1
Projective Geometry

1.1 Celestial Navigation

Consider you sailing in the middle of Atlantic ocean without GPS, where no ground landmark can be seen. Dominant visual features from cloud and wave are highly dynamic, which are not reliable measure to localize where I am.

Early Minoans of Crete (Bronze Age) have developed a navigation skill using pre-charted constellation to sail the open Mediterranean sea all the way up to the Egypt. The key idea is that the angle made by stars is invariant where the zenith can tell us about the current location.

Imaging that you are located at the green point in Figure 1.1. It is possible to identify the location by knowing the projected location of the zenith—the point in the sky or celestial sphere directly above an observer—onto the earth surface given time. However, this is challenging because reliably identifying the zenith is often difficult. Instead, you can use other pre-charted stars. Consider a star $\mathtt{A}$ that is visible from my location. As the star is infinitely far away from the Earth, the lines of projections of the star A are parallel, i.e., the angles α between $\mathbf{L}$ and $\mathbf{L}_{\mathsf{A}}$ and between $\mathbf{L}$ and $\mathbf{L}_{\mathsf{A}'}$ are preserved. Instead of measuring α, the angle $90° - \alpha$ between $\mathbf{L}_{\mathsf{A}'}$ and horizon can be reliably measured by Sextant as shown in Figure 1.2. Using α, the distance from my location to the projection of the star onto the Earth surface, $\mathbf{e}_{\mathsf{A}}$ is $r\alpha$ where r is the radius of the Earth. Any location at $r\alpha$ distance from $\mathbf{e}_{\mathsf{A}}$ can be my location, and therefore, there are infinite number of solutions. One more star can reduce the candidates to two, i.e., the intersection between two greate circles. Three star measurements can uniquely determine the global location. This celestial localization indicates that it is possible to know "where am I?" using points at infinity.

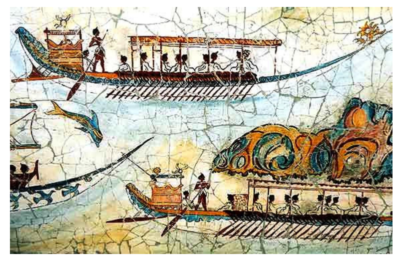


Figure 1.1: Minoan sailors used star locations to travel open Mediterranean sea.

Figure 1.2: Sextant is used to measure the angle between star/moon/sun and horizon.

1.2 2D Line and Point

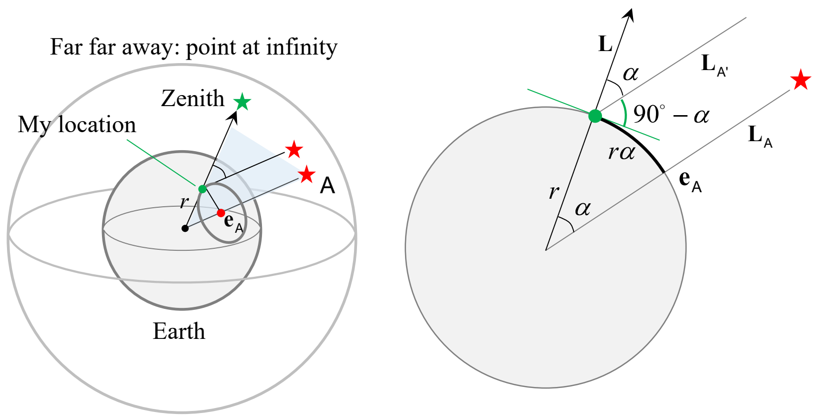


Figure 1.3: Geometry of celestial localization.

Consider a 2D point $(u, v)^{\mathsf{T}}$ in an image as shown in Figure 1.4. Any line passing through this point satisfies:

$$au + bv + c = 0 \tag{1.1}$$

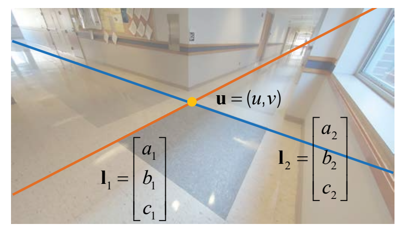


Figure 1.4: Two lines intersect at a point.

where a, b, and c are the line parameters, i.e., $-a/b$ and $-c/b$ are the slope and y-intercept of the line. The Equation (1.1) can be rewritten as a vector form:

$$\begin{aligned} \begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= \mathbf{l}^{\mathsf{T}} \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{u}^{\mathsf{T}} & 1 \end{bmatrix} \mathbf{l} \\ &= 0 \end{aligned} \tag{1.2}$$

where $\mathbf{l} = (a, b, c)^{\mathsf{T}}$ and $\mathbf{u} = (u, v)^{\mathsf{T}}$. Note that three variables describe the line up to scale—any scalar multiplication of line parameters results in an equivalent line, i.e., $\mathbf{l} \equiv \alpha \mathbf{l}$, or $(a, b, c)^{\mathsf{T}} \equiv (\alpha a, \alpha b, \alpha c)^{\mathsf{T}}$ where α is nonzero scalar.

A line can be uniquely defined by two distinctive points $\mathbf{u}_1 = (u_1, v_1)^{\mathsf{T}}$ and $\mathbf{u}_2 = (u_2, v_2)^{\mathsf{T}}$. The line must satisfy the following system of linear equations:

$$au_1 + bv_1 + c = \begin{bmatrix} \mathbf{u}_1^{\mathsf{T}} & 1 \end{bmatrix} \mathbf{l} = 0 \tag{1.3}$$

$$au_2 + bv_2 + c = \begin{bmatrix} \mathbf{u}_2^{\mathsf{T}} & 1 \end{bmatrix} \mathbf{l} = 0 \tag{1.4}$$

or

$$\begin{bmatrix} \mathbf{u}_1^{\mathsf{T}} & 1 \\ \mathbf{u}_2^{\mathsf{T}} & 1 \end{bmatrix} \mathbf{l} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{1.5}$$

Equation (1.8) can be thought of a homogeneous linear system:

$$\mathbf{A}\mathbf{x} = \mathbf{0} \tag{1.6}$$

where $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^m$ is a known matrix, $\mathbf{x} \in \mathbb{R}^n$ is unknown, and $\mathbf{0}$ is m dimensional zero matrix. In this case, $m = 2$ and $n = 3$.

The non-trivial solution of Equation (1.6) is the basis vector of the nullspace of $\mathbf{A}$:

$$\lambda \mathbf{l} = Null\left(\begin{bmatrix} \mathbf{u}_1^\mathsf{T} & 1 \\ \mathbf{u}_2^\mathsf{T} & 1 \end{bmatrix}\right) \tag{1.7}$$

where $Null$ computes the nullspace—orthogonal to the row space of $\mathbf{A}$. Note that there is a scalar λ in LHS as the nullspace basis vector is defined up to scale. For further detail, see Chapter **??**.

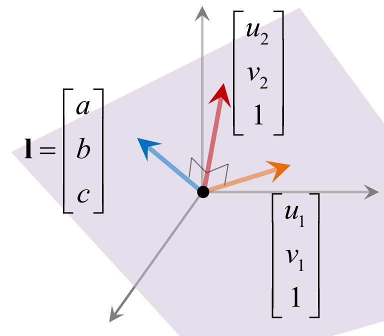


Figure 1.5: Cross product.

Note that there exists one dimensional null basis vector in general due to the rank theorem, i.e., $n = Rank(\mathbf{A}) + Dim(Null(\mathbf{A}))$ where $n = 3$, $Rank(\mathbf{A}) = 2$, and $Dim(\mathcal{H})$ computes the dimension of the subspace $\mathcal{H}$. When two points coincide ($\mathbf{u}_1 = \mathbf{u}_2$), two rows of $\mathbf{A}$ are linear dependent, i.e., $Dim(Null(\mathbf{A})) = 1$. Therefore, two null basis vector exists, which span all lines passing through the coincided points.

As vectors are three dimensional[1], the solution of Equation (1.5) can be also computed by the cross product between two points:

$$\lambda \mathbf{l} = \begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{u}_2 \\ 1 \end{bmatrix} \tag{1.8}$$

Geometrically, two 3D vectors formed by two points $(\mathbf{u}_1, 1)^\mathsf{T}$ and $(\mathbf{u}_2, 1)^\mathsf{T}$ span a plane as shown in Figure 1.5. The line solution $\mathbf{l}$ corresponds to the plane normal vector.

[1] Cross product can be defined only in 3D and 7D.

Now, let two lines, $\mathbf{l}_1$ and $\mathbf{l}_2$, intersect at the point $\mathbf{u}$ as shown in Figure 1.4. Similarly, the intersection point satisfies the following linear equations:

$$a_1 u + b_1 v + c_1 = \mathbf{l}_1^\mathsf{T} \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = 0 \tag{1.9}$$

$$a_2 u + b_2 v + c_2 = \mathbf{l}_2^\mathsf{T} \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = 0 \tag{1.10}$$

As two non-parallel lines meet at a point, $\mathbf{u}$ can be uniquely computed by two known lines, i.e., two equations and two unknowns (u, v) as shown in Figure 1.11:

$$\begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{l}_1^\mathsf{T} \\ \mathbf{l}_2^\mathsf{T} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{1.11}$$

where we color-code the linear systems of equations where the blue represents known matrix and the red represents unknowns.

Again, non-parallel two lines, $\mathbf{l}_1$ and $\mathbf{l}_2$ are linearly independent, i.e., $\mu\mathbf{l}_1 \neq \mathbf{l}_2$ where the intersection point can be computed by cross product of two lines:

$$\lambda \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = \mathbf{l}_1 \times \mathbf{l}_2 \tag{1.12}$$

where λ is the nonzero scalar.

1.3 *Skew-symmetric Representation of Cross Product*

The cross product between two 3D vectors is defined as:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \times \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = - \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} \times \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} y_1 z_2 - z_1 y_2 \\ z_1 x_2 - x_1 z_2 \\ x_1 y_2 - y_1 x_2 \end{bmatrix} \tag{1.13}$$

Equation (1.13) can be written as a matrix-vector multiplication by transforming a 3D vector to 3×3 matrix, which often makes notation simpler:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \times \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \left[\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}\right]_\times \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = - \left[\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}\right]_\times \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \tag{1.14}$$

where

$$\left[\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right]_\times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}. \tag{1.15}$$

$[\mathbf{x}]_\times$ is a skew-symmetric matrix, i.e., $[\mathbf{x}]_\times = -[\mathbf{x}]_\times^\mathsf{T}$. For instance, Equation (1.16) can be rewritten as:

$$\lambda \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = [\mathbf{l}_1]_\times \mathbf{l}_2 \tag{1.16}$$

MATLAB 1.1 (Line and point)**.** *Write a code to compute (1) the line passing two points and (2) the intersection point of two lines.*
Answer

GetPointFromTwoLines.m

```
function u = GetPointFromTwoLines(l1,l2)
u = Vec2Skew(l1)*l2;
u = u/u(3)
```

GetLineFromTwoPoints.m

```
function l = GetLineFromTwoPoints(u1,u2)
l = Vec2Skew(u1)*u2;
```

Vec2Skew.m

```
function skew = Vec2Skew(v)
skew = [0 -v(3) v(2);
      v(3) 0 -v(1);
      -v(2) v(1) 0];
```

1.4 *Line-point duality*

Equation (1.8) and (1.16) show significant analogy between line and point equations. The equations are identical if we switch the meaning of point and line. This is called *line-point duality*—there exists equivalent formula for line given point formula, or vice versa.

Consider a linear transformation $\mathbf{T} \in \mathbb{R}^{3\times3}$, i.e., Euclidean transformation, of a point:

$$\begin{bmatrix} \mathbf{u}_2 \\ 1 \end{bmatrix} = \mathbf{T} \begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix}. \tag{1.17}$$

By the duality, the corresponding transformation of line is:

$$\mathbf{l}_2 = \mathbf{T}^{-\mathsf{T}}\mathbf{l}_1, \tag{1.18}$$

because

$$\mathbf{l}_1^{\mathsf{T}} \begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix} = \left(\mathbf{l}_1^{\mathsf{T}}\mathbf{T}^{-1}\right)\left(\mathbf{T}\begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix}\right) \tag{1.19}$$

$$= \left(\mathbf{T}^{-\mathsf{T}}\mathbf{l}_1\right)^{\mathsf{T}}\left(\mathbf{T}\begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix}\right) = \mathbf{l}_2^{\mathsf{T}}\begin{bmatrix} \mathbf{u}_2 \\ 1 \end{bmatrix} \tag{1.20}$$

Example 1.1 (Line Euclidean transform)**.** *Consider that a point* $\mathbf{u}_1$ *that undergoes 30 degree rotational transformation about the center of image (image size: 1280×960), and is mapped to* $\mathbf{u}_2$*. Give the expression of the transformed line* $\mathbf{l}_2$ *in terms of* $\mathbf{l}_1$.
Solution A point in the image 1 undergoes the following transformation:

$$\begin{bmatrix} \mathbf{u}_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos 30^\circ & \sin 30^\circ & 640 \\ -\sin 30^\circ & \cos 30^\circ & 480 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix} \tag{1.21}$$

$$= \begin{bmatrix} \mathbf{R}_{30^\circ} & \mathbf{t} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix} \tag{1.22}$$

By the duality, the line undergoes the following transformation:

$$\mathbf{l}_2 = \begin{bmatrix} \mathbf{R}_{30^\circ} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}^{-\mathsf{T}} \mathbf{l}_1 \tag{1.23}$$

$$= \begin{bmatrix} \mathbf{R}_{30^\circ}^\mathsf{T} & \mathbf{0} \\ \mathbf{t}^\mathsf{T} & 1 \end{bmatrix}^{-1} \mathbf{l}_1 \tag{1.24}$$

$$= \begin{bmatrix} \mathbf{R}_{30^\circ} & \mathbf{0} \\ \mathbf{t}^\mathsf{T}\mathbf{R}_{30^\circ} & 1 \end{bmatrix} \mathbf{l}_1 \tag{1.25}$$

Show some more examples.

1.5 Geometric Interpretation of 2D Line and Point

Consider a line $\mathbf{l}$ joining two points $\mathbf{u}_1$ and $\mathbf{u}_2$ in a 2D image. These points can be transformed to a normalized coordinate:

$$\begin{bmatrix} \widehat{\mathbf{u}}_1 \\ 1 \end{bmatrix} = \mathbf{K}^{-1} \begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix} \tag{1.26}$$

$$\begin{bmatrix} \widehat{\mathbf{u}}_2 \\ 1 \end{bmatrix} = \mathbf{K}^{-1} \begin{bmatrix} \mathbf{u}_2 \\ 1 \end{bmatrix} \tag{1.27}$$

By the duality, the line can also be represented in the normalized coordinate, $\widehat{\mathbf{l}} = \mathbf{K}^\mathsf{T}\mathbf{l}$ where

$$\widehat{\mathbf{l}} = \begin{bmatrix} \widehat{\mathbf{u}}_2 \\ 1 \end{bmatrix} \times \begin{bmatrix} \widehat{\mathbf{u}}_2 \\ 1 \end{bmatrix} \tag{1.28}$$

Exercise 1.1. *Prove* $\widehat{\mathbf{l}} = \mathbf{K}^\mathsf{T}\mathbf{l}$ *by plugging Equation (1.27) into Equation (1.28).*

Recall that a point in a 2D image corresponds to a 3D ray emitted from the camera center as discussed in Section **??**. Two points in the image form a two 3D rays:

$$\mathbf{L}_1 = \lambda_1 \mathbf{K}^{-1} \begin{bmatrix} \mathbf{u}_1 \\ 1 \end{bmatrix} \tag{1.29}$$

$$\mathbf{L}_2 = \lambda_2 \mathbf{K}^{-1} \begin{bmatrix} \mathbf{u}_2 \\ 1 \end{bmatrix} \tag{1.30}$$

These two rays span a plane which passes through the origin and the 2D line $\widehat{\mathbf{l}}$ in the image as shown in Figure 1.6. Interestingly, the plane normal is $\widehat{\mathbf{l}}$ because

$$\widehat{\mathbf{l}} = \mathbf{L}_1 \times \mathbf{L}_2. \tag{1.31}$$

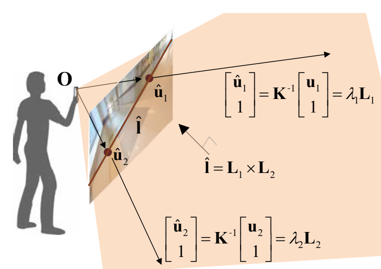


Figure 1.6: Geometric interpretation of line.

Therefore, a line in an image corresponds to a 3D plane where the line parameter indicates the plane normal.

Two non parallel lines meet at a point in 2D. These two lines correspond to two 3D planes represented by the line parameters $\widehat{\mathbf{l}}_1$ and $\widehat{\mathbf{l}}_2$. The intersection of these two 3D planes forms a 3D line that is perpendicular to two plane normals as shown in Figure 1.7:

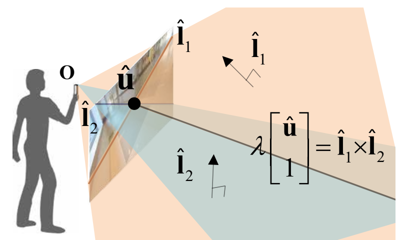


Figure 1.7: Geometric interpretation of line intersection.

$$\mathbf{L} = \widehat{\mathbf{l}}_1 \times \widehat{\mathbf{l}}_2 \tag{1.32}$$

This 3D line coincides with the ray passing through the intersecting point, $\widehat{\mathbf{u}}$, between 2D lines in the normalized image coordinate:

$$\mathbf{K}^{-1}\begin{bmatrix}\mathbf{u}\\1\end{bmatrix} = \begin{bmatrix}\widehat{\mathbf{u}}\\1\end{bmatrix} \tag{1.33}$$

$$= \widehat{\mathbf{l}}_1 \times \widehat{\mathbf{l}}_2 = \mathbf{L} \tag{1.34}$$

Example 1.2 (Zero offset). *What does a line with* $\widehat{\mathbf{l}}_0 = (a, b, 0)^\mathsf{T}$ *mean in 3D?*
Solution From the line equation, $\widehat{\mathbf{l}}_0$ passes through the principal point of the image. This line corresponds to a 3D plane whose normal is $\lambda\mathbf{L}_0 = \widehat{\mathbf{l}}_0$. This plane always passes through the camera optical axis $\mathbf{Z} = (0, 0, 1)^\mathsf{T}$ because

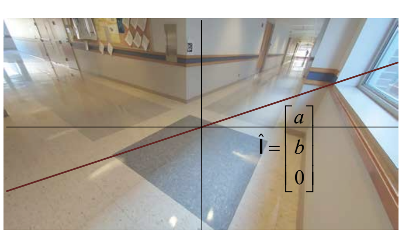
$$\mathbf{L}_0^\mathsf{T}\mathbf{Z} = 0 \tag{1.35}$$

Therefore, it is a plane perpendicular to the camera X-Y plane as shown in Figure 1.2.

When two 2D lines are parallel ($\mathbf{l}_1 = (a, b, c_1)^\mathsf{T}$ and $\mathbf{l}_2 = (a, b, c_2)^\mathsf{T}$), the intersection point $\mathbf{v}_\infty$ is no longer finite because:

$$\lambda\begin{bmatrix}\mathbf{v}_\infty\\1\end{bmatrix} = \mathbf{l}_1 \times \mathbf{l}_2 = (c_2 - c_1)\begin{bmatrix}-b\\a\\0\end{bmatrix} \tag{1.36}$$

This indicate $\mathbf{v}_\infty \to \infty$, i.e., parallel lines meet at a point at infinity.



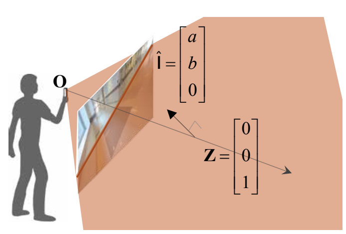


Figure 1.8: Zero offset

This point at infinity corresponds to a ray, $\mathbf{L}_\infty$, lying in the X-Y plane of the camera because it is perpendicular to the optical axis $\mathbf{Z} = (0,0,1)^\mathsf{T}$:

$$\mathbf{L}_\infty^\mathsf{T}\mathbf{Z} = \lambda\mathbf{K}^{-1}\mathbf{v}\mathbf{Z} = 0 \tag{1.37}$$

because $\mathbf{K}^{-1}\mathbf{v} = (\cdot,\cdot,0)^\mathsf{T}$.

We call such point $\mathbf{v}_\infty = (\cdot,\cdot,0)^\mathsf{T}$ as an *ideal point* that exists at infinity. Interestingly, there exists a line that passes through all ideal points:

$$\mathbf{l}_{\text{ideal}} = \mathbf{v}_\infty^1 \times \mathbf{v}_\infty^2 = \begin{bmatrix} a_1 \\ b_1 \\ 0 \end{bmatrix} \times \begin{bmatrix} a_1 \\ b_1 \\ 0 \end{bmatrix} = \lambda \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{1.38}$$

This ideal line corresponds to the camera's X-Y plane whose surface normal is $(0,0,1)^\mathsf{T}$ as shown in Figure 1.9.

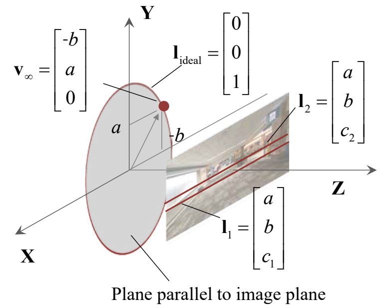


Figure 1.9: Zero offset.

1.6 Vanishing Point and Line

Two parallel 3D lines do not appear parallel in an image in general. These two lines meet at the point at infinity which becomes finite point in image as shown in Figure 1.10.

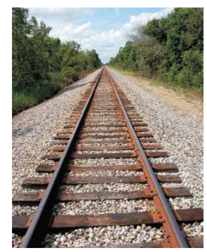


Figure 1.10: Two parallel lines in 3D meets at a vanishing point in an image.

Consider the 3D line on the ground plane $\mathbf{L}_1(\lambda_1) = (0, H, \lambda_1 Z)^\mathsf{T}$ that is emitted from the feet location and aligned with the world Z axis where H is the height of the camera. Assuming that the camera is at origin ($\mathbf{C} = \mathbf{0}$ and $\mathbf{R} = \mathbf{I}_3$). Each point in the line is projected onto the camera:

$$\lambda_1 Z \begin{bmatrix} p_x \\ \frac{Hf_y}{\lambda_1 Z} + p_y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ H \\ \lambda_1 Z \end{bmatrix} \tag{1.39}$$

The projection of the line is the line along the Y axis of image passing through the principal point $(p_x, p_y)^\mathsf{T}$, i.e., the X coordinate is constant. As $\lambda_1 \to \infty$, the projection approaches to $\mathbf{u}_\infty = (p_x, p_y)^\mathsf{T}$.

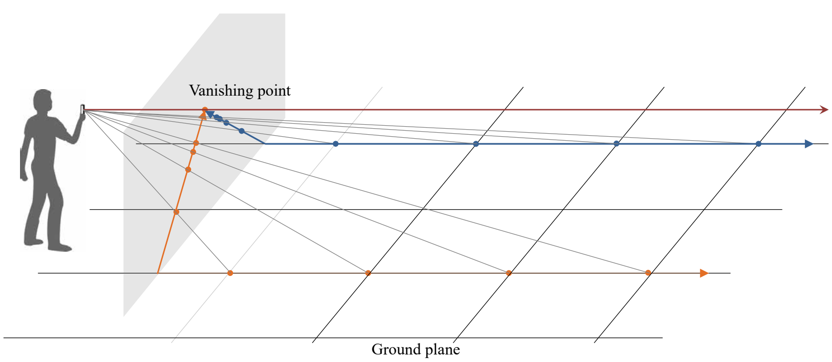


Figure 1.11: .

(a) Horizontal vanishing line (b) Vertical vanishing line

Another 3D line parallel to $\mathbf{L}_1$ can be written as $\mathbf{L}_2 = (X, H, \lambda_2 Z)^\mathsf{T}$. Each point in this line is projected onto the camera:

$$\lambda_1 Z \begin{bmatrix} \frac{X f_y}{\lambda_1 Z} + p_x \\ \frac{H f_y}{\lambda_1 Z} + p_y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ H \\ \lambda_1 Z \end{bmatrix} \tag{1.40}$$

This produces an oblique line in the image and approaches to $\mathbf{u}_\infty = (p_x, p_y)^\mathsf{T}$ as $\lambda_2 \to \infty$. This indicates that the projections of two parallel lines in 3D ($\mathbf{L}_1$ and $\mathbf{L}_2$) meet at $\mathbf{u}_\infty$.

Now consider a general line in 3D, i.e., $\mathbf{L}(\lambda) = (X, Y, Z)^\mathsf{T} + \lambda(d_x, d_y, d_z)^\mathsf{T}$ where $(d_x, d_y, d_z)^\mathsf{T}$ is the direction and $(X, Y, Z)^\mathsf{T}$ is offset. Any line with the same $(d_x, d_y, d_z)^\mathsf{T}$ is parallel. Each point in the line is projected to the camera:

$$\begin{bmatrix} f_x(X + \lambda d_x) + p_x(Z + \lambda d_z) \\ f_y(Y + \lambda d_y) + p_y(Z + \lambda d_z) \\ Z + \lambda d_z \end{bmatrix} = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X + \lambda d_x \\ Y + \lambda d_y \\ Z + \lambda d_z \end{bmatrix}$$

Therefore, the projection of 3D point at infinity is:

$$\lim_{\lambda \to \infty} \begin{bmatrix} \frac{f_x(X+\lambda d_x) + p_x(Z+\lambda d_z)}{Z+\lambda d_z} \\ \frac{f_y(Y+\lambda d_y) + p_y(Z+\lambda d_z)}{Z+\lambda d_z} \end{bmatrix} = \begin{bmatrix} f_x \frac{d_x}{d_z} + p_x \\ f_y \frac{d_y}{d_z} + p_y \end{bmatrix} = \mathbf{u}_\infty \tag{1.41}$$

This result shows that the point at infinity is not a function of the offset $(X, Y, Z)^\mathsf{T}$ but of the direction $(d_x, d_y, d_z)^\mathsf{T}$. Thus, all parallel lines with different offset meet at the same point called *vanishing point*.

For the 3D lines in the ground plane, $d_y = 0$, which results in:

$$\mathbf{u}_\infty^{\text{gnd}} = \begin{bmatrix} f_x \frac{d_x}{d_z} + p_x \\ p_y \end{bmatrix} \tag{1.42}$$

This indicates that the set of vanishing points produced by different directions are lined up along the X axis because p_y is constant, i.e., it forms the horizon in the image. Note that this is the case where the camera orientation is aligned with the ground plane coordinate.

Two vanishing lines can define a line called *vanishing line*. The horizon is an example of vanishing line. When the camera axis is not aligned with the ground plane, the vanishing line can be computed as follows:

$$\mathbf{l}_\infty = \begin{bmatrix} \mathbf{u}^1_\infty \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{u}^2_\infty \\ 1 \end{bmatrix} = \left(\mathbf{l}^1_1 \times \mathbf{l}^1_2\right) \times \left(\mathbf{l}^2_1 \times \mathbf{l}^2_2\right) \tag{1.43}$$

$\mathbf{u}^i_\infty$ is the vanishing point at the i^{th} direction and $\mathbf{l}^j_i$ is the projection of $\mathbf{L}^j_i$ where $\mathbf{L}^1_1 \parallel \mathbf{L}^1_2$ and $\mathbf{L}^2_1 \parallel \mathbf{L}^2_2$.

Figure 1.12: Horizon computation.

MATLAB 1.2 (Horizon). *Write a code to compute the horizon given a square pattern on the floor similar to Figure 1.12.*
Answer

GetHorizon.m

```
m11 = [2145;2120;1];
m12 = [2566;1191;1];
m13 = [1804;935;1];
m14 = [1050;1320;1];
m21 = m11; m22 = m14 ;m23 = m12; m24 = m13;

l11 = GetLineFromTwoPoints(m11,m12);
l12 = GetLineFromTwoPoints(m13,m14);

l21 = GetLineFromTwoPoints(m21,m22);
l22 = GetLineFromTwoPoints(m23,m24);

x1 = GetPointFromTwoLines(l11,l12);
x2 = GetPointFromTwoLines(l21,l22);

horizon = GetLineFromTwoPoints(x1, x2);
```

1.7 *Geometric Interpretation of Vanishing Line*

As discussed in Section 1.5, a line in an image corresponds to a plane in 3D where the normalized line parameter is the plane normal. Similarly, the vanishing line is mapped to the plane in 3D. In particular, the vanishing line created by 3D lines in the ground plane is mapped to the plane parallel to the ground plane because:

$$\mathbf{l} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{1.44}$$

which aligns with the Y axis of the camera where the ground normal is also $(1,0,0)^\mathsf{T}$. When the camera coordinate is not aligned with the origin, still the plane normal of horizon is aligned with the ground normal with camera height offset. Therefore, any point along the horizon has the same height with the camera height.

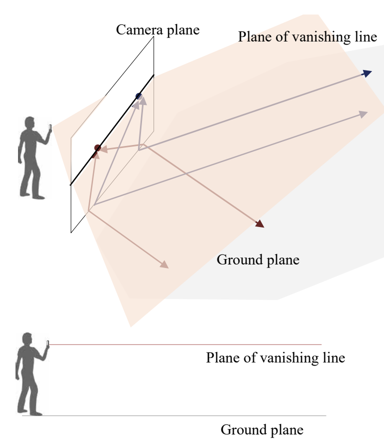


Figure 1.13: The heigh to the camera holder.

(a) Height 1 (b) Height 2

Example 1.3 (Where am I?)**.** *Two images in Figure 1.3 are taken at different heights: one was taken at the ground level and the other one was taken at the* 6^{th} *floor. Which one was taken at the ground level?*
Solution The horizon indicates the height of the camera. In the left image, the horizon passes through the middle of the cargo ship while the bottom of the ship in the right. The right image is taken at the lower level and therefore, ground level.

The vertical vanishing point, $\mathbf{v}_y$ is directly related to the horizon $\mathbf{l}_{\text{hor}}$ because the surface normal of the ground plane is aligned with the vertical direction:

$$\widehat{\mathbf{l}}_{\text{hor}} = \mathbf{K}^{\mathsf{T}}\mathbf{l}_{\text{hor}} = \lambda\mathbf{K}^{-1}\mathbf{v}_y \tag{1.45}$$

where $\lambda\mathbf{K}^{-1}\mathbf{v}_y$ is the ray towards vertical direction. Therefore, the horizon is

$$\mathbf{l}_{\text{hor}} = \mathbf{K}^{-\mathsf{T}}\mathbf{K}^{-1}\mathbf{v}_y \tag{1.46}$$

So knowing the vertical vanishing point implies knowing horizon, and vice versa.

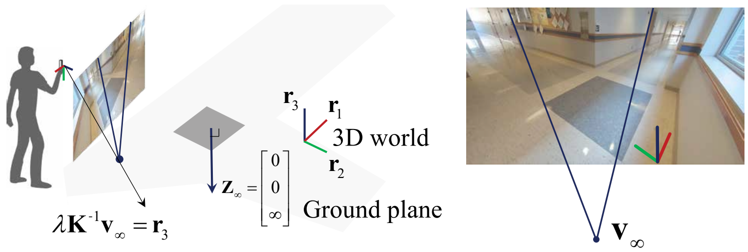
MATLAB 1.3 (Vertical vanishing point)**.** *Given the horizon from MATLAB code 1.2, write a code to compute the vertical vanishing point similar to Figure 1.12(b).*
Answer

1.8 Where am I?

From celestial navigation in Section 1.1, we learned that it is possible to localize my location using vanishing points (points at infinity). Intuitively, knowing horizon tells us about camera's pitch angle (rotation about X axis) and roll angle (rotation about Z axis). The slope of the horizon will tell us about the roll angle and the offset will tell us about the pitch angle. Equivalently, from Equation (1.46), knowing vertical vanishing point is indicative of roll and pitchn angle of the camera.

1.8.1 Localization Cue 1: Single Vanishing Point

Let us define the world coordinate where the ground plane is spanned by the world X-Y axis as shown in Figure 1.14(c), i.e., the surface normal of the



(c) Height 1 (d) Height 2

Figure 1.14: Camera localization with one vanishing point.

ground plane is aligned with Z axis of the world coordinate.

Consider a point in 3D along the world Z axis $\mathbf{Z} = (0, 0, Z)^\mathsf{T}$. The vertical vanishing point $\mathbf{v}_z$ is the projection of $\mathbf{Z}$ as $Z \rightarrow \infty$ onto the camera:

$$\lambda \begin{bmatrix} \mathbf{v}_z \\ 1 \end{bmatrix} = \lim_{Z\rightarrow\infty} \mathbf{K} \begin{bmatrix} | & | & | & | \\ \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \\ | & | & | & | \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ Z \\ 1 \end{bmatrix} \tag{1.47}$$

$$= \mathbf{K}\mathbf{r}_3. \tag{1.48}$$

This result accords with the intuition that the ray corresponding to the vertical vanishing point $\mathbf{K}^{-1} \begin{bmatrix} \mathbf{v}_z \\ 1 \end{bmatrix}$ is aligned with the world Z axis seen from the camera, $\mathbf{r}_3$.

To relate with roll and pitch angles, we parametrize the rotation matrix using roll θ_r, pitch θ_p, yaw θ_y angles:

$$\begin{bmatrix} | & | & | \\ \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \\ | & | & | \end{bmatrix} = \left(\mathbf{R}_{\theta_y}\mathbf{R}_{\theta_p}\mathbf{R}_{\theta_r}\right)^\mathsf{T} \tag{1.49}$$

$$= \begin{bmatrix} \cdot & \cdot & -\sin\theta_p \\ \cdot & \cdot & \cos\theta_p \sin\theta_r \\ \cdot & \cdot & \cos\theta_p \cos\theta_r \end{bmatrix} \tag{1.50}$$

where the rotation matrices for these angles are defined as

$$\mathbf{R}_{\theta_y} = \begin{bmatrix} \cos\theta_y & -\sin\theta_y & 0 \\ \sin\theta_y & \cos\theta_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.51}$$

$$\mathbf{R}_{\theta_p} = \begin{bmatrix} \cos\theta_p & 0 & \sin\theta_p \\ 0 & 1 & 0 \\ -\sin\theta_p & 0 & \cos\theta_p \end{bmatrix} \tag{1.52}$$

$$\mathbf{R}_{\theta_r} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_r & -\sin\theta_r \\ 0 & \sin\theta_r & \cos\theta_r \end{bmatrix} \tag{1.53}$$

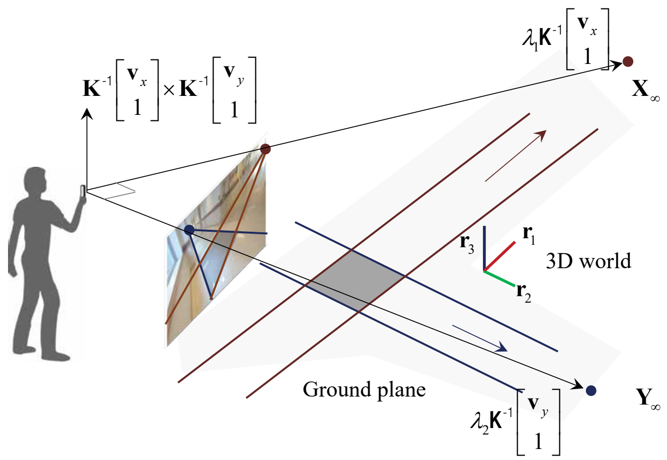


Figure 1.15: Vanishing point geometry.

Given a vertical vanishing point, or equivalently, horizon in an image, the camera's pitch and roll angles can be deduced by Equation (1.48) and (1.50):

$$\theta_p = \tan^{-1}\left(-\frac{r_1}{\sqrt{r_2^2 + r_3^2}}\right) \tag{1.54}$$

$$\theta_r = \tan^{-1}\frac{r_2}{r_3} \tag{1.55}$$

where

$$\frac{\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_z\\1\end{bmatrix}}{\left\|\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_z\\1\end{bmatrix}\right\|} = \begin{bmatrix}r_1\\r_2\\r_3\end{bmatrix} \tag{1.56}$$

Note that the normalization ensures that the resulting vector is a unit vector to represent an orthogonal rotation matrix.

1.8.2 Localization Cue 2: Two Perpendicular Vanishing Points

Knowing two perpendicular vanishing points in the ground plane tells us not only pitch and roll angles but also the yaw angle, i.e., these two vanishing points define X and Y axes of the world coordinate which allows us to compute full rotation matrix while only Z world axis was known in Section 1.8.1.

Similar to Equation (1.48), two perpendicular vanishing points ($\mathbf{v}_x$ and $\mathbf{v}_y$) can be written as:

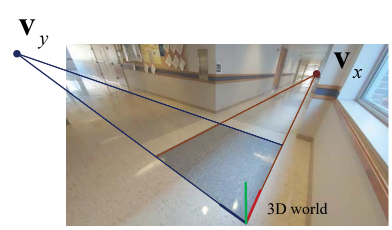


Figure 1.16: Camera localization with XY vanishing points.

$$\lambda_1\begin{bmatrix}\mathbf{v}_x\\1\end{bmatrix} = \mathbf{K}\mathbf{r}_1 \tag{1.57}$$

$$\lambda_2\begin{bmatrix}\mathbf{v}_y\\1\end{bmatrix} = \mathbf{K}\mathbf{r}_2 \tag{1.58}$$

This allows us to represent the camera rotation matrix:

$$\mathbf{r}_1 = \frac{\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_x\\1\end{bmatrix}}{\left\|\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_x\\1\end{bmatrix}\right\|}, \quad \mathbf{r}_2 = \frac{\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_y\\1\end{bmatrix}}{\left\|\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_y\\1\end{bmatrix}\right\|} \tag{1.59}$$

Since the world Z axis is orthogonal to $\mathbf{r}_1$ and $\mathbf{r}_2$,

$$\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2 \tag{1.60}$$

In 3D, the two perpendicular vanishing points form two rays ($\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_x\\1\end{bmatrix}$ and $\mathbf{K}^{-1}\begin{bmatrix}\mathbf{v}_y\\1\end{bmatrix}$) as shown in Figure 1.8.2. These two rays are aligned with X and Y axes of the world coordinate system, respectively. $\mathbf{r}_3$ is the surface normal of the ground plane which is orthogonal to the two rays. The vanishing points allows computing the rotation with respect to the world coordinate system that aligns with the ground plane. Unlike celestial navigation, the location, or translation $\mathbf{t}$ cannot be computed using two vanishing lines, i.e., there exists translation ambiguity. The main difference is that the celestial navigation knows the projection of the vanishing point on the Earth surface, which provides additional cue to identify the location.

MATLAB 1.4 (Camera calibration)**.** *Using the orthogonality of the rotation matrix, it is possible to calibrate the focal length of the camera. Write a code to compute the focal length assuming the principal point is located at the center of image.*
Answer We can search the focal length such that it produces orthogonal rotation matrix, $\mathbf{R} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3\end{bmatrix}$, i.e., $\mathbf{R}^\mathsf{T}\mathbf{R} = \mathbf{R}\mathbf{R}^\mathsf{T} = \mathbf{I}_3$.

CameraCalibrationViaVanishingPoints.m

```
f = 10:5000

l11 = GetLineFromTwoPoints(m11,m12);
l12 = GetLineFromTwoPoints(m13,m14);

l21 = GetLineFromTwoPoints(m21,m22);
l22 = GetLineFromTwoPoints(m23,m24);

v1 = GetPointFromTwoLines(l11,l12);
v2 = GetPointFromTwoLines(l21,l22);

for i = 1 : length(f)
   K = [f(i) 0 size(im,2)/2;
       0 f(i) size(im,1)/2;
       0 0 1];

```

```
    r1 = inv(K)*v1/norm(inv(K)*v1);
    r2 = inv(K)*v2/norm(inv(K)*v2);
    r3 = Vec2Skew(r1)*r2;
    R = [r1 r2 r3];

    err(i) = norm(eye(3)-R'*R);
end

[~,best] = min(err);
best_f = f(best);
```

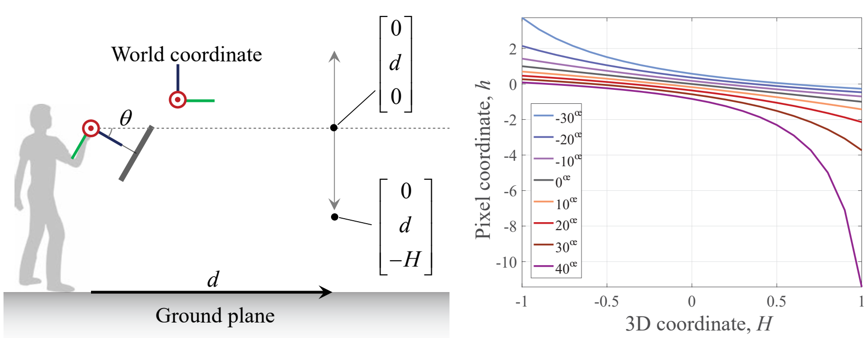
1.9 *Single View Metrology*

The same person in Figure 1.17 appears differently: the left person looks smaller than the right. This picture is framed in an *Ames Room*—a room that creates visual illusion due to perspective projection. The room layout from top view is actually trapezoidal while the objects and visual patterns such as wall painting and checkerboard floor are designed to project on the pinhole in the same way as the rectangular room that is more natural to humans. For instance, the table on the left side is bigger and farther than right side in reality while they appear the same. As a result, the person on the left is seen relatively smaller than one on the right. This Ames Room illustrates that the size measured from an image is not precise to represent the real object size due to the perspective projection.

Figure 1.17: Ames room

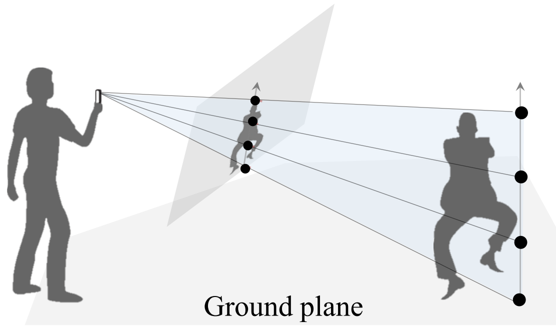
Consider a camera located at the world origin. Given depth d, the point along the vertical direction in 3D is project on the camera whose orientation is θ angle as shown in Figure 1.18(a):

$$\lambda \begin{bmatrix} 0 \\ h \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\sin\theta & -\cos\theta \\ 0 & \cos\theta & -\sin\theta \end{bmatrix} \begin{bmatrix} 0 \\ d \\ -H \end{bmatrix} \tag{1.61}$$

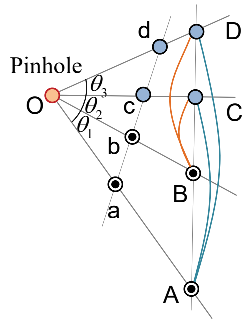


(a) Height 1 (b) Height 2

Figure 1.18: dfa



(a) Height 1



(b) Height 2

Figure 1.19: adfa

where we assume the intrinsic parameter as $\mathbf{K} = \mathbf{I}_3$. This results in

$$h = \frac{-d\sin\theta + H\cos\theta}{d\cos\theta + H\sin\theta} \tag{1.62}$$

The height in 3D H has nonlinear relationship with the pixel height h in general. For the special case of $\theta = 0$, $h = -H/d$ and therefore linear. This nonlinear relationship is shown in Figure 1.18(b). Unless the Y axis of the camera is perfectly aligned with the ground plane normal, the 3D height cannot be directly measured from the image assuming the camera orientation θ is unknown.

1.9.1 Cross Ratio

Although the perspective projection distort the size ratio nonlinearly, there is a measure called *cross ratio* that is invariant to the perspective projection.

Theorem 1. *Given four points* A, B, C, *and* D *on a line, not equal, the cross ratio is preserved under perspective transformation, i.e.,*

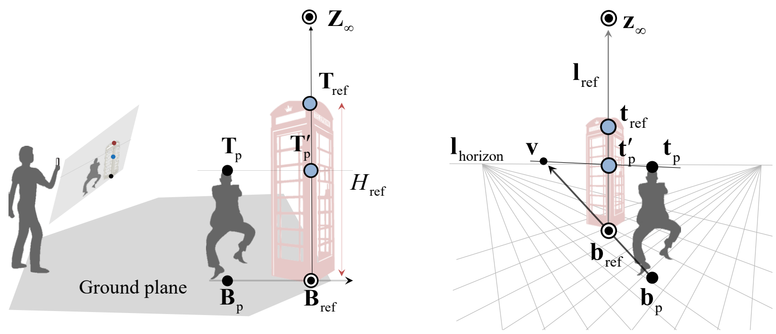
$$\frac{|\overline{\mathsf{AC}}|}{|\overline{\mathsf{AD}}|}\frac{|\overline{\mathsf{BD}}|}{|\overline{\mathsf{BC}}|} = \frac{|\overline{\mathsf{ac}}|}{|\overline{\mathsf{ad}}|}\frac{|\overline{\mathsf{bd}}|}{|\overline{\mathsf{bc}}|}. \tag{1.63}$$

where a, b, c, *and* d *are four points on another line after the perspective transformation.*

Proof. The area of the triangle $\triangle\mathsf{OAB}$ is

$$|\triangle\mathsf{OAB}| = \frac{1}{2}|\overline{\mathsf{AB}}|h = \frac{1}{2}d_{\mathsf{A}}d_{\mathsf{B}}\sin\theta_1 \tag{1.64}$$

where h is the height of the triangle given $\overline{\mathsf{AB}}$ baseline. Likewise, from the



(a) Height 1 (b) Height 2

Figure 1.20: adfa

equation of triangle area, the baseline can be related with angle:

$$|\overline{\mathsf{AC}}| = \frac{d_{\mathsf{A}} d_{\mathsf{C}} \sin(\theta_1 + \theta_2)}{h} \tag{1.65}$$

$$|\overline{\mathsf{AD}}| = \frac{d_{\mathsf{A}} d_{\mathsf{D}} \sin(\theta_1 + \theta_2 + \theta_3)}{h} \tag{1.66}$$

$$|\overline{\mathsf{BC}}| = \frac{d_{\mathsf{B}} d_{\mathsf{C}} \sin\theta_2}{h} \tag{1.67}$$

$$|\overline{\mathsf{BD}}| = \frac{d_{\mathsf{B}} d_{\mathsf{D}} \sin(\theta_2 + \theta_3)}{h} \tag{1.68}$$

This allows writing the cross ratio in terms of pure angles:

$$\frac{|\overline{\mathsf{AC}}|}{|\overline{\mathsf{AD}}|} \frac{|\overline{\mathsf{BD}}|}{|\overline{\mathsf{BC}}|} = \frac{\sin(\theta_1 + \theta_2)}{\sin(\theta_1 + \theta_2 + \theta_3)} \frac{\sin(\theta_2 + \theta_3)}{\sin\theta_2} \tag{1.69}$$

The height h and edges d_{A}, d_{B}, d_{C}, and d_{D} are canceled out. Therefore, the cross ratio is invariant to the line defining the four points. Four points defined by any line satisfy the cross ratio (any perspective transformation. □

1.9.2 *Perspective Metrology*

Since the cross ratio is preserved after the perspective projection, it is possible to use the cross ratio to measure the size of an object given a known reference size[2] such as height of a person, building, or furniture.

[2] The intrinsic parameter **K** encapsulates the mapping from 3D metric space and pixel space where the physical scale of scene is impossible to recover given an image.

Consider a person in an image whose height is unknown. Given a vanishing point on the horizon, it is possible to predict the image of the person, i.e., size, as moving towards the vanishing points. If there exists another 3D object such as English phone booth, we can compare their heights by finding a vanishing line passing through their ground projection points. In other words, we can project the person onto the phone booth, or project the phone booth to the person to compare their heights because we can predict the size given the vanishing point.

This cross ratio allows us to compute the height of the person H_{p} where the height of the phone booth H_{ref} is known (reference object). A key idea is to co-locate two objects such that they can form a line along the height direction, i.e., the bottom and top points of both objects are colinear in the image and 3D. In conjunction with vertical point at infinity, this co-location allows to relate the line in the image with the line in 3D using the cross ratio.

Using the phone booth bottom and top points, the line along the vertical vanishing point is:

$$\mathbf{l}_{\rm ref} = \begin{bmatrix} \mathbf{b}_{\rm ref} \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{t}_{\rm ref} \\ 1 \end{bmatrix} \tag{1.70}$$

From other vertical lines in the image, the vertical vanishing point $\mathbf{z}_\infty$ along $\mathbf{l}_{\rm ref}$ can be computed. This point in the corresponds to $\mathbf{Z}_\infty = (0, 0, \infty)^\mathsf{T}$ in 3D.

By co-locating the person at the phone booth position $\mathbf{b}_{\rm ref}$, we can predict the person's head location by projecting onto the phone booth vertical line $\mathbf{l}_{\rm ref}$:

$$\begin{bmatrix} \mathbf{t}'_{\rm p} \\ 1 \end{bmatrix} = \mathbf{l}_{\rm ref} \times \left(\begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{t}_{\rm p} \\ 1 \end{bmatrix} \right) \tag{1.71}$$

where

$$\lambda \begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix} = \mathbf{l}_{\rm horizon} \times \left(\begin{bmatrix} \mathbf{b}_{\rm p} \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{b}_{\rm ref} \\ 1 \end{bmatrix} \right) \tag{1.72}$$

Now $\{\mathbf{b}_{\rm ref}, \mathbf{t}'_{\rm p}, \mathbf{t}_{\rm ref}, \mathbf{z}_\infty\} \leftrightarrow \{\mathbf{B}_{\rm ref}, \mathbf{T}'_{\rm p}, \mathbf{T}_{\rm ref}, \mathbf{Z}_\infty\}$ form lines in the image and 3D, respectively. This two lines with pinhole location allows us to apply the cross ratio:

$$\frac{\|\mathbf{b}_{\rm ref} - \mathbf{t}_{\rm ref}\|}{\|\mathbf{b}_{\rm ref} - \mathbf{t}'_{\rm p}\|} \frac{\|\mathbf{z}_\infty - \mathbf{t}'_{\rm p}\|}{\|\mathbf{z}_\infty - \mathbf{t}_{\rm ref}\|} = \frac{\|\mathbf{B}_{\rm ref} - \mathbf{T}_{\rm ref}\|}{\|\mathbf{B}_{\rm ref} - \mathbf{T}'_{\rm p}\|} \frac{\|\mathbf{Z}_\infty - \mathbf{T}'_{\rm p}\|}{\|\mathbf{Z}_\infty - \mathbf{T}_{\rm ref}\|} \tag{1.73}$$

$$= \frac{\|\mathbf{B}_{\rm ref} - \mathbf{T}_{\rm ref}\|}{\|\mathbf{B}_{\rm ref} - \mathbf{T}'_{\rm p}\|} \frac{\infty}{\infty} \tag{1.74}$$

$$= \frac{\|\mathbf{B}_{\rm ref} - \mathbf{T}_{\rm ref}\|}{\|\mathbf{B}_{\rm ref} - \mathbf{T}'_{\rm p}\|} \tag{1.75}$$

Therefore, the height of the person can be computed:

$$\|\mathbf{B}_{\rm p} - \mathbf{T}_{\rm p}\| = \|\mathbf{B}_{\rm ref} - \mathbf{T}'_{\rm p}\| \tag{1.76}$$

$$= \|\mathbf{B}_{\rm ref} - \mathbf{T}_{\rm ref}\| \frac{\|\mathbf{b}_{\rm ref} - \mathbf{t}'_{\rm p}\|}{\|\mathbf{b}_{\rm ref} - \mathbf{t}_{\rm ref}\|} \frac{\|\mathbf{z}_\infty - \mathbf{t}_{\rm ref}\|}{\|\mathbf{z}_\infty - \mathbf{t}'_{\rm p}\|} \tag{1.77}$$

$$= H_{\rm ref} \frac{\|\mathbf{b}_{\rm ref} - \mathbf{t}'_{\rm p}\|}{\|\mathbf{b}_{\rm ref} - \mathbf{t}_{\rm ref}\|} \frac{\|\mathbf{z}_\infty - \mathbf{t}_{\rm ref}\|}{\|\mathbf{z}_\infty - \mathbf{t}'_{\rm p}\|} \tag{1.78}$$

MATLAB 1.5 (Height computation). *Given the image in the Figure* **??** *and the lady's height* $h_{\rm lady} = 1.6m$*, write a code to compute her friend in the back. Use the same code to compute the height of the street lamps. Are they the same?*
Answer

Figure 1.21: Cross ratio

ComputeHeightFromCrossRatio.m

```
sh_f = [1504;1447;1];
sh_h = [1468;730;1];
jp_f = [1997;1175;1];
jp_h = [1997;695;1];

l11 = GetLineFromTwoPoints(m1,m2);
l12 = GetLineFromTwoPoints(m3,m4);
l21 = GetLineFromTwoPoints(m1,m3);
l22 = GetLineFromTwoPoints(m2,m4);

v1 = GetPointFromTwoLines(l11,l12);
v2 = GetPointFromTwoLines(l21,l22);
l = GetLineFromTwoPoints(v1,v2);

line_sh_jp_f = GetLineFromTwoPoints(sh_f, jp_f);
v = GetPointFromTwoLines(line_sh_jp_f, l);

line_jp_h_v = GetLineFromTwoPoints(jp_head, v);
line_sh = GetLineFromTwoPoints(sh_h, sh_f);
p3 = GetPointFromTwoLines(line_jp_head_v, line_sh);
p2 = sh_h;
p1 = sh_f;

l31 = GetLineFromTwoPoints(m5,m6);
l32 = GetLineFromTwoPoints(m7,m8)
v3 = GetPointFromTwoLines(l31,l32);
p4 = v3;
```