates three matrices $(\mathsf{U}, \mathsf{D}, \mathsf{V})$ such that $\mathsf{A} = \mathsf{U}\mathsf{D}\mathsf{V}^T$. Here, D is a diagonal matrix of singular values and U, V are orthonormal matrices. Typically, SVD algorithms order the singular values in descending order down the diagonal of D and so the required solution, corresponding to a singular value of zero, is extracted as the last column of V. Due to the homogeneous form of Eq. 2.11, the solution is determined up to a non-zero scale factor, which is acceptable because H is only defined up to scale. Often a unit scale is chosen (i.e., $||\mathbf{h}|| = 1$), and this scaling is returned automatically in the columns of V.

In general, a larger number of correspondences than the minimum will not exactly satisfy the same homography because of image noise. In this case, a least squares solution to $\mathbf{h}$ can be determined in an over-determined system of linear equations. We follow the same procedure as above but this time the data matrix is of size $2n \times 9$ where $n > 4$ is the number of correspondences. When we apply SVD, we still select the last column of V corresponding to the smallest singular value in D. (Note that, in this case, the smallest singular value will be non-zero.)

Data normalization prior to the application of SVD is *essential* to give stable estimates [38]. The basic idea is to translate and scale both image and world coordinates to avoid orders of magnitude difference between the columns of the data matrix. Image points are translated so that their centroid is at the origin and scaled to give a root-mean-squared (RMS) distance of $\sqrt{2}$ from that origin, so that the "average" image point has coordinates of unity magnitude. Scene points should be normalized in a similar way except that they should be scaled to give an RMS distance of $\sqrt{3}$.

When using homogeneous coordinates, the normalizations can be applied using matrix operators N_i, N_s, such that new *normalized* coordinates are given as

$$\mathbf{x}_n = \mathsf{N}_\mathsf{i}\mathbf{x}, \ \mathbf{X}_n = \mathsf{N}_s\mathbf{X}$$

[8] No three points collinear.

for the image points and scene points, respectively. Suppose that the homography computed from normalized coordinates is $\tilde{\mathsf{H}}$, then the homography relating the original coordinates of the correspondences is given as

$$\mathsf{H} = \mathsf{N}_i^{-1}\tilde{\mathsf{H}}\mathsf{N}_s.$$

2.4.2 Basic Calibration

From the known planar scene target and the resulting image, a scene-to-image planar homography can be estimated as described in the previous subsection. Suppose that we describe such a homography as a set of 3×1 column vectors, i.e., $\mathsf{H} = [\mathbf{h}_1\ \mathbf{h}_2\ \mathbf{h}_3]$, then comparing this to Eq. 2.7 we have

$$\lambda_H \mathbf{h}_1 = \mathsf{K}\mathbf{r}_1,\ \lambda_H \mathbf{h}_2 = \mathsf{K}\mathbf{r}_2, \tag{2.12}$$

where λ_H is a scale factor, accounting for the particular scale of an estimated homography. Noting that the columns of the rotation matrix, $\mathbf{r}_1$, $\mathbf{r}_2$ are orthonormal,

$$\mathbf{r}_1^T\mathbf{r}_2 = \mathbf{h}_1^T\mathsf{K}^{-T}\mathsf{K}^{-1}\mathbf{h}_2 = 0, \tag{2.13}$$

$$\mathbf{r}_1^T\mathbf{r}_1 = \mathbf{r}_2^T\mathbf{r}_2 \Rightarrow \mathbf{h}_1^T\mathsf{K}^{-T}\mathsf{K}^{-1}\mathbf{h}_1 = \mathbf{h}_2^T\mathsf{K}^{-T}\mathsf{K}^{-1}\mathbf{h}_2. \tag{2.14}$$

These equations provide one constraint each on the intrinsic parameters.

We construct a symmetric matrix B such that

$$\mathsf{B} = \mathsf{K}^{-T}\mathsf{K}^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}.$$

Let the i^{th} column vector of H be $\mathbf{h}_i = [h_{1i}, h_{2i}, h_{3i}]^T$, we have

$$\mathbf{h}_i^T\mathsf{B}\mathbf{h}_j = \mathbf{v}_{ij}^T\mathbf{b},$$

where

$$\mathbf{v}_{ij} = [h_{1i}h_{1j},\ h_{1i}h_{2j} + h_{2i}h_{1j},\ h_{2i}h_{2j},\ h_{3i}h_{1j} + h_{1i}h_{3j},\ h_{3i}h_{2j} + h_{2i}h_{3j},\ h_{3i}h_{3j}]^T$$

and $\mathbf{b}$ is the vector containing six independent entries of the symmetric matrix B:

$$\mathbf{b} = [B_{11},\ B_{12},\ B_{22},\ B_{13},\ B_{23},\ B_{33}]^T.$$

Therefore, the two constraints in Eqs. 2.13 and 2.14 can be rewritten as

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^T \end{bmatrix} \mathbf{b} = \mathbf{0}.$$

If n images of the planar calibration grid are observed, n sets of these equations can be stacked into a matrix–vector equation as

$$\mathsf{V}\mathbf{b} = \mathbf{0},$$

where V is a $2n \times 6$ matrix. Although a minimum of three planar views allows us to solve for $\mathbf{b}$, it is recommended to take more and form a least squares solution. In this case, the solution for $\mathbf{b}$ is the eigenvector of $\mathsf{V}^T\mathsf{V}$ associated with the smallest eigenvalue. Once $\mathbf{b}$ is estimated, we know the matrix B up to some unknown scale factor, λ_B, and all of the intrinsic camera parameters can be computed by expanding the right-hand side of $\mathsf{B} = \lambda_B \mathsf{K}^{-T}\mathsf{K}^{-1}$ in terms of its individual elements. Although this is somewhat laborious, it is straightforward algebra of simultaneous equations, where five intrinsic camera parameters plus one unknown scale factor can be derived from the six parameters of the symmetric matrix B. Zhang [98] presents the solution:

$$\begin{aligned}
y_0 &= \frac{(B_{12}B_{13} - B_{11}B_{23})}{(B_{11}B_{22} - B_{12}^2)} \\
\lambda_B &= B_{33} - \frac{[B_{13}^2 + y_0(B_{12}B_{13} - B_{11}B_{23})]}{B_{11}} \\
\alpha_x &= \sqrt{\frac{\lambda_B}{B_{11}}} \\
\alpha_y &= \sqrt{\frac{\lambda_B B_{11}}{(B_{11}B_{22} - B_{12}^2)}} \\
s &= \frac{-B_{12}\alpha_x^2\alpha_y}{\lambda_B} \\
x_0 &= \frac{s y_0}{\alpha_y} - \frac{B_{13}\alpha_x^2}{\lambda_B}.
\end{aligned}$$

Once K is known, the extrinsic camera parameters for each image can be computed using Eq. 2.12:

$$\begin{aligned}
\mathbf{r}_1 &= \lambda_H \mathsf{K}^{-1}\mathbf{h}_1 \\
\mathbf{r}_2 &= \lambda_H \mathsf{K}^{-1}\mathbf{h}_2 \\
\mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \\
\mathbf{t} &= \lambda_H \mathsf{K}^{-1}\mathbf{h}_3,
\end{aligned}$$

where here

$$\lambda_H = \frac{1}{||\mathsf{K}^{-1}\mathbf{h}_1||} = \frac{1}{||\mathsf{K}^{-1}\mathbf{h}_2||}.$$

The vectors $\mathbf{r}_1, \mathbf{r}_2$ will not be exactly orthogonal and so the estimated rotation matrix does not exactly represent a rotation. Zhang [98] suggests performing SVD on the estimated rotation matrix so that $\mathsf{USV}^T = \mathsf{R}$. Then the closest pure rotation matrix in terms of Frobenius norm to that estimated is given as $\mathsf{R}' = \mathsf{UV}^T$.

2.4.3 Refined Calibration

After computation of the linear solution described above, it can be iteratively refined via a non-linear least squares minimization using the *Levenberg–Marquardt* (LM) algorithm. As previously mentioned, the camera parameters can be extended at this stage to include an estimation for the lens distortion parameters, to give us the following minimization:

$$\hat{\mathbf{p}} = \min_{\mathbf{p}} \left\{ \sum_{i=1}^{n} \sum_{j=1}^{m} ||\mathbf{x}_{i,j} - \hat{\mathbf{x}}_{i,j}(\mathsf{K}, k_1, k_2, \mathsf{R}_i, \mathbf{t}_i, \mathbf{X}_j)||^2 \right\},$$

where $\mathbf{x}_{i,j}$ is the image of world point $\mathbf{X}_j$ in image i and $\hat{\mathbf{x}}_{i,j}$ is the predicted projection of the same world point according to Eq. 2.7 (using estimated intrinsic and extrinsic camera parameters) followed by radial distortion according to Eq. 2.6.

The vector $\mathbf{p}$ contains all of the free parameters within the planar projection (homography) function plus two radial distortion parameters k_1 and k_2 as described in Sect. 2.3.3. Initial estimates of these radial distortion parameters can be set to zero. LM iteratively updates all parameters according to the equation:

$$\begin{aligned} \mathbf{p}_{k+1} &= \mathbf{p}_k + \delta\mathbf{p}_k \\ \delta\mathbf{p}_k &= -(\mathsf{J}^T\mathsf{J} + \lambda_J diag(\mathsf{J}^T\mathsf{J}))^{-1}\mathsf{J}^T\mathbf{e}, \end{aligned}$$

where J is the Jacobian matrix containing the first derivatives of the residual $\mathbf{e}$ with respect to each of the camera parameters.

Thus computation of the Jacobian is central to LM minimization. This can be done either numerically or with a custom routine, if analytical expressions for the Jacobian entries are known. In the numerical approach, each parameter is incremented and the function to be minimized (the least squares error function in this case) is computed and divided by the increment, which should be the maximum of 10^{-6} and $10^{-4} \times |p_i|$, where p_i is some current parameter value [38]. In the case of providing a custom Jacobian function, the expressions are long and complicated in the case of camera calibration, and so the use of a symbolic mathematics package can help reduce human error in constructing the partial differentials.

Note that there are LM implementations available on many platforms, for example, in MATLAB's optimization toolbox, or the C/C++ *levmar* package. A detailed discussion of iterative estimation methods including LM is given in Appendix 6 of Hartley and Zisserman's book [38].

2.4.4 Calibration of a Stereo Rig

As a mathematical convenience, it is common practice to choose the optical center of one camera to be the origin of a stereo camera's 3D coordinate system. Then, the relative rigid location of the other camera, $[\mathsf{R},\ \mathbf{t}]$, within this frame, along with both sets of intrinsic parameters, is required to generate a pair of projection matrices and hence a pair of 3D rays from corresponding image points that intersect at their common scene point.

The previous two subsections showed how we can calculate the intrinsic parameters for any single camera. If we have a stereo pair, which is our primary interest, then we would compute a pair of intrinsic parameter matrices, one for the left camera and one for the right. In most cases, the two cameras are the same model, and hence we would expect the two intrinsic parameter matrices to be very similar.

Also, we note that, for each chessboard position, two sets of extrinsic parameters, $[\mathsf{R},\ \mathbf{t}]$, are generated, one for the left camera's position relative to the calibration plane and one for the right. Clearly, each left–right pair of extrinsic parameters should have approximately[9] the same relationship, which is due to the fixed rigid rotation and translation of one camera relative to another in the stereo rig.

Once two sets of intrinsic parameters and *one* set of extrinsic parameters encoding the *relative* rigid pose of one camera relative to another has been computed, the results are often refined in a global stereo optimization procedure, again using the Levenberg–Marquardt approach. To reduce n sets of relative extrinsic parameters to one set, we could choose the set associated with the closest calibration plane or compute some form of average.

All parameter estimates, both intrinsic and extrinsic, can be improved if the optimization is now performed over a minimal set of parameters, since the extrinsic parameters are reduced from 12 (two rotations and two translations) to 6 (one relative rotation and one relative translation) per calibration grid location. This approach ensures global rigidity of the stereo rig going from left-to-right camera. An implementation of global stereo optimization to refine stereo camera parameters is given in the Caltech camera calibration toolbox for MATLAB [11].

[9] "Approximately", because of noise in the imaged corner positions supplied to the calibration process.

2.5 Two-View Geometry

3D reconstruction from an image pair must solve two problems: the correspondence problem and the reconstruction problem.

- *Correspondence problem.* For a point $\mathbf{x}$ in the left image, which is the corresponding point $\mathbf{x}'$ in the right image, where $\mathbf{x}$ and $\mathbf{x}'$ are images of the same physical scene point $\mathbf{X}$?
- *Reconstruction problem.* Given two corresponding points $\mathbf{x}$ and $\mathbf{x}'$, how do we compute the 3D coordinates of scene point $\mathbf{X}$?

Of these problems, the correspondence problem is significantly more difficult as it is a search problem whereas, for a stereo camera of known calibration, reconstruction to recover the 3D measurements is a simple geometric mechanism. Since we have sets of three unique points, $(\mathbf{x}, \mathbf{x}', \mathbf{X})$, this mechanism is called triangulation (not to be confused with surface mesh triangulation, described in Chap. 6).

This section is designed to give the reader a good general grounding in two-view geometry and estimation of the key two-view geometric relations that can be useful even when extrinsic or intrinsic camera calibration information is not available.[10] As long as the concept of epipolar geometry is well understood, the remaining main sections of this chapter can be easily followed.

2.5.1 Epipolar Geometry

Epipolar geometry establishes the relationship between two camera views. When we have calibrated cameras and we are dealing with metric image coordinates, it is dependent only on the relative pose between the cameras. When we have uncalibrated cameras and we are dealing with pixel-based image coordinates, it is additionally dependent on the cameras' intrinsic parameters; however, it is independent of the scene.

Once the epipolar geometry is known, for any image point in one image, we know that its corresponding point (its match) in the other image must lie on a line, which is known as the *epipolar line* associated with the original point. This epipolar constraint greatly reduces the correspondence problem from a 2D search over the whole image to a 1D search along the epipolar line only, and hence reduces computational cost and ambiguities.

The discussion here is limited to two-view geometry only. A similar constraint called the *trifocal tensor* is applicable for three views but is outside the scope of this chapter. For further information on the trifocal tensor and n-view geometries, please refer to [38].

[10] Extrinsic parameters are always not known in a structure from motion problem; they are part of what we are trying to solve for. Intrinsic parameters may or may not be known, depending on the application.

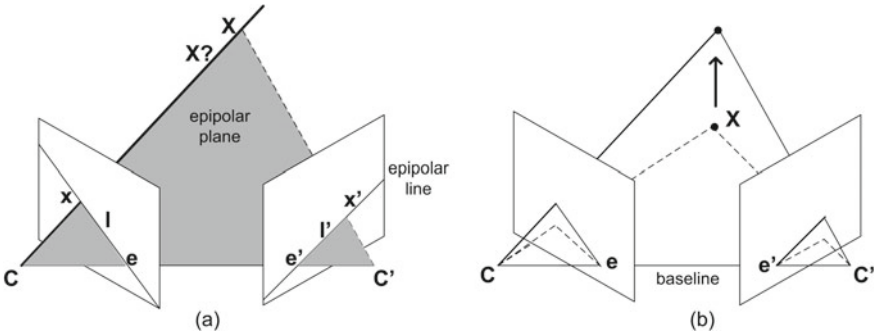

Fig. 2.8 **a** The epipolar geometry establishes the relationship between the two camera views. **b** The epipolar planes rotate around the baseline, and all epipolar lines intersect at the epipole

As shown in Fig. 2.8a, the image points $\mathbf{x}$ and $\mathbf{x}'$, world point $\mathbf{X}$, and the camera centers are co-planar, and this plane is called the epipolar plane, which is shaded in the figure. If we only know $\mathbf{x}$, how is the corresponding point $\mathbf{x}'$ constrained? The line $\mathbf{l}'$ is the intersection of the epipolar plane with the second image plane. $\mathbf{l}'$ is called the epipolar line, which is the image in the second view of the ray back-projected from $\mathbf{x}$. As the point $\mathbf{x}'$ lies on $\mathbf{l}'$, the correspondences search does not need to cover the entire image but can be restricted only to the line $\mathbf{l}'$. In fact, if any point on epipolar line $\mathbf{l}$ has a corresponding point in the second image, it must lie on epipolar line $\mathbf{l}'$ and vice versa. Thus $\mathbf{l}$ and $\mathbf{l}'$ are called *conjugate* epipolar lines.

The epipole is the point of intersection of the line joining the camera centers with the image plane. The epipole $\mathbf{e}$ is the projection of the second camera center on the first image, while the epipole $\mathbf{e}'$ is the projection of the first camera center on the second image.

In essence, two-view epipolar geometry describes the intersection of the image planes with the pencil of planes having the baseline as the pencil axis, as illustrated in Fig. 2.8b. Note that the baseline is the line joining the two camera centers.[11] All epipolar lines intersect at the epipole of the respective image to give a pencil of epipolar lines in each image. Note that the epipoles are not necessarily within the boundaries of the image. A special case is when the cameras are oriented in the same direction, and they are separated by a translation parallel to both image planes. In this case, the epipoles are at infinity and the epipolar lines are parallel. Furthermore, if the translation is in the X direction only and the cameras have the same intrinsic parameters, the conjugate epipolar lines lie on the same image rows. This is an ideal set up when we search for correspondences between the two images. However, we may prefer some camera vergence to improve the field-of-view overlap between the two cameras for certain distance and, in this case, the images need to be warped so that the epipolar lines become horizontal again. This *rectification* process is discussed later in the chapter.

[11] The length of the baseline is the magnitude of the extrinsic translation vector, $\mathbf{t}$.

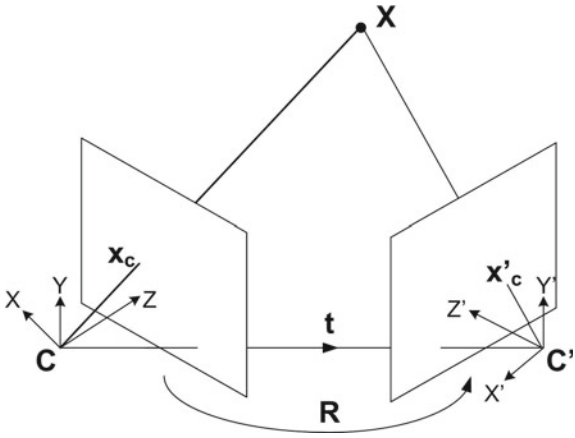

Fig. 2.9 The essential matrix $\mathsf{E} = [\mathbf{t}]_x\mathsf{R}$ encodes the epipolar geometry. It is used to relate the correspondences $\mathbf{x_c}$ and $\mathbf{x_c}'$ between two images, when these image locations are expressed in metric units. If pixel-based coordinates are used (for example, if intrinsic camera parameters are unknown) epipolar geometry is encoded by the fundamental matrix F

The epipolar constraint can be represented algebraically by a 3×3 matrix called the *fundamental matrix* (F), when we are dealing with raw pixel coordinates, and by the *essential matrix* (E) when the intrinsic parameters of the cameras are known and we are dealing with metrically expressed coordinates (e.g., millimeters) in the image plane.

2.5.2 Essential and Fundamental Matrices

Both the essential and fundamental matrices derive from a simple co-planarity constraint. For simplicity it is best to look at the epipolar relation using the essential matrix first and then adapt it using the camera intrinsic parameters to obtain a relation for pixel-based image coordinates, which involves the fundamental matrix.

Referring to Fig. 2.9, we have a world point $\mathbf{X}$ that projects to points $\mathbf{x}_c$ and $\mathbf{x}'_c$ in the image planes. These image plane points are expressed as 3-vectors, so that they are effectively the 3D positions of the imaged points expressed metrically in their own camera frame, hence the subscript c. (Note also that they can be regarded as normalized homogeneous image coordinates, with the scale set to the focal length, f, although any non-zero scale would suffice.) We know that the three vectors $\mathbf{C}\mathbf{x}_c$, $\mathbf{C}'\mathbf{x}'_c$ and $\mathbf{t}$ are co-planar, so we can choose one of the two camera frames to express this co-planarity, using the scalar triple product. If we choose the right frame (primed), then we must rotate vector $\mathbf{C}\mathbf{x}_c$ using rotation matrix R, to give

$$\mathbf{x}_c'^T(\mathbf{t} \times \mathsf{R}\mathbf{x}_c) = 0.$$

Expressing the cross-product with $\mathbf{t}$ by the multiplication with the skew-symmetric matrix $[\mathbf{t}]_x$, we have

$$\mathbf{x}'_c[\mathbf{t}]_x\mathsf{R}\mathbf{x}_c = 0,$$

where

$$[\mathbf{t}]_x = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

and thus we have

$$\mathsf{E} = [\mathbf{t}]_x \mathsf{R} \tag{2.15}$$

and

$$\mathbf{x}_c'^T \mathsf{E} \mathbf{x}_c = 0. \tag{2.16}$$

Thus the essential matrix encapsulates only extrinsic parameters, namely, the rotation and translation associated with the *relative* pose of the two cameras. The implication of this is that, in applications where R and $\mathbf{t}$ have not been computed in a calibration procedure, they may be recoverable from E, which will be discussed further in Sect. 2.8.2 in the context of structure from motion.

In many practical situations, we also need to deal with uncalibrated cameras where the intrinsic parameters are unknown (i.e., the mapping between metric image coordinates and raw pixel values is unknown). The shifting and scaling operations required for this conversion can be encapsulated in matrices K and K', as follows:

$$\mathbf{x} = \mathsf{K}\mathbf{x}_c, \quad \mathbf{x}' = \mathsf{K}'\mathbf{x}_c',$$

where K and K' are the 3×3 matrices containing the intrinsic camera parameters for the two cameras. Inserting these relations into Eq. 2.16 gives

$$\mathbf{x}'^T \mathsf{K}'^{-T} \mathsf{E} \mathsf{K}^{-1} \mathbf{x} = 0$$

$$\mathbf{x}'^T \mathsf{F} \mathbf{x} = 0$$

thus

$$\mathsf{F} = \mathsf{K}'^{-T} \mathsf{E} \mathsf{K}^{-1} = \mathsf{K}'^{-T} [\mathbf{t}]_x \mathsf{R} \mathsf{K}^{-1}$$

and we can see that the fundamental matrix encapsulates both intrinsic and extrinsic parameters. The interpretation of the epipolar constraint given by the fundamental matrix is that, if points $\mathbf{x}$ and $\mathbf{x}'$ correspond, then $\mathbf{x}'$ must lie on the epipolar line given by $\mathbf{l}' = \mathsf{F}\mathbf{x}$ and therefore the dot product between $\mathbf{x}'$ and $\mathsf{F}\mathbf{x}$ is zero.

Some key properties of the fundamental matrix are summarized below:

- If F is the fundamental matrix between camera P and camera P', then F^T is the fundamental matrix between camera P' and camera P.
- F is a projective mapping taking a point to a line. If $\mathbf{l}$ and $\mathbf{l}'$ are corresponding (i.e., conjugate) epipolar lines, then any point $\mathbf{x}$ on $\mathbf{l}$ maps to the same line $\mathbf{l}'$. Hence, there is no inverse mapping (zero determinant, rank 2).

- F has seven degrees of freedom. While a 3×3 homogeneous matrix has eight independent ratios, there is also an additional constraint that the determinant of F is zero (F is rank 2), which further removes one degree of freedom.
- For any point $\mathbf{x}$ in the first image, the corresponding epipolar line in the second image is $\mathbf{l}' = \mathsf{F}\mathbf{x}$. Similarly, $\mathbf{l} = \mathsf{F}^T\mathbf{x}'$ represents the epipolar line in the first image corresponding to $\mathbf{x}'$ in the second image.
- The epipoles are determined as the left and right nullspaces of the fundamental matrix. This is evident, since each epipole is on every epipolar line in their respective image. This is written as $\mathbf{e}'^T\mathbf{l}' = \mathbf{e}'^T\mathsf{F}\mathbf{x} = 0 \ \ \forall\mathbf{x}$, hence $\mathbf{e}'^T\mathsf{F} = 0$. Similarly $\mathbf{l}^T\mathbf{e} = \mathbf{x}'^T\mathsf{F}\mathbf{e} = 0 \ \ \forall\mathbf{x}'$, hence $\mathsf{F}\mathbf{e} = 0$.
- The SVD (*Singular Value Decomposition*) of F is given as

$$\mathsf{F} = \mathsf{U}\, diag(\sigma_1, \sigma_2, 0)\, \mathsf{V}^T,$$

  where $\mathsf{U} = [\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}']$, $\mathsf{V} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{e}]$. Thus finding the column in V that corresponds to the zero singular value gives a simple method of computation of the epipoles from the fundamental matrix.
- For cameras with some vergence (epipoles not at infinity) to give camera projection matrices: $\mathsf{P} = \mathsf{K}[\mathsf{I}|0]$ and $\mathsf{P}' = \mathsf{K}'[\mathsf{R}|\mathbf{t}]$, then we have $\mathsf{F} = \mathsf{K}'^{-T}[\mathbf{t}]_\times\mathsf{R}\mathsf{K}^{-1} = [\mathsf{K}'\mathbf{t}]_\times\mathsf{K}'\mathsf{R}\mathsf{K}^{-1} = \mathsf{K}'^{-T}\mathsf{R}\mathsf{K}^T[\mathsf{K}\mathsf{R}^T\mathbf{t}]_\times$ [38].

2.5.3 The Fundamental Matrix for Pure Translation

If the two identical cameras ($\mathsf{K} = \mathsf{K}'$) are separated by a pure translation ($\mathsf{R} = \mathsf{I}$), the fundamental matrix has a simple form, which can be shown to be [38]

$$\mathsf{F} = [\mathsf{K}\mathbf{t}]_x = [\mathbf{e}']_x = \begin{bmatrix} 0 & -e'_z & e'_y \\ e'_z & 0 & -e'_x \\ -e'_y & e'_x & 0 \end{bmatrix}.$$

In this case, the epipoles are at the same location in both images. If the translation is parallel to the image plane, the epipoles are at infinity with $e_z = e'_z = 0$ and the epipolar lines are parallel in both images. When discussing rectilinear stereo rigs and rectification later, we will be particularly interested in the case when the translation is parallel to the camera's x-axis, in which case the epipolar lines are parallel and horizontal and thus correspond to image scan (raster) lines. In this case $\mathbf{e}' = [1, 0, 0]^T$ and the fundamental matrix is

$$\mathsf{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

and hence the relationship between corresponding points $\mathbf{x}$ and $\mathbf{x}'$ is given by $\mathbf{x}'^T \mathsf{F}\mathbf{x} = 0$ which reduces to $y = y'$.

2.5.4 Computation of the Fundamental Matrix

As the fundamental matrix is expressed in terms of corresponding image points, F can be computed from image correspondences alone. No camera calibration information is needed, and pixel coordinates are used directly. Note that there are degenerate cases in the estimation of F. These occur in two common and well-known instances: (i) when the relative pose between the two views can be described by a pure rotation and (ii) when the scene is planar. For now we consider scenarios where such degeneracies do not occur and we return to them later.

By expanding $\mathbf{x}'^T \mathsf{F}\mathbf{x} = 0$ where $\mathbf{x} = [x, y, 1]^T$ and $\mathbf{x}' = [x', y', 1]^T$ and

$$\mathsf{F} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}$$

we obtain

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0.$$

As each feature correspondence provides one equation, for n correspondences, we get the following set of linear equations:

$$\begin{bmatrix} x'_1x_1, & x'_1y_1, & x'_1, & y'_1x_1, & y'_1y_1, & y'_1, & x_1, & y_1, & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n, & x'_ny_n, & x'_n, & y'_nx_n, & y'_ny_n, & y'_n, & x_n, & y_n, & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = \mathbf{0} \qquad (2.17)$$

or more compactly,

$$\mathsf{A}\mathbf{f} = \mathbf{0},$$

where A is termed the data matrix and $\mathbf{f}$ is the vector of unknown elements of F.

The eight-point algorithm[12] [47] can be used as a very simple method to solve for F linearly using eight correspondences. As this is a homogeneous set of equations, $\mathbf{f}$ can only be determined up to a scale factor. With eight correspondences, Eq. 2.17 can be solved by linear methods, where the solution is the nullspace of A. (This can be found from the column in V that corresponds to the *zero* singular value in D in the singular value decomposition $\mathsf{A} = \mathsf{UDV}^T$.) However, a solution with a minimal set of correspondences is often inaccurate, particularly if the correspondences are not well spread over the images, or they may not provide enough strong constraints if some of them are near-collinear or co-planar. It is preferable to use more than eight correspondences, then the least squares solution for $\mathbf{f}$ is given by the singular vector corresponding to the *smallest* singular value of A.

Note that this approach is similar to that for determining the homography matrix, discussed earlier in Sect. 2.4.1. As with that approach, it is essential to normalize the pixel coordinates of each image before applying SVD [38, 39], using a mean-centering translation and a scaling so that the RMS distance of the points to the origin is $\sqrt{2}$. When using homogeneous coordinates, this normalization can be applied using matrix operators N, N', such that new *normalized* image coordinates are given as $\mathbf{x}_n = \mathsf{N}\mathbf{x}$, $\mathbf{x}'_n = \mathsf{N}'\mathbf{x}'$.

In general, the solution for F_n (the subscript n now denotes that we have based the estimate on normalized image coordinates) will not have zero determinant (its rank will be 3 and not 2), which means that the epipolar lines will not intersect at a single point. In order to enforce this, we can apply SVD a second time, this time to the initially estimated fundamental matrix so that $\mathsf{F}_n = \mathsf{UDV}^T$. We then set the smallest singular value (in the third row and third column of D) to zero to produce matrix D' and update the estimate of the fundamental matrix as $\mathsf{F}_n = \mathsf{UD}'\mathsf{V}^T$.

Of course, the estimate of F_n maps points to epipolar lines in the normalized image space. If we wish to search for correspondences within the original image space, we need to de-normalize the fundamental matrix estimate as $\mathsf{F} = \mathsf{N}'^T\mathsf{F}_n\mathsf{N}$.

Typically, there are many correspondences between a pair of images, including mostly inliers but also some outliers. This is inevitable, since matching is a local search and ambiguous matches exist, which will be discussed further in Sect. 2.7. Various robust methods for estimating the fundamental matrix, which address the highly corrupting effect of outliers, are compared in [86]. In order to compute F from these correspondences automatically, a common method is to use a robust statistics technique called Random Sample Consensus (RANSAC) [32], which we now outline:

1. Extract features in both images, for example, from a corner detector [37].
2. Perform feature matching between images (usually over a local area neighborhood) to obtain a set of potential matches or *putative correspondences*.
3. Repeat the following steps N times:
   - Select eight putative correspondences randomly.
   - Compute F using these eight points, as described above.

[12] There are several other approaches, such as the seven-point algorithm.

Table 2.1 Number of samples required to get at least one good sample with 99% probability for various sample sizes s and outlier fraction ϵ.

Sample size s	$\epsilon = 10\%$	$\epsilon = 20\%$	$\epsilon = 30\%$	$\epsilon = 40\%$	$\epsilon = 50\%$
4	5	9	17	34	72
5	6	12	26	57	146
6	7	16	37	97	293
7	8	20	54	163	588
8	9	26	78	272	1177

- Find the number of inliers[13] that support F.

4. Find the F with the highest number of inliers (largest support) among the N trials.
5. Use this F to look for additional matches outside the search range used for the original set of putative correspondences.
6. Re-compute a least squares estimate of F using all inliers.

Note that re-computing F in the final step may change the set of inliers, as the epipolar lines are adjusted. Thus, a possible refinement is to iterate computation of a linear least squares estimate of F and its inliers, until a stable set of inliers is achieved or some maximum number of iterations is reached. The refinement achieved is often considered to be not worth the additional computational expense if processing time is considered important or if the estimate of F is to be used as the starting point for more advanced iterative non-linear refinement techniques, described later.

In the RANSAC approach, N is the number of trials (putative F computations) needed to get at least one good sample with a high probability (e.g., 99%). How large should N be? The probability p of getting a good sample is given by

$$p = 1 - (1 - (1 - \epsilon)^s)^N,$$

where ϵ is the fraction of outliers (incorrect feature correspondences) and s is the number of correspondences selected for each trial. The above equation can be re-arranged as

$$N = \frac{log(1 - p)}{log(1 - (1 - \epsilon)^s)}. \tag{2.18}$$

The number of samples required for various sample sizes and outlier fractions based on Eq. 2.18 are shown in Table 2.1. It can be seen that the number of samples gets higher as the outlier fraction increases.

By repeatedly selecting a group of correspondences, the inlier support would be high for a correct hypothesis in which all the correspondences within the sample

[13] An inlier is a putative correspondence that lies within some threshold of its expected position predicted by F. In other words, image points must lie within a threshold from their epipolar lines generated by F.

size, s, are correct. This allows the robust removal of outliers and the computation of F using inliers only. As the fraction of outliers may not be known in advance, an adaptive RANSAC method can be used where the number of outliers at each iteration is used to re-compute the total number of iterations required.

As the fundamental matrix has only seven degrees of freedom, a minimum of seven correspondences are required to compute F. When there are only seven correspondences, $det(\mathsf{F}) = 0$ constraint also needs to be imposed, resulting in a cubic equation to solve, and hence may produce up to three solutions and all three must be tested for support. The advantage of using seven correspondences is that fewer trials are required to achieve the same probability of getting a good sample, as illustrated in Table 2.1.

Fundamental matrix refinement techniques are often based on the Levenberg–Marquardt algorithm, such that some non-linear cost function is minimized. For example, a geometric cost function can be formulated as the sum of the squared distances between image points and the epipolar lines generated from their associated corresponding points and the estimate of F. This is averaged over both points in a correspondence and over all corresponding points (i.e., all those that agree with the estimate of F). The minimization can be expressed as

$$\mathsf{F} = \min_{\mathsf{F}} \left(\frac{1}{N} \Sigma_{i=1}^{N} \left(d(\mathbf{x}_i', \ \mathsf{F}\mathbf{x}_i)^2 + d(\mathbf{x}_i, \ \mathsf{F}^T \mathbf{x}_i')^2 \right) \right),$$

where $d(\mathbf{x}, \mathbf{l})$ is the distance of a point $\mathbf{x}$ to a line $\mathbf{l}$, expressed in pixels. For more details of this and other non-linear refinement schemes, the reader is referred to [38].

2.5.5 Two Views Separated by a Pure Rotation

If two views are separated by a pure rotation around the camera center, the baseline is zero, the epipolar plane is not defined, and a useful fundamental matrix cannot be computed. In this case, the back-projected rays from each camera cannot form a triangulation to compute depth. This lack of depth information is intuitive because, under rotation, all points in the same direction move across the image in the same way, regardless of their depth. Furthermore, if the translation magnitude is small, the epipolar geometry is close to this degeneracy and computation of the fundamental matrix will be highly unstable.

In order to model the geometry of correspondences between two rotated views, a *homography*, described by a 3×3 matrix H, should be estimated instead. As described earlier, a homography is a projective transformation (*projectivity*) that maps points to points and lines to lines. For two identical cameras ($\mathsf{K} = \mathsf{K}'$), the scene-to-image projections are

$$\mathbf{x} = \mathsf{K}[\mathsf{I}|\mathbf{0}]\mathbf{X}, \quad \mathbf{x}' = \mathsf{K}[\mathsf{R}|\mathbf{0}]\mathbf{X}$$

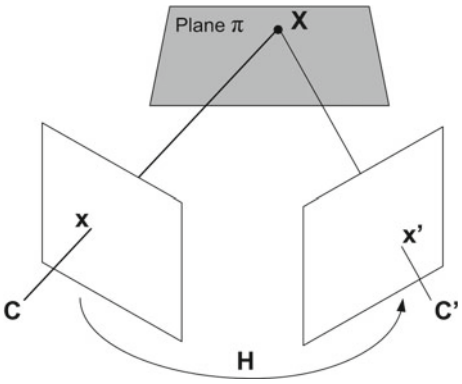

Fig. 2.10 The homography induced by a plane $\boldsymbol{\pi}$, where a point $\mathbf{x}$ in the first image can be transferred to the point $\mathbf{x}'$ in the second image

hence

$$\mathbf{x}' = \mathsf{KRK}^{-1}\mathbf{x} = \mathsf{H}\mathbf{x}. \tag{2.19}$$

We can think of this homography as a mapping of image coordinates onto normalized coordinates (centered on the principal point at a unit metric distance from the camera). These points are rotated and then multiplying by K generates the image coordinates on the focal plane of the second, rotated camera.

2.5.6 *Two Views of a Planar Scene*

A homography should also be estimated for planar scenes where correspondences cannot uniquely define the epipolar geometry and hence the fundamental matrix. Similar to Eq. 2.7, the *2D-to-2D* projection of the world plane $\boldsymbol{\pi}$ in Fig. 2.10 to the left and right images is given by

$$\lambda_x \mathbf{x} = \mathsf{H}_x \mathbf{X}, \qquad \lambda_{x'} \mathbf{x}' = \mathsf{H}_{x'} \mathbf{X},$$

where H_x, $\mathsf{H}_{x'}$ are 3×3 homography matrices (homographies) and $\mathbf{x}$, $\mathbf{x}'$ are homogeneous image coordinates. The planar homographies form a group, and hence we can form a composite homography as $\mathsf{H} = \mathsf{H}_{x'}\mathsf{H}_x^{-1}$ and it is straightforward to show that

$$\lambda \mathbf{x}' = \mathsf{H}\mathbf{x}.$$

Figure 2.10 illustrates this mapping from $\mathbf{x}$ to $\mathbf{x}'$ and we say that a homography is induced by the plane $\boldsymbol{\pi}$. Homography estimation follows the same approach as was described in Sect. 2.4.1 for a scene-to-image planar homography (replacing $\mathbf{X}$ with $\mathbf{x}$ and $\mathbf{x}$ with $\mathbf{x}'$ in Eqs. 2.8 to 2.10).

Note that a minimum of four correspondences (no three points collinear in either image) are required because, for the homography, each correspondence generates a pair of constraints. Larger numbers of correspondences allow a least squares solution to an over-determined system of linear equations. Again suitable normalizations are required before SVD is applied to determine the homography.

A RANSAC-based technique can also be used to handle outliers, similar to the fundamental matrix estimation method described in Sect. 2.5.4. By repeatedly selecting the minimal set of four correspondences randomly to compute H and counting the number of inliers, the H with the largest number of inliers can be chosen. Additional matches that are not in the original set of putative correspondences can be obtained using the best H. Then, H can be re-computed using all supporting matches in a linear least squares minimization using SVD.

Finally, we note that, as in the case of the fundamental matrix, a non-linear optimization can be applied to refine the homography solution, if required by the application. The interested reader is referred to [38] for the details of the geometric cost function to be minimized.

2.6 Rectification

Typically, in a stereo rig, the cameras are horizontally displaced and rotated toward each other by an equal amount (verged), in order to overlap their fields of view. In this case, epipolar lines lie at a variety of angles across the two images, complicating the search for correspondences. In contrast, if these cameras had their principal axes parallel to each other (no vergence) and the two cameras had identical intrinsic parameters, conjugate (corresponding) epipolar lines would lie along the same horizontal scanline in each image, as observed in Sect. 2.5.3. This configuration is known as a standard rectilinear stereo rig. Clearly, it is desirable to retain the improved stereo viewing volume associated with verged cameras and yet have the simplicity of correspondence search associated with a rectilinear rig.

To achieve this, we can warp or *rectify* the raw images associated with the verged system such that corresponding epipolar lines become collinear and lie on the same scanline. A second advantage is that the equations for 3D reconstruction are very simply related to image disparity after image rectification, since they correspond to those of a simple rectilinear stereo rig. This triangulation computation is described later in the chapter.

Rectification can be achieved either with camera calibration information, for example, in a typical stereo application, or without calibration information, for example, in a typical structure from motion application. We discuss the calibrated case in the following subsection and give a brief mention of uncalibrated approaches in Sect. 2.6.2.

2.6.1 Rectification with Calibration Information

Here, we assume a calibrated stereo rig, where we know both the intrinsic and the extrinsic parameters. Knowing this calibration information gives a simple rectification approach, where we find an image mapping that generates, from the original images, a pair of images that would have been obtained from a rectilinear rig. Of course, the field of view of each image is still bound by the real original cameras, and so the rectified images tend to be a different shape than the originals (e.g., slightly trapezoidal in a verged stereo rig).

Depending on the lenses used and the required accuracy of the application, it may be considered necessary to correct for radial distortion, using estimated parameters k_1 and k_2 from the calibration. To do the correction, we employ Eq. 2.6 in order to compute the unknown, undistorted pixel coordinates, $[x, y]^T$, from the known distorted coordinates, $[x_d, y_d]^T$. There are various ways to solve this non-linear equation, such as approximation of the Taylor series with the first-order derivatives, a look-up table, or an iterative solution where the distorted pixel coordinates can be used as the initial estimate.

Assuming some vergence, we wish to map the image points onto a pair of (virtual) image planes that are parallel to the baseline and in the same plane. Thus we can use the homography structure in Eq. 2.19 that warps images between a pair of rotated views. Given that we already know the intrinsic camera parameters, we need to determine the rotation matrices associated with the rectification of the left and right views. We will assume that the origin of the stereo system is at the optical center of the left camera and calibration information gives $[\mathsf{R},\ \mathbf{t}]$ to define the rigid position of the right camera relative to this. To get the rotation matrix that we need to apply to image points of the left camera, we define the rectifying rotation matrix as

$$\mathsf{R}_{rect} = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{bmatrix},$$

where $\mathbf{r}_i,\ i = 1 \ldots 3$ are a set of mutually orthogonal unit vectors. The first of these is in the direction of the epipole or, equivalently, the direction of the translation to the right camera, $\mathbf{t}$. (This ensures that epipolar lines will be horizontal in the rectified image.) Hence, the unit vector that we require is

$$\mathbf{r}_1 = \frac{\mathbf{t}}{||\mathbf{t}||}.$$

The second vector $\mathbf{r}_2$ is orthogonal to the first and obtained as the cross-product of $\mathbf{t}$ and the original left optical axis $[0,\ 0,\ 1]^T$ followed by a normalization to unit length to give

$$\mathbf{r}_2 = \frac{1}{\sqrt{t_x^2 + t_y^2}}[-t_y,\ t_x,\ 0]^T.$$

The third vector is mutually orthogonal to the first two and so is computed using the cross-product as $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$.

Given that the real right camera is rotated relative to the real left camera, we need to apply a rotation $\mathsf{R}\mathsf{R}_{rect}$ to the image points of the right camera. Hence, applying homographies to left and right image points, using the form of Eq. 2.19, we have

$$\mathbf{x}_{rect} = \mathsf{K}\mathsf{R}_{rect}\mathsf{K}^{-1}\mathbf{x}$$

$$\mathbf{x}'_{rect} = \mathsf{K}'\mathsf{R}\mathsf{R}_{rect}\mathsf{K}'^{-1}\mathbf{x}',$$

where K and K' are the 3×3 matrices containing the intrinsic camera parameters for the left and right cameras, respectively. Note that, even with the same make and model of camera, we may find that the focal lengths associated with K and K' are slightly different. Thus we need to scale one rectified image by the ratio of focal lengths in order to place them on the same focal plane.

As the rectified coordinates are, in general, not integer, *resampling* using some form of interpolation is required. The rectification is often implemented in reverse, so that the pixel values in the new image plane can be computed as a bilinear interpolation of the four closest pixels values in the old image plane. Rectified images give a very simple triangulation reconstruction procedure, which is described later in Sect. 2.8.1.2.

2.6.2 Rectification Without Calibration Information

When calibration information is not available, rectification can be achieved using an estimate of the fundamental matrix, which is computed from correspondences within the raw image data. A common approach is to compute a pair of rectifying homographies for the left and right images [38, 54] so that the fundamental matrix associated with the rectified images is the same form as that for a standard rectilinear rig and the "new cameras" have the same intrinsic camera parameters. Since such rectifying homographies map the epipoles to infinity ($[1, 0, 0]^T$), this approach fails when the epipole lies within the image. This situation is common in structure from motion problems, when the camera translates in the direction of its Z-axis. Several authors have tackled this problem by directly resampling the original images along their epipolar lines, which are specified by an estimated fundamental matrix. For example, the image is reparameterized using polar coordinates around the epipoles to reduce the search ambiguity to half epipolar lines [67, 68]. Figure 2.11 shows an example of an image pair before and after rectification for this scheme, where the corresponding left and right features lie on the same image row afterward. Specialized rectifications exist, for example, [17] that allow image matching over large forward translations of the camera although, in this scheme, rotations are not catered for.

Fig. 2.11 An image pair before rectification (**a**) and after rectification (**b**). The overlay shows that the corresponding left and right features lie on the same image row after rectification. Figure courtesy of [68]

2.7 Finding Correspondences

Finding correspondences is an essential step for 3D reconstruction from multiple views. The correspondence problem can be viewed as a search problem, which asks, given a pixel in the left image, which is the corresponding pixel in the right image? Of course there is something of a circular dependency here. We need to find correspondences to determine the epipolar geometry, yet we need the epipolar geometry to find (denser) correspondences in an efficient manner. The RANSAC sampling approach described earlier showed us how to break into this loop. Once we have the epipolar geometry constraint, the search space is reduced from a 2D search to the epipolar line only.

The following assumptions underpin most methods for finding correspondences in image pairs. These assumptions hold when the distance of the world point from the cameras is much larger than the baseline.

- Most scene points are visible from both viewpoints.
- Corresponding image regions are similar.

Two questions are involved: what is a suitable image element to match and what is a good similarity measure to adopt? There are two main classes of correspondence algorithms: correlation-based and feature-based methods. Correlation-based methods recover dense correspondences where the element to match is an image window centered on some pixel and the similarity measure is the correlation between the windows. Feature-based methods typically establish sparse correspondences where the element to match is an image feature and the similarity measure is the distance between descriptors of the image features.

2.7.1 Correlation-Based Methods

If the element to match is only a single image pixel, ambiguous matches exist. Therefore, windows are used for matching in correlation-based methods, and the similarity criterion is a measure of the correlation between the two windows. A larger

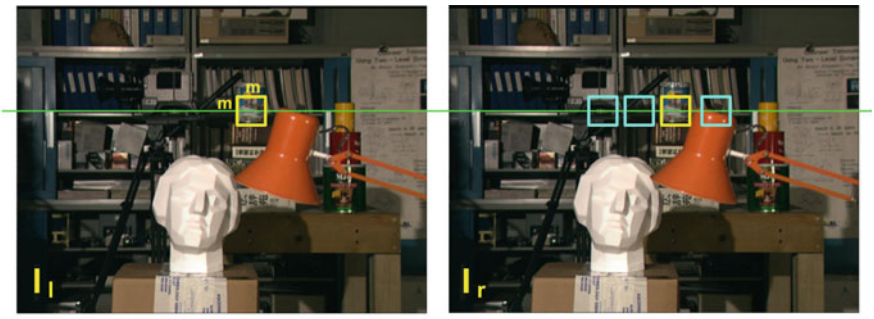

Fig. 2.12 Correlation-based methods look for the matching image window between the left and right rectified images. An m by m window centering at the pixel is used for correlation. (Raw image pair courtesy of the Middlebury Stereo Vision Page [74], originally sourced from Tsukuba University.)

window gives larger image context which can reduce the probability of ambiguities, but this has its own problems which will be discussed in Sect. 2.8.1.1. The selected correspondence is given by the window that maximizes a similarity criterion or minimizes a dissimilarity criterion within a search range. Once a match is found, the offset between the two windows can be computed, which is called the *disparity* from which the depth can be recovered. Some commonly used criteria for correlation-based methods are described next.

Based on the rectified images in Fig. 2.12, we define the window function, where m, an odd integer, is the image window size so that

$$W_m(x, y) = \{(u, v) | x - \frac{(m-1)}{2} \leq u \leq x + \frac{(m-1)}{2}, y - \frac{(m-1)}{2} \leq v \leq y + \frac{(m-1)}{2}\}. \tag{2.20}$$

The dissimilarity can be measured by the *Sum of Squared Differences* (SSD) cost, for instance, which is the intensity difference as a function of disparity d:

$$SSD(x, y, d) = \sum_{(u,v) \in W_m(x,y)} [I_l(u, v) - I_r(u - d, v)]^2,$$

where I_l and I_r refer to the intensities of the left and right images, respectively.

If two image windows correspond to the same world object, the pixel values of the windows should be similar, and hence the SSD value would be relatively small. As shown in Fig. 2.12, for each pixel in the left image, correlation-based methods would compare the SSD measure for pixels within a search range along the corresponding epipolar line in the right image. The disparity value that gives the lowest SSD value indicates the best match.

A slight variation of SSD is the *Sum of Absolute Differences* (SAD) where the absolute values of the differences are added instead of the squared values:

$$SAD(x, y, d) = \sum_{(u,v)\in W_m(x,y)} |I_l(u, v) - I_r(u - d, v)|.$$

This cost measure is less computationally expensive as it avoids the multiplication operation required for SSD. On the other hand, the SSD cost function penalizes the large intensity difference more due to the squaring operation.

The intensities between the two image windows may vary due to illumination changes and non-Lambertian reflection. Even if the two images are captured at the same time by two cameras with identical models, non-Lambertian reflection and differences in the gain and sensitivity can cause variation in the intensity. In these cases, SSD or SAD may not give a low value even for the correct matches. For these reasons, it is a good idea to normalize the pixels in each window. A first level of normalization would be to ensure that the intensities in each window are zero-mean. A second level of normalization would be to scale the zero-mean intensities so that they either have the same range or, preferably, unit variance. This can be achieved by dividing each pixel intensity by the standard deviation of window pixel intensities, after the zero-mean operation, i.e., normalized pixel intensities are given as

$$I_n = \frac{I - \bar{I}}{\sigma_I},$$

where $\bar{I}$ is the mean intensity and σ_I is the standard deviation of window intensities. While SSD measures the dissimilarity and hence the smaller the better, *Normalized Cross-Correlation* (NCC) measures the similarity and hence, the larger the better. Again, the pixel values in the image window are normalized first by subtracting the average intensity of the window so that only the relative variation would be correlated. The NCC measure is computed as follows:

$$NCC(x, y, d) = \frac{\sum_{(u,v)\in W_m(x,y)} (I_l(u, v) - \overline{I_l})(I_r(u - d, v) - \overline{I_r})}{\sqrt{\sum_{(u,v)\in W_m(x,y)} (I_l(u, v) - \overline{I_l})^2 (I_r(u - d, v) - \overline{I_r})^2}},$$

where

$$\overline{I_l} = \frac{1}{m^2} \sum_{(u,v)\in W_m(x,y)} I_l(u, v), \quad \overline{I_r} = \frac{1}{m^2} \sum_{(u,v)\in W_m(x,y)} I_r(u, v).$$

2.7.2 Feature-Based Methods

Rather than matching each pixel, feature-based methods only search for correspondences to a sparse set of features, such as those located by a repeatable, well-localized interest point detector (e.g., a corner detector). Apart from locating the features, feature extraction algorithms also compute some sort of feature descriptors for their representation, which can be used for the similarity criterion. The correct correspon-

dence is given by the most similar feature pair, the one with the minimum distance between the feature descriptors.

Stable features are preferred in feature-based methods to facilitate matching between images. Typical examples of image features are edge points, lines, and corners. For example, a feature descriptor for a line could contain the length, the orientation, coordinates of the mid-point, or the average contrast along the edge line. A problem with linear features is that the matching can be poorly localized along the length of a line particularly if a linear feature is fragmented (imagine a smaller fragment from the left image sliding along a larger fragment from the right image). This is known as the aperture problem, referring to the fact that a *local* match "looks through" a small aperture.

As a consequence, point-based features that are well localized in two mutually orthogonal directions have been preferred by researchers and practitioners in the field of computer vision. For example, the Harris corner detector [37] extracts points that differ as much as possible from neighboring points. This is achieved by looking for high curvatures in two mutually orthogonal directions, as the gradient is ill-defined in the neighborhood of corners. The corner strength or the grayscale values in a window region around each corner could be used as the descriptor. Another corner detector SUSAN [81] detects features based on the size, centroid, and second moments of the local areas. As it does not compute image derivatives, it is robust to noise and does not require image smoothing.

Wide baseline matching refers to the situation where the two camera views differ considerably. Here, matching has to operate successfully over more difficult conditions, since there are larger geometric and photometric variations between the images.

In recent years, many interest point detection algorithms have been proposed that are scale-invariant and viewpoint invariant to a certain extent which facilitates wide baseline matching. An interest point refers to an image feature that is stable under local and global perturbations, and the local image structure is rich in terms of local image contents. These features are often described by a distinctive feature descriptor which is used as the similarity criterion. They can be used even when epipolar geometry is not yet known, as such distinctive descriptors allow correspondences to be searched over the whole image relatively efficiently.

For example, the *Scale-Invariant Feature Transform* (SIFT) [48] and the *Speeded-Up Robust Feature* (SURF) [5] are two traditional features which were developed for image feature generation in object recognition applications. The SIFT feature is described by a local image vector with 128 elements, which is invariant to image translation, scaling, rotation, and partially invariant to illumination changes and affine or 3D projections. Fig. 2.13 shows an example of matching SIFT features across large baseline and viewpoint variation. It can be seen that most matches are correct, thanks to the invariance and discriminative nature of SIFT features. Extraction of such traditional features has been shown to be accurate and stable across views, but they are rather slow to compute compared to more recent binary descriptors.

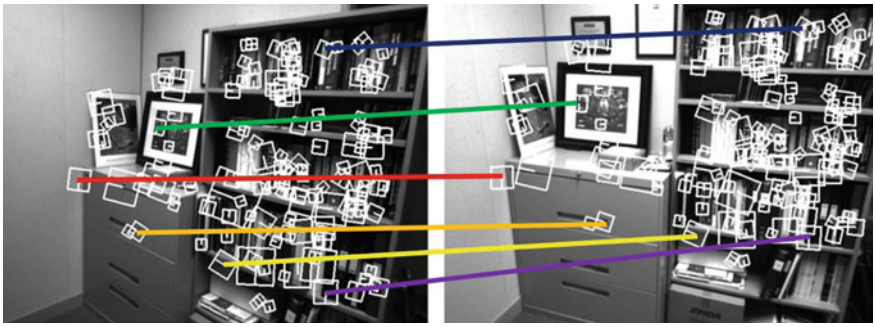

Fig. 2.13 Wide baseline matching between two images with SIFT. The size and orientation of the squares correspond to the scale and orientation of the matching SIFT features. The color lines highlight some of the matches between the two images

These include BRIEF [15], BRISK [45], ORB [73], and FREAK [2], which are more suitable for real-time performance on portable hardware of relatively limited computational power.

2.8 3D Reconstruction

Different types of 3D reconstruction can be obtained based on the amount of *a priori* knowledge available, as illustrated in Table 2.2. The simplest method to recover 3D information is stereo where the intrinsic and extrinsic parameters are known, and the *absolute metric* 3D reconstruction can be obtained. This means we can determine the actual dimensions of structures, such as *height of door*=1.93 m.

For structure from motion, if no such prior information is available, only a projective 3D reconstruction can be obtained. This means that 3D structure is known only up to an arbitrary projective transformation so we know, for example, how many planar faces the object has and what point features are collinear, but we do not know anything about the scene dimensions and angular measurements within the scene. If intrinsic parameters are available, the projective 3D reconstruction can be upgraded to a metric reconstruction, where the 3D reconstruction is known up to a scale factor (i.e., a scaled version of the original scene). There is more detail to this hierarchy of

Table 2.2 Different Types of 3D Reconstruction

A Priori Knowledge	3D Reconstruction
Intrinsic and extrinsic parameters	Absolute 3D reconstruction
Intrinsic parameters only	Metric 3D reconstruction (up to a scale factor)
No information	Projective 3D reconstruction

reconstruction than we can present here (for example, *affine* 3D reconstruction lies between the metric and projective reconstructions) and we refer the interested reader to [38].

2.8.1 Stereo

Stereo vision refers to the ability to infer information on the 3D structure and distance of a scene from two or more images taken from different viewpoints. The disparities of all the image points form the *disparity map*, which can be displayed as an image. If the stereo system is calibrated, the disparity map can be converted to a 3D point cloud representing the scene.

The discussion here focuses on binocular stereo for two image views only. Please refer to [80] for a survey of multiple-view stereo methods that reconstruct a complete 3D model instead of just a single disparity map, which generates *range image* information only. In such a 3D imaging scenario, there is at most one depth per image plane point, rear-facing surfaces, and other self-occlusions are not imaged and the data is sometimes referred to as 2.5D.

2.8.1.1 Dense Stereo Matching

The aim of dense stereo matching is to compute disparity values for all the image points from which a dense 3D point cloud can be obtained. Correlation-based methods provide dense correspondences, while feature-based methods only provide sparse correspondences. Dense stereo matching is challenging as textureless regions do not provide information to distinguish the correct matches from the incorrect ones. The quality of correlation-based matching results depends highly on the amount of texture available in the images and the illumination conditions. Repetitive texture does not help though as the matching would be ambiguous.

Figure 2.14 shows a sample disparity map after dense stereo matching. The disparity map is shown in the middle with disparity values encoded in grayscale level. The brighter pixels refer to larger disparities which means the object is closer. For example, the ground pixels are brighter than the building pixels. An example of correspondences is highlighted in red in the figure. The pixel itself and the matching pixel are marked and linked to the right image. The length of the line corresponds to the disparity value highlighted in the disparity map.

Comparing image windows between two images could be ambiguous. Various matching constraints can be applied to help reduce the ambiguity, such as

- Epipolar constraint,
- Ordering constraint,
- Uniqueness constraint, and
- Disparity range constraint.

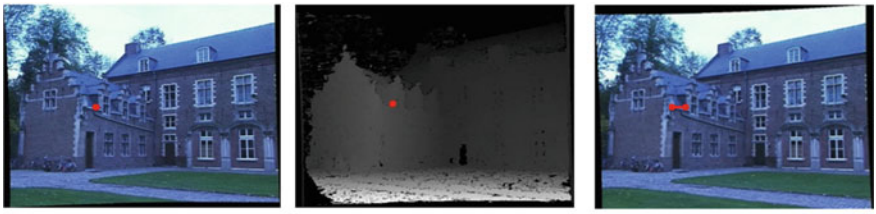

Fig. 2.14 A sample disparity map (**b**) obtained from the left image (**a**) and the right image (**c**). The disparity value for the pixel highlighted in red in the disparity map corresponds to the length of the line linking the matching features in the right image. Figure courtesy of [68]

The epipolar constraint reduces the search from 2D to the epipolar line only, as has been described in Sect. 2.5. The ordering constraint means that if pixel b is to the right of a in the left image, then the correct correspondences a' and b' must also follow the same order (i.e., b' is to the right of a' in the right image). This constraint fails if there is occlusion.

The uniqueness constraint means that each pixel has at most one corresponding pixel. In general, there is a one-to-one correspondence for each pixel, but there is none in the case of occlusion or noisy pixels. The left-right consistency check can be applied to ensure that the same match is obtained via both left-to-right and right-to-left matching.

The disparity range constraint limits the disparity search range according to the prior information of the expected scene. Maximum disparity sets how close the object can be, while the minimum disparity sets how far the object can be. Zero disparity refers to objects at infinity.

One important parameter for these correlation-based methods is the window size m in Eq. 2.20. While using a larger window size provides more intensity variation and hence more context for matching, this may cause problems around the occlusion area and at object boundaries, particularly for wide baseline matching.

Figure 2.15 shows the effect of window size on the resulting disparity map. The disparity map in the middle is for a window size of 3×3. It can be seen that, while it captures details well, it is very noisy, as the smaller window provides less information for matching. The disparity map on the right is for a window size of 15×15. It can be seen that while it looks very clean, the boundaries are not well defined. Moreover, the use of a larger window size typically increases the processing time as more pixels need to be correlated. The best window size is a trade-off between these two effects and is dependent on the level of fine detail in the scene.

For local methods, disparity computation at a given point depends on the intensity value within a local window only. The best matching window is indicated by the lowest dissimilarity measure or the highest similarity measure which uses information in the local region only. As pixels in an image are correlated (they may belong to the same object, for instance), global methods could improve the stereo matching quality by making use of information outside the local window region.

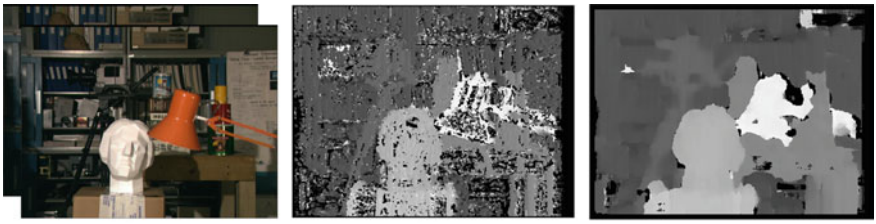

Fig. 2.15 The effect of window size on correlation-based methods: (**a**) input images (**b**) disparity map for a small correlation window (**c**) disparity map for a large correlation window. (Raw image pair courtesy of the Middlebury Stereo Vision Page [74], originally sourced from Tsukuba University.)

Global methods perform optimization across the image and are often formulated as an energy minimization problem. Dynamic programming approaches [6, 8, 19] compute the minimum-cost path through the matrix of all pairwise matching costs between two corresponding scanlines so that the best set of matches that satisfy the ordering constraint can be obtained. Dynamic programming utilizes information along each scanline independently; therefore, it may generate results that are not consistent across scanlines.

The *Graph Cuts* [12, 44] optimization technique makes use of information across the whole image and produces high-quality disparity maps. There is a trade-off between stereo matching quality and the processing time. Global methods such as this, max flow [72], and belief propagation [83, 84] produce better disparity maps than local methods, but they are computationally intensive.

Apart from the algorithm itself, the processing time also depends on the image resolution, the window size, and the disparity search range. The higher the image resolution, the more pixels need to be processed to produce the disparity map. The similarity measure needs to correlate more pixels for a larger window size. The disparity search range affects how many such measures need to be computed in order to find the correct match.

Hierarchical stereo matching methods have been proposed by down-sampling the original image into a pyramid [7, 70]. Dense stereo matching is first performed on the lowest resolution image, and disparity ranges can be propagated back to the finer resolution image afterward. This coarse-to-fine hierarchical approach allows fast computation to deal with a large disparity range, as a narrower disparity range can be used for the original image. Moreover, the more precise disparity search range helps to obtain better matches in the low texture areas.

The Middlebury webpage [74] provides standard datasets with ground truth information for researchers to benchmark their algorithms so that the performance of various algorithms can be evaluated and compared. A wide spectrum of dense stereo matching algorithms has been benchmarked, as illustrated in Fig. 2.16 [75]. Researchers can submit results of new algorithms which are ranked based on various metrics, such as RMS error between computed disparity map and ground truth map, percentage of bad matching pixels, and so on. It can be observed from Fig. 2.16 that

Fig. 2.16 Comparative disparity maps for the top 15 dense stereo matching algorithms in [75] in decreasing order of performance. The top left disparity map is the ground truth. Performance here is measured as the percentage of bad matching pixels in regions where there are no occlusions. This varies from 1.15% in algorithm 19 to 5.23% in algorithm 1. Algorithms marked with a $*$ were implemented by the authors of [75], who present a wider range of algorithms in their publication. Figure courtesy of [75]

it is very difficult to understand algorithmic performance by qualitative inspection of disparity maps and the quantitative measures presented in [75] are required.

2.8.1.2 Triangulation

When the corresponding left and right image points are known, two rays from the camera centers through the left and right image points can be back-projected. The two rays and the stereo baseline lie on a plane (the epipolar plane) and form a triangle; hence, the reconstruction is termed "triangulation". Here, we describe triangulation for a rectilinear arrangement of two views or, equivalently, two rectified views.

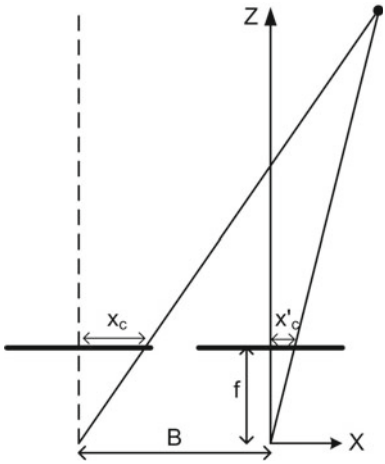

Fig. 2.17 The stereo geometry becomes quite simple after image rectification. The world coordinate frame is arbitrarily centered on the right camera. B is the stereo baseline, and f is the focal length. Disparity is given by $d = x_c - x'_c$

After image rectification, the stereo geometry becomes quite simple as shown in Fig. 2.17, which shows the top-down view of a stereo system composed of two pinhole cameras. The necessary parameters, such as baseline and focal length, are obtained from the original stereo calibration. The following two equations can be obtained based on the geometry:

$$x'_c = f\frac{X}{Z}$$

$$x_c = f\frac{X+B}{Z},$$

where x'_c and x_c are the corresponding horizontal image coordinates (in metric units) in the right and left images, respectively, f is the focal length, and B is the baseline distance.

Disparity d is defined as the difference in horizontal image coordinates between the corresponding left and right image points, given by

$$d = x_c - x'_c = \frac{fB}{Z}.$$

Therefore,

$$Z = \frac{fB}{d} \tag{2.21}$$

$$X = \frac{Zx'_c}{f}, \quad Y = \frac{Zy'_c}{f},$$

where y'_c is the vertical image coordinates in the right image.

This shows that the 3D world point can be computed once disparity is available: $(x'_c, y'_c, d) \mapsto (X, Y, Z)$. Disparity maps can be converted into depth maps using these equations to generate a 3D point cloud. It can be seen that triangulation is straightforward compared to the earlier stages of computing the two-view relations and finding correspondences.

Stereo matches are found by seeking the minimum of some cost functions across the disparity search range. This computes a set of disparity estimates in some discretized space, typically integer disparities, which may not be accurate enough for 3D recovery. 3D reconstruction using such quantized disparity maps leads to many thin layers of the scene. Interpolation can be applied to obtain sub-pixel disparity accuracy, such as fitting a curve to the SSD values for the neighboring pixels to find the peak of the curve, which provides more accurate 3D world coordinates.

By taking the derivatives of Eq. 2.21, the standard deviation of depth, which represents uncertainty of depth estimation, is given by

$$\Delta Z = \frac{Z^2}{Bf} \Delta d,$$

where Δd is the standard deviation of the disparity. This equation shows that the depth uncertainty increases quadratically with depth. Therefore, stereo systems typically are operated within a limited range. If the object is far away, the depth estimation becomes more uncertain. The depth error can be reduced by increasing the baseline, focal length, or image resolution. However, each of these has detrimental effects. For example, increasing the baseline makes matching harder as features appear less similar and causes viewed objects to self-occlude, increasing the focal length reduces the depth of field, and increasing image resolution increases processing time and data bandwidth requirements. Thus, we can see that design of stereo cameras typically involves a range of performance trade-offs, where trade-offs are selected according to the application requirements.

Figure 2.18 compares the depth uncertainty for three stereo configurations assuming a disparity standard deviation of 0.1 pixel. A stereo camera with higher resolution (dashed line) provides better accuracy than the one with lower resolution (dotted line). A stereo camera with a wider baseline (solid line) provides better accuracy than the one with a shorter baseline (dashed line).

A quick and simple method to evaluate the accuracy of 3D reconstruction is to place a highly textured planar target at various depths from the sensor, fit a least squares plane to the measurements, and measure the residual RMS error. In many cases, this gives us a good measure of depth repeatability, unless there are significant systematic errors, for example, from inaccurate calibration of stereo camera parameters. In this case, more sophisticated processes and ground truth measurement equipment are required. Capturing images of a target of known size and shape at various depths, such as a textured cube, can indicate how reconstruction performs when measuring in all three spatial dimensions.

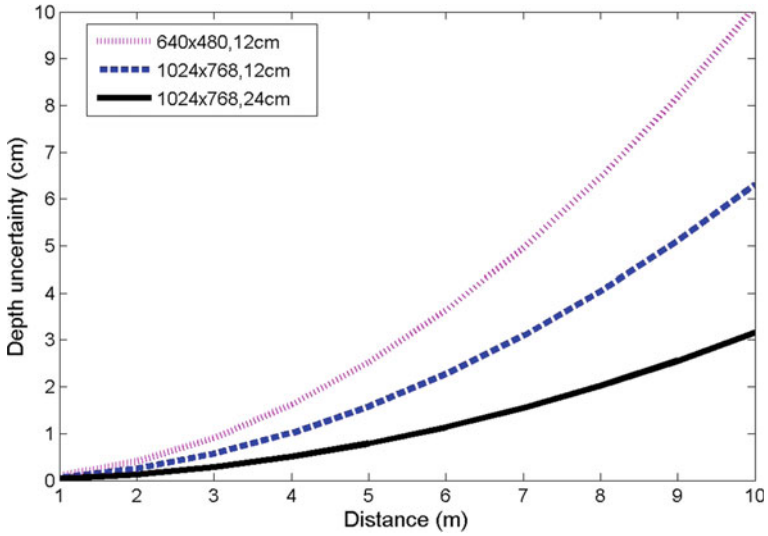

Fig. 2.18 A plot illustrating the stereo uncertainty with regard to image resolution and baseline distance. A larger baseline and higher resolution provide better accuracy, but each of these has other costs

2.8.2 Structure from Motion

Structure from motion (SfM) is the simultaneous recovery of 3D structure and camera relative pose (position and orientation) from image correspondences, and it refers to the situation where images are captured by a moving camera. There are three sub-problems in structure from motion.

- Correspondence: which elements of an image frame correspond to which elements of the next frame.
- Ego-motion and reconstruction: determination of camera motion (sometimes called ego-motion) and structure of the observed world.
- Segmentation: extraction of regions corresponding to one or more moving objects.

The third sub-problem is a relatively recent problem in structure from motion, where some objects in the scene may have moved between frames. For dynamic scenes, features belonging to moving objects could be identified and removed as outliers. Alternatively one could consider an environment to contain an unknown number (n) of independently moving objects and a static environment as $n + 1$ SfM sub-problems, each having their own F matrix. However, for the following discussion, we assume that the scene is static, without any moving objects.

By matching features between frames, we obtain at least eight correspondences from which the fundamental matrix can be recovered as described in Sect. 2.5.4. Without camera calibration parameters, only the projective reconstruction can be obtained where orthogonal lines in the world may not be reconstructed as orthogonal.

While this may be useful by itself, most practical applications require at least metric reconstruction where the reconstructed 3D model is a scaled version of the real scene.

Metric reconstruction requires camera intrinsic parameters which can be estimated from the images themselves using self-calibration (auto-calibration) techniques [38, 55] developed in recent years. Such methods exploit some prior information of the scene itself such as parallel lines, vanishing points, and so on. For better accuracy and more robustness, the camera intrinsic parameters can be obtained with a calibration procedure using a known calibration grid, as discussed in Sect. 2.4.

Once the camera intrinsic parameters are known, the essential matrix E can be computed from the fundamental matrix. According to Eq. 2.15, the motion can be recovered from E, where $\mathbf{t}$ is determined up to a scale factor only (since we can multiply Eq. 2.16 by an arbitrary non-zero scale factor). The physical insight into this is that the same image disparity between a pair of views can occur for a point close to the camera positions and a point n-times the distance away with n-times the translation. Effectively, we have scaled similar triangles in the triangulation-based reconstruction process.

SVD can be applied to extract $\mathbf{t}$ and R from E as follows [38]. Application of SVD gives the factorization $\mathsf{E} = \mathsf{U}\mathsf{D}\mathsf{V}^T$. By defining

$$\mathsf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathsf{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

the solution is given by

$$\mathsf{R} = \mathsf{U}\mathsf{W}\mathsf{V}^T \; or \; \mathsf{U}\mathsf{W}^T\mathsf{V}^T$$

$$\mathbf{t} = \pm \mathbf{u}_3,$$

where $\mathbf{u}_3$ is the third column of matrix U. With two possible choices of R and $\mathbf{t}$, there are four possible solutions. Testing with a single point to determine if it is in front of both cameras is sufficient to decide among the four different solutions. For further details, please refer to [38].

Once $\mathbf{t}$ (up to scale) and R have been extracted from E, the sparse scene structure can be recovered by computing the intersection between the back-projected rays. In general, due to measurement noise, these will not intersect in 3D space. The simplest solution is to compute the mid-point of the shortest perpendicular line between the two rays. However, a refined solution is to choose a reconstructed scene point $\mathbf{X}$, such that it minimizes the sum of square errors between the actual image positions and their positions predicted by their respective camera projection matrices. The scene structure is only determined up to a scale factor but in some applications this could be constrained, for example, if some measurement is known in the scene, or the translation can be estimated from the wheel odometry of a mobile robot. In summary, this method first estimates the intrinsic camera parameters (or uses an existing calibration) after which the extrinsic camera parameters are recovered. Both

the intrinsic and extrinsic camera parameters are then used to compute the scene structure.

Alternatively, *bundle adjustment*[14] offers a more accurate method that simultaneously optimizes the 3D structure and the 6-DOF camera pose (extrinsic camera parameters) for each view in an image sequence [88]. Sometimes the intrinsic camera parameters are also refined in the procedure. This is a batch process that iteratively refines the camera parameters and the 3D structure in order to minimize the sum of the reprojection errors. (A reprojection error is the Euclidean distance between an image feature and its reprojection into the image plane after computing the 3D world coordinate and the camera pose associated with that image point.) Since a specific reprojection error is only dependent on its own scene point and own viewpoint, the structure of the equations is sparse. Thus, even though bundle adjustments are thought to be fairly computationally expensive, exploitation of sparse linear algebra algorithms can significantly mitigate this. Such procedures are referred to as *sparse bundle adjustment*.

Using consecutive video frames gives poor 3D accuracy due to the very short baseline. An image pair formed by a larger time increment would provide better 3D information. However, if the time increment is too large, the camera could have moved significantly and it would be harder to establish correct correspondences. One possible solution to this is to track features over several short baseline frames using a small, local area-based search, before computing 3D from a pair of frames tracked over a significantly longer baseline.

2.9 Deep Learning for Passive 3D Imaging

Recently, deep learning has proved to provide the best solutions to many problems in image classification, speech recognition, and natural language processing. Essentially, application of this powerful technology involves formulation of optimization problems that have well-behaved differentials, thus enabling gradient descent to function well. Deep learning has also been applied successfully to passive 3D imaging in two modalities: improving stereo matching and inferring depth from monocular imagery.

2.9.1 Deep Learning for Stereo Matching

A Convolutional Neural Network (CNN) was trained to predict how well two image patches match and it was used to compute the stereo matching cost [96]. The proposed network consisted of eight layers and achieved the best performance in the KITTI

[14]Bundle adjustment methods appeared several decades ago in the photogrammetry literature and are now used widely in the computer vision community.

dataset [35]. While [96] focused on comparing patches in narrow baseline stereo, a general similarity function was learned from image data in [95] to handle a broader set of appearance changes so that it can be used in more challenging set of applications including wide baseline stereo. Various neural network architectures such as Siamese and 2-channel models were explored, showing that they exhibited good performance and outperformed the state-of-the-art on wide baseline stereo matching. A dot product layer was exploited to compute the inner product between the two representations of a Siamese architecture, which achieved an order of magnitude faster computation [50]. The network was trained by treating the problem as multi-class classification where the classes are all possible disparities. As a result, correlations between different disparities can be captured implicitly, whereas the previous approaches performed independent binary predictions on image patches.

2.9.2 Deep Learning for Monocular Reconstruction

Depth was estimated from a single image by integrating both global and local information from various cues [27]. A multi-scale CNN approach was proposed by employing two deep network stacks: one for a coarse global prediction based on the entire image and another to refine the prediction locally. A scale-invariant error was applied to help measure depth relations rather than scale. The proposed method achieved state-of-the-art results on both NYU Depth and KITTI datasets. A deep structured learning scheme was proposed to learn the unary and pairwise potentials of continuous Conditional Random Field (CRF) in a unified deep CNN framework [46]. Using a faster model based on fully convolutional networks and a novel superpixel pooling method resulted in 10 times speedup while producing similar prediction accuracy. The proposed method outperformed state-of-the-art results on various indoor and outdoor scenes. While both [27] and [46] require vast quantities of ground truth depth data for training, an unsupervised approach was proposed in [36] to perform single image depth estimation without the use of ground truth depth data for training. Disparity images were generated by training the CNN with an image reconstruction loss using binocular stereo data. A novel training loss that enforced left–right consistency was employed to improve performance and robustness. The proposed method produced state-of-the-art results for monocular depth estimation on KITTI dataset, even outperforming supervised methods that have been trained with ground truth depth data. Recently, Wu et al. [92] described how to learn shape priors for monocular 3D completion. Their approach integrates deep generative models with adversarially learned shape priors, where the learned priors act as a regularizer, penalizing the model if its output is unrealistic.

2.10 Passive Multiple-View 3D Imaging Systems

Examples of passive multiple-view 3D imaging systems and their applications will now be presented, including stereo cameras, people counting, 3D modeling, and visual SLAM. 3D modeling systems generate photo-realistic 3D models from sequences of images and have a wide range of applications. For mobile robot applications, passive multiple-view 3D imaging systems are used for localization, building maps, and obstacle avoidance.

2.10.1 Stereo Cameras

Stereo cameras can be custom-built by mounting two individual cameras on a rigid platform separated by a fixed baseline. However, it is important that, for non-static scenes or for mobile platforms, the two cameras are synchronized so that they capture images at the same time. In order to obtain absolute 3D information, as discussed earlier in Table 2.2, the stereo camera needs to be calibrated to recover the intrinsic and extrinsic parameters. It is also critical that the relative camera pose does not change over time; otherwise, re-calibration would be required.

Commercial off-the-shelf (COTS) stereo vision systems have been emerging in recent years. These cameras often have a fixed baseline and are pre-calibrated by the vendor. Typically, they are nicely packaged and convenient to use. There are a number of pre-calibrated COTS stereo camera available such as Tara from E-con Systems with 6 cm baseline [26] and DUO MC from Code Laboratories with 3 cm baseline [25]. The FLIR Bumblebee camera [33] is another example with 12 cm baseline, which comes pre-calibrated, and an application programming interface (API) is provided to configure the camera and grab images, as well as rectify the images and perform dense stereo matching. FLIR also offers Bumblebee XB3 which is a three-sensor multi-baseline stereo camera which has both 12 cm and 24 cm baselines available for stereo processing. The wide baseline provides more precision at longer range, while the narrow baseline improves close range matching and minimum-range limitation.

It is desirable to obtain disparity maps in real time in many applications, for example, obstacle detection for mobile robots. Dense stereo matching can be highly parallelized; therefore, such algorithms are highly suitable to run on graphics processing units (GPUs) to free up the CPU for other tasks. GPUs have a parallel throughput architecture that supports executing many concurrent threads, providing immense speedup for highly parallelized algorithms. A dense stereo matching algorithm has been implemented on a commodity graphics card [94] to perform several hundred millions of disparity evaluations per second. This corresponds to 20 Hz for 512×512 image resolution with 32 disparity search range.

Stereolabs ZED stereo camera [82] is a stereo camera with 12 cm baseline for depth sensing and motion tracking. It relies on the use of NVIDIA GPU for processing and provides real-time depth data as well as camera pose. Stereolabs also offers a mini

Fig. 2.19 FLIR Bumblebee XB3 stereo camera with three cameras provides both narrow and wide baseline configurations (left) and Stereolabs ZED stereo camera (right)

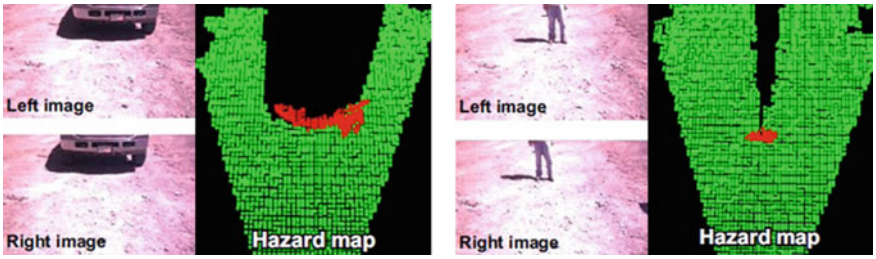

Fig. 2.20 Examples of hazard detection using stereo images: a truck (left) and a person (right). Hazards are shown in red on the hazard map

version with 6 cm baseline for augmented reality application. Two of the COTS stereo cameras mentioned are shown in Fig. 2.19.

Stereo cameras are often used for obstacle/hazard detection in mobile robots as they can capture the 3D scene instantaneously. Figure 2.20 shows the stereo images from Bumblebee XB3 and the resulting hazard maps for a truck and a person, respectively. Correlation-based matching is performed to generate a dense 3D point cloud. Clusters of point cloud that are above the ground plane are considered as hazards.

2.10.2 People Counting

People counting for retail analytics is a key market for passive stereo cameras. The stereo cameras are mounted on the ceiling of retail store entrances, pointing down to count the number of people walking in and out of the store. Based on the disparity or 3D point cloud, people are detected and tracked as they walk across the camera field-of-view. Enter/exit counts are incremented when the tracks cross a pre-defined count line. Such information can be used to generate retail analytics such as conversion rates (proportion of customers who make a purchase), to optimize staff scheduling, and so on.

Stereo cameras provide better counting accuracy than monocular cameras, as they are more tolerant of lighting changes and can avoid counting shadows. Moreover, the 3D data can be used to distinguish between people and shopping carts as well as between adults and children. Currently, there are a number of successful stereo-based

Fig. 2.21 Examples of stereo-based people counters for retail analytics: FLIR Brickstream camera (left) and Xovis PC2 camera (right)

people counters offered by various vendors including FLIR, Xovis, Axper, and Hella. Two of these are shown in Fig. 2.21.

2.10.3 3D Modeling

The creation of photo-realistic 3D models of observed scenes has been an active research topic for many years. Such 3D models are very useful for both visualization and measurements in various applications such as planetary rovers, defense, mining, forensics, archaeology, and virtual reality.

Pollefeys et al. [68] and Nister [60] presented systems which create surface models from a sequence of images taken with a hand-held video camera. The camera motion is recovered by matching corner features in the image sequence. Dense stereo matching is carried out between the frames. The input images are used as surface texture to produce photo-realistic 3D models. These monocular approaches only output a scaled version of the original scene but can be scaled with some prior information. Moreover, it requires a long processing time.

The objective of the DARPA Urbanscape project [56] is to develop a real-time data collection and processing system for the automatic geo-registered 3D reconstruction of urban scenes from video data. Multiple video streams as well as *Global Positioning System* (GPS) and *Inertial Navigation System* (INS) measurements are collected to reconstruct photo-realistic 3D models and place them in geo-registered coordinates. An example of a large-scale 3D reconstruction is shown in Fig. 2.22.

A stereo camera-based 3D vision system is capable of quickly generating calibrated photo-realistic 3D models of unknown environments. *Instant Scene Modeler* (iSM) can process stereo image sequences captured by an unconstrained hand-held stereo camera [77]. Dense stereo matching is performed to obtain 3D point clouds from each stereo pair. 3D point clouds from each stereo pair are merged together to obtain a color 3D point cloud. Furthermore, a surface triangular mesh is generated from the point cloud. This is followed by texture mapping, which involves mapping image textures to the mesh triangles. As adjacent triangles in the mesh may use different texture images, seamlines may appear unless texture blending is performed.

Fig. 2.22 An example of 3D modeling of urban scene from the Urbanscape project. Figure courtesy of [56]

Fig. 2.23 The user points the stereo camera freely at the scene of interest (left), and the photo-realistic 3D model of the scene is generated (right). Figure adapted from [77]

The resulting photo-realistic 3D models can be visualized from different views, and absolute measurements can be performed on the models. Figure 2.23 shows the user pointing the hand-held COTS stereo camera to freely scan the scene and the resulting photo-realistic 3D model, which is a textured triangular mesh.

For autonomous vehicles and planetary rovers, the creation of 3D terrain models of the environment is useful for visualization and path planning [4]. Moreover, the 3D modeling process achieves significant data compression, allowing the transfer of data as compact surface models instead of raw images. This is beneficial for planetary rover exploration due to the limited bandwidth available. Figure 2.24 shows a photo-realistic 3D model created from a moving autonomous vehicle that traveled in a desert in Nevada.

One of the key technologies required for planetary rover navigation is the ability to sense the nearby 3D terrain. Stereo cameras are suitable for planetary exploration, thanks to their low power, low mass requirements, and the lack of moving parts. The NASA *Mars Exploration Rovers* (MERs), named *Opportunity* and *Spirit*, use passive stereo image processing to measure geometric information about the environment [52]. This is done by matching and triangulating pixels from a pair of rectified stereo images to generate a 3D point cloud. Figure 2.25 shows an example of the

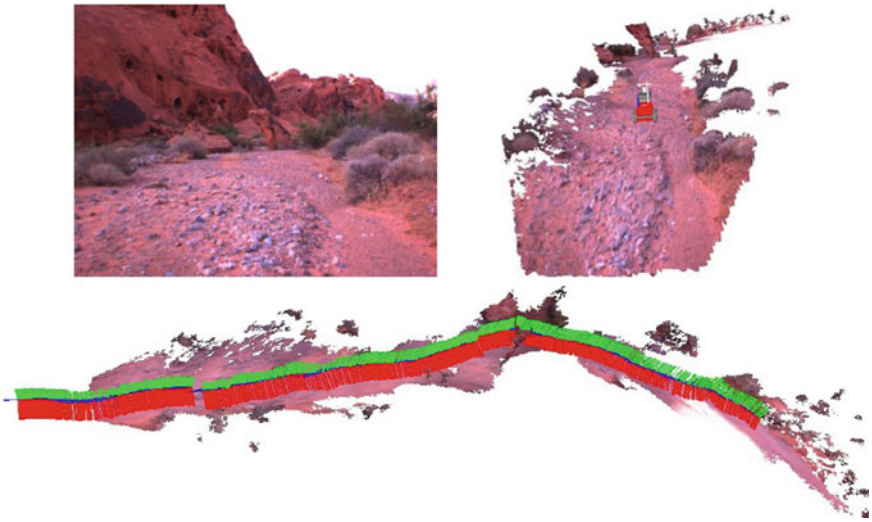

Fig. 2.24 First image of a sequence captured by an autonomous rover in a desert in Nevada (left). Terrain model generated with virtual rover model inserted (right). Resulting terrain model and rover trajectory (bottom). Figure courtesy of [4]

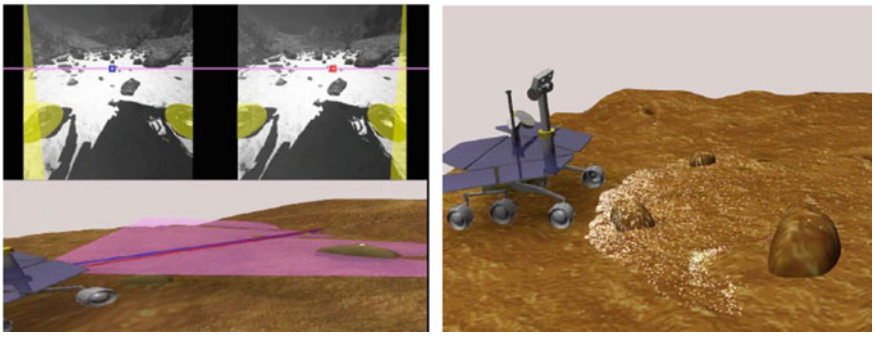

Fig. 2.25 Mars Exploration Rover stereo image processing (left) and the reconstructed color 3D point cloud (right), with a virtual rover model inserted. Figure courtesy of [52]

stereo images captured and the color 3D point cloud generated which represents the imaged terrain.

Documenting crime scenes is a tedious process that requires the investigators to record vast amounts of data by using video, still cameras, and measuring devices, and by taking samples and recording observations. With passive 3D imaging systems, 3D models of the crime scene can be created quickly without much disturbance to the crime scene. The police can also perform additional measurements using the 3D model after the crime scene is released. The 3D model can potentially be shown in court so that the judge and the jury can understand the crime scene better. Figure 2.26

Fig. 2.26 3D model of a mock crime scene obtained with a hand-held stereo camera. Figure courtesy of [78]

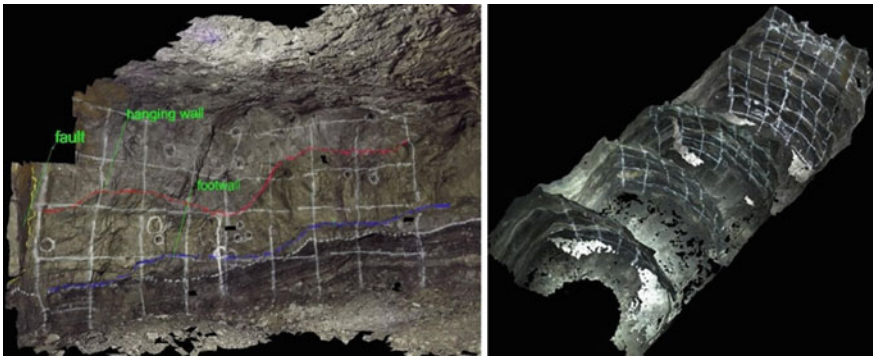

Fig. 2.27 Underground mine 3D model (left) and consecutive 3D models as the mine advances (right). The red and blue lines on the left are geological features annotated by geologists to help with the ore body modeling. Figure courtesy of [78]

shows a 3D reconstruction of a mock crime scene generated from a hand-held stereo sequence within minutes after acquisition [78].

Photo-realistic 3D models are useful for survey and geology in underground mining. The mine map can be updated after each daily drill/blast/ore removal cycle to minimize any deviation from the plan. In addition, the 3D models can also allow the mining companies to monitor how much ore is taken at each blast. Figure 2.27 shows a photo-realistic 3D model of an underground mine face annotated with geological features and consecutive 3D models of a mine tunnel created as the mine advances [78].

Airborne surveillance and reconnaissance are essential for successful military missions. *Unmanned Aerial Vehicles* (UAVs) are becoming the platform of choice

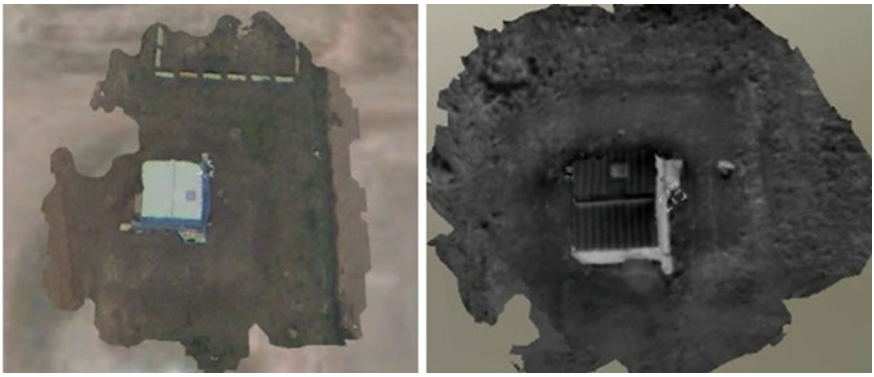

Fig. 2.28 3D reconstruction of a building on the ground using video (left) and using infrared video (right) captured by an UAV (Unmanned Aerial Vehicle). Figure courtesy of [76]

for such surveillance operations, and video cameras are among the most common sensors onboard UAVs. Photo-realistic 3D models can be generated from UAV video data to provide situational awareness as it is easier to understand the scene by visualizing it in 3D. The 3D model can be viewed from different perspectives and allow distance measurements and line-of-sight analysis. Figure 2.28 shows a 3D reconstruction of a building on the ground using video and infrared video captured by an UAV [76]. The photo-realistic 3D models are geo-referenced and can be visualized in 3D *Geographical Information System* (GIS) viewers such as Google Earth.

Deep learning has also been applied to 3D reconstruction. A novel recurrent neural network architecture was proposed for single- and multi-view 3D object reconstruction by learning a mapping from observations to their underlying 3D shape [18]. The network took one or more images of an object instance from arbitrary viewpoints to output a reconstruction of the object in the form of a 3D occupancy grid. Experimental results showed that the proposed framework outperformed the state-of-the-art method for single-view reconstruction. It worked well in situations where traditional SfM methods failed due to the lack of texture or wide baselines. An encoder–decoder network was proposed for single-view 3D object reconstruction with a novel projection loss defined by the perspective transformation [93]. The projection loss enabled unsupervised learning using 2D observation without ground truth 3D training data. A single network for multi-class 3D object reconstruction was trained, with generalization potential to unseen categories.

2.10.4 Visual SLAM

Mobile robot localization and mapping is the process of simultaneously tracking the position of a mobile robot relative to its environment and building a map of the

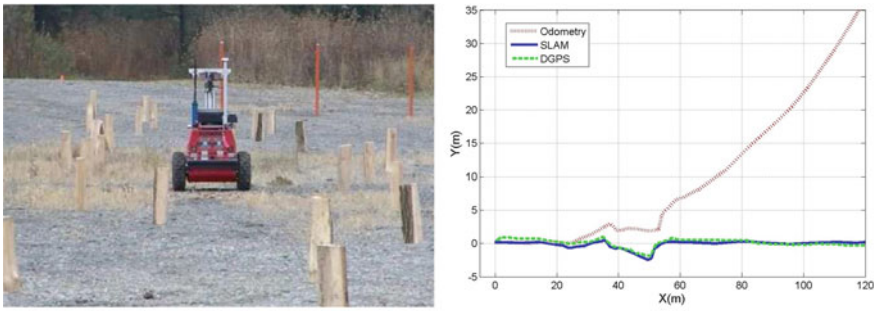

Fig. 2.29 (**a**) Autonomous rover on a gravel test site with obstacles (**b**) Comparison of the estimated path by SLAM, wheel odometry, and DGPS (Differential GPS). Figure courtesy of [4]

environment. Accurate localization is a prerequisite for building a good map and having an accurate map is essential for good localization. Therefore, *Simultaneous Localization and Mapping* (SLAM) is a critical underlying capability for successful mobile robot applications. To achieve a SLAM capability, high-resolution passive vision systems can capture images in milliseconds, and hence they are suitable for moving platforms such as mobile robots.

Stereo vision systems are commonly used on mobile robots, as they can measure the full six degrees of freedom (DOF) of the change in robot pose. This is known as visual odometry. By matching visual landmarks between frames to recover the robot motion, visual odometry is not affected by wheel slip and hence is more accurate than the wheel-based odometry. For outdoor robots with GPS receivers, visual odometry can also augment the GPS to provide better accuracy, and it is also valuable in environments where GPS signals are not available.

Unlike in 3D modeling where correlation-based dense stereo matching is typically performed, feature-based matching is sufficient for visual odometry and SLAM; indeed, it is preferable for real-time robotics applications, as it is computationally less expensive. Such features are used for localization, and a feature map is built at the same time.

The MERs *Opportunity* and *Spirit* are equipped with visual odometry capability [53]. An update to the rover's pose is computed by tracking the motion of autonomously selected terrain features between two pairs of stereo images. It has demonstrated good performance and successfully detected slip ratios as high as 125% even while driving on slopes as high as 31 degrees.

As SIFT features [48] are invariant to image translation, scaling, rotation, and fairly robust to illumination changes and affine or even mild projective deformation, they are suitable landmarks for robust SLAM. When the mobile robot moves around in an environment, landmarks are observed over time but from different angles, distances, or under different illuminations. SIFT features are extracted and matched between the stereo images to obtain 3D SIFT landmarks which are used for indoor SLAM [79] and for outdoor SLAM [4]. Figure 2.29 shows a field trial of

an autonomous vehicle at a gravel test site with obstacles and a comparison of rover localization results. It can be seen that the vision-based SLAM trajectory is much better than the wheel odometry and matches well with the *Differential GPS* (DGPS).

Monocular visual SLAM applications have been emerging in recent years which only require a single camera. The results are up to a scale factor but can be scaled with some prior information. A number of different approaches have gained in popularity, which can be categorized in two axes: direct versus indirect approaches as well as dense versus sparse methods [29]. Indirect approaches first pre-process the raw sensor measurements to generate an intermediate representation such as feature correspondences, which are then interpreted as noisy measurements to estimate geometry and camera motion. Direct approaches skip the pre-processing step and use the actual sensor values directly as measurement. In the case of passive vision, direct approaches optimize a photometric error, while indirect approaches optimize a geometric error.

Dense methods use and reconstruct all pixels in the 2D image, while sparse methods use and reconstruct only a selected set of feature points such as corners. In sparse formulation, there is no notion of neighborhood, as feature positions are conditionally independent given the camera poses and intrinsics. Dense methods exploit the connectedness of the image region to formulate a geometry prior, favoring smoothness, which is often necessary to make a dense world model. Such a geometric prior may introduce bias which could reduce long-term large-scale accuracy and may make it infeasible to have a consistent joint optimization in real time. The various visual SLAM approaches can be categorized into the following four categories:

- Sparse + indirect approach is the most widely used formulation where keypoint correspondences are used to estimate 3D geometry, such as MonoSLAM [23], PTAM [43], and ORB-SLAM [57].
- Dense + indirect approach estimates 3D geometry from a dense, regularized optical flow field, such as [71].
- Dense + direct approach employs a photometric error as well as a geometric prior to estimate dense geometry, such as DTAM [59] and LSD-SLAM [28].
- Sparse + direct approach optimizes a photometric error without incorporating a geometric prior, such as [29].

Given the recent advances in depth prediction from monocular imagery using CNNs, researchers have investigated how to improve visual SLAM with deep learning. CNN-predicted dense depth maps were fused together with depth measurements obtained from direct monocular SLAM [85]. The fusion scheme prioritized CNN depth prediction in image locations where monocular SLAM approaches tend to fail. CNN depth prediction also estimated the absolute scale of the reconstruction, hence overcoming one of the major limitations of monocular SLAM.

2.11 Passive Versus Active 3D Imaging Systems

Before concluding, we briefly compare passive multiple-view 3D imaging systems and their active imaging counterpart, as a bridge between this and the following chapter. Passive systems do not emit any illumination and only perceive the ambient light reflected from the scene. Typically, this is reflected sunlight when outdoors, or the light, reflected from standard room lighting when indoors. On the other hand, active systems include their own source of illumination, which has two main benefits:

- 3D structure can be determined in smooth, textureless regions. For passive stereo, it is impossible to extract features and correspondences in such circumstances.
- The correspondence problem either disappears, for example, a single spot of light may be projected at any one time, or is greatly simplified by controlling the structure of the projected light.

The geometric principle of determining depth from a light (or other EMR) projector (e.g., laser) and a camera is identical to the passive binocular stereo situation. The physical difference is that, instead of using triangulation applied to a pair of back-projected rays, we apply triangulation to the axis of the projected light and a single back-projected ray.

Compared with active approaches, passive systems are computationally intensive as the 3D data is computed from processing the images and matching image features. Moreover, the depth data could be noisier as it relies on the natural texture in the scene and ambient lighting condition. Unlike active scanning systems such as laser scanners, cameras could capture complete images in milliseconds; hence, they can be used as mobile sensors or operate in dynamic environments. The cost, size, mass, and power requirements of cameras are generally lower than those of active sensors.

2.12 Concluding Remarks

One of the key challenges for 3D vision researchers is to develop algorithms to recover accurate 3D information robustly under a wide range of illumination conditions, a task that can be done by humans so effortlessly. While 3D passive vision algorithms have been maturing over the years, this is still an active topic in the research community and at major computer vision conferences. Many algorithms perform reasonably well with test data but there are still challenges to handle scenes with uncontrolled illumination. Deep learning, in the form of many-layered neural networks, continues to have a big impact on this area. Passive 3D imaging systems are becoming more prevalent as cameras are getting cheaper and computers are fast enough to handle the intensive processing requirements. Thanks to hardware acceleration and GPUs, embedded real-time applications are becoming more robust, leading to a growing number of real-world applications.

After working through this chapter, you should be able to

- Explain the fundamental concepts and challenges of passive 3D imaging systems.
- Explain the principles of epipolar geometry.
- Solve the correspondence problem by correlation-based and feature-based techniques (using off-the-shelf feature extractors).
- Estimate the fundamental matrix from correspondences.
- Perform dense stereo matching and compute a 3D point cloud.
- Explain the principles of structure from motion.
- Explain how deep learning can be applied to 3D imaging.
- Provide example applications of passive 3D imaging systems.

2.13 Further Reading

Two-view geometry is studied extensively in [38], which also covers the equivalent of epipolar geometry for three or more images. The eight-point algorithm was proposed in [39] to compute the fundamental matrix, while the five-point algorithm was proposed in [61] for calibrated cameras. Reference [88] provides a good tutorial and survey on bundle adjustment, which is also covered in textbooks [30, 38] and a recent survey article [55].

Surveys such as [75] serve as a guide to the extensive literature on stereo imaging. Structure from motion is extensively covered in review articles such as [55]. A step-by-step guide to 3D modeling from images is described in detail in [51]. Non-rigid structure from motion for dynamic scenes is discussed in [87].

Multiple-view 3D vision, particularly in the context of deep learning, continues to be highly active research topics, and some of the major computer vision conferences include the International Conference on Computer Vision (ICCV), IEEE Conference on Computer Vision and Pattern Recognition (CVPR), and the European Conference on Computer Vision (ECCV). Some of the relevant major journals include International Journal of Computer Vision (IJCV), IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), and Image and Vision Computing (IVC).

The following websites provide comprehensive online resources for computer vision including 3D passive vision topics and are being updated regularly.

- CVonline [31] provides an online compendium of computer vision.
- VisionBib.Com [69] contains annotated bibliography on a wide range of computer vision topics, as well as references to available datasets.

Due to limited space, there are areas of passive 3D imaging that we were unable to cover. For example, we have assumed that we are using imaging in the medium of air, yet many applications require underwater imaging, such as Unmanned Underwater Vehicles (UUVs). A useful function of such UUVs is to perform a reconstruction of the seabed or some artefact on the seabed, such as a shipwreck. This poses several problems that we have not considered, such as the refractive properties of the air–glass–water interface affecting the camera's intrinsic parameters. Furthermore, images are blurred due to light scattering, and visibility is often poor due to suspended

particles. An interesting case study of the Xlendi Wreck in Malta is provided by Drap et al. [24], which employs underwater photogrammetry and object modeling.

Also, we did not extensively detail methods that can deal with missing parts due to insufficient texture for multiple-view matching, or due to self-occlusion. In applications when the class of object to be reconstructed is known, it is possible to learn a shape prior, for example, as a 3D Morphable Model (3DMM). Given a 3DMM, missing parts can then be inferred from the reconstructed parts. Such models are discussed extensively in a later chapter of this book. Furthermore, recent work has applied deep learning to the combined problem of monocular reconstruction and shape completion [92].

2.14 Software Resources

As 3D reconstruction is now a mature technology, there are a variety of software packages that can perform 3D reconstruction from a series of 2D images. Open-source 3D modeling packages include OpenMVG [63] and OpenMVS [64]. Open Multiple-View Geometry (OpenMVG) is a library for computer vision scientists targeted the multiple-view geometry community. It provides an easy access to classical problem solvers in multiple-view geometry, such as *structure from motion*, which recovers camera poses and a sparse 3D point cloud from a set of input images. Open Multi-View Stereo (OpenMVS) focuses on the last part of the reconstruction pipeline by providing a set of algorithms to recover the full surface of the scene. The input is a set of camera poses plus the sparse point cloud, and the output is a photo-realistic 3D model in the form of a textured mesh. There are other free open-source 3D reconstruction software packages, such as AliceVision's *Meshroom* [3]. Furthermore, *OpenCV* [62] is a free general open-source computer vision library that is useful in a wide range of passive 3D applications. There are several commercial 3D reconstruction packages available, sometimes called *photogrammetry* packages and sometimes with educational licenses. For example, the Agisoft company has the *Metashape* package [1], which is a development of their earlier *Photoscan* product. Another is *Pix4D* [66], which is targeted at drone-based aerial mapping.

2.15 Questions

1. What are the differences between passive and active 3D vision systems?
2. Name two approaches to recover 3D from single images and two approaches to recover 3D from multiple images.
3. What is the epipolar constraint and how can you use it to speed up the search for correspondences?
4. What are the differences between essential and fundamental matrices?
5. What is the purpose of rectification?

6. What are the differences between correlation-based and feature-based methods for finding correspondences?
7. What are the differences between local and global methods for dense stereo matching?
8. What are the differences between stereo and structure from motion?
9. What are the factors that affect the accuracy of stereo vision systems?
10. What are two use cases of applying deep learning to 3D imaging?

2.16 Exercises

Experimenting with stereo imaging requires that you have two images of a scene from slightly different viewpoints, with a good overlap between the views, and a significant number of well-distributed corner features that can be matched. You will also need a corner detector. There are many stereo image pairs and corner detector implementations available on the web [62]. Of course, you can collect your own images either with a pre-packaged stereo camera or with a pair of standard digital cameras. The following programming exercises should be implemented in a language of your choice.

1. *Fundamental matrix with manual correspondences*. Run a corner detector on the image pair. Use a point-and-click GUI to manually label around 20 well-distributed correspondences. Compute the fundamental matrix and plot the conjugate pair of epipolar lines on the images for each correspondence. Experiment with different numbers and combinations of correspondences, using a minimum of eight in the eight-point algorithm. Observe and comment on the sensitivity of the epipolar lines with respect to the set of correspondences chosen.
2. *Fundamental matrix estimation with outlier removal*. Add 4 incorrect corner correspondences to your list of 20 correct ones. Observe the effect on the computed fundamental matrix and the associated (corrupted) epipolar lines. Augment your implementation of fundamental matrix estimation with the RANSAC algorithm. Use a graphical overlay on your images to show that RANSAC has correctly identified the outliers, and verify that the fundamental matrix and its associated epipolar lines can now be computed without the corrupting effect of the outliers.
3. *Automatic feature correspondences*. Implement a function to *automatically* match corners between two images according to the *Sum of Squared Differences* (SSD) measure. Also, implement a function for the *Normalized Cross-Correlation* (NCC) measure. Compare the matching results with test images of similar brightness and also of different brightnesses.
4. *Fundamental matrix from automatic correspondences*. Use your fundamental matrix computation (with RANSAC) with the automatic feature correspondences. Determine the positions of the epipoles and, again, plot the epipolar lines.

The following additional exercises require the use of a stereo rig, which could be a pre-packaged stereo pair or a homemade rig with a pair of standard digital cameras. The cameras should have a small amount of vergence to overlap their fields of view.

5. *Calibration*. Create your own calibration target by printing off a chessboard pattern and pasting it to a flat piece of wood. Use a point-and-click GUI to semi-automate the corner correspondences between the calibration target and a set of captured calibration images. Implement a camera calibration procedure for a stereo pair to determine the intrinsic and extrinsic parameters of the stereo rig. If you have less time available, you may choose to use some of the calibration libraries available on the web [11, 62].
6. *Rectification*. Compute an image warping (homography) to apply to each image in the stereo image pair, such that conjugate epipolar lines are horizontal (parallel to the x-axis) and have the same y-coordinate. Plot a set of epipolar lines to check that this rectification is correct.
7. *Dense stereo matching*. Implement a function to perform local dense stereo matching between left and right rectified images, using NCC as the similarity measure, and hence generate a disparity map for the stereo pair. Capture stereo images for a selection of scenes with varying amounts of texture within them and at varying distances from the cameras, and compare their disparity maps.
8. *3D reconstruction*. Implement a function to perform a 3D reconstruction from your disparity maps and camera calibration information. Use a graphics tool to visualize the reconstructions. Comment on the performance of the reconstructions for different scenes and for different distances from the stereo rig.

References

1. Agisoft: Agisoft's Metashape. Photogrammetric processing of digital images and 3D spatial data generation. https://www.agisoft.com (2020). Accessed 3 Jan 2020
2. Alahi, A., Ortiz, R., Vandergheynst, P.: FREAK: Fast retina keypoint. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 510–517 (2012)
3. AliceVision: AliceVision: Photogrammetric Computer Vision Framework. https://alicevision.org/#meshroom (2020). Accessed 3 Jan 2020
4. Barfoot, T., Se, S., Jasiobedzki, P.: Vision-based localization and terrain modelling for planetary rovers. In: A. Howard, E. Tunstel (eds.) Intelligence for Space Robotics, pp. 71–92. TSI Press, Beijing (2006)
5. Bay, H., Ess, A., Tuytelaars, T., Gool, L.V.: Speeded-up robust features (SURF). Comput. Vis. Image Underst. **110**(3), 346–359 (2008)
6. Belhumeur, P.N.: A bayesian approach to binocular steropsis. Int. J. Comput. Vis. **19**(3), 237–260 (1996)
7. Bergen, J.R., Anandan, P., Hanna, K.J., Hingorani, R.: Hierarchical model-based motion estimation. In: G. Sandini (ed.) Computer Vision— ECCV'92, pp. 237–252 (1992)
8. Birchfield, S., Tomasi, C.: Depth discontinuities by pixel-to-pixel stereo. Int. J. Comput. Vis. **35**(3), 269–293 (1999)
9. Blanz, V., Vetter, T.: Face recognition based on fitting a 3D morphable model. IEEE Trans. Pattern Anal. Mach. Intell. **25**(9), 1063–1074 (2003)

10. Booth, J., Roussos, A., Ponniah, A., Dunaway, D., Zafeiriou, S.: Large scale 3D morphable models. Int. J. Comput. Vis. **126**(2), 233–254 (2018)
11. Bouguet, J.Y.: Camera calibration toolbox for MATLAB. http://www.vision.caltech.edu/bouguetj/calib_doc/ (2015). Accessed 3 Jan 2020
12. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **23**(11), 1222–1239 (1999)
13. Brown, D.H.: Decentering distortion of lenses. Photom. Eng. **32**(3), 444–462 (1966)
14. Brown, D.H.: Close-range camera calibration. Photom. Eng. **37**(8), 855–866 (1971)
15. Calonder, M., Lepetit, V., Strecha, C., Fua, P.: BRIEF: Binary robust independent elementary features. In: Computer Vision—ECCV 2010, pp. 778–792. Springer, Berlin, Heidelberg (2010)
16. Chen, Z., Wu, C., Liu, Y., Pears, N.E.: 3D Euclidean reconstruction of buildings from uncalibrated image sequences. Int. J. Shape Modell. **10**(1), 115–131 (2004)
17. Chen, Z., Pears, N.E., Liang, B.: Monocular obstacle detection using reciprocal-polar rectification. Image Vis. Comput. **24**(12), 1301–1312 (2006)
18. Choy, C., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In: European Conference on Computer Vision (ECCV), pp. 628–644 (2016)
19. Cox, I.J., Hingorani, S.L., Rao, S.B., Maggs, B.M.: A maximum likelihood stereo algorithm. Comput. Vis. Image Underst. **63**(3), 542–567 (1996)
20. Coxeter, H.: Projective Geometry, 2nd edn. Springer, Berlin (2003)
21. Criminisi, A., Reid, I., Zisserman, A.: Single view metrology. Int. J. Comput. Vis. **40**(2), 123–148 (2000)
22. Dai, H., Pears, N., Smith, W.A.P., Duncan, C.: A 3D morphable model of craniofacial shape and texture variation. In: IEEE International Conference on Computer Vision (ICCV), pp. 3104–3112 (2017)
23. Davison, A.J., Reid, I.D., Molton, N.D., Stasse, O.: MonoSLAM: Real-time single camera SLAM. IEEE Trans. Pattern Anal. Mach. Intell. **29**(6), 1052–1067 (2007)
24. Drap, P., Merad, D., Hijazi, B., Gaoua, L., Nawaf, M., Saccone, M., Chemisky, B., Seinturier, J., Sourisseau, J.C., Gambin, T., Castro, F.: Underwater photogrammetry and object modeling: A case study of xlendi wreck in malta. Sensors **15**, 30351–30384 (2015)
25. duo3d: The DUO MC ultra-compact, configurable stereo camera. https://duo3d.com/product/duo-mc-lv1 (2020). Accessed 3 Jan 2020
26. e-con systems: Tara—USB 3.0 Stereo Vision Camera. https://www.e-consystems.com/3D-USB-stereo-camera.asp (2020). Accessed 3 Jan 2020
27. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Advances in Neural Information Processing Systems 27 (NIPS), pp. 2366–2374 (2014)
28. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM. In: Computer Vision – ECCV 2014, pp. 834–849. Springer International Publishing, Berlin (2014)
29. Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. IEEE Trans. Pattern Anal. Mach. Intell. **40**(3), 611–625 (2018)
30. Faugeras, O., Luong, Q.: The Geometry of Multiple Images. MIT Press, Cambridge, MA (2001)
31. Fischer, R.: CVonline: The Evolving, Distributed, Non-Proprietary, On-Line Compendium of Computer Vision. http://homepages.inf.ed.ac.uk/rbf/CVonline/ (1999–2020). Accessed 10 Mar 2020
32. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
33. FLIR systems: Bumblebee 3D cameras. https://www.flir.com/iis/machine-vision/stereo-vision (2020). Accessed 10 Mar 2020
34. Garding, J.: Shape from texture for smooth curved surfaces in perspective projection. J. Math. Imaging Vis. **2**, 630–638 (1992)
35. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. Int. J. Robot. Res. **32**, 1231–1237 (2013)

36. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
37. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proceedings of the 4th Alvey Vision Conference, pp. 147–151 (1988)
38. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, Cambridge. ISBN: 0521540518 (2004)
39. Hartley, R.I.: In defense of the eight-point algorithm. IEEE Trans. Pattern Anal. Mach. Intell. **19**(6), 580–593 (1997)
40. Horn, B.K.P., Brooks, M.J. (eds.): Shape from Shading. MIT Press, Cambridge, MA, USA (1989)
41. Horn, B.K., Schunck, B.G.: Determining optical flow. Artif. Intell. **17**(1), 185–203 (1981)
42. Huang, R., Smith, W.: A shape-from-shading framework for satisfying data-closeness and structure-preserving smoothness constraints. In: Proceedings of the British Machine Vision Conference (BMVC) (2009)
43. Klein, G., Murray, D.: Parallel tracking and mapping for small ar workspaces. In: 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 225–234 (2007)
44. Kolmogorov, V., Zabih, R.: Computing visual correspondence with occlusions using graph cuts. In: Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, vol. 2, pp. 508–515 (2001)
45. Leutenegger, S., Chli, M., Siegwart, R.Y.: BRISK: Binary robust invariant scalable keypoints. In: 2011 International Conference on Computer Vision, pp. 2548–2555 (2011)
46. Liu, F., Shen, C., Lin, G., Reid, I.: Learning depth from single monocular images using deep convolutional neural fields. IEEE Trans. Pattern Anal. Mach. Intell. **38**(10), 2024–2039 (2016)
47. Longuet-Higgins, H.: A computer algorithm for reconstructing a scene from two projections. In: Fischler, M.A., Firschein, O. (eds.) Readings in Computer Vision, pp. 61–62. Morgan Kaufmann, San Francisco (CA) (1987)
48. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**, 91–110 (2004)
49. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision (ijcai). In: Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81), pp. 674–679 (1981)
50. Luo, W., Schwing, A.G., Urtasun, R.: Efficient deep learning for stereo matching. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5695–5703 (2016)
51. Ma, Y., Soatto, S., Kosecka, J., Sastry, S.S.: An Invitation to 3-D Vision: From Images to Geometric Models. Springer, Berlin (2003)
52. Maimone, M., Biesiadecki, J., Tunstel, E., Cheng, Y., Leger, C.: Surface navigation and mobility intelligence on the mars exploration rovers. In: A. Howard, E. Tunstel (eds.) Intelligence for Space Robotics, pp. 45–69. TSI Press, San Antonio (2006)
53. Maimone, M., Cheng, Y., Matthies, L.: Two years of visual odometry on the mars exploration rovers. J. Field Robot. **24**(3), 169–186 (2007)
54. Mallon, J., Whelan, P.F.: Projective rectification from the fundamental matrix. Image Vis. Comput. **23**(7), 643–650 (2005)
55. Moons, T., Gool, L.V., Vergauwen, M.: 3d reconstruction from multiple images part 1: Principles. Found. Trends Comput. Graph. Vis. **4**(4), 287–404 (2010)
56. Mordohai, P., Frahm, J.M., Akbarzadeh, A., Engels, C., Gallup, D., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewenius, H., Towles, H., Welch, G., Yang, R., Pollefeys, M., Nistér, D.: Real-time video-based reconstruction of urban environments. In: Proceedings of 3DARCH: 3D Virtual Reconstruction and Visualization of Complex Architectures (2007)
57. Mur-Artal, R., Montiel, J.M.M., Tardós, J.D.: ORB-SLAM: A versatile and accurate monocular SLAM system. IEEE Trans. Robot. **31**(5), 1147–1163 (2015)
58. Nayar, S.K., Nakagawa, Y.: Shape from focus. IEEE Trans. Pattern Anal. Mach. Intell. **16**(8), 824–831 (1994)

59. Newcombe, R.A., Lovegrove, S.J., Davison, A.J.: Dtam: Dense tracking and mapping in real-time. In: 2011 International Conference on Computer Vision, pp. 2320–2327 (2011)
60. Nister, D.: Automatic passive recovery of 3D from images and video. In: Proceedings 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004, pp. 438–445 (2004)
61. Nister, D.: An efficient solution to the five-point relative pose problem. IEEE Trans. Pattern Anal. Mach. Intell. **26**(6), 756–770 (2004)
62. OpenCV: Open source Computer Vision library. https://opencv.org/ (2020). Accessed 4 Jan 2020
63. OpenMVG: Open Multiple View Geometry library. https://github.com/openMVG/openMVG/ (2020). Accessed 10 Mar 2020
64. OpenMVS: Open Multi-View Stereo reconstruction library (Accessed Jan 4th, 2020). https://github.com/cdcseacave/openMVS/ (2020). Accessed 4 Mar 2020
65. Pentland, A.P.: A new sense for depth of field. IEEE Trans. Pattern Anal. Mach. Intell. **9**(4), 523–531 (1987)
66. Pix4D: Pix4D Photogrammetry software suite for drone mapping. https://www.pix4d.com/ (2020). Accessed 4 Jan 2020
67. Pollefeys, M., Koch, R., Van Gool, L.: A simple and efficient rectification method for general motion. Proc. Seventh IEEE Int. Conf. Comput. Vis. **1**, 496–501 (1999)
68. Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. Int. J. Comput. Vis. **59**(3), 207–232 (2004)
69. Price, K.: VisionBib.com: Computer Vision Information Pages. http://www.visionbib.com/index.php (2020). Accessed 10 Mar 2020
70. Quam, L.H.: Hierarchical warp stereo. In: Fischler, M.A., Firschein, O. (eds.) Readings in Computer Vision, pp. 80–86. Morgan Kaufmann, San Francisco (CA) (1987)
71. Ranftl, R., Vineet, V., Chen, Q., Koltun, V.: Dense monocular depth estimation in complex dynamic scenes. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4058–4066 (2016)
72. Roy, S., Cox, I.J.: A maximum-flow formulation of the n-camera stereo correspondence problem. In: International Conference on Computer Vision, pp. 492–499 (1998)
73. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to SIFT or SURF. In: International Conference on Computer Vision, pp. 2564–2571 (2011)
74. Scharstein, D., Szeliski, R., Hirschmüller, H.: The Middlebury stereo vision page. http://vision.middlebury.edu/stereo (2020). Accessed 3 Jan 2020
75. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vis. **47**(1), 7–42 (2002)
76. Se, S., Firoozfam, P., Goldstein, N., Dutkiewicz, M., Pace, P.: Automated uav-based video exploitation for mapping and surveillance. In: International Society for Photogrammetry and Remote Sensing (ISPRS) Commission I Symposium, vol. 38 (2010)
77. Se, S., Jasiobedzki, P.: Photo-realistic 3D model reconstruction. In: Proceedings IEEE International Conference on Robotics and Automation (ICRA), pp. 3076–3082 (2006)
78. Se, S., Jasiobedzki, P.: Stereo-vision based 3D modeling and localization for unmanned vehicles. Int. J. Intell. Control Syst. **13**(1), 47–58 (2008)
79. Se, S., Lowe, D., Little, J.: Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. Int. J. Robot. Res. **21**(8), 735–758 (2002)
80. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 519–528 (2006)
81. Smith, S.M., Brady, J.M.: Susan—A new approach to low level image processing. Int. J. Comput. Vis. **23**(1), 45–78 (1997)
82. Stereolabs: The Stereolab ZED camera. https://www.stereolabs.com/zed/ (2020). Accessed 4 Jan 2020
83. Sun, Jian, Zheng, Nan-Ning, Shum, Heung-Yeung: Stereo matching using belief propagation. IEEE Trans. Pattern Anal. Mach. Intell. **25**(7), 787–800 (2003)

84. Tappen, Freeman: Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. Proc. Ninth IEEE Int. Conf. Comput. Vis. **2**, 900–906 (2003)
85. Tateno, K., Tombari, F., Laina, I., Navab, N.: CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
86. Torr, P., Murray, D.: The development and comparison of robust methods for estimating the fundamental matrix. Int. J. Comput. Vis. **24**(3), 271–300 (1997)
87. Torresani, L., Hertzmann, A., Bregler, C.: Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. IEEE Trans. Pattern Anal. Mach. Intell. **30**(5), 878–892 (2008)
88. Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle adjustment—A modern synthesis. In: Vision Algorithms: Theory and Practice, LNCS, pp. 298–375 (2000)
89. Tsai, R.: A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. IEEE J. Robot. Autom. **3**(4), 323–344 (1987)
90. White, R., Forsyth, D.A.: Combining cues: Shape from shading and texture. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 2, pp. 1809–1816 (2006)
91. Woodham, R.J.: Analysing images of curved surfaces. Artif. Intell. **17**(1), 117–140 (1981)
92. Wu, J., Zhang, C., Zhang, X., Zhang, Z., Freeman, W., Tenenbaum, J.: Learning shape priors for single-view 3D completion and reconstruction. ECCV **2018**, 673–691 (2018)
93. Yan, X., Yang, J., Yumer, E., Guo, Y., Lee, H.: Perspective transformer nets: Learning single-view 3D object reconstruction without 3D supervision. In: Advances in Neural Information Processing Systems (NIPS), vol. 29, pp. 1696–1704 (2016)
94. Yang, R., Pollefeys, M.: A versatile stereo implementation on commodity graphics hardware. Real-Time Imaging **11**(1), 7–18 (2005)
95. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pp. 4353–4361 (2015)
96. Zbontar, J., LeCun, Y.: Computing the stereo matching cost with a convolutional neural network. In: IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 1592–1599 (2015)
97. Zhang R., Tsai P.-S., Cryer, J.E., Shah, M.: Shape-from-shading: A survey. IEEE Trans. Pattern Anal. Mach. Intell. **21**(8), 690–706 (1999)
98. Zhang, Z.: A flexible new technique for camera calibration. IEEE Trans. Pattern Anal. Mach. Intell. **22**(11), 1330–1334 (2000)