CHAPTER 2

Cameras

2.1 CAMERAS

2.1.1 The Pinhole Camera

A *pinhole camera* is a light-tight box with a very small hole in the front (Figure 2.1). Think about a point on the back of the box. The only light that arrives at that point must come through the hole, because the box is light-tight. If the hole is very small, then the light that arrives at the point comes from only one direction. This means that an inverted image of a scene appears at the back of the box (Figure 2.1). An appropriate sensor (CMOS sensor; CCD sensor; light sensitive film) at the back of the box will capture this image.

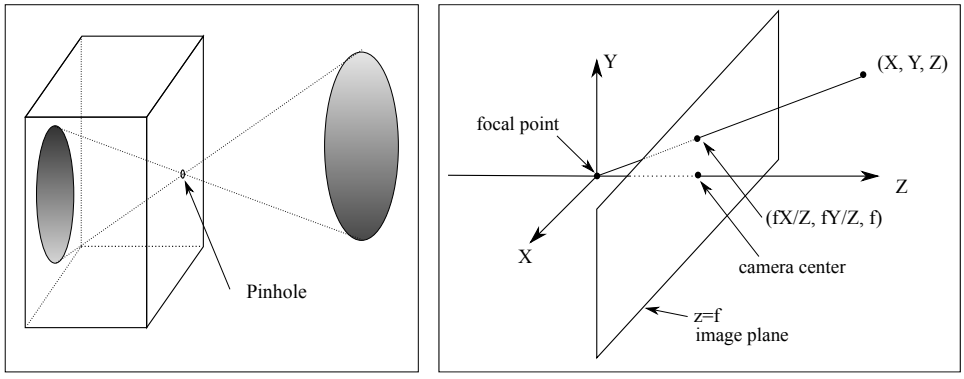

FIGURE 2.1: *The pinhole imaging model. On the* **left***, a light-tight box with a pinhole in it views an object. The only light that a point on the back of the box sees comes through the very small pinhole, so that an inverted image is formed on the back face of the box. On the* **right***, the usual geometric abstraction. The box doesn't affect the geometry, and is omitted. The pinhole has been moved to the back of the box, so that the image is no longer inverted. The image is formed on the plane $z = f$, by convention. Notice the coordinate system is* left-handed*, because the camera looks down the z-axis. This is because most people's intuition is that z* increases *as one moves into the image. The text provides some more detail on this point.*

Pinhole camera models produce an upside-down image. This is easily dealt with in practice (turn the image the right way up). An easy way to account for this is to assume the sensor is *in front* of the hole, so that the image is not upside-down. One could not build a camera like this (the sensor blocks light from the hole) but it is a convenient abstraction. There is a standard model of this camera, in a standard coordinate system. The coordinate system is left-handed even though coordinate systems in 3D are usually right-handed coordinate systems. This is because most

people's intuition is that z *increases* as one moves into the image. The pinhole – usually called the *focal point* – is at the origin, and the sensor is on the plane $z = f$. This plane is the *image plane*, and f is the *focal length*. We ignore any camera body and regard the image plane as infinite.

Under this highly abstracted camera model, almost any point in 3D will map to a point in the image plane. We *image* a point in 3D by constructing a ray through the 3D point and the focal point, and intersecting that ray with the image plane. The focal point has an important, distinctive, property: It cannot be imaged, and it is the only point that cannot be imaged.

Similar triangles yields that the point (X, Y, Z) in 3D is imaged to

$$(fX/Z, fY/Z, f)$$

on the sensor (Figure 2.1). Notice that the z-coordinate is the same for each point on the image plane, so it is quite usual to ignore it and use the model

$$(X, Y, Z) \to (fX/Z, fY/Z).$$

The projection process is known as *perspective projection*. The point where the z-axis intersects the image plane (equivalently, where the ray through the focal point perpendicular to the image plane intersects the image plane) is the *camera center*. Remarkably, in almost every publication in computer vision the camera is expressed in left-handed coordinates and everything else works in right-handed coordinates. The exercises demonstrate that there is no real difficulty here.

Remember this: *Most practical cameras can be modelled as a pinhole camera. A pinhole camera with focal length f maps*

$$(X, Y, Z) \to (fX/Z, fY/Z).$$

Figure 2.1 shows important terminology (focal point; image plane; camera center).

2.1.2 Perspective Effects

Perspective projection has a number of important properties, summarized as:

- lines project to lines;
- more distant objects are smaller;
- lines that are parallel in 3D meet in the image;
- planes have horizons;
- planes image as half-planes.

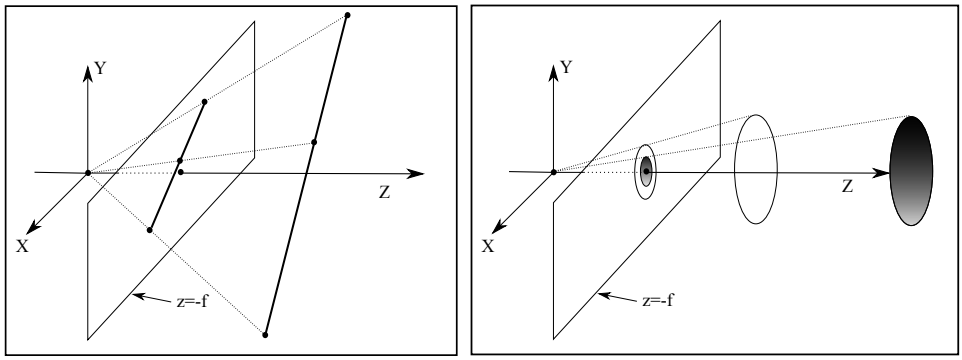

FIGURE 2.2: *Perspective projection maps almost any 3D line to a line in the image plane (***left***). Some rays from the focal point to points on the line are shown as dotted lines. The family of all such rays is a plane, and that plane must intersect the image plane in a line as long as the 3D line does not pass through the focal point. On the* **right***, two 3D objects viewed in perspective projection; the more distant object appears smaller in the image.*

Lines project to lines: Almost every line in 3D maps to a line in the image. You can see this by noticing that the image of the 3D line is formed by intersecting rays from the focal point to each point on the 3D line with the image plane. But these rays form a plane, so we are intersecting a plane with the image plane, and so obtain a line (Figure 2.2). The exceptions are the 3D lines through the focal point – these project to points.

More distant objects are smaller: The further away an object is in 3D, the smaller the image of that object, because of the division by Z (Figure 2.2).

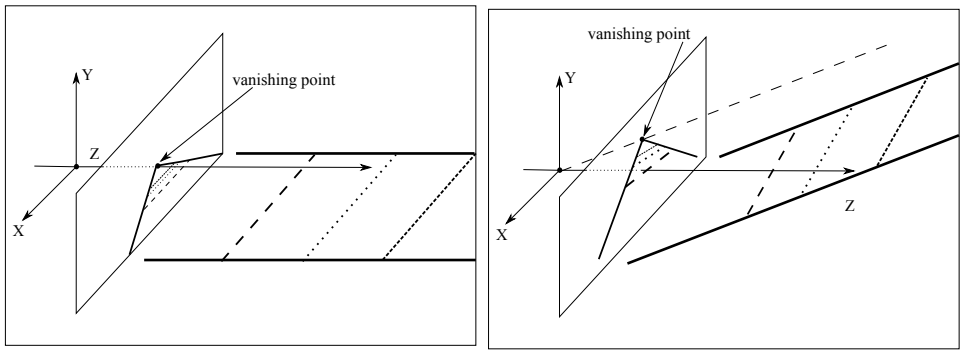

FIGURE 2.3: *Perspective projection maps a set of parallel lines to a set of lines that meet in a point. On the* **left***, a set of lines parallel to the z-axis, with "railway sleepers" shown. As these sleepers get further away, they get smaller in the image, meaning the projected lines must meet. The vanishing point (the point where they meet) is obtained by intersecting the ray parallel to the lines and through the focal point with the image plane. On the* **right***, a different pair of parallel lines with a different vanishing point. The figure establishes that, if there are more than two lines in the set of parallel lines, all will meet at the vanishing point.*

Lines that are parallel in 3D meet in the image: Now think about a set of infinitely long parallel railroad tracks. The sleepers supporting the tracks are all the same size. Distant sleepers are smaller than nearby sleepers, and arbitrarily distant sleepers are arbitrarily small. This means that parallel lines will meet in the image. The point at which the lines in a collection of parallel lines meet is known as the *vanishing point* for those lines (Figure 2.3). The vanishing point for a set of parallel lines can be obtained by intersecting the ray from the focal point and parallel to those lines with the image plane (Figure 2.3).

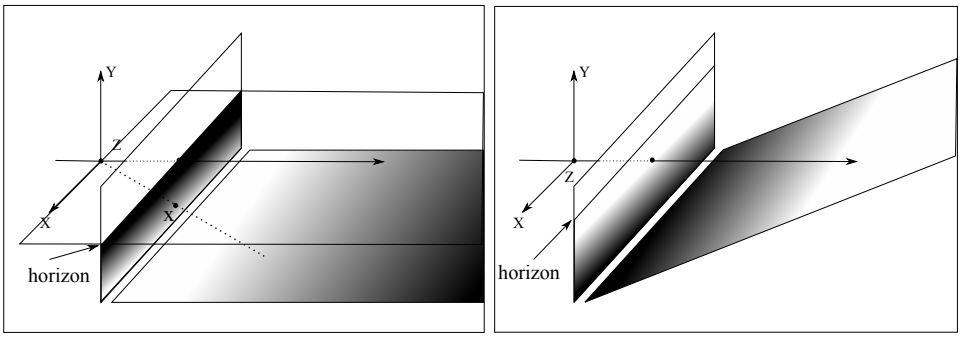

FIGURE 2.4: **Left** *shows a plane in 3D (in this case, $y = -1$). The intersection of the plane through the focal point parallel to the 3D plane (in this case, $y = 0$) and the image plane, forms an image line called the horizon. This line cuts the image plane into two parts. Construct the ray through the focal point and a point* $\mathbf{x}$ *in the image plane. For* $\mathbf{x}$ *on one side of the horizon, this ray will intersect the 3D plane in the half space $z > 0$ (and so in front of the camera, shown here). If* $\mathbf{x}$ *is on the other side of the horizon, the intersection will be in the half space $z < 0$ (and so behind the camera, where it cannot be seen).* **Right** *shows a different 3D plane with a different horizon. The gradients on the planes indicate roughly where points on the 3D plane appear in the image plane (light points map to light, dark to dark).*

Planes have horizons: Now think about the image of a plane. As Figure 2.5 shows, the plane through the focal point and parallel to that plane produce a line in the image, known as the *horizon* of the plane.

Planes image as half-planes: For an abstract perspective camera, any point on the plane can be imaged to a point on the image plane. In practical cameras, we cannot image points that lie behind the camera in 3D. Now cast a ray through the focal point and some point $\mathbf{x}$ in the image plane. If $\mathbf{x}$ is on one side of the horizon, the ray will hit the plane in the $z > 0$ half space and so we can see the plane. If it is on the other side, it will hit the plane in the $z < 0$ half space, so we cannot see the plane.

Remember this: *Under perspective projection:*

- *points project to points;*
- *lines project to lines;*
- *more distant objects are smaller;*
- *lines that are parallel in 3D meet in the image;*
- *planes have horizons;*
- *planes image as half-planes.*

2.1.3 Scaled Orthographic Projection and Orthographic Projection

Under some circumstances, perspective projection can be simplified. Assume the camera views a set of points which are close to one another compared with the distance to the camera. Write $\mathbf{X}_i = (X_i, Y_i, Z_i)$ for the i'th point, and assume that $Z_i = Z(1+\epsilon_i)$, where ϵ_i is quite small. In this case, the distance to the set of points is much larger than the *relief* of the points, which is the distance from nearest to furthest point. The i'th point projects to $(fX_i/Z_i, fY_i/Z_i)$, which is approximately $(f(X_i/Z)(1-\epsilon_i), f(Y_i/Z)(1-\epsilon_i))$. Ignoring ϵ_i because it is small, we have the projection model

$$(X, Y, Z) \to (f/Z)(X, Y) = s(X, Y).$$

This model is usually known as *scaled orthograpic projection*. The model applies quite often. One important example is pictures of people. Very often, all body parts are roughly the same distance from the camera — think of a side view of a pedestrian seen from a motor car. Scaled orthographic projection applies in such cases. It is not always an appropriate model. For example, when a person is holding up a hand to block the camera's view, perspective effects can be significant (Figure **??**).

Occasionally, it is useful to rescale the camera (or assume that $f/Z = 1$), yielding $(X, Y, Z) \to (X, Y)$. This is known as *orthographic projection*.

Remember this: *Scaled orthographic projection maps*

$$(X, Y, Z) \to s(X, Y)$$

where s is some scale. The model applies when the distance to the points being viewed is much greater than their relief. Many views of people have this property.

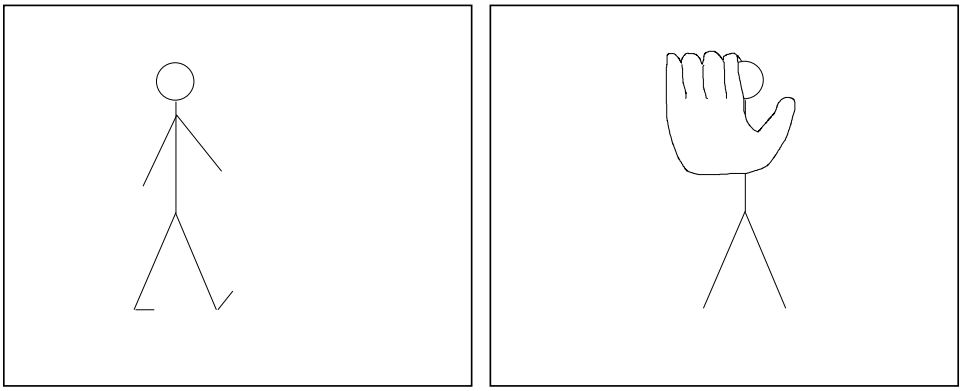

FIGURE 2.5: *The pedestrian on the* **left** *is viewed from some way away, so the distance to the pedestrian is much larger than the change in depth over the pedestrian. In this case, which is quite common for views of people, scaled orthography will apply. The celebrity on the* **right** *is holding a hand up to prevent the camera viewing their face; the hand is quite close to the camera, and the body is an armslength away. In this case, perspective effects are strong. The hand looks big because it is close, and the head looks small because it is far.*

2.1.4 Lenses

One practical version of a pinhole camera is a *camera obscura* – the box is built as a room, and you can stand in the room and see the view on the back wall (some examples are at `https://www.atlasobscura.com/lists/camera-obscura-places`; the internet yields amusing disputes about the correct plural form of the term). You can also build a simple pinhole camera with a matchbox, some tape, a pin, and some light sensitive film do the trick. Getting good images takes trouble, however.

A large hole in front of the camera will cause the image at the back to be brighter, but blurrier, because each point on the sensor will average light over all directions that happen to go through the hole. If the hole is smaller, the image will get sharper, but darker. In practical cameras, achieving an image that is both bright and focused is the job of the lens system. There may be one or several lenses that light passes through before reaching the sensor at the back of the camera. Each of these lenses is built from refracting materials. The shape and position of the lenses, together with the refractive index of the materials they are built of, determine the path that light follows through the lens system. Generally, the lens system is designed to collect as much light as possible at the input and produce a focused image on the image plane. Remarkably, the many or most lens systems result in an imaging geometry that can be modelled with a pinhole camera model, and lens system effects are ignored in all but quite specialized applications of computer vision.

Lens systems are designed and modelled using geometric optics, but lens designs always involve compromises. The result is that cameras with lenses differ from pinhole cameras in some ways that are worth knowing about, although they are not always important. First, in an abstract pinhole camera, all objects at what-

ever distance are in focus. Geometric optics means that a lens with this property admits very little light, so it is common to work with cameras that have a limited *depth of field* – the range of distances to the camera over which objects are in focus on the image plane. Second, manufacturing difficulties and cost considerations mean that lenses will have various *aberrations*. The net effect of most aberrations is a tendency to defocus some objects under some circumstances, but *chromatic aberrations* can cause colors to be less crisp and objects to have "halos" of color. Chromatic aberration occurs because light of different wavelengths takes slightly different paths through a refracting object. Various lens coatings can correct chromatic aberration, but the resulting lens system will be more expensive. Third, in most lens systems, the periphery of the image tends to be brighter than it would be in a pure pinhole camera. For more complex lens systems, an effect in the lens known as *vignetting* can darken the periphery somewhat. Finally, lenses may cause *geometric distortions* of the image. The most noticeable effect of these distortions is that straight lines in the world may project to curves in the image. Most common is *barrel distortion*, where a square is imaged as a bulging barrel; *pincushion distortion*, where the square bulges in rather than out, can occur (Figure **??**).

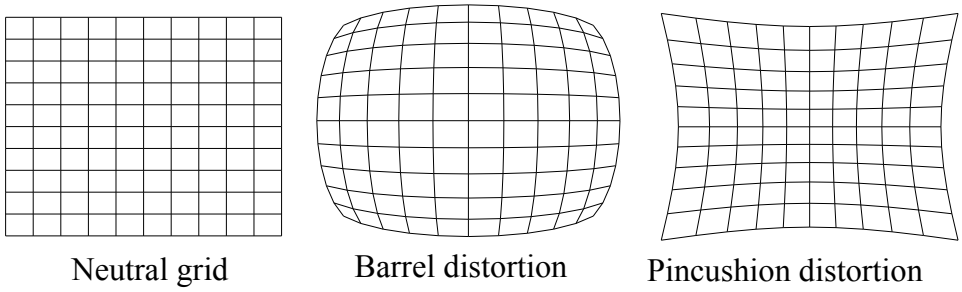

FIGURE 2.6: *On the* **left** *a neutral grid observed in a non-distorting lens (and viewed frontally to prevent any perspective distortion).* **Center** *shows the same grid, viewed in a lens that produces barrel distortion.* **Right***, the same grid, now viewed in a lens that produces pincushion distortion.*

2.2 SIMPLE PROJECTIVE GEOMETRY

Draw a pattern on a plane, then view that pattern with a perspective camera. The distortions you observe are more interesting than are predicted by simple rotation, translation and scaling. For example, if you drew parallel lines, you might see lines that intersect at a vanishing point – this doesn't happen under rotation, translation and scaling. *Projective geometry* can be used to describe the set of transformations produced by a perspective camera.

2.2.1 Homogeneous Coordinates and Projective Spaces

The coordinates that every reader will be most familiar with are known as *affine coordinates*. In affine coordinates, a point on the plane is represented by 2 numbers, a point in 3D is represented with 3 numbers, and a point in k dimensions is rep-

resented with k numbers. Now adopt the convention that a point in k dimensions is represented by $k+1$ numbers *not all of which are zero*. Two representations $\mathbf{X}_1$ and $\mathbf{X}_2$ represent the same point (write $\mathbf{X}_1 \equiv \mathbf{X}_2$) if there is some $\lambda \neq 0$ so that

$$\mathbf{X}_1 = \lambda \mathbf{X}_2.$$

These coordinates are known as *homogeneous coordinates*, and will offer a particularly convenient representation of perspective projection.

Remember this: *In homogeneous coordinates, a point in a k dimensional space is represented by $k+1$ coordinates $(X_1, \ldots, X_{k+1})$, together with the convention that*

$$(X_1, \ldots, X_{k+1}) \equiv \lambda(X_1, \ldots, X_{k+1}) \text{ for } \lambda \neq 0.$$

The space represented by $k+1$ homogeneous coordinates is different from the space represented by k affine coordinates in important but subtle ways. We start with a 1D space. In homogenous coordinates, we represent a point on a 1D space with two coordinates, so (X_1, X_2) (by convention, homogeneous coordinates are written with capital letters). Two sets of homogeneous coordinates (U_1, U_2) and (V_1, V_2) represent different points if there is no $\lambda \neq 0$ such that $\lambda(U_1, U_2) = (V_1, V_2)$. Now consider the set of all the distinct points, which is known as the *projective line*. Any point on an ordinary line (the *affine line*) has a corresponding point on the projective line. In affine coordinates, a point on the affine line is given by a single coordinate x. This point can be identified with the point on the projective line given by $(X_1, X_2) = \lambda(x, 1)$ (for $\lambda \neq 0$) in homogeneous coordinates. Notice that the projective line has an "extra point" – $(X_1, 0)$ are the homogeneous coordinates of a single point (check this), but this point would be "at infinity" on the affine line.

Example: 2.1 *Seeing the point at infinity*

You can actually see the point at infinity. Recall that lines that are parallel in the world can intersect in the image at a vanishing point. This vanishing point turns out to be the image of the point "at infinity" on the parallel lines. For example, on the plane $y = -1$ in the camera coordinate system, draw two lines $(1, -1, t)$ and $(-1, -1, t)$ (these lines are in Figure 25.2). Now these lines project to $(f1/t, f(-1/t), f)$ and $(f(-1/t), f(-1/t), f)$ on the image plane, and their vanishing point is $(0, 0, f)$. This vanishing point occurs when the parameter t reaches infinity. The exercises work this example in homogeneous coordinates.

There isn't anything special about the point on the projective line given by $(X_1, 0)$. You can see this by identifying x on the affine line with $(X_1, X_2) = \lambda(1, x)$

(for $\lambda \neq 0$). Now $(X_1, 0)$ is a point like any other, and $(0, X_2)$ is "at infinity". A little work establishes that there is a 1-1 mapping between the projective line and a circle (exercises).

Higher dimensional spaces follow the same pattern. In affine coordinates, a point in a k dimensional affine space (eg an *affine plane*; *affine 3D space*; etc) is given by k coordinates $(x_1, x_2, \ldots, x_k)$. The space described by $k+1$ homogeneous coordinates is a *projective space* (a *projective plane*; *projective 3D space*; etc). A point $(x_1, x_2, \ldots, x_k)$ in a k dimensional affine space can be identified with $(X_1, X_2, \ldots, X_{k+1}) = \lambda(x_1, x_2, \ldots, x_k, 1)$ (for $\lambda \neq 0$) in the k dimensional projective space. The points in the projective space given by $(X_1, X_2, \ldots, 0)$ have no corresponding points in the affine space. Notice that this set of points is a $k-1$ dimensional space in homogeneous coordinates. When $k = 2$, this set is a projective line, and is referred to as the *line at infinity*, and the whole space is known as the *projective plane*. As the exercises show, you can see the line at infinity: the horizon of a plane in the image is actually the image of the line at infinity in that plane.

When $k = 3$, this set is itself a projective plane, and is known as the *plane at infinity*; the whole space is sometimes known as *projective 3-space*. Notice this means that 3D projective space is obtained by "sewing" a projective plane to the 3D affine space we are accustomed to. The piece of the projective space "at infinity" isn't special, using the same argument as above. The particular line (resp. plane) that is "at infinity" is chosen by the homogeneous coordinate you divide by. There is an established convention in computer vision of dividing by the last homogeneous coordinate and talking about the line at infinity and the plane at infinity.

Remember this: *The k dimensional space represented by $k+1$ homogeneous coordinates is a projective space. You can represent a point $(x_1, \ldots, x_k$ in affine k space in this projective space as $(x_1, \ldots, x_k, 1)$. Not every point in the projective space can be obtained like this – the points $(X_1, \ldots, X_k, 0)$ are "extra". These points form a projective $k-1$ space which is thought of as being "at infinity". Important cases are $k = 1$ (the projective line with a point at infinity); $k = 2$ (the projective plane with a line at infinity).*

2.2.2 Lines and Planes in Projective Space

Lines on the affine plane form one important example of homogeneous coordinates. A line is the set of points (x, y) where $ax + by + c = 0$. We can use the coordinates (a, b, c) to represent a line. If $(d, e, f) = \lambda(a, b, c)$ for $\lambda \neq 0$ (which is the same as $(d, e, f) \equiv (a, b, c)$), then (d, e, f) and (a, b, c) represent the same line. This means the coordinates we are using for lines are homogeneous coordinates, and the family of lines in the affine plane is a projective plane. Notice that encoding lines using affine coordinates must leave out some lines. For example, if we insist on using $(u, v, 1) = (a/c, b/c, 1)$ to represent lines, the corresponding equation of the line

would be $ux + vy + 1 = 0$. But no such line can pass through the origin – our representation has left out every line through the origin.

Lines on the projective plane work rather like lines on the affine plane. Write the points on our line using homogeneous coordinates to get

$$(x, y, 1) = (X_1/X_3, X_2/X_3, 1)$$

or equivalently (X_1, X_2, X_3) where $X_1 = xX_3$, $X_2 = yX_3$. Substitute to find the equation of the corresponding line on the projective plane, $aX_1 + bX_2 + cX_3 = 0$, or $\mathbf{a}^T\mathbf{X} = 0$. There is an interesting point here. A set of three homogenous coordinates can be used to describe either a point on the projective plane or a line on the projective plane.

Remember this: *A line on the projective plane is the set of points* $\mathbf{X}$ *such that*

$$\mathbf{a}^T\mathbf{X} = 0.$$

Here $\mathbf{a}$ *is a vector of homogeneous coordinates specifying the particular line.*

Remember this: *Write* $\mathbf{P}_1$ *and* $\mathbf{P}_2$ *for two points on the projective plane that are represented in homogeneous coordinates and are different. From the exercises, the line through these two points is given by*

$$\mathbf{a} = \mathbf{P}_1 \times \mathbf{P}_2.$$

From the exercises, a parametrization of this line is given by

$$U\mathbf{P}_1 + V\mathbf{P}_2.$$

Planes in projective 3-space work rather like lines on the projective plane. The locus of points (x, y, z) where $ax + by + cz + d = 0$ is a plane in affine 3-space. Because (a, b, c, d) and $\lambda(a, b, c, d)$ give the same plane, we have that (a, b, c, d) are homogeneous coordinates for a plane in 3D. We can write the points on the plane using homogeneous coordinates to get

$$(x, y, z, 1) = (X_1/X_4, X_2/X_4, X_3/X_4, 1)$$

or equivalently

$$(X_1, X_2, X_3, X_4) \text{ where } X_1 = xX_4, X_2 = yX_4, X_3 = zX_4.$$

Substitute to find the equation of the corresponding plane in projective 3-space $aX_1 + bX_2 + cX_3 + dX_4 = 0$ or $\mathbf{a}^T\mathbf{X} = 0$. A set of four homogenous coordinates

can be used to describe either a point in projective 3-space or a plane in projective 3-space.

Remember this: *A plane in projective 3D is the set of points* $\mathbf{X}$ *such that*

$$\mathbf{a}^T\mathbf{X} = 0.$$

Here $\mathbf{a}$ *is a vector of homogeneous coordinates specifying the particular plane.*

Remember this: *Write* $\mathbf{P}_1$, $\mathbf{P}_2$ *and* $\mathbf{P}_3$ *for three points in projective 3D that are represented in homogeneous coordinates, are different points, and are not collinear. From the exercises, the plane through these points is given by*

$$\mathbf{a} = \mathit{NullSpace}\left(\begin{bmatrix} \mathbf{P}_1^T \\ \mathbf{P}_2^T \\ \mathbf{P}_3^T \end{bmatrix}\right).$$

From the exercises, a parametrization of this plane is given by

$$U\mathbf{P}_1 + V\mathbf{P}_2 + W\mathbf{P}_3.$$

2.2.3 Homographies

Write $\mathbf{X} = (X_1, X_2, X_3)$ for the coordinates of a point on the projective plane. Now consider $\mathbf{V} = \mathcal{M}\mathbf{X}$, where $\mathcal{M}$ is a 3×3 matrix with non-zero determinant. We can interpret $\mathbf{V}$ as a point on the projective plane, and in fact $\mathcal{M}$ is a mapping from the projective plane to itself. There is something to check here. Write $\mathcal{M}(\mathbf{X})$ for the point that $\mathbf{X}$ maps to, etc. Because $\mathbf{X} \equiv \lambda\mathbf{X}$ (for $\lambda \neq 0$), we must have that $\mathcal{M}(\mathbf{X}) \equiv \mathcal{M}(\lambda\mathbf{X})$ otherwise one point would map to several points. But

$$\mathcal{M}(\mathbf{X}) = \mathcal{M}\mathbf{X} \equiv \lambda\mathcal{M}\mathbf{X} = \mathcal{M}(\lambda\mathbf{X})$$

so $\mathcal{M}$ is a mapping. Such mappings are known as *homographies*. You should check that $\mathcal{M}^{(-1)}$ is the inverse of $\mathcal{M}$, and is a homography. You should check that $\mathcal{M}$ and $\lambda\mathcal{M}$ represent the same homography. Homographies are interesting to us because any view of a plane by a perspective (or orthographic) camera is a homography, and a variety of useful tricks rest on understanding homographies.

Any homography will map every line to a line. Write $\mathbf{a}$ for the line in the projective plane whose points satisfy $\mathbf{a}^T\mathbf{X} = 0$. Now apply the homography $\mathcal{M}$ to those points to get $\mathbf{V} = \mathcal{M}\mathbf{X}$. Notice that

$$\mathbf{a}^T\mathcal{M}^{(-1)}\mathbf{V} = \mathbf{a}^T\mathbf{X} = 0,$$

so that the line $\mathbf{a}$ transforms to the line $\mathcal{M}^{(-T)}\mathbf{a}$. Homographies are easily inverted.

Remember this: *A homography is a mapping from the projective plane to the projective plane. Assume $\mathcal{M}$ is a 3×3 matrix with non-zero determinant; then the homography represented by $\mathcal{M}$ maps the point with homogeneous coordinates $\mathbf{X}$ to the point with homogeneous coordinates $\mathcal{M}\mathbf{X}$. The two matrices $\mathcal{M}$ and $\lambda\mathcal{M}$ represent the same homography, and the inverse of this homography is represented by $\mathcal{M}^{-1}$. The homography represented by $\mathcal{M}$ will map the line represented by $\mathbf{a}$ to the line represented by $\mathcal{M}^{-T}\mathbf{a}$.*

2.3 CAMERA MATRICES AND TRANSFORMATIONS

2.3.1 Perspective and Orthographic Camera Matrices

In affine coordinates in the camera coordinate system of Figure 25.2, we could write perspective projection as $(X, Y, Z) \to (fX/Z, fY/Z)$. Now write the 3D point in homogeneous coordinates, so

$$\mathbf{X} = (X_1, X_2, X_3, X_4) \text{ where } X_1 = XX_4, \text{ etc.}$$

Write the point in the image plane in homogeneous coordinates as well, to obtain

$$\mathbf{I} = (I_1, I_2, I_3) \text{ where } I_1 = f(X/Z)I_3 \text{ and } I_2 = f(Y/Z)I_3.$$

So we could use

$$\mathbf{I} = (fX, fY, Z) \equiv (fX/Z, fY/Z, 1) \equiv (fX_1/X_4, fX_2/X_4, X_3/X_4) \equiv (fX_1, fX_2, X_3).$$

Notice that (X, Y, Z) is a natural choice of homogeneous coordinates for the point in the image plane. This means that, in homogeneous coordinates, we can represent perspective projection as

$$(X_1, X_2, X_3, X_4) \to (fX_1, fX_2, X_3) \equiv (X_1, X_2, (1/f)X_3)$$

or

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

where the matrix is known as the *perspective camera matrix* (write $\mathcal{C}_p$). Notice that this representation preserves the property that the focal point of the camera cannot be imaged, and is the only such point. The focal point can be represented in homogeneous coordinates by $(0, 0, 0, T)$, for $T \neq 0$. This maps to $(0, 0, 0)$, which is meaningless in homogeneous coordinates. You should check no other point maps to $(0, 0, 0)$. Notice also that changing f just scales the image, as you would expect. It is quite common to absorb the effects of f into the scaling caused by change to camera coordinates, which is equivalent to assuming that $f = 1$.

Remember this: *The perspective camera matrix is*

$$\mathcal{C}_p = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix}.$$

It is quite common to assume $f = 1$.

In affine coordinates, in the right coordinate system and assuming that the scale is chosen to be one, scaled orthographic perspective can be written as $(X, Y, Z) \rightarrow (X, Y)$. Following the argument above, we obtain in homogeneous coordinates

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

where the matrix is known as the *orthographic camera matrix* (write $\mathcal{C}_o$).

Remember this: *The orthographic camera matrix is*

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

2.3.2 Cameras in World Coordinates

The camera matrix describes a perspective (resp. orthographic) projection for a camera in a specific coordinate system – the focal point is at the origin, the camera is looking down the z-axis, and so on. In the more general case, the camera is placed somewhere in world coordinates looking in some direction, and we need to account for this. Furthermore, the camera matrix assumes that points in the camera are reported in a specific coordinate system. The pixel locations reported by a practical camera might not be in that coordinate system. For example, many cameras place the origin at the top left hand corner. We need to account for this effect, too.

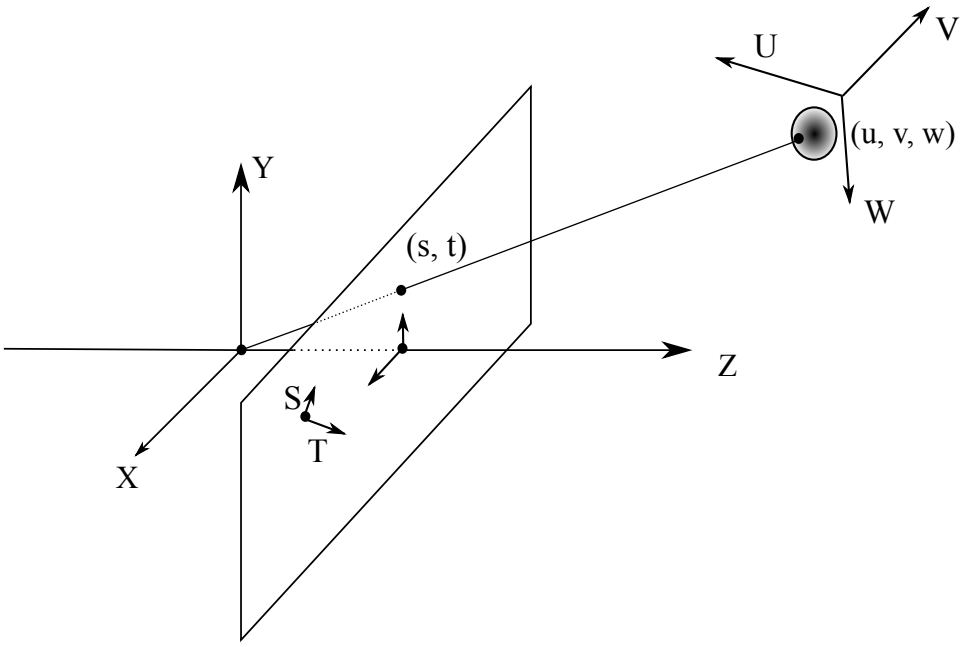

FIGURE 2.7: *A perspective camera (in its own coordinate system, given by X, Y and Z axes) views a point in world coordinates (given by (u, v, w) in the UVW coordinate system) and reports the position of points in ST coordinates. We must model the mapping from (u, v, w) to (s, t), which consists of a transformation from the UVW coordinate system to the XYZ coordinate system followed by a perspective projection followed by a transformation to the ST coordinate system.*

A general perspective camera transformation can be written as:

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} \text{Transformation} \\ \text{mapping image} \\ \text{plane coords to} \\ \text{pixel coords} \end{bmatrix} \mathcal{C}_p \begin{bmatrix} \text{Transformation} \\ \text{mapping world} \\ \text{coords to camera} \\ \text{coords} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

$$= \mathcal{T}_i \mathcal{C}_p \mathcal{T}_e \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

The parameters of $\mathcal{T}_i$ are known as *camera intrinsic parameters* or *camera intrinsics*, because they are part of the camera, and typically cannot be changed. The parameters of $\mathcal{T}_e$ are known as *camera extrinsic parameters* or *camera extrinsics*, because they can be changed.

2.3.3 Camera Extrinsic Parameters

The transformation $\mathcal{T}_e$ represents a rigid motion (equivalently, a *Euclidean transformation*, which consists of a 3D rotation and a 3D translation). In affine coordinates, any Euclidean transformation maps the vector $\mathbf{x}$ to

$$\mathcal{R}\mathbf{x} + \mathbf{t}$$

where $\mathcal{R}$ is an appropriately chosen 3D rotation matrix (check the endnotes if you can't recall) and $\mathbf{t}$ is the translation. Any map of this form is a Euclidean transformation. You should confirm the transformation that maps the vector $\mathbf{X}$ representing a point in 3D in homogeneous coordinates to

$$\lambda \begin{bmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{X}$$

represents a Euclidean transformation, but in homogeneous coordinates. It follows that any map of this form is a Euclidean transformation. Because $\mathcal{T}_e$ represents a Euclidean transformation, it must have this form. The exercises explore some properties of $\mathcal{T}_e$.

2.3.4 Camera Intrinsic Parameters

Camera intrinsic parameters must model a possible coordinate transformation in the image plane from projected world coordinates (write (x, y)) to pixel coordinates (write (u, v)), together with a possible change of focal length. This change is caused by the image plane being further away from, or closer to, the focal point. The coordinate transformation is not arbitrary (Figure 2.8). Typically, the origin of the pixel coordinates is usually not at the camera center. Write Δx for the step in the image plane from pixel (i, j) to $(i + 1, j)$ and Δy for the step to $(i, j + 1)$. For many cameras, Δx is different from Δy – such cameras have *non-square pixels*, and $\Delta x/\Delta y$ is known as the *aspect ratio* of the pixel. Furthermore, Δx is not usually one unit in world coordinates.

The transformation $\mathcal{T}_i$ is typically parametrized as

$$\begin{bmatrix} as & k & c_x \\ 0 & s & c_y \\ 0 & 0 & 1/f \end{bmatrix}.$$

Here (c_x, c_y) is the location of the camera center in pixel coordinates; s is a scale, and $1/s$ is the size of the spacing between pixels in world units; a is the aspect ratio of the pixels; k is the skew; and f is the focal length of the camera (Figure 2.8). The parameters c_x, c_y, s, a and k can often be obtained from camera manufacturer literature, or can be calibrated. Notice there is an interaction between $1/f$ and s. When there is no need to distinguish between scaling caused by pixel size and scaling caused by change in focal length, $\mathcal{T}_i$ can be parametrized as

$$\begin{bmatrix} as & k & c_x \\ 0 & s & c_y \\ 0 & 0 & 1 \end{bmatrix}.$$

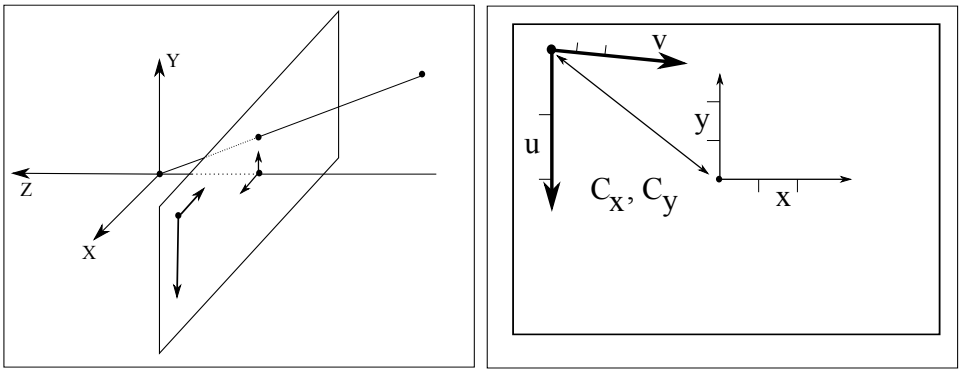

FIGURE 2.8: *The camera reports pixel values in pixel coordinates, which are not the same as world coordinates. The camera intrinsics represent the transformation between world coordinates and pixel coordinates. On the* **left**, *a camera (as in Figure 2.1), with the camera coordinate system shown in heavy lines. On the* **right**, *a more detailed view of the image plane. The camera coordinate axes are marked* (u, v) *and the image coordinate axes* (x, y). *It is hard to determine* f *from the figure, and we will conflate scaling due to* f *with scaling resulting from the change to camera coordinates. The camera coordinate system's origin is not at the camera center, so* (c_x, c_y) *are not zero. I have marked unit steps in the coordinate system with ticks. Notice that the* v*-axis is not perpendicular to the* u*-axis (so* k *-* the skew *- is not zero). Ticks in the* u, v *axes are not the same distance apart as ticks on the* x, y *axes, meaning that* s *is not one. Furthermore,* u *ticks are further apart than* v *ticks, so that* a *is not one.*

Remember this: *A general perspective camera can be written in homogeneous coordinates as:*

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \mathcal{T}_i \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathcal{T}_e \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

$$= \begin{bmatrix} as & k & c_x \\ 0 & s & c_y \\ 0 & 0 & 1/f \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

A common alternative parametrization applies when there is no need to distinguish between scaling caused by change of focal length and scaling resulting from the change to camera coordinates. This is:

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} as & k & c_x \\ 0 & s & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

where $\mathcal{R}$ *is a rotation matrix.*

By the arguments above, a general orthographic camera transformation can be written as:

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \mathcal{T}_i \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathcal{T}_e \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

PROBLEMS

2.1. We construct the vanishing point of a pair of parallel lines in homogeneous coordinates.

(a) Show that the set of points in homogeneous coordinates in 3D given by $(s, -s, t, s)$ (for s, t parameters) form a line in 3D.

(b) Now image the line $(s, -s, t, s)$ in 3D in a standard perspective camera with focal length 1. Show the result is the line $(s, -s, t)$ in the image plane.

(c) Now image the line $(-s, -s, t, s)$ in 3D in a standard perspective camera with focal length 1. Show the result is the line $(-s, -s, t)$ in the image plane.

(d) Show that the lines $(s, -s, t)$ and $(-s, -s, t)$ intersect in the point $(0, 0, t)$.

2.2. We construct the horizon of a plane for a standard perspective camera with focal length 1. Write $\mathbf{a} = [a_1, a_2, a_3, a_4]^T$ for the coefficients of the plane, so that for every point $\mathbf{X}$ on the plane we have $\mathbf{a}^T\mathbf{X} = 0$.

(a) Show that the plane given by $\mathbf{u} = [a_1, a_2, a_3, 0]$ is parallel to the plane given by $\mathbf{a}$, and passes through $(0, 0, 0, 1)$.

(b) Write the points on the image plane $(u, v, 1) \equiv (U, V, W)$ in homogeneous coordinates. Show that the horizon of the plane is the set of points $\mathbf{u}$ in the image plane given by $\mathbf{l}^T\mathbf{u} = 0$, where $\mathbf{l} = [a_1, a_2, a_3]^T$.

2.3. A pinhole camera with focal point at the origin and image plane at $z = f$ views two parallel lines $\mathbf{u} + t\mathbf{w}$ and $\mathbf{v} + t\mathbf{w}$. Write $\mathbf{w} = [w_1, w_2, w_3]^T$, etc.

(a) Show that the vanishing point of these lines, on the image plane, is given by $(f\frac{w_1}{w_3}, f\frac{w_2}{w_3})$.

(b) Now we model a family of pairs of parallel lines, by writing $\mathbf{w}(r, s) = r\mathbf{a} + s\mathbf{b}$, for any (r, s). In this model, $\mathbf{u} + t\mathbf{w}(r, s)$ and $\mathbf{v} + t\mathbf{w}(r, s)$ are the pair of lines, and (r, s) chooses the direction. First, show that this family of vectors lies in a plane. Now show that the vanishing point for the (r, s)'th pair is $(f\frac{ra_1+sb_1}{ra_3+sb_3}, f\frac{ra_2+sb_2}{ra_3+sb_3})$.

(c) Show that the family of vanishing points $(f\frac{ra_1+sb_1}{ra_3+sb_3}, f\frac{ra_2+sb_2}{ra_3+sb_3})$ lies on a straight line in the image. Do this by constructing $\mathbf{c}$ such that $\mathbf{c}^T\mathbf{a} = \mathbf{c}^T\mathbf{b} = 0$. Now write $(x(r, s), y(r, s)) = (-f\frac{ra_1+sb_1}{ra_3+sb_3}, -f\frac{ra_2+sb_2}{ra_3+sb_3})$ and show that $c_1x(r, s) + c_2y(r, s) + c_3 = 0$.

2.4. All points on the projective plane with homogeneous coordinates $(U, V, 0)$ lie "at infinity" (divide by zero). As we have seen, these points form a projective line.

(a) Show this line is represented by the vector of coefficients $(0, 0, C)$.

(b) A homography $\mathcal{M} = \left[\mathbf{m}_1^T; \mathbf{m}_2^T; \mathbf{m}_3^T\right]$ is applied to the projective plane. Show that the line whose coefficients are $\mathbf{v}_3$ maps to the line at infinity.

(c) Now write the homography as $\mathcal{M} = \left[\mathbf{m}'_1, \mathbf{m}'_2, \mathbf{m}'_3\right]$ (so $\mathbf{m}'$ are columns). Show that the homography maps the points at infinity to a line given in parametric form as $s\mathbf{m}'_1 + t\mathbf{m}'_2$.

(d) Now write $\mathbf{n}$ for a non-zero vector such that $\mathbf{n}^T\mathbf{m}'_1 = \mathbf{n}^T\mathbf{m}'_2 = 0$. Show that, for any point $\mathbf{x}$ on the line given in parametric form as $s\mathbf{m}'_1 + t\mathbf{m}'_2$, we have $\mathbf{n}^T\mathbf{x} = 0$. Is $\mathbf{n}$ unique?

(e) Use the results of the previous subexercises to show that for any given line, there are some homographies that map that line to the line at infinity.

(f) Use the results of the previous subexercises to show that for any given line, there are some homographies that map the line at infinity to that line.

2.5. We will show that there is no significant difference between choosing a right-handed camera coordinate system and a left-handed camera coordinate system. Notice that, in a right handed camera coordinate system (where the camera looks down the negative z-axis rather than the positive z-axis) the image plane is at $z = -f$.

(a) Show that, in a right-handed coordinate system, a pinhole camera maps

$$(X, Y, Z) \to (-fX/Z, -fY/Z).$$

(b) Show that the argument in the text yields a camera matrix of the form

$$\mathcal{C}'_p = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/f & 0 \end{bmatrix}.$$

(c) Show that, if one allows the scale in $\mathcal{T}_i$ to be negative, one could still use

$$\mathcal{C}_p = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix}$$

as a camera matrix.

CHAPTER 3

Camera Calibration

3.1 A CAMERA AT FIXED HEIGHT

A particularly interesting family of homographies occurs when the camera moves at fixed height above a ground plane, and the ground plane is at right angles to the image plane. This is a good model for a camera on (say) an autonomous car or a taxiing aircraft. Figure 3.1 shows the notation, etc. Here the focal length is f, the ground plane is the plane $y = -h$ in the camera coordinate system (remember, z is depth into the scene). Remarkably, we can calibrate the camera with elementary geometric reasoning in a configuration like this, at least for simple cameras. We make no attempt to distinguish between scaling caused by the focal length f and scaling caused by s.

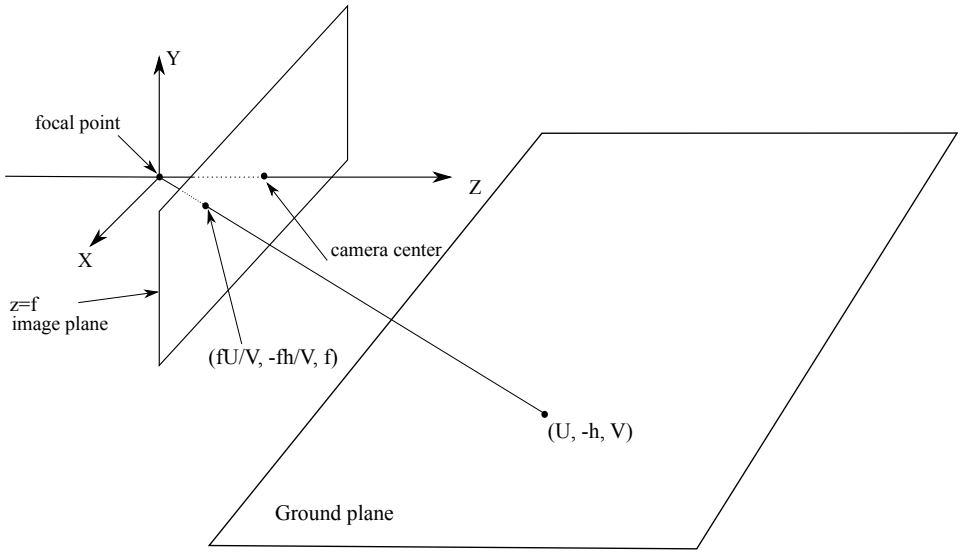

FIGURE 3.1: *A perspective camera with its image plane at right angles to a ground plane ($y = -h$), imaging a point on the ground plane.*

3.1.1 Geometry

Then the point $(u, -h, v)$ on the ground plane intersects the image plane at $(fu/v, -fh/v, f)$. These are affine coordinates for a point in 3D. Homogeneous coordinates for the point on the image plane are $(fu/v, -fh/v, 1)$ or equivalently $(fu, -fh, v)$. Similarly, homogeneous coordinates for the point on the ground plane are $(u, v, 1)$.

The first step is to simplify notation. Notice that the effect of a change in focal length f is to scale the image. The camera intrinsics will also scale the image, and we have no reason to distinguish between scaling due to focal length and that due to the focal point. As a result, we set $f = 1$, and let the scaling from camera intrinsics account for the effects of the focal length.

In this case, the homography from the ground plane to the image is

$$\mathcal{T}_{\mathrm{g}\to\mathrm{i}} = \begin{bmatrix} s & k & c_x \\ 0 & s & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -h \\ 0 & 1 & 0 \end{bmatrix}$$

where the matrix on the left comes from the camera intrinsics, and the matrix on the right is the mapping from ground plane to image plane. You should check that, in this geometry, the horizon of the ground plane is horizontal in the image and passes through c_y (Figure 3.1) should help, so we can determine c_y from an image. Write (i_x, i_y) for the affine coordinates of a point in the image. If we ensure that the horizon is the line $i_y = 0$ (which we can do by a simple subtraction), then $c_y = 0$. In these coordinates, we have

$$\mathcal{T}_{\mathrm{g}\to\mathrm{i}} = \begin{bmatrix} as & k & c_x \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -h \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} as & c_x & -hk \\ 0 & 0 & -sh \\ 0 & 1 & 0 \end{bmatrix}$$

The homography from the image to the ground plane is the inverse, so

$$\mathcal{T}_{\mathrm{i}\to\mathrm{g}} = \begin{bmatrix} \frac{1}{as} & -\frac{k}{as^2} & -\frac{c_x}{as} \\ 0 & 0 & 1 \\ 0 & \frac{-1}{sh} & 0 \end{bmatrix}.$$

The parameters we don't know here are a, s, k, h and c_x. Now imagine we choose some values – say $a = 1$, $s = 1$, $k = 0$, $h = 1$, $c_x = 0$. This gives an easy homography from the image to the ground plane, that is

$$\mathcal{T}'_{\mathrm{i}\to\mathrm{g}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}.$$

Map from the image to the ground plane using this homography. This will give us the pattern on the ground plane, but it will be distorted because we used the wrong values of the parameters. We can ask *how* the ground plane is distorted by different choices of parameters. In turn, if we know some properties *of the ground plane pattern* we can estimate our unknown parameters to ensure these properties hold. Write

$$(g_x(i_x, i_y, a, s, k, h, c_x), g_y(i_x, i_y, a, s, k, h, c_x), 1)$$

for the ground plane coordinates of an image plane point whose homogeneous coordinates are $(i_x, i_y, 1)$. You should check (in the exercises) that

$$\begin{bmatrix} g_x(i_x, i_y, a, s, k, h, c_x) \\ g_y(i_x, i_y, a, s, k, h, c_x) \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{h}{a} & -\frac{hk}{as} & \frac{c_x h}{a} \\ 0 & sh & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} g_x(i_x, i_y, 1, 1, 0, 1, 0) \\ g_y(i_x, i_y, 1, 1, 01, 0) \\ 1 \end{bmatrix}.$$

3.1.2 Calibration

Now assume we have a set of points $\mathbf{p}_i$ on the ground plane, and we can find the corresponding points $\mathbf{q}_i$ on the image plane. Apply the homography $\mathcal{T}_{\mathrm{i}\to\mathrm{g}}$ to the points in the image plane to get $\mathbf{r}_i$. The map $\mathbf{p}_i \to \mathbf{r}_i$ is

$$\mathcal{T}_{\mathbf{p}\to\mathbf{r}} = \begin{bmatrix} \frac{h}{a} & -\frac{hk}{as} & \frac{c_x h}{a} \\ 0 & sh & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0}^t & 1 \end{bmatrix} = \mathcal{T}_{\mathrm{int}}\mathcal{T}_{\mathrm{ext}}$$

where $\mathcal{T}_{\mathrm{ext}}$ is a rigid motion in the ground plane (we don't really know that the focal point is above the origin in the ground plane) and $\mathcal{T}_{\mathrm{int}}$ accounts for the fact that we don't know the camera calibration either. Notice that $\mathcal{T}_{\mathbf{p}\to\mathbf{r}}$ is an affine transformation, so we can recover it using procedure 23.1, which will yield a 2×2 matrix $\mathcal{M}$ and a translation vector $\mathbf{u}$.

We cannot recover $\frac{c_x h}{a}$ from $\mathbf{u}$ because we don't know what $\mathbf{t}$ is. However, we have

$$\mathcal{M} = \begin{bmatrix} \frac{h}{a} & -\frac{hk}{as} \\ 0 & sh \end{bmatrix} \mathcal{R}$$

and we can factor $\mathcal{M}$ using an RQ factorization (see procedure 26.1). Doing so yields $\frac{h}{a}$, $\frac{k}{s}$ and sh. These are enough for some applications. For example, we may be able to obtain a from manufacturing specifications, yielding h and so s and k. Furthermore, this calibration is enough to estimate rigid motion over the ground plane. For frames n and $n+1$, we compute $\mathcal{T}_{\mathrm{ext}}$, and the motion over the ground plane is $\mathcal{T}_{\mathrm{ext},n+1}\mathcal{T}^{-1}_{\mathrm{ext},n}$. Of course, if we move the $\mathbf{q}_i$ may no longer be visible in the image. Once we have the camera intrinsics, we do not need them. We could reconstruct the ground plane pattern in frame n, then reconstruct the ground plane pattern in frame $n+1$. The camera movement is the rotation and translation that make the two patterns line up best. There are a variety of procedures for doing so (Section 25.2), but the general idea should be clear.

3.2 CAMERA CALIBRATION FROM A 3D REFERENCE

Camera calibration involves estimating the intrinsic parameters of the camera, and perhaps lens parameters if needed, from one or more images. There are numerous strategies, all using the following recipe: build a *calibration object*, where the positions of some points (*calibration points*) are known; view that object from one or more viewpoints; obtain the image locations of the calibration points; and solve an optimization problem to recover camera intrinsics and perhaps lens parameters. As one would expect, much depends on the choice of calibration object. If all the calibration points sit on an object, the extrinsics will yield the *pose* (for position and orientation) of the object with respect to the camera. We use a two step procedure: formulate the optimization problem, then find a good starting point.

3.2.1 Formulating the Optimization Problem

The optimization problem is relatively straightforward to formulate. Notation is the main issue. We have N reference points $\mathbf{s}_i = [s_{x,i}, s_{y,i}, s_{z,i}]$ with known position in some reference coordinate system in 3D. The measured location in the image for

the i'th such point is $\hat{\mathbf{t}}_i = \left[\hat{t}_{x,i}, \hat{t}_{y,i}\right]$. There may be measurement errors, so the $\hat{\mathbf{t}}_i = \mathbf{t}_i + \xi_i$, where ξ_i is an error vector and $\mathbf{t}$ is the unknown true position. We will assume the magnitude of error does not depend on direction in the image plane (it is *isotropic*), so it is natural to minimize the squared magnitude of the error

$$\sum_i \xi_i^T \xi_i.$$

The main issue here is writing out expressions for ξ_i in the appropriate coordinates. Write $\mathcal{T}_i$ for the intrinsic matrix whose u, v'th component will be i_{uv}; $\mathcal{T}_e$ for the extrinsic transformation, whose u, v'th component will be e_{uv}. Recalling that $\mathcal{T}_i$ is lower triangular, and engaging in some manipulation, we obtain

$$\sum_i \xi_i^T \xi_i = \sum_i (t_{x,i} - p_{x,i})^2 + (t_{y,i} - p_{y,i})^2$$

where

$$\begin{aligned} p_{x,i} &= \frac{i_{11} g_{x,i} + i_{12} g_{y,i} + i_{13} g_{i,3}}{g_{i,3}} \\ p_{y,i} &= \frac{i_{22} g_{x,i} + i_{23} g_{i,3}}{g_{i,3}} \end{aligned}$$

and

$$\begin{aligned} g_{x,i} &= e_{11} s_{x,i} + e_{12} s_{y,i} + e_{13} s_{z,i} + e_{14} \\ g_{y,i} &= e_{21} s_{x,i} + e_{22} s_{y,i} + e_{23} s_{z,i} + e_{24} \\ g_{z,i} &= e_{31} s_{x,i} + e_{32} s_{y,i} + e_{33} s_{z,i} + e_{34} \end{aligned}$$

(which you should check as an exercise). This is a constrained optimization problem, because $\mathcal{T}_e$ is a Euclidean transformation. The constraints here are

$$1 - \sum_v e_{j,1v}^2 = 0 \text{ and } 1 - \sum_v e_{j,2v}^2 = 0 \text{ and } 1 - \sum_v e_{j,3v}^2 = 0$$
$$\sum_v e_{j,1v} e_{j,2v} = 0 \text{ and } \sum_v e_{j,1v} e_{j,3v} = 0 \text{ and } 1 - \sum_v e_{j,2v} e_{j,3v} = 0 \quad .$$

We might just throw this into a constrained optimizer (review Section 25.2), but good behavior requires a good start point. This can be obtained by a little manipulation, which I work through in the next section. Some readers may prefer to skip this at first (or even higher) reading because it's somewhat specialized, but it shows how the practical application of some tricks worth knowing.

3.2.2 Setting up a Start Point

Write $\mathbf{C}_j^T$ for the j'th row of the camera matrix, and $\mathbf{S}_i = [s_{x,i}, s_{y,i}, s_{z,i}, 1]^T$ for homogeneous coordinates representing the i'th point in 3D. Then, assuming no errors in measurement, we have

$$\hat{t}_{x,i} = \frac{\mathbf{C}_1^T \mathbf{S}_i}{\mathbf{C}_3^T \mathbf{S}_i} \text{ and } \hat{t}_{y,i} = \frac{\mathbf{C}_2^T \mathbf{S}_i}{\mathbf{C}_3^T \mathbf{S}_i},$$

which we can rewrite as

$$\mathbf{C}_3^T\mathbf{S}_i\hat{t}_{x,i} - \mathbf{C}_1^T\mathbf{S}_i = 0 \text{ and } \mathbf{C}_3^T\mathbf{S}_i\hat{t}_{y,i} - \mathbf{C}_2^T\mathbf{S}_i = 0.$$

We now have two homogenous linear equations in the camera matrix elements for each pair (3D point, image point). There are a total of 12 degrees of freedom in the camera matrix, meaning we can recover a least squares solution from six point pairs. The solution will have the form $\lambda\mathcal{P}$ where λ is an unknown scale and $\mathcal{P}$ is a known matrix. This is a natural consequence of working with homogeneous equations, but also a natural consequence of working with homogeneous coordinates. You should check that if $\mathcal{P}$ is a projection from projective 3D to the projective plane, $\lambda\mathcal{P}$ will yield the same projection as long as $\lambda \neq 0$.

This is enough information to recover the focal point of the camera. Recall that the focal point is the single point that images to $[0,0,0]^T$. This means that if we are presented with a 3×4 matrix claiming to be a camera matrix, we can determine what the focal point of that camera is without fuss – just find the null space of the matrix. Notice that we do not need to know λ to estimate the null space.

Remember this: *Given a 3×4 camera matrix $\mathcal{P}$, the homogeneous coordinates of the focal point of that camera are given by $\mathbf{X}$, where $\mathcal{P}\mathbf{X} = [0,0,0]^T$*

There is an important relationship between the focal point of the camera and the extrinsics. Assume that, in the world coordinate system, the focal point can be represented by $\left[\mathbf{f}^T, 1\right]^T$. This point must be mapped to $[0,0,0,1]^T$ by $\mathcal{T}_e$. Because we can recover $\mathbf{f}$ from $\mathcal{P}$ easily, we have an important constraint on $\mathcal{T}_e$, given in the box.

Remember this: *Assume camera matrix $\mathcal{P}$ has null space $\lambda\mathbf{u} = \lambda\left[\mathbf{f}^T, 1\right]^T$. Then we must have $\mathcal{T}_e\mathbf{u} = [0,0,0,1]^T$, so we must have*

$$\mathcal{T}_e = \begin{bmatrix} \mathcal{R} & -\mathcal{R}\mathbf{f} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

This means that, if we know $\mathcal{R}$, we can recover the translation from the focal point. We must now recover the intrinsic transformation and $\mathcal{R}$ from what we know.

$$\lambda\mathcal{P} = \mathcal{T}_i \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathcal{R} & -\mathcal{R}\mathbf{f} \\ \mathbf{0}^T & 1 \end{bmatrix} = \begin{bmatrix} \mathcal{T}_i\mathcal{R} & -\mathcal{T}_i\mathcal{R}\mathbf{f} \end{bmatrix}$$

We do not know λ, but we do know $\mathcal{P}$. Now write $\mathcal{P}_l$ for the left 3×3 block of $\mathcal{P}$, and recall that $\mathcal{T}_i$ is upper triangular and $\mathcal{R}$ orthonormal. The first question is the sign of λ. We expect $\mathsf{Det}(\mathcal{R}) = 1$, and $\mathsf{Det}(\mathcal{T}_i) > 0$, so $\mathsf{Det}(\mathcal{P}_l)$ should be positive. This yields the sign of λ – choose a sign $s \in \{-1, 1\}$ so that $\mathsf{Det}(s\mathcal{P}_l)$ is positive.

We can now factor $s\mathcal{P}_l$ into an upper triangular matrix $\mathcal{T}$ and an orthonormal matrix $\mathcal{Q}$. This is an RQ factorization (Section 25.2). Recall we could not distinguish between scaling caused by the focal length and scaling caused by pixel scale, so that

$$\mathcal{T}_i = \begin{bmatrix} as & k & c_x \\ 0 & s & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

In turn, we have $\lambda = s(1/t_{33})$, $c_y = (t_{23}/t_{33})$, $s = (t_{22}/t_{33})$, $c_x = (t_{13}/t_{33})$, $k = (t_{12}/t_{33})$, and $a = (t_{11}/t_{22})$.

Procedure: 3.1 *Calibrating a Camera using 3D Reference Points*

For N reference points $\mathbf{s}_i = [s_{x,i}, s_{y,i}, s_{z,i}]$ with known position in some reference coordinate system in 3D, write the measured location in the image for the i'th such point $\hat{\mathbf{t}}_i = \left[\hat{t}_{x,i}, \hat{t}_{y,i}\right]$. Now minimize

$$\sum_i \xi_i^T \xi_i = \sum_i (\hat{t}_{x,i} - p_{x,i})^2 + (\hat{t}_{y,i} - p_{y,i})^2$$

where

$$\begin{aligned} p_{x,i} &= \frac{i_{11}g_{x,i} + i_{12}g_{y,i} + i_{13}g_{i,3}}{g_{i,3}} \\ p_{y,i} &= \frac{i_{22}g_{x,i} + i_{23}g_{i,3}}{g_{i,3}} \end{aligned}$$

and

$$\begin{aligned} g_{x,i} &= e_{11}s_{x,i} + e_{12}s_{y,i} + e_{13}s_{z,i} + e_{14} \\ g_{y,i} &= e_{21}s_{x,i} + e_{22}s_{y,i} + e_{23}s_{z,i} + e_{24} \\ g_{z,i} &= e_{31}s_{x,i} + e_{32}s_{y,i} + e_{33}s_{z,i} + e_{34} \end{aligned}$$

subject to:

$$1 - \sum_v e_{j,1v}^2 = 0 \text{ and } 1 - \sum_v e_{j,2v}^2 = 0 \text{ and } 1 - \sum_v e_{j,3v}^2 = 0$$

$$\sum_v e_{j,1v}e_{j,2v} = 0 \text{ and } \sum_v e_{j,1v}e_{j,3v} = 0 \text{ and } 1 - \sum_v e_{j,2v}e_{j,3v} = 0 \quad .$$

Use the start point of procedure 3.2

Procedure: 3.2 *Calibrating a Camera using 3D Reference Points: Start Point*

Estimate the rows of the camera matrix $\mathbf{C}_i$ using at least six points and

$$\mathbf{C}_3^T \mathbf{S}_i \hat{t}_{x,i} - \mathbf{C}_1^T \mathbf{S}_i = 0 \text{ and } \mathbf{C}_3^T \mathbf{S}_i \hat{t}_{y,i} - \mathbf{C}_2^T \mathbf{S}_i = 0.$$

Write $\lambda\mathcal{P}$ for the 1D family of solutions to this set of homogeneous linear equations, organized into 3×4 matrix form. Compute the vector $\mathbf{n} = \left[\mathbf{f}^T, 1\right]$ such that $\mathcal{P}\mathbf{n}$. Write $\mathcal{P}_l$ for the left 3×3 block of $\mathcal{P}$. Choose $s \in \{-1, 1\}$ such that $\mathsf{Det}\left(s\mathcal{P}_l\right) > 0$. Use RQ factorization to obtain $\mathcal{T}$ and $\mathcal{Q}$ such that $s\mathcal{P}_l = \mathcal{T}\mathcal{Q}$. Then the start point for the intrinsic parameters is:

$$\begin{bmatrix} a \\ s \\ k \\ c_x \\ c_y \end{bmatrix} = \begin{bmatrix} (t_{11}/t_{22}) \\ (t_{22}/t_{33}) \\ (t_{12}/t_{33}) \\ (t_{13}/t_{33}) \\ (t_{23}/t_{33}) \end{bmatrix}$$

and for $\mathcal{T}_e$ is:

$$\begin{bmatrix} \mathcal{Q} & -\mathcal{Q}\mathbf{f} \\ \mathbf{0} & 1 \end{bmatrix}.$$

3.3 CAMERA CALIBRATION FROM PLANE REFERENCES

3D calibration objects can be inconvenient in practice. It is possible to calibrate a camera with a plane calibration object, but you need to have several views. Plane patterns are easy to make and easy to disseminate. Obtain a plane pattern with a set of easily localized points (a checkerboard is good) where the locations of those points on the plane are known in world units. So if one is using a checkerboard, one might know that the checks are square and 10cm on edge, for example. Lay this down flat, and take a set of images of it from different views. In each view the calibration points should be visible.

For each view, we will compute the homography from the calibration object's plane to the image plane using point correspondences (Section 25.2). It turns out that these homographies yield constraints on the camera matrix (Section 25.2) and these constraints yield a camera estimate (Section 25.2). This estimate is a start point for an optimization problem (Section 25.2, very much on the lines of Section 25.2).

3.3.1 Estimating Homographies from Data

Estimating a homography from data is useful, and relatively easy to do. We are given a set of source points $\mathbf{S}_i$ in one plane and target points $\mathbf{T}_i$ in a second plane, and we must determine the homography. In all cases of interest, the points will be supplied in affine coordinates, rather than homogeneous coordinates, and we convert to homogeneous coordinates by attaching a 1, as before. Write m_{ij} for the

i, j'th element of matrix $\mathcal{M}$. In affine coordinates, a homography $\mathcal{M}$ will map (s_x, s_y) to (t_x, t_y) where

$$t_x = \frac{m_{11}s_x + m_{12}s_y + m_{13}}{m_{31}s_x + m_{32}s_y + m_{33}} \text{ and } t_y = \frac{m_{21}s_x + m_{22}s_y + m_{23}}{m_{31}s_x + m_{32}s_y + m_{33}}$$

You should use these expressions to check:

- if $m_{31} = m_{32} = 0$, the homography is an affine transformation;
- if $m_{31} = m_{32} = 0, m_{11}^2 + m_{21}^2 = m_{21}^2 + m_{22}^2$ and $m_{11}m_{12} + m_{21}m_{22} = 0$, then the homography is a scaled euclidean transformation;
- and if $m_{31} = m_{32} = 0, m_{11}^2 + m_{21}^2 = m_{21}^2 + m_{22}^2 = m_{33}$ and $m_{11}m_{12} + m_{21}m_{22} = 0$, then the homography is a euclidean transformation.

In most cases of interest, the coordinates of the points are not measured precisely, so we observe $\hat{\mathbf{t}} = \mathbf{t} + \xi$, where ξ is some noise vector and $\mathbf{t}$ contains the true (and unknown) coordinates of the point. The error will be in affine coordinates – for example, in the image plane – which justifies working in affine rather than homogeneous coordinates. In most cases of interest, the probability distribution of ξ does not depend on orientation (it is *isotropic*), and so we can use straightforward least squares. The homography can then be estimated by minimizing

$$\sum_i \xi^T \xi = \sum_i \left(\xi_x^2 + \xi_y^2\right)$$

where

$$\begin{aligned} \xi_x &= (\hat{t}_{x,i} - \frac{m_{11}s_{x,i} + m_{12}s_{y,i} + m_{13}}{m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}})^2 \\ \xi_y &= (\hat{t}_{y,i} - \frac{m_{21}s_{x,i} + m_{22}s_{y,i} + m_{23}}{m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}})^2 \end{aligned}$$

using standard methods (Levenberg-Marquardt is favored; Chapter 25.2). If the error happens to be normally distributed and isotropic, this cost function is proportional to the negative log-likelihood of the error (exercises), so this approach is sometimes known as *maximum likelihood* . Experience teaches that this optimization is not well behaved without a strong start point.

There is an easy construction for a good start point. Notice that the equations for the homography mean that

$$\begin{aligned} \hat{t}_{x,i}(m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}) - m_{11}s_{x,i} + m_{12}s_{y,i} + m_{13} &= 0 \\ \text{and} \\ \hat{t}_{y,i}(m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}) - m_{21}s_{x,i} + m_{22}s_{y,i} + m_{23} &= 0 \end{aligned}$$

so each corresponding pair of points $\mathbf{s}_i$, $\mathbf{t}_i$ yields two *homogeneous* linear equations in the coefficients of the homography. They are homogeneous because scaling $\mathcal{M}$ doesn't change what it does to points (check this if you're uncertain). If we obtain sufficient points, we can solve the resulting system of homogeneous linear equations. Four point correspondences yields an unambiguous solution; more than four – which is better – can be dealt with by least squares. The resulting estimate of $\mathcal{M}$ has a good reputation as a start point for a full optimization.

Procedure: 3.3 *Estimating a Homography from Data*

Given N known source points $\mathbf{s}_i = (s_{x,i}, s_{y,i})$ in affine coordinates and N corresponding target points $\mathbf{T}_i$ with measured locations $(\hat{t}_{x,i}, \hat{t}_{y,i})$, estimate the homography $\mathcal{M}$ with i, j'th element m_{ij} by minimizing:

$$\sum_i \xi^T \xi = \sum_i \left(\xi_x^2 + \xi_y^2\right)$$

where

$$\begin{aligned} \xi_x &= (\hat{t}_{x,i} - \frac{m_{11}s_{x,i} + m_{12}s_{y,i} + m_{13}}{m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}})^2 \\ \xi_y &= (\hat{t}_{y,i} - \frac{m_{21}s_{x,i} + m_{22}s_{y,i} + m_{23}}{m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}})^2. \end{aligned}$$

Obtain a start point by solving the set of homogeneous linear equations

$$\begin{aligned} \hat{t}_{x,i}(m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}) - m_{11}s_{x,i} + m_{12}s_{y,i} + m_{13} &= 0 \\ \text{and} \\ \hat{t}_{y,i}(m_{31}s_{x,i} + m_{32}s_{y,i} + m_{33}) - m_{21}s_{x,i} + m_{22}s_{y,i} + m_{23} &= 0. \end{aligned}$$

3.3.2 Constraining Intrinsics with Homographies

Each map from the pattern to an image is a homography. Choose the world coordinate system so that the pattern lies on the plane $z = 0$. Doing so just changes the camera *extrinsics*, so no generality has been lost, but it allows us to write the homography in a useful form. Recall the camera is

$$\mathcal{T}_i \mathcal{C}_p \mathcal{T}_{e,j}$$

where $\mathcal{T}_{e,j}$ is the euclidean transformation giving the extrinsics for the j'th view. This is applied to a set of points $(s_{x,i}, s_{y,i}, 0, 1)$. In turn, the homography for the j'th view must have the form

$$\lambda_j \mathcal{M}_j = \mathcal{T}_i \left[\mathbf{r}_{1,j}, \mathbf{r}_{2,j}, \mathbf{t}_j\right]$$

(where $\mathbf{r}_{1,j}$, $\mathbf{r}_{2,j}$ are the first two *columns* of the rotation matrix in $\mathcal{T}_{e,j}$ and $\mathbf{t}_j$ is the translation). We do not know λ_j (which is non-zero) because scaling the homography matrix yields the same homography. Now write $\mathcal{N}_j = \mathcal{T}_i^{(-1)} \mathcal{M}_j = [\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3]$. We must have that

$$\mathbf{n}_1^T \mathbf{n}_1 - \mathbf{n}_2^T \mathbf{n}_2 = 0 \text{ and } \mathbf{n}_1^T \mathbf{n}_2 = 0.$$

These equations constrain the unknown values of $\mathcal{T}_i$, and we get two for each homography. In turn, with sufficient views (and so homographies), we can estimate $\mathcal{T}_i$.

3.3.3 Estimating Intrinsics from Homographies

In the j'th view of the plane calibration object, we recover a homography $\mathcal{M}_j$. Now write $\mathcal{N}_j = \mathcal{T}_i^{(-1)}\mathcal{M}_j = [\mathbf{n}_{j,1}, \mathbf{n}_{j,2}, \mathbf{n}_{j,3}]$. We know from Section 25.2 that

$$\mathbf{n}_{j,1}^T\mathbf{n}_{j,1} - \mathbf{n}_{j,2}^T\mathbf{n}_{j,2} = 0 \text{ and } \mathbf{n}_{j,1}^T\mathbf{n}_{j,2} = 0.$$

Now write $\mathcal{A} = (\mathcal{T}_i^{(-T)}\mathcal{T}_i^{(-1)})$ (which is unknown). These two constraints are linear homogenous equations in the entries of $\mathcal{A}$, which is 3×3 but symmetric, and so has 6 unknown parameters. If we have 3 homographies, we will have 6 constraints, and can use least squares to recover a 1D family of solutions $\lambda\mathcal{B}$, where $\mathcal{B}$ is *known* and λ is a scale. We now need to find $\mathcal{T}_i$ and λ so that $\lambda\mathcal{B}$ is close to $(\mathcal{T}_i^{(-T)}\mathcal{T}_i^{(-1)})$.

There are constraints here. Write $\mathcal{U} = \mathcal{T}_i^{(-1)}$. Recall $\mathcal{T}_i$ is upper triangular, and $i_{33} = 1$. This means that $\mathcal{U}$ is upper triangular, and $u_{33} = 1$. We will find $\mathcal{U}$ and λ by finding $\mathcal{V}$ such that $\mathcal{V}^T\mathcal{V}$ is closest to $\mathcal{B}$, then computing $\mathcal{U} = (1/v_{33})\mathcal{V}$.

Finding $\mathcal{V}$ is straightforward. We obtain the closest symmetric matrix to $\mathcal{B}$, then apply a Cholesky factorization (Section 25.2). The factorization could be modified if a very small number appears on the diagonal, but this event is most unlikely. We now invert $\mathcal{U}$ to obtain an estimate $\mathcal{E}$ of $\mathcal{T}_i$. Recall this has the form

$$\begin{bmatrix} as' & k' & c_x' \\ 0 & s' & c_y' \\ 0 & 0 & 1 \end{bmatrix}.$$

so we have $c_x' = e_{13}$, $c_y' = e_{23}$, $s' = e_{22}$, $a = e_{11}/e_{22}$ and $k' = e_{12}$. This is usually an acceptable start point for optimization.

3.3.4 Estimating Extrinsics from Homographies

We have an estimate of the camera intrinsics, and now need an estimate of the extrinsics for each view. Recall from Section **??** that

$$\lambda_j\mathcal{M}_j = \mathcal{T}_i\left[\mathbf{r}_{1,j}, \mathbf{r}_{2,j}, \mathbf{t}_j\right]$$

(where $\mathbf{r}_{1,j}$, $\mathbf{r}_{2,j}$ are the first two *columns* of the rotation matrix in $\mathcal{T}_{e,j}$ and $\mathbf{t}_j$ is the translation). We have estimates of $\mathcal{M}_j$ and of $\mathcal{T}_i$, but we do not know λ_j. We can solve for λ_j by noticing that the first two columns of

$$\lambda_j\mathcal{T}_i^{-1}\mathcal{M}_j = \lambda_j\mathcal{Q}_j = \lambda_j\left[\mathbf{q}_{1,j}, \mathbf{q}_{2,j}, \mathbf{q}_{3,j}\right]$$

are unit vectors, and are normal to one another. For example, we might estimate

$$\lambda_j = \sqrt{\frac{2}{\mathbf{q}_{1,j}^T\mathbf{q}_{1,j} + \mathbf{q}_{2,j}^T\mathbf{q}_{2,j}}}$$

and from this follows the estimate

$$\mathcal{T}_{e,j} = \begin{bmatrix} \lambda_j\mathbf{q}_{1,j} & \lambda_j\mathbf{q}_{2,j} & \lambda_j^2\mathbf{q}_{1,j} \times \mathbf{q}_{2,j} & \lambda_j\mathbf{q}_{3,j} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.3.5 Formulating the Optimization Problem

As in Section 25.2, we will calibrate the camera by solving an optimization problem. The optimization problem is relatively straightforward to formulate, and follows the same lines as that in Section 25.2. The main difference with that section is that all calibration points will lie on the plane $z = 0$ in world coordinates, and we will have more than one view of that plane. Write $\mathbf{t}_{ij} = [t_{x,ij}, t_{y,ij}]$ for the measured x, y position in the image plane of the i'th reference point in the j'th view. We have that $\mathbf{t}_{ij} = \hat{\mathbf{t}}_{ij} + \xi_{ij}$, where ξ_{ij} is an error vector and $\hat{\mathbf{t}}_{ij}$ is the true (unknown) position. Again, assume the error is isotropic, so it is natural to minimize

$$\sum_{ij} \xi_{ij}^T \xi_{ij}.$$

The main issue here is writing out expressions for ξ_{ij} in the appropriate coordinates. Write $\mathcal{T}_i$ for the intrinsic matrix whose u, v'th component will be i_{uv}; $\mathcal{T}_{e,j}$ for the j'th extrinsic transformation, whose u, v'th component will be e_{uv}; and $\mathbf{s}_i = [s_{x,i}, s_{y,i}, 0]$ for the known coordinates of the i'th reference point in the coordinate frame of the reference points. Recalling that $\mathcal{T}_i$ is lower triangular, and engaging in some manipulation, we obtain

$$\sum_{ij} \xi_{ij}^T \xi_{ij} = \sum_{i} (t_{x,ij} - p_{x,ij})^2 + (t_{y,ij} - p_{y,ij})^2$$

where

$$\begin{aligned} p_{x,ij} &= \frac{i_{11} g_{x,ij} + i_{12} g_{y,ij} + i_{13} g_{z,ij}}{g_{z,ij}} \\ p_{y,ij} &= \frac{i_{22} g_{x,ij} + i_{23} g_{z,ij}}{g_{z,ij}} \end{aligned}$$

and

$$\begin{aligned} g_{x,ij} &= e_{11,j} s_{x,i} + e_{12,j} s_{y,i} + e_{14,j} \\ g_{y,ij} &= e_{21,j} s_{x,i} + e_{22,j} s_{y,i} + e_{24,j} \\ g_{z,ij} &= e_{31,j} s_{x,i} + e_{32,j} s_{y,i} + e_{34,j} \end{aligned}$$

(which you should check as an exercise – notice the missing $s_{z,i}$ terms!). This is a constrained optimization problem, because $\mathcal{T}_e$ is a Euclidean transformation. The constraints here are

$$\begin{aligned} 1 - \sum_v e_{j,1v}^2 = 0 \text{ and } 1 - \sum_v e_{j,2v}^2 = 0 \text{ and } 1 - \sum_v e_{j,3v}^2 = 0 \\ \sum_v e_{j,1v} e_{j,2v} = 0 \text{ and } \sum_v e_{j,1v} e_{j,3v} = 0 \text{ and } 1 - \sum_v e_{j,2v} e_{j,3v} = 0 \end{aligned} .$$

As in Section 25.2, we could just throw this into a constrained optimizer (review Section 25.2), but good behavior requires a good start point.

Procedure: 3.4 *Calibrating a Camera from Multiple Homographies*

Procedure: 3.5 *Calibrating a Camera from Multiple Homographies: Start Point*

3.4 CALIBRATING THE EFFECTS OF LENS DISTORTION

Now assume the lens applies some form of geometric distortion, as in Section 25.2. There are now strong standard models of the major lens distortions (Section 25.2). We will now estimate lens parameters, camera intrinsics and camera extrinsics from a view of a calibration object (as in Section 25.2; note the methods of Section 25.2 apply to this problem too). As in those sections, we use a two step procedure: formulate the optimization problem (Section 25.2), then find a good starting point (Section 25.2).

3.4.1 Modelling Geometric Lens Distortion

Geometric distortions caused by lenses are relatively easily modelled by assuming the lens causes (x, y) in the image plane to map to $(x+\delta x, y+\delta y)$ in the image plane. We seek a model for $\delta x, \delta y$ that has few parameters and that captures the main effects. A natural model of barrel distortion is that points are "pulled" toward the camera center, with points that are further from the center being "pulled" more. Similarly, pincushion distortion results from points being "pushed" away from the camera center, with distant points being pushed further (Figure **??**).

Set up a polar coordinate system (r, θ) in the image plane using the image center as the origin. The figure and description suggest that barrel and pincushion distortion can be described by a map $(r, \theta) \rightarrow (r + \delta r, \theta)$. We model δr as a polynomial in r. Brown and Conrady [] established the model $\delta r = k_1 r^3 + k_2 r^5$ as sufficient for a wide range of distortions, and we use $(r, \theta) \rightarrow (r + k_1 r^3 + k_2 r^5, \theta)$ for unknown k_1, k_2. We must map this model to image coordinates to obtain a map $(x, y) \rightarrow (x + \delta x, y + \delta y)$. Since $\cos\theta = x/r$, $\sin\theta = y/r$, we have $(x, y) \rightarrow (x + x(k_1(x^2 + y^2) + k_2(x^2 + y^2)^2), y + y(k_1(x^2 + y^2) + k_2(x^2 + y^2)^2))$. Figure 3.2 shows distortions resulting from different choices of k_1 and k_2. This model is known as a *radial distortion model*.

More sophisticated lens distortion models account for the lens being off-center. This causes *tangential distortion* (Figure 3.3). The most commonly used model of tangential distortion is a map $(x, y) \rightarrow (x + p_1(x^2 + y^2 + 2x^2) + 2p_2xy, y + p_2(x^2 + y^2 + 2y^2) + 2p_1xy)$ (derived from []; more detail in, for example []).

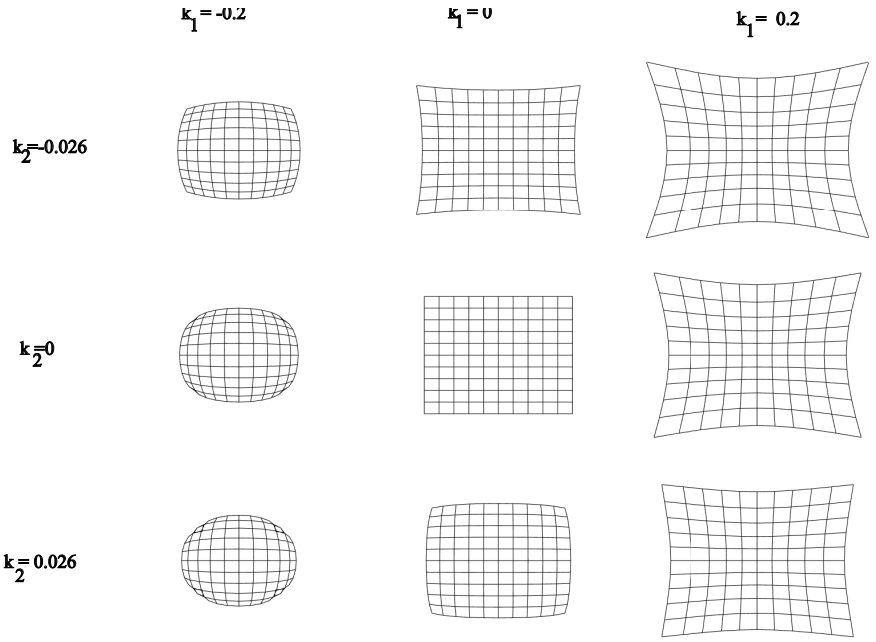

FIGURE 3.2: *The effects of k_1 and k_2 on a neutral grid (***center***), showing how the parameters implement various barrel or pincushion distortions. Notice how k_2 slightly changes the shape of the curves that k_1 produces from straight lines in the grid.*

Remember this: *A full lens distortion model is*

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x + x(k_1(x^2+y^2) + k_2(x^2+y^2)^2) + p_1(x^2+y^2+2x^2) + 2p_2xy \\ y + y(k_1(x^2+y^2) + k_2(x^2+y^2)^2) + p_2(x^2+y^2+2y^2) + 2p_1xy \end{pmatrix}.$$

for k_1, k_2, p_1, p_2 parameters. It is common to ignore tangential distortion and focus on radial distortion by setting $p_1 = p_2 = 0$.

3.4.2 Formulating the Optimization Problem

The optimization problem is relatively straightforward to formulate. Notation is the main issue. Write $\mathbf{t}_i = [t_{x,i}, t_{y,i}]$ for the measured x, y position in the image plane of the i'th reference point. We have that $\mathbf{t}_i = \hat{\mathbf{t}}_i + \xi_i$, where ξ_i is an error vector and $\hat{\mathbf{t}}$ is the true (unknown) position. Again, assume the error is isotropic,

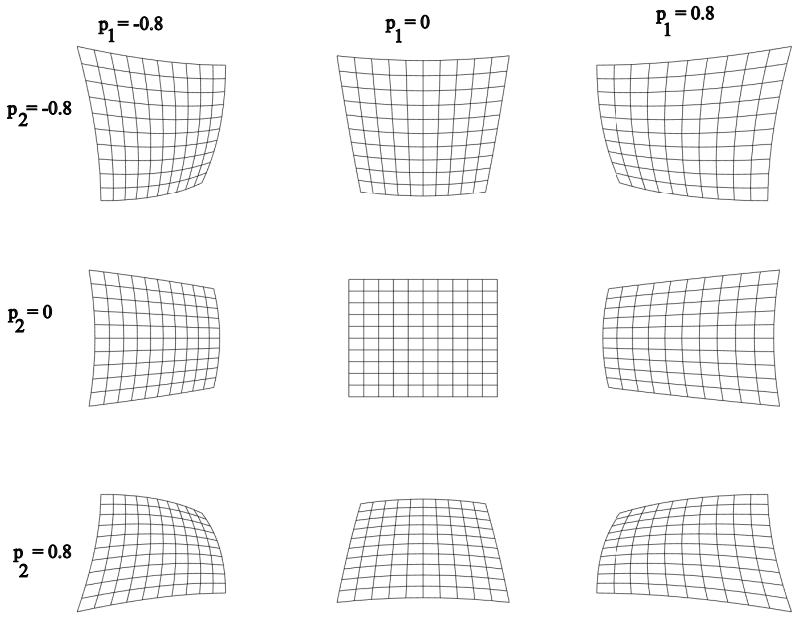

FIGURE 3.3: *The effects of p_1 and p_2 on a neutral grid (***center***), showing how the parameters implement various distortions. These parameters model effects that occur because the lens is off-center; note the grid "turning away" from the lens.*

so it is natural to minimize

$$\sum_i \xi_i^T \xi_i.$$

The main issue here is writing out expressions for $\xi_{i,j}$ in the appropriate coordinates. Write $\mathcal{T}_i$ for the intrinsic matrix whose u, v'th component will be i_{uv}; $\mathcal{T}_e$ for the extrinsic transformation, whose u, v'th component will be e_{uv}; and $\mathbf{s}_i = [s_{x,i}, s_{y,i}, s_{z,i}]$ for the known coordinates of the i'th reference point in the coordinate frame of the reference points. Recalling that $\mathcal{T}_i$ is lower triangular, and engaging in some manipulation, we obtain

$$\sum_i \xi_i^T \xi_i = \sum_i (t_{x,i} - l_{x,i})^2 + (t_{y,i} - l_{y,i})^2$$

where

$$\begin{aligned} l_{x,i} &= p_{x,i} + p_{x,i}(k_1(p_{x,i}^2 + p_{y,i}^2) + k_2(p_{x,i}^2 + p_{y,i}^2)^2) + p_1(p_{x,i}^2 + p_{y,i}^2 + 2p_{x,i}^2) + 2p_2 p_{x,i} p_{y,i} \\ l_{y,i} &= p_{y,i} + p_{y,i}(k_1(p_{x,i}^2 + p_{y,i}^2) + k_2(p_{x,i}^2 + p_{y,i}^2)^2) + p_2(p_{x,i}^2 + p_{y,i}^2 + 2p_{y,i}^2) + 2p_1 p_{x,i} p_{y,i} \end{aligned}$$

$$\begin{aligned} p_{x,i} &= \frac{i_{11}g_{x,i} + i_{12}g_{y,i} + i_{13}g_{i,3}}{g_{i,3}} \\ p_{y,i} &= \frac{i_{22}g_{x,i} + i_{23}g_{i,3}}{g_{i,3}} \end{aligned}$$

and

$$\begin{aligned} g_{x,i} &= e_{11}s_{x,i} + e_{12}s_{y,i} + e_{13}s_{z,i} + e_{14} \\ g_{y,i} &= e_{21}s_{x,i} + e_{22}s_{y,i} + e_{23}s_{z,i} + e_{24} \\ g_{z,i} &= e_{31}s_{x,i} + e_{32}s_{y,i} + e_{33}s_{z,i} + e_{34} \end{aligned}$$

(which you should check as an exercise). This is a constrained optimization problem, because $\mathcal{T}_e$ is a Euclidean transformation. The constraints here are

$$1 - \sum_v e_{j,1v}^2 = 0 \text{ and } 1 - \sum_v e_{j,2v}^2 = 0 \text{ and } 1 - \sum_v e_{j,3v}^2 = 0$$
$$\sum_v e_{j,1v}e_{j,2v} = 0 \text{ and } \sum_v e_{j,1v}e_{j,3v} = 0 \text{ and } 1 - \sum_v e_{j,2v}e_{j,3v} = 0 \quad .$$

We might just throw this into a constrained optimizer (review Section 25.2), but good behavior requires a good start point. This can be obtained by a little manipulation, which I work through in the next section. Some readers may prefer to skip this at first (or even higher) reading because it's somewhat specialized, but it shows how the practical application of some tricks worth knowing.