Cameras, Coordinates, and Calibration

6

This chapter describes three basic components of a computer vision system. The geometry and photometry of the used cameras needs to be understood (to some degree). For modelling the projective mapping of the 3D world into images and for the steps involved in camera calibration, we have to deal with several coordinate systems. By calibration we map recorded images into normalized (e.g. geometrically rectified) representations, thus simplifying subsequent vision procedures.

Insert 6.1 (Niépce and the First Photograph) *The world's first photograph* (*image is in the public domain*) *was taken in* 1826 *in France by N. Niépce* (1765–1833). *It shows a view from a workroom on his farm at Le Gras*:

During eight hours of exposure time (*note*: *buildings are illuminated by the sun from the right and from the left*), *the photograph was captured on a* 20×25 *cm oil-treated bitumen*.

R. Klette, *Concise Computer Vision*, Undergraduate Topics in Computer Science,
DOI 10.1007/978-1-4471-6320-6_6, © Springer-Verlag London 2014

Fig. 6.1 A drawing (in the public domain) of a camera obscura in the 17th century "Sketchbook on military art, including geometry, fortifications, artillery, mechanics, and pyrotechnics". Outside objects are projected top-down through a small hole onto a wall in the dark room

6.1 Cameras

The principle of a *camera obscura* is illustrated in Fig. 6.1. A small hole in the wall of a dark room projects the outside world top-down. This was known for thousands of years (e.g. about 2500 years ago in China), but it took till the beginning of the 19th century that projected images also were recorded on a medium, thus "frozen in time". By inserting a lens into the hole, the brightness and clarity of camera obscuras improved in the 16th century.

This section discusses features of cameras that may help you in the decision process which camera(s) should be used in your research or application. It also provides basic models for a single camera or a stereo-camera system, to be used in the following chapters.

6.1.1 Properties of a Digital Camera

A digital camera uses one or several *matrix sensors* for recording a projected image. See Fig. 6.2, left. A sensor matrix is an $N_{cols} \times N_{rows}$ array of *sensor elements* (phototransistors), produced either in *charge-coupled device* (CCD) or *complementary metal-oxide semiconductor* (CMOS) technology. The first digital camera was Sony's *Mavica* in 1981, after which other digital cameras were manufactured.

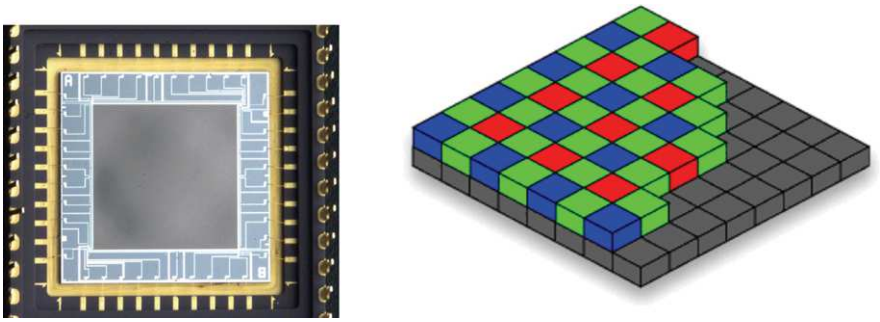


Fig. 6.2 *Left*: A sensor matrix. The individual cells are so tiny that they cannot be seen here, even after zooming in. *Right*: A sketch of the Bayer pattern

Fig. 6.3 The analysis of a car crash test (here at Daimler A.G., Sindelfingen, Germany) was based (in 2006) on high-resolution images captured at 1,000 pps

Computer Vision Cameras Computer vision benefits from the use of high-quality cameras. Important properties are, for example, the colour accuracy, reduced lens distortion, ideal aspect ratio, high spatial (also called high-definition) image resolution, large bit depth, a high dynamic range (i.e. value accuracy in dark regions of an image as well as in bright regions of the same image), and high speed of frame transfer. See Fig. 6.3 for an example of an application requiring high-quality cameras (e.g. for answering the question: "Did the mannequin's head hit the steering wheel?").

Computer vision cameras are typically permanently connected to a computer (via a video port or a frame grabber) and require software for frame capture or camera control (e.g., for time synchronization, panning, tilting, or zooming).

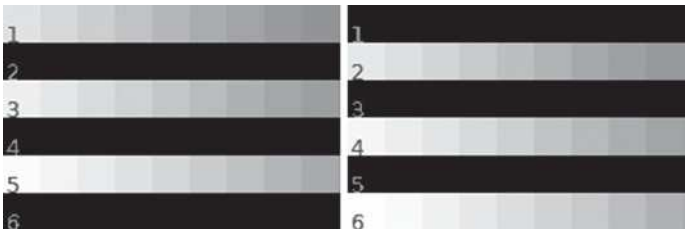


Fig. 6.4 Half-frames defined by either odd (*left*) or even (*right*) row indices

Digital Video Digital cameras provide normally both options of recording still images or video data. For a given camera, spatial times temporal resolution is typically a constant. For example, a camera which captures $7{,}680 \times 4{,}320$ (i.e. 33 Mpx) at 60 fps, records 1.99 Gpx (Gigapixels) per second. The same camera may also support to record $2{,}560 \times 1{,}440$ (i.e. 3.7 Mpx) at 540 fps, which also means 1.99 Gpx per second.

Interlaced digital video scans subsequent frames either at odd or even lines of the image sensor; see Fig. 6.4. Analog video introduced this technique for reducing transmission bandwidth. Reasons for interlaced video scans disappear with today's imaging technology.

Interlaced video is in particular disturbing for automated video analysis. Ways of combining both half-frames into one full frame can be (e.g.) by linear interpolation or simply by doubling rows in one half-frame.

Each frame contains the entire image in *progressive video*. This not only leads to better visual video quality, it also provides an appropriate input for video analysis.

Image Resolution and Bit Depth Each phototransistor is an $a \times b$ rectangular cell (e.g. a and b are about 2 μm each). Ideally, the *aspect ratio* a/b should be equal to 1 (i.e. square cells).

The image resolution $N_{cols} \times N_{rows}$ (= number of sensor elements) is commonly specified in *Megapixels* (Mpx). For example, a 4-Mpx camera has $\approx 4{,}000{,}000$ pixels in an image format such as $3:4$ or $9:16$. Without further mentioning, the number of pixels means "colour pixels". For example, Kodak offered in 1991 its DCS-100, which had a 1.3-Mpx sensor array.

A large number of pixels alone does not yet ensure image quality. As a simple example, more pixels means in general a smaller sensor area per pixel, thus less light per sensor area and a worse signal-to-noise ratio (SNR). The point-spread function of the optics used has to ensure that a larger number of pixels does not simply lead to additional noise in images. For computer vision applications, it is also often important to have more than just 8 bits per pixel value in one channel (e.g. it is of benefit to have 16 bits per pixel in a grey-level image when doing motion or stereo analysis).

Bayer Pattern or Beam Splitting The *Bayer pattern* (named after B. Bayer at Eastman Kodak) is commonly used on consumer digital cameras. One colour pixel is actually captured by four sensor elements, two for Green and one for Red and

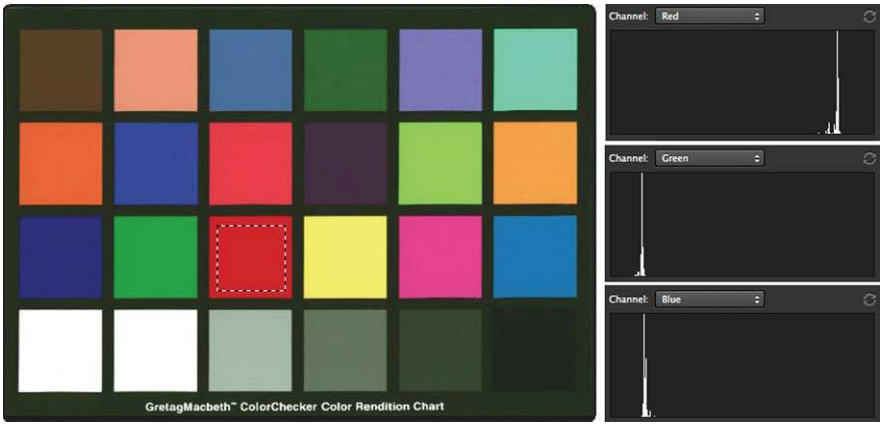


Fig. 6.5 A selected window in the red patch and histograms for the R, G, and B channels

Blue each. See Fig. 6.2, right. A sensor array of size $N_{cols} \times N_{rows}$ then actually records only colour images of resolution $\frac{N_{cols}}{2} \times \frac{N_{rows}}{2}$. Values R, G, and B at one recorded pixel are recorded at locations being one sensor element apart.

Alternatively, a *beam splitter* (e.g. using two dichroic prisms) is used in high-quality digital colour cameras to split light into three beams of differing wavelengths, one for the Red, one for the Green, and one for the Blue component. In this case, three $N_{cols} \times N_{rows}$ sensor arrays are used, and values R, G, and B at one pixel then actually correspond to the same pixel location.

Colour Accuracy A *colour checker* is a chart of squares showing different grey-levels or colour values. For example, see the colour checker from Macbeth™ in Figs. 1.32 and 6.5.

Example 6.1 (Colour Accuracy by Histogram Analysis) For evaluating colour accuracy, take an image of such a chart, preferably under diffuse illumination (for reducing the impact of lighting on colour appearance). Position a window within one patch of the acquired image. The histogram of such a window (if a colour patch, then three histograms for R, G, and B channels) should describe a "thin peak" for a camera with high colour accuracy. See Fig. 6.5.

Means of windows within different patches should relate (relatively, due to illumination effects) to each other as specified by the norm RGB values of those patches, provided by the producer of the colour checker.

Lens Distortion Optic lenses contribute the *radial lens distortion* to the projection process when capturing images, also known as the *barrel transform* or *pincushion transform*; see Fig. 6.6.

If a rectangular planar region is captured such that the projection centre is orthogonally in front of the centre of the region, then the region should ideally appear as a rectangle.

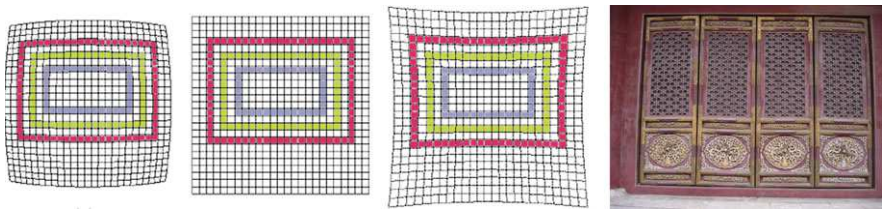


Fig. 6.6 *Left to right*: An image grid distorted by a barrel transform, an ideal rectangular image, an image grid distorted by a pincushion transform, and a projective and lens distortion combined in one image

Fig. 6.7 Grey-level bar going linearly up from *Black* (value 0) to *White* (value $G_{\max}$)

Example 6.2 (Quantifying Lens Distortion) By capturing a regular grid (e.g., a checker board), the deviation of captured lines from the ideal of straight lines can be used to characterize the lens distortion of a camera. Effects of lens distortion depend on the distance to the test pattern and appear often together with projective distortions. See Fig. 6.6, right.

Linearity of a Camera Cameras are often designed in a way that they correspond to the perceived brightness in the human eye, which in nonlinear. For image analysis purposes, we either turn off the nonlinearity of created values, or, if not possible, it might be desirable to know a correction function for mapping captured intensities into linearly distributed intensities.

Patches of grey values, such as the bottom row of patches (representing grey levels) on the Macbeth™ colour checker or the linear bar in Fig. 6.7, can be used for testing the *linearity* of the measured intensity values $M = (R + G + B)/3$.

Assume that a black patch results in the mean intensity value $u_{\min}$ ($= 0$ in the ideal case) and a white patch results in the mean intensity value $u_{\max}$ ($= G_{\max}$ in the ideal case). Now consider a patch which is a % white (i.e. $(100 - a)$ % black). Then this patch should get the corresponding linear value

$$u_{\min} + \frac{a}{100}(u_{\max} - u_{\min}) \tag{6.1}$$

between $u_{\min}$ and $u_{\max}$. Deviations from this expectation define correction values.

6.1.2 Central Projection

Ignoring radial distortions caused by the used optical lens, a projection through a small hole can be described by the theoretical *model of a pinhole camera*.

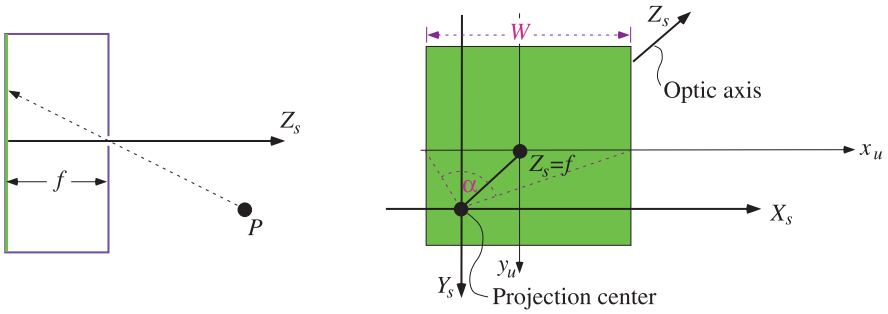


Fig. 6.8 *Left*: A sketch of an existing pinhole camera; a point P is projected through the hole (the projection centre) onto the image plane at distance f behind the hole; the projected image appears top-down. *Right*: A model of a pinhole camera, with an image of width W and viewing angle α; in the model we assume that the image plane is between world and projection centre

Model of a Pinhole Camera In this model, the diameter of the hole is assumed to be "very close" to zero. Existing pinhole cameras, also known as "shoebox cameras" (using either film or sensor matrices for image recording; see the web for photo examples) use indeed very small pinholes and long exposure times.

See Fig. 6.8, right, for the model of a pinhole camera. The pinhole is here called the *projection centre*. For avoiding top-down reversed images, the model has the projection centre behind the image plane.

We assume a right-hand $X_s Y_s Z_s$ camera coordinate system.[1] The Z_s-axis points into the world, called the *optic axis*. Because we exclude the consideration of radial distortion, we have *undistorted projected points* in the image plane with coordinates x_u and y_u. The distance f between the $x_u y_u$ image plane and the projection centre is the *focal length*.

In cases where the value of f is not (even in some abstract sense) defined by a focal length of a camera, it can also be called the *projection parameter*.

An ideal pinhole camera has a *viewing angle* (see Fig. 6.8, right) of

$$\alpha = 2 \arctan \frac{W}{2f}$$

The focal length f typically starts at 14 mm and can go up to multiples of 100 mm. For example, for $W = 36$ mm and $f = 14$ mm, the horizontal viewing angle equals about $\alpha = 104.25°$.[2] This model of a pinhole camera uses notions of optics in an abstract sense; it disregards the wave nature of light by assuming ideal geometric rays. It also assumes that objects are in focus, whatever their distance is to the camera. If projecting a visible surface point at close range, under practical circumstances we

[1] The subscript "s" comes from "sensor"; the camera is a particular sensor for measuring data in the 3D world. A laser range-finder or radar are other examples of sensors.

[2] For readers who prefer to define a *wide angle* accurately: let it be any angle greater than this particular $\alpha = 104.25°$, with 360° as an upper bound.

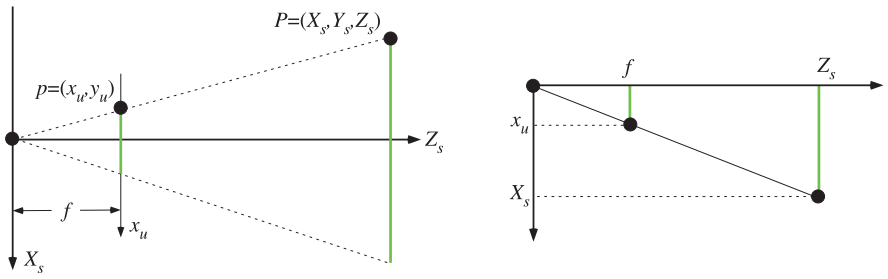


Fig. 6.9 *Left*: The central projection in the X_sZ_s plane for focal length f. *Right*: An illustration of the ray theorem for x_u to X_s and f to Z_s

would have to focus an applied camera to this range (the parameter f of the camera increases to some $f + z$ this way).

Central Projection Equations The $X_sY_sZ_s$ Cartesian camera coordinate system can be used for representing any point in the 3D world. A visible point $P = (X_s, Y_s, Z_s)$ in the world is mapped by the *central projection* into a pixel location $p = (x_u, y_u)$ in the undistorted image plane; see Fig. 6.9, left. The ray theorem of elementary geometry tells us that f to Z_s (of point P) is the same as x_u (of pixel location p) to X_s (of point P), with analogous ratios in the Y_sZ_s plane. Thus, we have that

$$x_u = \frac{fX_s}{Z_s} \quad \text{and} \quad y_u = \frac{fY_s}{Z_s} \tag{6.2}$$

In the following we make use repeatedly of those two equations in (6.2).

The Principal Point Figure 6.8, right, illustrates that the optical axis intersects the image somewhere close to its centre. In our assumed xy image coordinate system (see Fig. 1.1) we have the coordinate origin in the upper left corner of the image, and not somewhere close to its centre, as it occurs for the x_uy_u image coordinates.

Let (c_x, c_y) be the intersection point of the optical axis with the image plane in xy coordinates. This point (c_x, c_y) is called the *principal point* in the xy image plane, and it needs to be determined by camera calibration. It follows that

$$(x, y) = (x_u + c_x, y_u + c_y) = \left(\frac{fX_s}{Z_s} + c_x, \frac{fY_s}{Z_s} + c_y \right) \tag{6.3}$$

The pixel location (x, y) in our 2D xy image coordinate system also has the 3D coordinates $(x - c_x, y - c_y, f)$ in the $X_sY_sZ_s$ camera coordinate system.

6.1.3 A Two-Camera System

For understanding the 3D geometry of a scene, it is convenient to use more than just one camera. Stereo vision requires two or more cameras.

Fig. 6.10 *Left*: A stereo camera rig on a suction pad with indicated base distance b. *Right*: A forward-looking stereo camera system integrated into a quadcopter

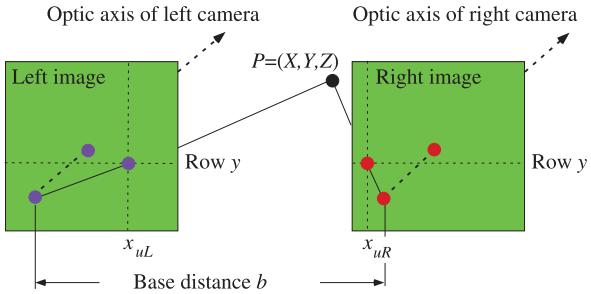


Fig. 6.11 The canonical (or standard) stereo geometry

Stereo Camera System If we use two or more cameras in a computer vision application, then they should be as identical as possible for avoiding unnecessary difficulties. Calibration will then allow us to have virtually two identical copies of the same camera. The *base distance* b is the translational distance between the projection centres of two cameras. See Fig. 6.10, left. Figure 6.10, right, shows a quadcopter where the forward-looking integrated stereo camera system has a base distance of 110 mm (a second down-looking stereo camera system in this quadcopter has a base distance of 60 mm).

After calibrating two "nearly parallel" cameras, the base distance b is the only remaining parameter defining the relative pose of one camera with respect to the other.

Canonical Stereo Geometry As a result of calibration (to be described later), assume that we have two virtually identical cameras perfectly aligned as illustrated in Fig. 6.11. We describe each camera by using the model of a pinhole camera. The *canonical stereo geometry* of two cameras (also known as the *standard stereo geometry*) is characterized by having an identical copy of the camera on the left translated by the distance b along the X_s-axis of the $X_sY_sZ_s$ camera coordinate system of the left camera. The projection centre of the left camera is at $(0, 0, 0)$ and the projection centre of the cloned right camera is at $(b, 0, 0)$. In other words, we have

1. two coplanar images of identical size $N_{cols} \times N_{rows}$,

Fig. 6.12 Omnidirectional cameras. *Left*: A fish-eye camera. *Right*: A digital camera with a hyperboloidal-shaped mirror

2. parallel optic axes,
3. an identical effective focal length f, and
4. collinear image rows (i.e., row y in one image is collinear with row y in the second image).

By applying the central projection equations of (6.2) for both cameras, a 3D point $P = (X_s, Y_s, Z_s)$ in the $X_s Y_s Z_s$ coordinate system of the left camera is mapped into undistorted image points

$$p_{uL} = (x_{uL}, y_{uL}) = \left(\frac{f \cdot X_s}{Z_s}, \frac{f \cdot Y_s}{Z_s} \right) \tag{6.4}$$

$$p_{uR} = (x_{uR}, y_{uR}) = \left(\frac{f \cdot (X_s - b)}{Z_s}, \frac{f \cdot Y_s}{Z_s} \right) \tag{6.5}$$

in the left and right image planes, respectively. Calibration has to provide accurate values for b and f for being able to use those equations when doing stereo vision.

6.1.4 Panoramic Camera Systems

Panoramic imaging sensor technology contributes to computer vision, computer graphics, robot vision, or arts. Panoramic camera systems can either record a wide-angle image in one shot or are designed for recording multiple images, to be stitched or combined into one wide-angle image.

Omnidirectional Camera System Omnidirectional camera systems observe a 360-degree field of view; see Fig. 6.12 for two examples of such cameras. Omnidirectional imaging can be classified into catadioptric or dioptric systems.[3] A catadioptric system is a combination of a quadric mirror and a conventional camera; a

[3] *Catadioptric*: pertaining to, or involving both the reflection and the refraction of light; *dioptric*: relating to the refraction of light.

Fig. 6.13 *Upper row*: Original fish-eye images (180-degree fields of view, showing Prague castle and a group of people). *Lower row*: Resulting panoramic images

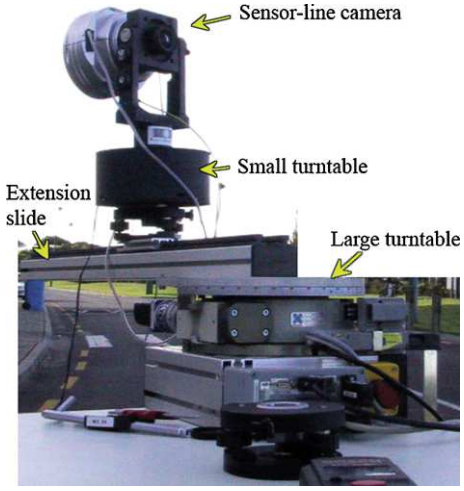


Fig. 6.14 An experimental rotating sensor-line camera configuration using a sensor-line camera mounted on a small turntable (for selecting a fixed viewing angle ω with respect to the normal of the rotation circle defined by the big turntable), which is on an extension slide, thus allowing us to chose a fixed distance R from the rotation centre of the big turntable

dioptric system has a specially designed refractor, which controls the angles of rays passing through the optical lens of the camera.

Figure 6.13 shows examples of recorded images. A mapping of a captured wide-angle field of view into a *cylindric panorama* is a solution to support common subsequent image analysis. Single-centre cylindric images possess perspective-like appearance and suppress circular distortion as given in catadioptric or dioptric images.

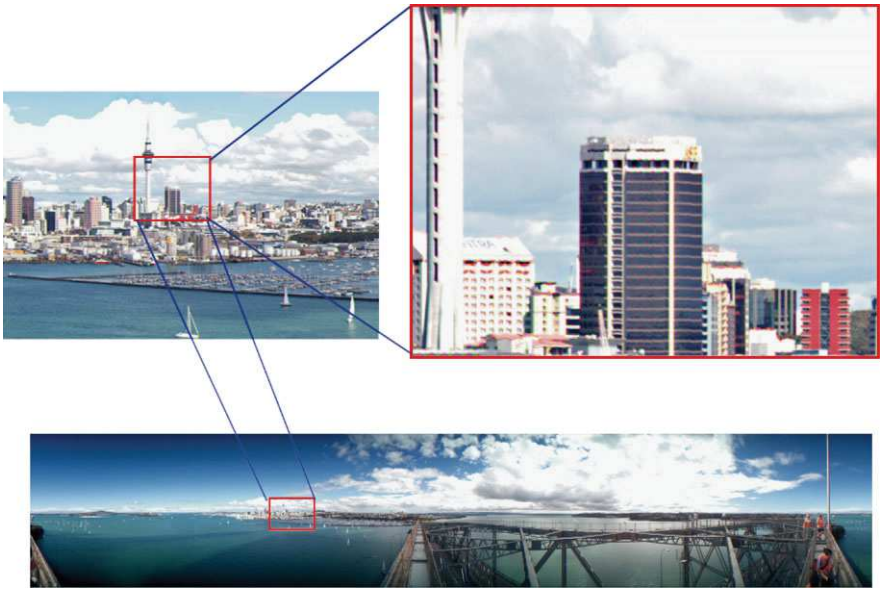


Fig. 6.15 A panoramic image of Auckland CBD recorded in 2001 from the top of Auckland Harbour Bridge using the sensor-line camera shown in Fig. 6.14

Rotating Sensor-Line Camera System A rotating sensor-line camera produces cylindric panoramas when used in a configuration as illustrated in Fig. 6.14. The configuration is basically characterized by radius R and viewing angle ω.

The sensor-line camera records in one shot $1 \times N_{rows}$ pixels. It records subsequently (say, N_{cols} times) images during one rotation, thus allowing us to merge those N_{cols} line-images into one $N_{cols} \times N_{rows}$ array-image.

A benefit is that the length N_{rows} of the sensor line can be several thousands of pixels. The rotating sensor-line camera may record 360° panoramic images within a time frame needed for taking N_{cols} individual shots during one full rotation. Figure 6.15 shows an example of a $56{,}580 \times 10{,}200$ panorama captured in 2002 by a rotating sensor-line camera. The technology records dynamic processes in subsequent single-line images, which might be disturbing or desirable, depending on interests.

Stereo Vision with a Rotating Sensor-Line Camera System A rotating sensor-line camera system can record a *stereo panorama* by rotating it once with a viewing angle ω and then again with a viewing angle $-\omega$, thus recording two cylindric array-images during both rotations, which define a stereo pair of images. See Fig. 6.16. If using a matrix-camera (i.e. of standard pinhole type), then it is sufficient to rotate this camera once and to compose panoramic images for a symmetric pair of angles ω and $-\omega$ just from a pair of image columns symmetric to the principle point of the camera used.

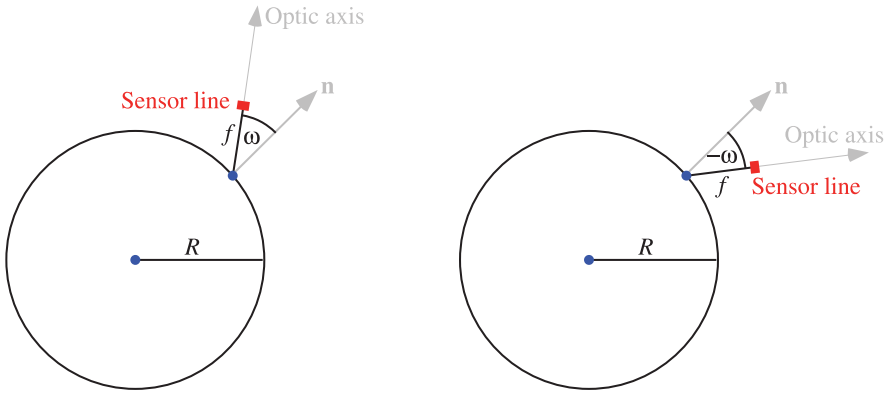


Fig. 6.16 *Left*: A top-view on a sensor-line camera with focal length f rotating at distance R to rotation centre and with viewing angle ω to normal of the rotation circle. *Right*: All the same but with viewing angle $-\omega$

Insert 6.2 (Panoramic Vision) *This is a very active field of research and applications, and we only provide two references here for further reading, the books* [K. Daniilidis and R. Klette, eds. Imaging Beyond the Pinhole Camera. Springer, Dordrecht, 2007] *and* [F. Huang, R. Klette, and K. Scheibe. Panoramic Imaging. Wiley, West Sussex, England, 2008].

6.2 Coordinates

This section discusses world coordinates, which are used as reference coordinates for cameras or objects in the scene. We also detail homogeneous coordinates, which provide a way to perform coordinate transforms uniformly by matrix multiplication (after extending the 3D world coordinate system by one more coordinate axis).

6.2.1 World Coordinates

We have cameras and 3D objects in the scenes to be analysed by computer vision. It is convenient to assume an $X_w Y_w Z_w$ *world coordinate system* that is not defined by a particular camera or other sensor. A camera coordinate system $X_s Y_s Z_s$ needs then to be described with respect to the chosen world coordinates; see Fig. 6.17. Figure 6.18 shows the world coordinate system at a particular moment during a camera calibration procedure.

Affine Transform An *affine transform* of the 3D space maps straight lines into straight lines and does not change ratios of distances between three collinear points. The mathematical representation of an affine transform is by a *linear transform*

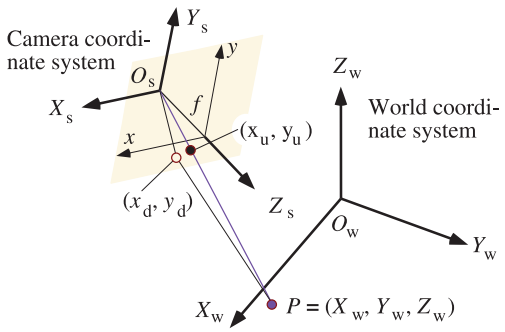


Fig. 6.17 Camera and world coordinate systems

defined by a matrix multiplication and a translation. For example, we may first apply a translation $\mathbf{T} = [t_1, t_2, t_3]^T$ followed by a rotation

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \mathbf{R}_1(\alpha) \cdot \mathbf{R}_2(\beta) \cdot \mathbf{R}_3(\gamma) \tag{6.6}$$

where

$$\mathbf{R}_1(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \tag{6.7}$$

$$\mathbf{R}_2(\beta) = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \tag{6.8}$$

$$\mathbf{R}_3(\gamma) = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.9}$$

are the individual rotations about the three coordinate axes, with *Eulerian rotation angles* α, β, and γ, one for each axis. A translation preceded by rotation would lead to a different rotation matrix and a different translation vector in general.

Observation 6.1 *Rotation and translation in the* 3*D space are uniquely determined by six parameters* α, β, γ, t_1, t_2, *and* t_3.

World and Camera Coordinates World and camera coordinates are transformed into each other by a linear (or affine) transform. Consider the affine transform of a point in the 3D space, given as $P_w = (X_w, Y_w, Z_w)$ in world coordinates, into a representation $P_s = (X_s, Y_s, Z_s)$ in camera coordinates. Besides this *coordinate notation* for points used so far, we also use the *vector notation*, such as

$P_w = [X_w, Y_w, Z_w]^T$ for a point P_w. We have that

$$(X_s, Y_s, Z_s)^T = \mathbf{R} \cdot \left[(X_w, Y_w, Z_w)^T + \mathbf{T}\right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \cdot \begin{bmatrix} X_w + t_1 \\ Y_w + t_2 \\ Z_w + t_3 \end{bmatrix} \tag{6.10}$$

for a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{T}$, which need to be specified by calibration. Note that $P_w = (X_w, Y_w, Z_w)$ and $P_s = (X_s, Y_s, Z_s)$ denote the *same* point in the 3D Euclidean space, just with respect to different 3D coordinate systems.

By multiplying the matrix and the vector in (6.10) we obtain that

$$X_s = r_{11}(X_w + t_1) + r_{12}(Y_w + t_2) + r_{13}(Z_w + t_3) \tag{6.11}$$

$$Y_s = r_{21}(X_w + t_1) + r_{22}(Y_w + t_2) + r_{23}(Z_w + t_3) \tag{6.12}$$

$$Z_s = r_{31}(X_w + t_1) + r_{32}(Y_w + t_2) + r_{33}(Z_w + t_3) \tag{6.13}$$

Projection from World Coordinates into an Image Assume that a point $P_w = (X_w, Y_w, Z_w)$ in the 3D scene is projected into a camera and visible at an image point (x, y) in the xy coordinate system. The affine transform between world and camera coordinates is as defined in (6.11) to (6.13). Using (6.3), we have in camera coordinates that

$$\begin{bmatrix} x - c_x \\ y - c_y \\ f \end{bmatrix} = \begin{bmatrix} x_u \\ y_u \\ f \end{bmatrix} = f \begin{bmatrix} X_s/Z_s \\ Y_s/Z_s \\ 1 \end{bmatrix} = f \begin{bmatrix} \frac{r_{11}(X_w+t_1)+r_{12}(Y_w+t_2)+r_{13}(Z_w+t_3)}{r_{31}(X_w+t_1)+r_{32}(Y_w+t_2)+r_{33}(Z_w+t_3)} \\ \frac{r_{21}(X_w+t_1)+r_{22}(Y_w+t_2)+r_{23}(Z_w+t_3)}{r_{31}(X_w+t_1)+r_{32}(Y_w+t_2)+r_{33}(Z_w+t_3)} \\ 1 \end{bmatrix} \tag{6.14}$$

where we also model the shift in the image plane by $(c_x, c_y, 0)$ into the principal point in undistorted image coordinates.

6.2.2 Homogeneous Coordinates

In general, it is of benefit to use *homogeneous coordinates* rather than just inhomogeneous coordinates (as so far in this text). Just to mention one benefit: in homogeneous coordinates, the subsequent steps of matrix multiplication and vector addition in an affine transform, as, for example, in (6.10), reduce to just one matrix multiplication.

Homogeneous Coordinates in the Plane We first introduce homogeneous coordinates in the plane before moving on to the 3D space. Instead of using only coordinates x and y, we add a third coordinate w. Assuming that $w \neq 0$, (x', y', w) represents now the point $(x'/w, y'/w)$ in the usual 2D inhomogeneous coordinates; the scale of w is unimportant, and thus we call (x', y', w) *homogeneous coordinates* for a 2D point $(x'/w, y'/w)$. Obviously, we can decide to use only $w = 1$ for representing points in the 2D plane, with $x = x'$ and $y = y'$.

Of course, you noticed that there is also the option to have $w = 0$. Homogeneous coordinates $(x, y, 1)$ define existing points, and coordinates $(x, y, 0)$ define *points at infinity*.

Lines in the Plane A straight line in the plane is now represented by the equation

$$a \cdot x + b \cdot y + 1 \cdot c = [a, b, c]^T \cdot [x, y, 1] = 0 \tag{6.15}$$

Consider two straight lines $\gamma_1 = (a_1, b_1, c_1)$ and $\gamma_2 = (a_2, b_2, c_2)$ in the plane, represented in the introduced homogeneous representation. They intersect at the point

$$\gamma_1 \times \gamma_2 = (b_1c_2 - b_2c_1, a_2c_1 - a_1c_2, a_1b_2 - a_2b_1) \tag{6.16}$$

given in homogeneous coordinates. Formula (6.16) is also known as the *cross product* of two vectors. For parallel lines, we have that $a_1b_2 = a_2b_1$; the parallel lines intersect at a point at infinity. The calculus using homogeneous coordinates applies uniformly for existing points as well as for points at infinity.

Consider two different points $p_1 = (x_1, y_1, w_1)$ and $p_2 = (x_2, y_2, w_2)$; they define (i.e. are incident with) the line $p_1 \times p_2$. For example, assume that one of the two points is at infinity, say $p_1 = (x_1, y_1, 0)$. Then we have with $p_1 \times p_2 = (y_1w_2, x_1w_2, x_1y_2 - x_2y_1)$ an existing straight line. The point $p_1 = (x_1, y_1, 0)$ is at infinity in direction $[x_1, y_1]^T$.

If both points are at infinity, i.e. $w_1 = w_2 = 0$, then we have a straight line $p_1 \times p_2 = (0, 0, x_1y_2 - x_2y_1)$ at infinity; note that $x_1y_2 \neq x_2y_1$ for $p_1 \neq p_2$ in this case.

Example 6.3 Consider a point $p = (x, y)$ and a translation $\mathbf{t} = [t_1, t_2]^T$ in inhomogeneous 2D coordinates. The multiplication

$$\begin{bmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{bmatrix} \cdot [x, y, 1]^T = [x + t_1, y + t_2, 1]^T \tag{6.17}$$

results in the point $(x + t_1, y + t_2)$ in inhomogeneous coordinates.

Observation 6.2 *Homogeneous coordinates allow us to perform uniquely defined calculations in the plane also covering the cases that we were not able to express before in our calculus when using only inhomogeneous xy coordinates.*

Homogeneous Coordinates in 3D Space A point $(X, Y, Z) \in \mathbb{R}^3$ is represented by (X', Y', Z', w) in homogeneous coordinates, with $(X, Y, Z) = (X'/w, Y'/w, Z'/w)$. Affine transforms can now be represented by 4×4 matrix multiplications.

Example 6.4 Consider a point $P = (X, Y, Z)$ and a translation $\mathbf{t} = [t_1, t_2, t_3]^T$ in inhomogeneous 3D coordinates. The multiplication

$$\begin{bmatrix} 1 & 0 & 0 & t_1 \\ 0 & 1 & 0 & t_2 \\ 0 & 0 & 1 & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot [X, Y, Z, 1]^T = [X + t_1, Y + t_2, Z + t_3, 1]^T \tag{6.18}$$

results in the point $(X + t_1, Y + t_2, Z + t_3)$ in inhomogeneous coordinates.

Now consider an affine transform defined by rotation and translation, as given in (6.10). The 4×4 matrix multiplication

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot [X, Y, Z, 1]^T = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \cdot [X, Y, Z, 1]^T = [X_s, Y_s, Z_s, 1]^T \tag{6.19}$$

results in the point (X_s, Y_s, Z_s) in inhomogeneous coordinates.

By means of (6.19) we also introduced a notation of 4×4 matrices by means of a 3×3 submatrix $\mathbf{R}$, a column 3-vector $\mathbf{t}$, and a row 3-vector $\mathbf{0}^T$. We will use such a notation sometimes in the following.

6.3 Camera Calibration

Camera calibration specifies *intrinsic* (i.e. camera-specific) and *extrinsic* parameters of a given one- or multi-camera configuration.

Intrinsic or internal parameters are the (effective) focal length, dimensions of the sensor matrix, sensor cell size or aspect ratio of sensor height to width, radial distortion parameters, coordinates of the principal point, or the scaling factor. Extrinsic parameters are those of the applied affine transforms for identifying poses (i.e. location and direction) of cameras in a world coordinate system.

This section provides an overview on calibration such that you can perform camera calibration with calibration software as available on the net, with sufficient background knowledge for understanding what is happening *in principle*. The section is not detailing any particular calibration method, which is outside the scope of this textbook.

6.3.1 A User's Perspective on Camera Calibration

A camera-producer specifies normally some internal parameters (e.g. the physical size of sensor cells). The given data are often not accurate enough for being used in a computer vision application.

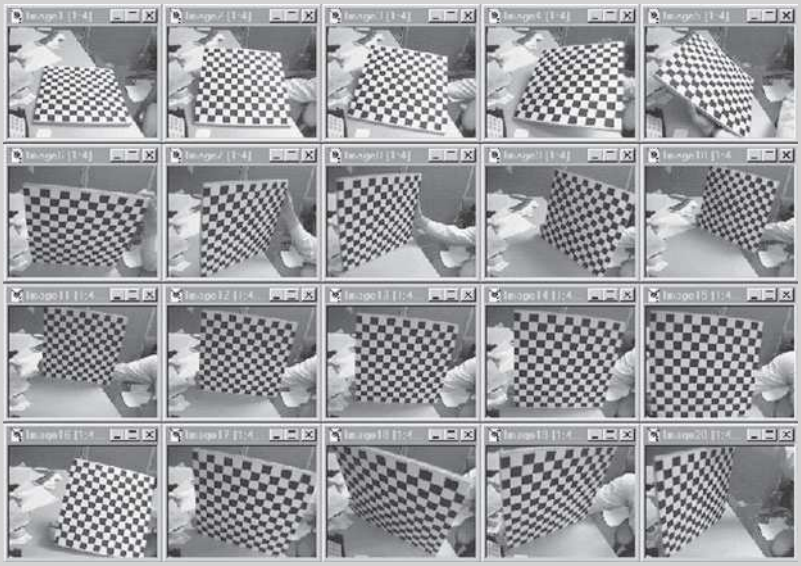
A Quick Guide For camera calibration, we use geometric patterns on 2D or 3D surfaces that we are able to measure very accurately. For example, we can use a calibration rig that is either attached to walls (i.e. permanently positioned) or dynamically moving in front of the camera system while taking multiple images; see Fig. 6.18, right. The used geometric patterns are recorded, localized in the resulting images, and their appearance in the image grid is compared with the available measurements about their geometry in the real world.

Calibration may be done by dealing with only one camera (e.g. of a multi-camera system) at a time assuming that cameras are static or that we only calibrate internal parameters.

Typically, we have a movable multi-camera system in computer vision, and we follow a multi-camera approach for calibration, aiming at calibrating internal and external parameters. Recording may commence after having the parameters needed for calibration specified and the appropriate calibration rig and software at hand. Calibration needs to be redone from time to time.

When calibrating a multi-camera system, all cameras need to be exactly time-synchronized, especially if the calibration rig moves during the procedure.

Insert 6.3 (Calibration Software) *There is calibration software available online, such as the C sources provided by J.-Y. Bouget: a calibration rig is recorded under various poses*



and processed as described on www.vision.caltech.edu/bouguetj/calib_doc/ or in the `OpenCV` *library.*

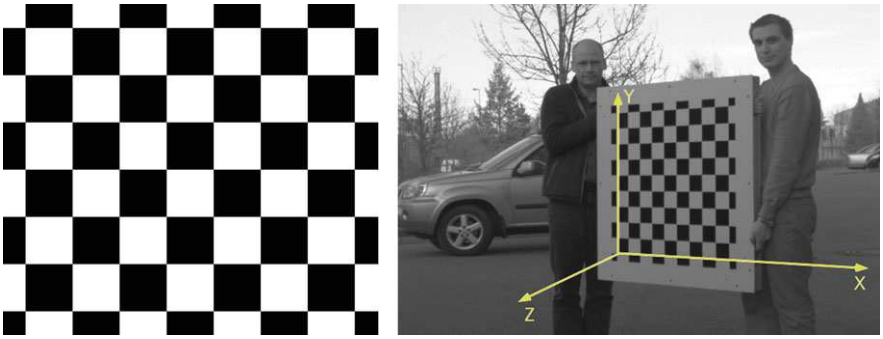


Fig. 6.18 *Left*: A 2D checkerboard pattern as commonly used for camera calibration. *Right*: A portable calibration rig; visible light reflections on the pattern would be a drawback when analysing recorded images of a calibration rig. Where is the world coordinate system?

Every new placement of the calibration rig in the 3D scene defines a different world coordinate system. Calibration provides internal and relative (i.e., one camera to other cameras) parameters. It is convenient for calibration accuracy if the calibration rig "fills" a captured image.

Involved Transforms Each camera comes with its own camera coordinate system, having the origin at its projection centre as shown in Fig. 6.8, right. The calibration rig is commonly used for defining the world coordinates at the moment when taking an image (see Fig. 6.18, right). We need to consider the following transforms:

1. a coordinate transform from world coordinates (X_w, Y_w, Z_w) into camera coordinates (X_s, Y_s, Z_s),
2. a central projection of (X_s, Y_s, Z_s) into undistorted image coordinates (x_u, y_u),
3. the lens distortion involved, mapping (x_u, y_u) into the actually valid (i.e. distorted) coordinates (x_d, y_d); see Fig. 6.17,
4. a shift of $x_d y_d$ coordinates defined by the principal point (x_c, y_c), defining the sensor coordinates (x_s, y_s), and finally,
5. the mapping of sensor coordinates (x_s, y_s) into image memory coordinates (x, y) (i.e. the actual address of a pixel), as specified in Fig. 1.1.

Lens Distortion The mapping from a 3D scene into 2D image points combines a perspective projection and a deviation from the model of a pinhole camera, caused by radial lens distortion (see Sect. 6.1.1).

A (simplified) rule: Given a lens-distorted image point $p_d = (x_d, y_d)$, we can obtain the corresponding undistorted image point $p_u = (x_u, y_u)$ as follows:

$$x_u = c_x + (x_d - c_x)\left(1 + \kappa_1 r_d^2 + \kappa_2 r_d^4 + e_x\right) \tag{6.20}$$

$$y_u = c_y + (y_d - c_y)\left(1 + \kappa_1 r_d^2 + \kappa_2 r_d^4 + e_y\right) \tag{6.21}$$

for principal point (c_x, c_y) and $r_d = \sqrt{(x_d - c_x)^2 + (y_d - c_y)^2}$.

The errors e_x and e_y are insignificant and can be assumed to be zero. There is experimental evidence that approximating these series with only two lower-order components κ_1 and κ_2 corrects more than 90 % of the radial distortion. Using only the first-order radial symmetric distortion parameter κ_1 allows a precision of about 0.1 pixels in the image sensor array.

Lens distortion needs to be calibrated for each of the cameras, and this process may be separated from the remaining calibration processes. After having lens distortion corrected, the camera may be viewed as being an implementation of the pinhole-camera model.

Designing a Calibration Method First, we need to define the set of parameters to be calibrated and a corresponding camera model having those parameters involved. For example, if the radial distortion parameters κ_1 and κ_2 need to be calibrated, then the camera model needs to include (6.20) and (6.21).

If we already know the radial distortion parameters and have used those for mapping recorded distorted images into undistorted images, then we can use equations such as (6.14). Points (X_w, Y_w, Z_w) on the calibration rig (e.g. corners of the squares) or *calibration marks* (i.e. special marks in the 3D scene where calibration takes place) are known by their physically measured world coordinates. For each point (X_w, Y_w, Z_w), we need to identify the corresponding point (x, y) (if possible, with subpixel accuracy) that is the projection of (X_w, Y_w, Z_w) in the image plane. Having, for example, 100 different points (X_w, Y_w, Z_w), this defines 100 equations in the form of (6.14), where only c_x, c_y, f, r_{11} to r_{33}, t_1, t_2, and t_3 appear as unknowns. We have an overdetermined equational system and need to find a "clever" optimization procedure for solving it for those few unknowns.

We can decide to refine our camera model. For example, we like to make a difference between a focal length f_x in the x-direction and a focal length f_y in the y-direction; we also like to include the edge length e_x and e_y of the sensor cells in the used sensor matrix in the camera, the transition from camera coordinates in world units to homogeneous camera coordinates in pixel units, or also a shearing factor s for evaluating the orthogonality of the recorded image array.

All such parameters can be used to add further details into the basic equation (6.14). Accordingly, the resulting equational systems will become more complex have more unknowns.

Thus, we briefly summarize the *general procedure*: known positions (X_w, Y_w, Z_w) in the world are related to identifiable locations (x, y) in recorded images. The equations defining our camera model then contain X_w, Y_w, Z_w, x, and y as known values and intrinsic or extrinsic camera parameters as unknowns. The resulting equation system (necessarily nonlinear due to central projection or radial distortion) needs to be solved for the specified unknowns, where over-determined situations provide for stability of a used numeric solution scheme.

We do not discuss any further such equation systems or solution schemes in this textbook.

Manufacturing a Calibration Board A rigid board wearing a black and white checkerboard pattern is common. It should have 7×7 squares at least. The squares need to be large enough such that their minimum size, when recorded on the image plane during calibration, is 10×10 pixels at least (i.e. having a camera with an effective focal length f, this allows us to estimate the size of $a \times a$ cm for each square assuming a distance of b m between the camera and board).

A rigid and planar board can be achieved by having the calibration grid, e.g. printed onto paper or by using equally sized black squares, those glued onto a rigid board. This method is relatively cheap and reliable.

This grid can be created with any image-creating tool as long as the squares are all exactly of the same size.

Localizing Corners in the Checkerboard For the checkerboard, *calibration marks* are the corners of the squares, and those can be identified by approximating intersection points of grid lines, thus defining the corners of the squares potentially with subpixel accuracy.

For example, assume 10 vertical and 10 horizontal grid lines on a checkerboard, as it is the case in Fig. 6.18, right. Then this should result in $10 + 10$ peaks in the $d\alpha$ Hough space for detecting line segments. Each peak defines a detected grid line, and the intersection points of those define the corners of the checkerboard in the recorded image.

Applying this method requires that lens distortion has been removed from the recorded images prior to applying the Hough-space method. Images with lens distortion will show bended lines rather than perfectly straight grid lines.

Localizing Calibration Marks A calibration pattern can also be defined by marks such as circular or square dots. For example, this is a popular choice if cameras are calibrated in the same location, thus calibration marks can be permanently painted on walls or other static surfaces.

Assume that we identify an image region S of pixels as the area that shows a calibration mark, say, in grey levels. The position of the calibration mark can then be identified at subpixel accuracy by calculating the centroid of this region, as already discussed in Example 3.7.

6.3.2 Rectification of Stereo Image Pairs

Consider a two-camera recording system as discussed in Sect. 6.1.3. This is the common input for stereo vision, a basic procedure in computer vision for obtaining distance data by just using visual information.

The complexity of stereo vision is mainly defined by the task to identify the corresponding points in pairs of input images, recorded by two cameras. This task is the subject of Chap. 8. For reducing this complexity, it is convenient to warp the recorded image pairs such that it appears that they are actually recorded in canonical

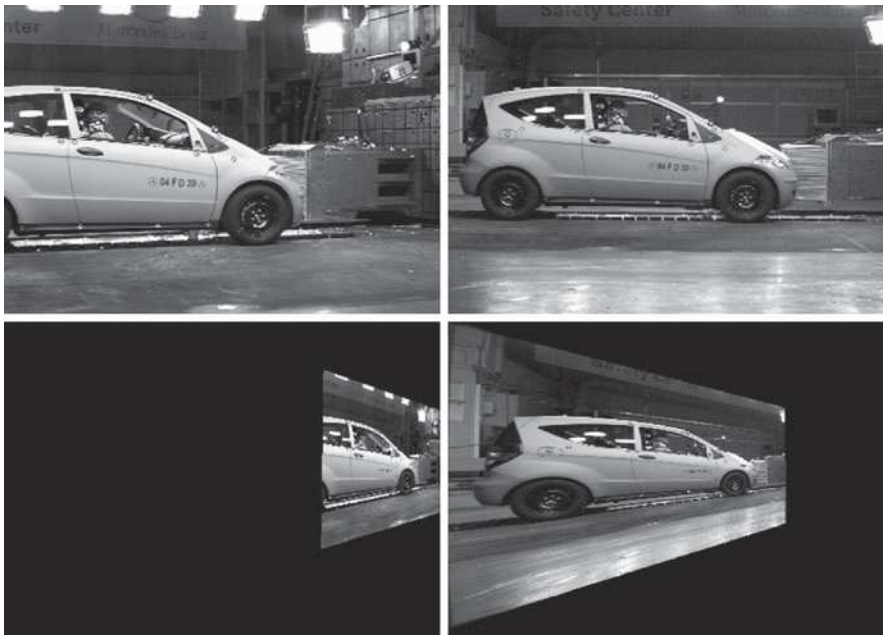


Fig. 6.19 *Top*: Images recorded by two cameras with significant differences in extrinsic parameters and also (insignificant) differences in intrinsic parameters. *Bottom*: Two geometrically rectified images taken at different viewing locations by two cameras installed in a crash-test hall

stereo geometry (by a pair of identical cameras). We call this process short *geometric rectification*, without further mentioning the context of stereo vision (geometric rectification is also of relevance in other contexts).

A Multi-camera System Often it is actually insufficient to use just two cameras for applying computer vision to complex environments or processes. For example, in a crash-test hall in the automotive industry there are many high-definition very fast cameras installed for recording the few seconds of a crash test from different viewing angles. Figure 6.19 illustrates two geometrically rectified images. How to achieve this?

We will answer this question in this subsection. We do not restrict the discussion to just a left camera and a right camera. We consider a general case of a Camera i or Camera j, where the numbers i and j identify different cameras in a multi-camera system.

The Camera Matrix As (examples of) intrinsic camera parameters of Camera i, we consider here

1. the edge lengths e_i^x and e_i^y of camera sensor cells (defining the aspect ratio),
2. a skew parameter s_i,
3. the coordinates of the principal point $\mathbf{c}_i = (c_i^x, c_i^y)$ where the optical axis of Camera i and image plane intersect, and

4. the focal length f_i.

We assume that the lens distortion has been calibrated before and does not need to be included anymore in the set of intrinsic parameters.

Instead of the simple equation (6.14), defining a camera model just based on the intrinsic parameters f, c_x and c_y, we have now a refined projection equation in 4D homogeneous coordinates, mapping a 3D point $P = (X_w, Y_w, Z_w)$ into the image coordinates $p_i = (x_i, y_i)$ of the ith camera (as defined by Fig. 1.1) as follows:

$$k \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_i/e_i^x & s_i & c_i^x & 0 \\ 0 & f_i/e_i^y & c_i^y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_i & -\mathbf{R}_i^T \mathbf{t}_i \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$
$$= [\mathbf{K}_i | \mathbf{0}] \cdot \mathbf{A}_i \cdot [X_w, Y_w, Z_w, 1]^T \tag{6.22}$$

where $\mathbf{R}_i$ and $\mathbf{t}_i$ denote the rotation matrix and translation vector in 3D inhomogeneous world coordinates, and $k \neq 0$ is a scaling factor.

By means of (6.22) we defined a 3×3 matrix $\mathbf{K}_i$ of intrinsic camera parameters, and a 4×4 matrix $\mathbf{A}_i$ of extrinsic parameters (of the affine transform) of Camera i. The 3×4 *camera matrix*

$$\mathbf{C}_i = [\mathbf{K}_i | \mathbf{0}] \cdot \mathbf{A}_i \tag{6.23}$$

is defined by 11 parameters if we allow for an arbitrary scaling of parameters; otherwise, it is 12.

Common Viewing Direction for Rectifying Cameras i and j We identify a common viewing direction for Cameras i and j, replacing the given viewing directions along the optical axes of those two cameras. Let Π be a plane perpendicular to the baseline vector $\mathbf{b}_{ij}$ from the projection centre of Camera i to the projection centre of Camera j. See Fig. 6.20.

We project the unit vectors $\mathbf{z}_i^\circ$ and $\mathbf{z}_j^\circ$ of both optical axes into Π, which results in vectors $\mathbf{n}_i$ and $\mathbf{n}_j$, respectively. The algebraic relations are as follows:

$$\mathbf{n}_i = \left(\mathbf{b}_{ij} \times \mathbf{z}_i^\circ\right) \times \mathbf{b}_{ij} \quad \text{and} \quad \mathbf{n}_j = \left(\mathbf{b}_{ij} \times \mathbf{z}_j^\circ\right) \times \mathbf{b}_{ij} \tag{6.24}$$

We could also have used $\mathbf{b}_{ji}$ in both equations, but then uniformly four times.

Aiming at a "balanced treatment" of both cameras, we use the bisector of $\mathbf{n}_i$ and $\mathbf{n}_j$ for defining the unit vector

$$\mathbf{z}_{ij}^\circ = \frac{\mathbf{n}_i + \mathbf{n}_j}{\|\mathbf{n}_i + \mathbf{n}_j\|_2} \tag{6.25}$$

of the common direction.

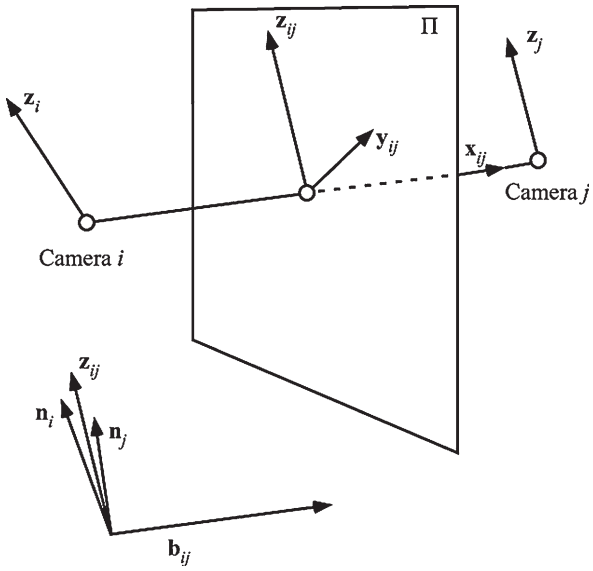


Fig. 6.20 An illustration for calculating the common viewing direction for Cameras i and j

Consider the unit vector $\mathbf{x}_{ij}^{\circ}$ in the same direction as $\mathbf{b}_{ij}$, and the unit vector $\mathbf{y}_{ij}^{\circ}$ is finally defined by the constraint of ensuring (say) a left-hand 3D Cartesian coordinate system. Formally, we have that

$$\mathbf{x}_{ij}^{\circ} = \frac{\mathbf{b}_{ij}}{\|\mathbf{b}_{ij}\|_2} \quad \text{and} \quad \mathbf{y}_{ij}^{\circ} = \mathbf{z}_{ij} \times \mathbf{x}_{ij}^{\circ} = -\mathbf{x}_{ij}^{\circ} \times \mathbf{z}_{ij} \tag{6.26}$$

In general, for any vectors $\mathbf{a}$ and $\mathbf{b}$, $(\mathbf{a}, \mathbf{b}, \mathbf{a} \times \mathbf{b})$ defines a left-hand tripod. We have the left-hand tripod $(\mathbf{x}_{ij}^{\circ}, \mathbf{z}_{ij}^{\circ} \times \mathbf{x}_{ij}^{\circ}, \mathbf{z}_{ij}^{\circ})$ because

$$\mathbf{x}_{ij}^{\circ} \times \left(\mathbf{z}_{ij}^{\circ} \times \mathbf{x}_{ij}^{\circ}\right) = \mathbf{z}_{ij}^{\circ}\left(\mathbf{x}_{ij}^{\circ} \cdot \mathbf{x}_{ij}^{\circ}\right) - \mathbf{x}_{ij}^{\circ}\left(\mathbf{x}_{ij}^{\circ} \cdot \mathbf{z}_{ij}^{\circ}\right) = \mathbf{z}_{ij}^{\circ} \tag{6.27}$$

and $(\mathbf{x}_{ij}^{\circ}, \mathbf{x}_{ij}^{\circ} \times \mathbf{z}_{ij}^{\circ}, \mathbf{z}_{ij}^{\circ})$ would be a right-hand tripod.

The two images of Camera i and Camera j need to be modified as though both would have been taken in the direction $\mathbf{R}_{ij} = (\mathbf{x}_{ij}\mathbf{y}_{ij}\mathbf{z}_{ij})^T$, instead of the actually used directions $\mathbf{R}_i$ and $\mathbf{R}_j$.

Producing the Rectified Image Pair The rotation matrices that rotate both cameras into their new (virtual) viewing direction are as follows:

$$\mathbf{R}_i^* = \mathbf{R}_{ij}\mathbf{R}_i^T \quad \text{and} \quad \mathbf{R}_j^* = \mathbf{R}_{ij}\mathbf{R}_j^T \tag{6.28}$$

In general, when rotating any camera around its projection centre about the matrix $\mathbf{R}$, the image is transformed by a rotation *homography* (i.e. a recalculated projective transformation)

$$\mathbf{H} = \mathbf{K} \cdot \mathbf{R} \cdot \mathbf{K}^{-1} \tag{6.29}$$

where $\mathbf{K}$ is the 3×3 matrix of intrinsic parameters of this camera. The matrix $\mathbf{K}^{-1}$ transfers pixel coordinates into camera coordinates in world units, the matrix $\mathbf{R}$ rotates them into the common plane, and the matrix $\mathbf{K}$ transfers them back into pixel coordinates.

A rectified image is calculated, pixel by pixel, using

$$p = \mathbf{H}^{-1}\,\hat{p} \tag{6.30}$$

such that the new value at pixel location $\hat{p}$ is calculated based on the original image values in a neighbourhood of a point p (which is in general not exactly a pixel location), using (e.g.) bilinear interpolation.

Creating an Identical Twin Assume that we want to have the image of Camera j after rotation homography with respect to the parameters of Camera i, i.e. we create an identical copy of Camera i at the pose of Camera j. For ensuring this effect, we simply apply the rotation homography

$$\mathbf{H}_{ij} = \mathbf{K}_i \cdot \mathbf{R}_j^* \cdot \mathbf{K}_j^{-1} \tag{6.31}$$

which first transforms by $\mathbf{K}_j^{-1}$ the points in the jth image plane into a "normalized" coordinate system, then we apply $\mathbf{R}_j^*$ to perform the desired rotation, and, finally, $\mathbf{K}_i$ for transforming the rotation result according to the parameters of Camera i.

Insert 6.4 (Fundamental and Essential Matrix of Stereo Vision) *Following a publication by H.C. Longuet-Higgins in* 1981, *Q.T. Luong identified in* 1992 *in his PhD thesis two matrices, called the fundamental matrix and essential matrix, which describe binocular stereo geometry, either with including the characteristics of the used cameras or without, respectively.*

Fundamental and Essential Matrix We go back to having just a left and a right camera. Let p_L and p_R be *corresponding stereo points*, i.e. the projections of a 3D point P in the left and right image planes. Assume that p_L and p_R are given in homogeneous coordinates. Then we have that

$$p_R^T \cdot \mathbf{F} \cdot p_L = 0 \tag{6.32}$$

for some 3×3 matrix $\mathbf{F}$, defined by the configuration (i.e. intrinsic and extrinsic parameters) of the two cameras for *any* pair p_L and p_R of corresponding stereo points.

This matrix $\mathbf{F}$ is known as the *fundamental matrix*, sometimes also called the *epipolar matrix* or the *bifocal tensor*. For example, $\mathbf{F} \cdot p_L$ defines a line in the image plane of the right camera, and any stereo point corresponding to p_L needs to be on that line. (This is an *epipolar line*, and we discuss such a line later in the book.)

The matrix $\mathbf{F}$ is of rank 2 and uniquely defined (by the left and right cameras) up to a scaling factor. In general, seven pairs of corresponding points (in general position) are sufficient to identify the matrix $\mathbf{F}$. Interestingly, there is the relation

$$\mathbf{F} = \mathbf{K}_R^{-T} \cdot \mathbf{R}[\mathbf{t}]_\times \cdot \mathbf{K}_L^{-1} \tag{6.33}$$

for camera matrices $\mathbf{K}_R$ and $\mathbf{K}_L$. We go from pixel coordinates to camera coordinates in world units. Here, $[\mathbf{t}]_\times$ is the *cross product matrix* of a vector $\mathbf{t}$, defined by $[\mathbf{t}]_\times \cdot \mathbf{a} = \mathbf{t} \times \mathbf{a}$, or

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix} \tag{6.34}$$

The matrix

$$\mathbf{E} = \mathbf{R}[\mathbf{t}]_\times \tag{6.35}$$

is also known as the *essential matrix*; it has five degrees of freedom, and it is uniquely defined (by the left and right cameras) up to scaling.

Insert 6.5 (Geometry of Multi-Camera Systems) *This chapter provided a basic introduction into geometric issues related to single- or multiple-camera systems. Standard references for geometric subjects in computer vision are, for example, the books* [R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision, Second Edition. Cambridge University Press, Cambridge, 2004] *and* [K. Kanatani. Geometric Computation for Machine Vision. Oxford University Press, Oxford, 1993].

6.4 Exercises

6.4.1 Programming Exercises

Exercise 6.1 (Evaluation of Cameras) Evaluate (at least) two different digital cameras with respect to the properties of

1. colour accuracy,
2. linearity, and
3. lens distortion.

Even two cameras of the same brand might be of interest; it is expected that they result in (slightly) different properties.

Design, calculate and print at least one test image (test pattern) for each of those three properties. Try to provide a motivation why you decided for your test images. Obviously, printer quality will influence your tests.

Take measurements repeatedly (say, under varying conditions, such as having your test images, or further test objects, at different distances to the camera, or under varying light conditions). This needs to be followed by a statistical analysis (calculate the means and standard deviations for your measurement series).

Fig. 6.21 *Top*: Registered images. *Bottom*: Stitched images. Results of a project in 1998 at the University of Auckland

Note that property comparisons will typically not lead to a simple result such as "this camera is better than the other one", rather to a more sophisticated comparison such as "for this criterion, this camera performs better under given circumstances."

Exercise 6.2 (Image Stitching) Use a common (i.e. array- or pinhole-type) digital camera on a tripod and record a series of images by rotating the camera: capture each image such that the recorded scene overlaps to some degree with the scene recorded in the previous image.

Write your own simple and straightforward program of *stitching* those images onto the surface of one cylinder or into a rectangular panorama. The first step is *image registration*, i.e. spatially aligning the recorded images and defining cuts between those. See Fig. 6.21 for an illustration of registration and stitching steps.

(*Image stitching* is an area where many solutions have been published already; check the net for some inspirations.)

For comparison, use a commercial or freely available stitching software (note that often this already comes with the camera or is available via the camera web support) for mapping the images into a 360° panorama.

Compare the results with the results of your own image stitching program.

Exercise 6.3 (Stereo Panorama) Rotate a video camera on a tripod for generating stereo panoramas as described at the end of Sect. 6.1.4. The two columns generating your views for ω and $-\omega$ need to be chosen carefully; they need to be symmetric with respect to the principal point of your video camera. If not, the generated two-view panoramas do not have proper geometry for being stereo-viewable.

Regarding the chosen radius R and viewing angle ω, there are two recommended values that maximize the number of disparity levels between the closest object of interest at distance D_1 and the furthest object of interest at distance D_2. These two (uniquely defined) values can be looked up in the second book listed in Insert 6.2. You may also experiment with taking different values R and ω.

The generated stereo panoramas become stereo-viewable by using anaglyphs. Anaglyphic images are generated by combining the channel for Red from one image with the channels for Blue and Green from the other image. The order of filters in available anaglyphic eyeglasses decides which image contributes which the channel. Demonstrate your recorded stereo panoramas using anaglyphic eyeglasses.

Exercise 6.4 (Detection of Calibration Marks) Record images for camera calibration using a checkerboard pattern as illustrated in Insert 6.3. Implement or use an available program for line detection (see Sect. 3.4.1) and detect the corners in the recorded images at subpixel accuracy. Discuss the impact of radial lens distortion on your detected corners.

Exercise 6.5 (Building a Pinhole Camera) This is not a programming exercise, but a practical challenge. It is a suggestion for those who like to experience the simplest possible camera, which you can build yourself, basically just using a shoebox and photo-sensitive paper: check out on the net for "pinhole cameras", where there are some practical hints, and there is also an interesting collection of photos online recorded with those cameras.

6.4.2 Non-programming Exercises

Exercise 6.6 Check this chapter for equations given in inhomogeneous coordinates. Express all those in homogeneous coordinates.

Exercise 6.7 Specify the point at infinity on the line $31x + 5y - 12 = 0$. Determine the homogeneous equation of this line. What is the intersection point of this line with the line $31x + 5y - 14 = 0$ at infinity?

Generalize by studying the lines $ax + by + c_1 = 0$ and $ax + by + c_2 = 0$ for $c_1 \neq c_2$.

Exercise 6.8 Consider a camera defined by the following 3×4 camera matrix:

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Compute the projections of the following 3D points (in world coordinates) with this camera:

$$\mathbf{P}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{P}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}, \quad \mathbf{P}_3 = \begin{bmatrix} 3 \\ 2 \\ 1 \\ 1 \end{bmatrix}, \quad \text{and} \quad \mathbf{P}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Exercise 6.9 Let $p_R = [x, y, 1]^\top$ and $p_L = [x', y', 1]^\top$. The equation

$$p_R^T \cdot \mathbf{F} \cdot p_L = 0$$

is equivalently expressed by

$$\begin{bmatrix} xx' & xy' & x & yx' & yy' & y & x' & y' & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{21} \\ F_{31} \\ F_{12} \\ F_{22} \\ F_{32} \\ F_{13} \\ F_{23} \\ F_{33} \end{bmatrix} = 0$$

where F_{ij} are the elements of the fundamental matrix $\mathbf{F}$. Now assume that we have at least eight pairs of corresponding pixels, defining the matrix equation

$$\begin{bmatrix} x_1x_1' & x_1y_1' & x_1 & y_1x_1' & y_1y_1' & y_1 & x_1' & y_1' & 1 \\ x_2x_2' & x_2y_2' & x_2 & y_2x_2' & y_2y_2' & y_2 & x_2' & y_2' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_nx_n' & x_ny_n' & x_n & y_nx_n' & y_ny_n' & y_n & x_n' & y_n' & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{21} \\ \vdots \\ F_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

for $n \geq 8$, expressed in short as $\mathbf{A} \cdot \mathbf{f} = \mathbf{0}$. Solve this equation for the unknowns F_{ij}, considering noise or inaccuracies in pairs of corresponding pixels.

Exercise 6.10 Show that the following is true for any nonzero vector $\mathbf{t} \in \mathbb{R}^3$:

1. $[\mathbf{t}]_x \cdot \mathbf{t} = 0$,
2. the rank of matrix $[\mathbf{t}]_x$ is 2,
3. the rank of the essential matrix $\mathbf{E} = \mathbf{R}[\mathbf{t}]_x$ is 2,
4. the fundamental matrix $\mathbf{F}$ is derived from the essential matrix $\mathbf{E}$ by the formula $\mathbf{F} = \mathbf{K}_R^{-T}\mathbf{E}\mathbf{K}_L^{-1}$,
5. the rank of the fundamental matrix $\mathbf{F}$ is 2.