Pinhole cameras

- Abstract camera model - box with a small hole in it
- Pinhole cameras work in practice
Real Pinhole Cameras

Pinhole too big - many directions are averaged, blurring the image

Pinhole too small - diffraction effects blur the image

Generally, pinhole cameras are dark, because a very small set of rays from a particular point hits the screen.

The reason for lenses

Lenses gather and focus light, allowing for brighter images.
The thin lens

Thin Lens Properties:

1. A ray entering parallel to optical axis goes through the focal point.
2. A ray emerging from focal point is parallel to optical axis
3. A ray through the optical center is unaltered

\[ \frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \]

Note that, if the image plane is very small and/or \( z \gg z' \), then \( z' \) is about \( f \).
Field of View

• The effective diameter of a lens \( (d) \) is the portion of a lens actually reachable by light rays.

• The effective diameter and the focal length determine the field of view:

\[
\tan w = \frac{d}{(2f')}
\]

• \( w \) is the half the total angular “view” of a lens system.

• Another fact is that in practice points at different distances are imaged, leading to so-called “circles of confusion” of size \( \frac{d}{z} \frac{|z'-z|}{z'} \) where \( z \) is the nominal image plane and \( z' \) is the focusing distance given by the thin lens equation.

• The “depth of field” is the range of distances that produce acceptably focused images. Depth of field varies inversely with focal length and lens diameter.

---

Lens Realities

Real lenses have a finite depth of field, and usually suffer from a variety of defects

vignetting

Spherical Aberration
Standard Camera Coordinates

- By convention, we place the image in front of the optical center
  - typically we approximate by saying it lies one focal distance from the center
  - in reality this can't be true for a finite size chip!

- Optical axis is z axis pointing outward

- X axis is parallel to the scanlines (rows) pointing to the right!

- By the right hand rule, the Y axis must point downward

- Note this corresponds with indexing an image from the upper left to the lower right, where the X coordinate is the column index and the Y coordinate is the row index.

The equation of projection

- Equating \( z' \) and \( f \)
  - We have, by similar triangles, that \( (x, y, z) \rightarrow (-f \frac{x}{z}, -f \frac{y}{z}, -f) \)
  - Ignore the third coordinate, and flip the image around to get:

\[
(x, y, z) \rightarrow \left( f \frac{x}{z}, f \frac{y}{z} \right)
\]
Distant objects are smaller

Parallel lines meet
common to draw film plane in front of the focal point

A Good Exercise: Show this is the case!
Some Useful Geometry

- In 3D space
  - points:
    - Cartesian point \((x,y,z)\)
    - Projective pt \((x,y,z,w)\) with convention that \(w\) is a scale factor
  - lines:
    - a point \(p\) on the line and unit vector \(v\) for direction
      - for minimal parameterization, \(p\) is closest point to origin
    - Alternative, a line is the intersection of two planes (see below)
  - planes
    - a point \(p\) on the plane and a unit normal \(n\) s.t. \(n \cdot (p' - p) = 0\)
      - multiplying through, also \(n \cdot p' - d = 0\), where \(d\) is distance of closest pt to origin.
    - any vector \(n \cdot q = 0\) where \(q\) is a projective pt
      - note, for two planes, the intersection is two equations in 4 unknowns up to scale --- i.e. a one-dimensional subspace, or a line
    - Note that planes and points are dual --- in the above, I can equally think of \(n\) or \(q\) as the normal (resp. point).

- In 2D space
  - points:
    - Cartesian point \((x,y)\)
    - Projective pt \((x,y,w)\) with convention that \(w\) is a scale factor
  - lines
    - a point \(p\) on the line and a unit normal \(n\) s.t. \(n \cdot (p' - p) = 0\)
      - multiplying through, also \(n \cdot p' - d = 0\), where \(d\) is distance of closest pt to origin.
    - any vector \(n \cdot q = 0\) where \(q\) is a projective pt
      - note, for two lines, the intersection is two equations in 3 unknowns up to scale --- i.e. a one-dimensional subspace, or a point
    - note that points and lines are dual --- I can think of \(n\) or \(q\) as the normal (resp. point)
Some Projective Concepts

• The vector \( p = (x,y,z,w)' \) is equivalent to the vector \( kp \) for nonzero \( k \)
  – note the vector \( p = 0 \) is disallowed from this representation

• The vector \( v = (x,y,z,0)' \) is termed a “point at infinity”; it corresponds to a direction

• In \( \mathbb{P}^2 \),
  – given two points \( p_1 \) and \( p_2 \), \( l = p_1 \times p_2 \) is the line containing them
  – given two lines, \( l_1 \) and \( l_2 \), \( p = l_1 \times l_2 \) is point of intersection
  – A point \( p \) lies on a line \( l \) if \( p \times l = 0 \) (note this is a consequence of the triple product rule)
  – \( l = (0,0,1) \) is the “line at infinity”
  – it follows that, for any point \( p \) at infinity, \( l \times p = 0 \), which implies that points at infinity lie on the line at infinity.

Some Projective Concepts

• The vector \( p = (x,y,z,w)' \) is equivalent to the vector \( kp \) for nonzero \( k \)
  – note the vector \( p = 0 \) is disallowed from this representation

• The vector \( v = (x,y,z,0)' \) is termed a “point at infinity”; it corresponds to a direction

• In \( \mathbb{P}^3 \),
  – A point \( p \) lies on a plane \( l \) if \( p \times l = 0 \) (note this is a consequence of the triple product rule; there is an equivalent expression in determinants)
  – \( l = (0,0,0,1) \) is the “plane at infinity”
  – it follows that, for any point \( p \) at infinity, \( l \times p = 0 \), which implies that points at infinity lie on the line at infinity.
Some Projective Concepts

- The vector \( p = (x,y,z,w)' \) is equivalent to the vector \( kp \) for nonzero \( k \)
  - note the vector \( p = 0 \) is disallowed from this representation
- The vector \( v = (x,y,z,0)' \) is termed a “point at infinity”; it corresponds to a direction
- Plücker coordinates
  - In general, a representation for a line through points \( p_1 \) and \( p_2 \) is given by all possible 2x2 determinants of \( [p_1 \ p_2] \) (an \( n \) by \( 2 \) matrix)
    - \( u = (l_{14}, l_{15}, l_{16}, l_{17}, l_{18}) \) are the Plücker coordinates of the line passing through the two points.
    - if the points are not at infinity, then this is also the same as \( (p_2 - p_1, p_1 \times p_2) \)
  - The first 3 coordinates are the direction of the line
  - The second 3 are the normal to the plane (in \( \mathbb{R}^3 \)) containing the origin and the points
  - In general, a representation for a plane passing through three points \( p_1, p_2 \) and \( p_3 \) are the determinants of all 3 by 3 submatrices \( [p_1 \ p_2 \ p_3] \)
    - let \( l_{i,j} \) mean the determinant of the matrix of matrix formed by the rows \( i \) and \( j \)
    - \( P = (l_{234}, l_{134}, l_{142}, l_{123}) \)
    - Note the three points are colinear if all four of these values are zero (hence the original 3x4 matrix has rank 2, as we would expect).
  - Two lines are colinear if we create the 4x4 matrix \( [p_1, p_2, p_1', p_2'] \) where the \( p \)'s come from one line, and the \( p \)'s come from another.

Parallel lines meet

- First, show how lines project to images.
- Second, consider lines that have the same direction (are parallel)
- Third, consider the degenerate case of lines parallel in the image
  - (by convention, the vanishing point is at infinity!)

A Good Exercise: Show this is the case!
Vanishing points

• Another good exercise (really follows from the previous one): show the form of projection of *lines* into images.

• Each set of parallel lines (=direction) meets at a different point
  – The vanishing point for this direction

• Sets of parallel lines on the same plane lead to collinear vanishing points.
  – The line is called the horizon for that plane

The Camera Matrix

• Homogenous coordinates for 3D
  – four coordinates for 3D point
  – equivalence relation \((X,Y,Z,T) \text{ is the same as } (kX, kY, kZ, kT)\)

• Turn previous expression into HC’s
  – HC’s for 3D point are \((X,Y,Z,T)\)
  – HC’s for point in image are \((U,V,W)\)

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix} =
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & f \\
0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
X \\
Y \\
Z \\
T
\end{pmatrix}
\]

\[ (U,V,W) \rightarrow \left( \frac{U}{W}, \frac{V}{W} \right) = (u,v) \]
Orthographic projection

Suppose I let \( f \) go to infinity; then

\[ u = x \]
\[ v = y \]

The model for orthographic projection

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
X \\
Y \\
Z \\
T
\end{pmatrix}
\]
Weak perspective

- Issue
  - perspective effects, but not over the scale of individual objects
  - collect points into a group at about the same depth, then divide each point by the depth of its group
  - Adv: easy
  - Disadv: wrong

\[ u = sx \]
\[ v = sy \]
\[ s = f / Z^* \]

The model for weak perspective projection

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix}
= \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & Z^*/f & Z
\end{pmatrix}
\begin{pmatrix}
X \\
Y \\
Z
\end{pmatrix}
\]
The Affine Camera

- Choose a nominal point \( x_0, y_0, z_0 \) and describe projection relative to that point

- \( u = f \left[ \frac{x_0}{z_0} + \frac{x-x_0}{z_0} \right] = f (a_1 x + a_2 z + d_1) \)
- \( v = f \left[ \frac{y_0}{z_0} + \frac{y-y_0}{z_0} \right] = f (a_3 y + a_4 z + d_2) \)

- gathering up

- \( A = [a_1, 0, a_2, 0, a_3, a_4] \)
- \( d = [d_1, d_2] \)
- \( u = A P + d \)

Geometric Transforms

In general, a point in n-D space transforms by

\[ P' = \text{rotate}(point) + \text{translate}(point) \]

In 2-D space, this can be written as a matrix equation:

\[
\begin{pmatrix}
x' \\
y'
\end{pmatrix} =
\begin{pmatrix}
\cos(\theta) & -\sin(\theta) \\
\sin(\theta) & \cos(\theta)
\end{pmatrix}
\begin{pmatrix}
x \\
y
\end{pmatrix} +
\begin{pmatrix}
tx \\
ty
\end{pmatrix}
\]

In 3-D space (or n-D), this can generalized as a matrix equation:

\[ p' = R p + T \quad \text{or} \quad p = R^t (p' - T) \]
Geometric Transforms

Now, using the idea of homogeneous transforms, we can write:

\[
p' = \begin{pmatrix} R & T \\ 0 & 0 & 0 & 1 \end{pmatrix} p
\]

R and T both require 3 parameters. These correspond to the 6 extrinsic parameters needed for camera calibration.

Intrinsic Parameters

*Intrinsic Parameters* describe the conversion from unit focal length metric to pixel coordinates (and the reverse)

\[
x_{\text{mm}} = -(x_{\text{pix}} - o_x) s_x \rightarrow -1/s_x x_{\text{mm}} + o_x = x_{\text{pix}}
\]

\[
y_{\text{mm}} = -(y_{\text{pix}} - o_y) s_y \rightarrow -1/s_y y_{\text{mm}} + o_y = y_{\text{pix}}
\]

or

\[
\begin{pmatrix} x \\ y \\ w \end{pmatrix}_{\text{pix}} = \begin{pmatrix} -1/s_x & 0 & o_x \\ 0 & -1/s_y & o_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ w \end{pmatrix}_{\text{mm}} = K_{\text{int}} p
\]

It is common to combine scale and focal length together as these are both scaling factors; note projection is unitless in this case!
The Camera Matrix

- Homogenous coordinates for 3D
  - four coordinates for 3D point
  - equivalence relation \((X,Y,Z,T)\) is the same as \((kX, kY, kZ,kT)\)
- Turn previous expression into HC’s
  - HC’s for 3D point are \((X,Y,Z,T)\)
  - HC’s for point in image are \((U,V,W)\)

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 / f & 0
\end{pmatrix} \begin{pmatrix}
X \\
Y \\
Z \\
T
\end{pmatrix} \quad (U,V,W) \rightarrow \left(\frac{U}{W},\frac{V}{W}\right) = (u,v)
\]

Camera parameters

- Summary:
  - points expressed in external frame
  - points are converted to canonical camera coordinates
  - points are projected
  - points are converted to pixel units

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix} = \text{Transformation representing intrinsic parameters} \quad \begin{pmatrix}
X \\
Y \\
Z
\end{pmatrix} \quad \text{Transformation representing projection model} \quad \begin{pmatrix}
X \\
Y \\
Z \\
T
\end{pmatrix} \quad \text{Transformation representing extrinsic parameters}
\]

point in pixel coords. \quad point in metric image coords. \quad point in cam. coords. \quad point in world coords.
Lens Distortion

• In general, lenses introduce minor irregularities into images, typically radial distortions:

\[
\begin{align*}
x &= x_d(1 + k_1 r^2 + k_2 r^4) \\
y &= y_d(1 + k_1 r^2 + k_2 r^4) \\
r^2 &= x_d^2 + y_d^2
\end{align*}
\]

• The values \(k_1\) and \(k_2\) are additional parameters that must be estimated in order to have a model for the camera system.

Summary: Other Models

• The orthographic and scaled orthographic cameras (also called weak perspective)
  – simply ignore \(z\)
  – differ in the scaling from \(x/y\) to \(u/v\) coordinates
  – preserve Euclidean structure to a great degree

• The affine camera is a generalization of orthographic models.
  – \(u = A p + d\)
  – \(A\) is 2 x 3 and \(d\) is 2x1
  – This can be derived from scaled orthography or by linearizing perspective about a point not on the optical axis

• The projective camera is a generalization of the perspective camera.
  – \(u' = M p\)
  – \(M\) is 3x4 nonsingular defined up to a scale factor
  – This just a generalization (by one parameter) from “real” model

• Both have the advantage of being linear models on real and projective spaces, respectively.
Related Transformation Models

- Euclidean models (homogeneous transforms); $b^p = b^T_a a^p$
- Similarity models: $b^p = s b^T_a a^p$
- Affine models: $b^p = b^K_a a^p$, $K = [A,t;0 0 0 1]$, $A \in \text{GL}(3)$
- Projective models: $b^p = b^M_a a^p$, $M \in \text{GL}(4)$
  - Ray models
  - Affine plane
  - Sphere

Model Stratification

<table>
<thead>
<tr>
<th>Transforms</th>
<th>Euclidean</th>
<th>Similarity</th>
<th>Affine</th>
<th>Projective</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotation</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>translation</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>uniform scaling</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>nonuniform scaling</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>shear</td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>perspective</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>composition of proj.</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Invariants</th>
<th>Euclidean</th>
<th>Similarity</th>
<th>Affine</th>
<th>Projective</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>angle</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ratios</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>parallelism</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>incidence/cross rat.</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>
Why Projective (or Affine or ...) 

- Recall in Euclidean space, we can define a change of coordinates by choosing a new origin and three orthogonal unit vectors that are the new coordinate axes
  - The class of all such transformation is SE(3) which forms a group
  - One rendering is the class of all homogeneous transformations
  - This does not model what happens when things are imaged (why?)
- If we allow a change in scale, we arrive at similarity transforms, also a group
  - This sometimes can model what happens in imaging (why?)
- If we allow the 3x3 rotation to be an arbitrary member of GL(3) we arrive at affine transformations (yet another group!)
  - This also sometimes is a good model of imaging
  - The basis is now defined by three arbitrary, non-parallel vectors
- The process of perspective projection does not form a group
  - that is, a picture of a picture cannot in general be described as a perspective projection
- Projective systems include perspectivities as a special case and do form a group
  - We now require 4 basis vectors (three axes plus an additional independent vector)
  - A model for linear transformations (also called collineations or homographies) on $P^n$ is $GL(n+1)$ which is, of course, a group

Camera calibration

- Issues:
  - what are intrinsic parameters of the camera?
  - what is the camera matrix? (intrinsic+extrinsic)
- General strategy:
  - view calibration object
  - identify image points
  - obtain camera matrix by minimizing error
  - obtain intrinsic parameters from camera matrix
- Most modern systems employ the multi-plane method
  - avoids knowing absolute coordinates of calibration points
- Error minimization:
  - Linear least squares
    - easy problem numerically
    - solution can be rather bad
  - Minimize image distance
    - more difficult numerical problem
    - solution usually rather good, but can be hard to find
      - start with linear least squares
  - Numerical scaling is an issue
The problem:
Compute the camera intrinsic (4 or more) and extrinsic parameters (6) using only observed camera data.

CAMERA CALIBRATION: A WARMUP

A simple way to get scale parameters; we can compute the optical center as the numerical center and therefore have the intrinsic parameters.
Calibration: Another Warmup

• Suppose we want to calibrate the affine camera and we know $u_i = A p_i + d$ for many pairs $i$

• $m$ is mean of $u$'s and $q$ is mean of $p$’s; note $m = A q + d$

• $U = [u_1 - m, u_2 - m, \ldots, u_n - m]$ and $P = [p_1 - q, p_2 - q, \ldots, p_n - q]$

• $U = A P \Diamond U P' (P P')^{-1} = A$

• $d$ is now mean of $u_i - A p_i$

Types of Calibration

• Photogrammetric Calibration
• Self Calibration
• Multi-Plane Calibration
Photogrammetric Calibration

- Calibration is performed through imaging a pattern whose geometry in 3d is known with high precision.

- PRO: Calibration can be performed very efficiently
- CON: Expensive set-up apparatus is required; multiple orthogonal planes.

- Approach 1: Direct Parameter Calibration
- Approach 2: Projection Matrix Estimation

The General Case

- Affine is “easy” because it is linear and unconstrained (note orthographic is harder because of constraints)

- Perspective case is also harder because it is both nonlinear and constrained

- Observation: optical center can be computed from the orthocenter of vanishing points of orthogonal sets of lines.
Basic Equations

\[ cT_w = (T_x, T_y, T_z)' \]

\[ cR_w = (R_x, R_y, R_z)' \]

\[ c_p = cR_w p + cT_w \]

\[ u = -f \frac{R_x p + T_x}{R_z p + T_z} \]

\[ v = -f \frac{R_y p + T_y}{R_z p + T_z} \]

Basic Equations

\[ u_{pix} = \frac{1}{s_x} u + o_x \]

\[ v_{pix} = \frac{1}{s_y} v + o_y \]

\[ \bar{u} = u_{pix} - o_x = -f \frac{R_x p + T_x}{R_z p + T_z} \]

\[ \bar{v} = v_{pix} - o_y = -f \frac{R_y p + T_y}{R_z p + T_z} \]
Basic Equations

\[ \bar{w}_i f_y (R_y p_i + T_y) = \bar{v}_i f_x (R_x p_i + T_x) \]
\[ \bar{w}_i (R_y p_i - T_y) - \bar{v}_i \alpha (R_x p_i + T_x) = 0 \]

\[ r = \alpha R_x \text{ and } w = \alpha T_x \]
\[ t = R_y \text{ and } s = T_y \]

one of these for each point

\[ A_i = (u_i p_i, u_i, -v_i p_i, -v_i) \text{ and } A[t, s, w, r]' = 0 \]

Properties of SVD

• Recall the singular values of a matrix are related to its rank.

• Recall that \( Ax = 0 \) can have a nonzero \( x \) as solution only if \( A \) is singular.

• Finally, note that the matrix \( V \) of the SVD is an orthogonal basis for the domain of \( A \); in particular the zero singular values are the basis vectors for the null space.

• Putting all this together, we see that \( A \) must have rank 7 (in this particular case) and thus \( x \) must be a vector in this subspace.

• Clearly, \( x \) is defined only up to scale.
Basic Equations

\[ A_i = (u_i p_i, u_i, -v_i p_i, -v_i) \] \text{ and } \\
\[ A[t, s, w, r]' = Am = 0 \]

Note that m is defined up a scale factor!

\[ A = UDV' \] and choose m as column of V corresponding to the smallest singular value

\[ ||t|| = |\gamma| \] gives scale factor for solution
\[ ||w|| = |\gamma| \alpha \]

We now know \( R_x \) and \( R_y \) up to a sign and \( \gamma \).
\( R_z = R_x \times R_y \)

We will probably use another SVD to orthogonalize this system \( (R = UDV'; \text{ set } D \text{ to } I \text{ and multiply}) \).
Last Details

• We still need to compute the correct sign.
  – note that the denominator of the original equations must be positive (points must be in front of the cameras)
  – Thus, the numerator and the projection must disagree in sign.
  – We know everything in numerator and we know the projection, hence we can determine the sign.

• We still need to compute \( T_z \) and \( f_x \)
  – we can formulate this as a least squares problem on those two values using the first equation.

\[
\bar{u} = -f_x \frac{R_x p + T_x}{R_z p + T_z} \rightarrow \\
\bar{u}(R_z p + T_z) = -f_x(R_x p + T_x) \\
f_x(R_x p + T_x) + \bar{u}T_z = -\bar{u}R_z p \\
A(f_x, T_z)' = b \rightarrow (f_x, T_z)' = (A' A)^{-1} A' b
\]

Direct Calibration: The Algorithm

1. Compute image center from orthocenter
2. Compute the A matrix (6.8)
3. Compute solution with SVD
4. Compute gamma and alpha
5. Compute R (and normalize)
6. Compute \( f_x \) and and \( T_z \)
7. If necessary, solve a nonlinear regression to get distortion parameters
Indirect Calibration: The Basic Idea

• We know that we can also just write
  – \( u_h = M p_h \)
  – \( x = (u/w) \) and \( y = (v/w) \), \( u_h = (u,v,1)' \)
  – As before, we can multiply through (after plugging in for \( u,v, \) and \( w \))

• Once again, we can write
  – \( A m = 0 \)

• Once again, we use an SVD to compute \( m \) up to a scale factor.

Getting The Camera Parameters

\[
M = \begin{bmatrix}
-f_x R_x + o_x R_z & -f_x T_x + o_x T_z \\
-f_y R_y + o_y R_z & -f_y T_y + o_y T_z \\
R_z & T_z
\end{bmatrix}
\]

We’ll write

\[
M = \begin{bmatrix}
q_1 \\
q_2 \\
q_3 \\
q_4'
\end{bmatrix}
\]
Getting The Camera Parameters

\[
M = \begin{bmatrix}
-f_x R_x + o_x R_z & -f_x T_x + o_y T_z \\
-f_y R_y + o_y R_z & -f_y T_y + o_y T_z \\
R_z & T_z
\end{bmatrix}
\]

We’ll write

\[
M = \begin{bmatrix}
q_1 \\
q_2 \\
q_3 \\
q_4'
\end{bmatrix}
\]

FIRST:

\[|q_3| \text{ is scale up to sign; divide by this value}\]

\[M_{3,4} \text{ is } T_z \text{ up to sign, but } T_z \text{ must be positive; if not divide } M \text{ by } -1\]

THEN:

\[
R_y = (q_2 - o_y R_z)/f_y
\]

\[
R_x = R_y \times R_z
\]

\[
T_x = -(q_{4,1} - o_x T_z)/f_x
\]

\[
T_y = -(q_{4,2} - o_y T_z)/f_y
\]

Finally, use SVD to orthogonalize the rotation,

Self-Calibration

- Calculate the intrinsic parameters solely from point correspondences from multiple images.
- Static scene and intrinsics are assumed.
- No expensive apparatus.
- Highly flexible but not well-established.
- Projective Geometry – image of the absolute conic.
Model Examples: Points on a Plane

- Normal vector \( n = (n_x, n_y, n_z, 0)' \); point \( P = (p_x, p_y, p_z, 1) \)
  - plane equation: \( n \cdot P = d \)
  - w/o loss of generality, assume \( n_z \neq 0 \)
  - Thus, \( p_z = a \cdot p_x + b \cdot p_y + c \); let \( B = (a, b, 0, c) \)
  - Define \( P' = (p_x, p_y, 0, 1) \)
  - \( P = P' + (0, 0, B \cdot P', 0) \)

- Affine: \( u = A \cdot P, A \) a 3 by 4 matrix
  - \( u = A_{1,2,4} \cdot P' + A_3 \cdot B \cdot P' = A_{3 \times 3} \cdot P_{3 \times 1} \)
  - Note that we can now "reproject" the points \( u \) and group the projections --- in short projection of projections stays within the affine group

- Projective \( p = M \cdot P, M \) a 4 by 3 matrix
  - \( p = M_{1,2,4} \cdot P' + M_3 \cdot B \cdot P' = M \cdot P_{3 \times 1} \)
  - Note that we can now "reproject" the points \( p \) and group the resulting matrices --- in short projections of projections stays within the projective group

Multi-Plane Calibration

- Hybrid method: Photogrammetric and Self-Calibration.
- Uses a planar pattern imaged multiple times (inexpensive).
- Used widely in practice and there are many implementations.
- Based on a group of projective transformations called homographies.

- \( m \) be a 2d point \([u \ v \ 1]'\) and \( M \) be a 3d point \([x \ y \ z \ 1]'\).

- Projection is

\[
\tilde{s}m = A[R \ T] \tilde{M}
\]
Review: Projection Model

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & f
\end{pmatrix} \begin{pmatrix}
X \\
Y \\
Z \\
T
\end{pmatrix}
\]

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix} = \begin{pmatrix}
f & 0 & 0 & 0 \\
0 & f & 0 & 0 \\
0 & 0 & 1 & 0
\end{pmatrix} \begin{pmatrix}
X \\
Y \\
Z \\
T
\end{pmatrix}
\]

\[
\begin{pmatrix}
U \\
V \\
W
\end{pmatrix}_{pix} = \begin{pmatrix}
s_u & 0 & o_u \\
0 & s_v & o_v \\
0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
U \\
V \\
W
\end{pmatrix}_{mm} = Ap
\]

10/15/04

CS 441, Copyright G.D. Hager
Result

• We know that

\[ \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} = sA \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \]

• From one homography, how many constraints on the intrinsic parameters can we obtain?
  – Extrinsic have 6 degrees of freedom.
  – The homography supplies 8 values.
  – Thus, we should be able to obtain 2 constraints per homography.

• Use the constraints on the rotation matrix columns…

Planar Homographies

• First Fundamental Theorem of Projective Geometry:
  – There exists a unique homography that performs a change of basis between two projective spaces of the same dimension.

\[ s\tilde{m} = H\tilde{M} \]

  – Notice that the homography is defined up to scale (s).

• In \( P(2) \), we have
  – \( p' = Hp \) for points \( p \)
  – \( u' = H^t u \) for lines \( u \)

• Note to define the homography, we need three basis vectors *plus* the unit point!
Planar Homographies

• First Fundamental Theorem of Projective Geometry:
  – There exists a unique homography that performs a change of basis between two projective spaces of the same dimension.
  
  \[
  s[u \ v \ 1]^T = A[r_1 \ r_2 \ r_3 \ t][X \ Y \ Z \ 1]^T \\
  s[u \ v \ 1]^T = A[r_1 \ r_2 \ r_3 \ t][X \ Y \ 0 \ 1]^T \\
  s[u \ v \ 1]^T = A[r_1 \ r_2 \ t][X \ Y \ 1]^T \\
  s[u \ v \ 1]^T = H[X \ Y \ 1]^T \\
  \]
  
  – Projection Becomes
  \[
  s\tilde{m} = H\tilde{M} \\
  \]
  
  – Notice that the homography is defined up to scale (s).

Estimating A Homography

• Here is what looks like a reasonable recipe for computing homographies:
  – Planar pts \((x_1; y_1; 1, x_2; y_2; 1, ..., x_n; y_n; 1) = X\)
  – Corresponding pts \((u_1; v_1; 1, u_2; v_2; 1, ..., u_n; v_n; 1) = U\)
  – \(U = HX\)
  – \(U X' (X X')^{-1} = H\)
  
  • The problem is that \(X\) will not be full rank (why?). So we’ll have to work a little harder ...
Computing Intrinsics

• Rotation Matrix is orthogonal:

\[ r_i^T r_j = 0 \]
\[ r_i^T r_i = r_j^T r_j \]

• Write the homography in terms of its columns:

\[ h_1 = sA r_1 \]
\[ h_2 = sA r_2 \]
\[ h_3 = sA t \]

Computing Intrinsics

• Derive the two constraints:

\[ h_1 = sA r_1 \]
\[ \frac{1}{s} A^{-1} h_1 = r_1 \]
\[ \frac{1}{s} A^{-1} h_2 = r_2 \]
\[ r_1^T r_2 = 0 \]
\[ h_1^T A^{-T} A^{-1} h_2 = 0 \]
\[ r_1^T r_1 = r_2^T r_2 \]
\[ h_1^T A^{-T} A^{-1} h_1 = h_2^T A^{-T} A^{-1} h_2 \]
Closed-Form Solution

Let \( B = A^{-T}A^{-1} = \)

\[
\begin{bmatrix}
\frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma}{\alpha^2\beta} \\
-\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma^2}{\alpha^2\beta^2} \\
\frac{\gamma^2}{\alpha^2\beta^2} - \frac{\gamma^2}{\alpha^2\beta^2} - \frac{\gamma}{\alpha^2\beta} & -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma}{\alpha^2\beta}
\end{bmatrix}
\]

• Notice \( B \) is symmetric, 6 parameters can be written as a vector \( b \).
• From the two constraints, we have \( h_i^T B h_j = v_{ij}^T \)

\[
\begin{bmatrix}
v^T_{ij} \\
(v_{11} - v_{22})^T
\end{bmatrix} b = 0;
\]

• Stack up \( n \) of these for \( n \) images and build a \( 2n*6 \) system.
• Solve with SVD (yet again).
• Extrinsic “fall-out” of the result easily.

Non-linear Refinement

• Closed-form solution minimized algebraic distance.
• Since full-perspective is a non-linear model
  – Can include distortion parameters (radial, tangential)
  – Use maximum likelihood inference for our estimated parameters.

\[
\sum_{i=1}^{n} \sum_{j=1}^{m} \left| m_{ij} - \hat{m}(A, R_k, T_k, M_j) \right|^2
\]
Multi-Plane Approach In Action

• …if we can get matlab to work…

Calibration Summary

• Two groups of parameters:
  – internal (intrinsic) and external (extrinsic)
• Many methods
  – direct and indirect, flexible/robust
• The form of the equations that arise here and the way they are solved is common in vision:
  – bilinear forms
  – $Ax = 0$
  – Orthogonality constraints in rotations

• Most modern systems use the method of multiple planes (matlab demo)
  – more difficult optimization over a large # of parameters
  – more convenient for the user
An Example Using Homographies

- Image rectification is the computation of an image as seen by a rotated camera
  - The computation of the planar reprojection is a homography
  - we’ll show later that depth doesn’t matter when rotating; for now we’ll just use intuition

![](image.png)

Rectification Using Homographies

- Pick a rotation matrix $R$ from old to new image

- Consider all points in the image *you want to compute*; then
  - construct pixel coordinates $x = (u, v, 1)$
  - $K$ maps unit focal length metric coordinates to pixel (normalized camera)
  - $x' = KR^tK^{-1}x \quad \diamond \quad x' = Hx$

- Sample a point $x'$ in the original image for each point $x$ in the new.
Bilinear Interpolation

- A minor detail --- new value $x' = (u',v',1)$ may not be integer

- let $u' = i + f_u$ and $v' = j+f_v$

- New image value $b = (1-f_u)((1-f_v)I(j,i) + f_v I(j+1,i)) + f_u((1-f_v)I(j,i+1) + f_v I(j+1,i+1))$

Rectification: Basic Algorithm

1. Create a mesh of pixel coordinates for the rectified image
2. Turn the mesh into a list of homogeneous points
3. Project *backwards* through the intrinsic parameters to get unit focal length values
4. Rotate these values back to the current camera coordinate system.
5. Project them *forward* through the intrinsic parameters to get pixel coordinates again.
   - Note equivalently this is the homography $KR'R'$ where $K$ is the intrinsic parameter matrix
6. Sample at these points to populate the rectified image
   - typically use bilinear interpolation in the sampling
Rectification Results

.2 rad
.4 rad
.6 rad

“Homework” Problems

- Derive the relationship between the Plucker coordinates of a line in space and its projection in Plucker coordinates

- Show that the projection of parallel lines meet at a point (and show how to solve for the point)

- Given two sets of points that define two projective bases, show how to solve for the homography that relates them.

- Describe a simple algorithm for calibrating an affine camera given known ground truth points and their observation --- how many points do you need?
Two-Camera Geometry

PlPr
TPr = R(Pl – T)

(Pl – T) · (T x Pl) = 0
Pr^t R (T x Pl) = 0
Pr^t E Pl = 0

where E = R sk(T)

sk(T) =

\[
\begin{pmatrix}
0 & -T_z & T_y \\
T_z & 0 & -T_x \\
-T_y & T_x & 0
\end{pmatrix}
\]

The matrix E is called the essential matrix and completely describes the epipolar geometry of the stereo pair.
Fundamental Matrix Derivation

Note that $E$ is invariant to the scale of the points, therefore we also have

$$p_r^t E \ p_l = 0$$

where $p$ denotes the (metric) image projection of $P$

Now if $K$ denotes the internal calibration, converting from metric to pixel coordinates, we have further that

$$r_r^t K^t E \ K^{-1} r_l = r_r^t F \ r_l = 0$$

where $r$ denotes the pixel coordinates of $p$. $F$ is called the fundamental matrix.