Projective Geometry

Lecture slides by Steve Seitz (mostly)
Lecture presented by Rick Szeliski

Final project ideas
Discussion by Steve Seitz and Rick Szeliski

Projective geometry

Uses of projective geometry

- Drawing
- Measurements
- Mathematics for projection
- Undistorting images
- Focus of expansion
- Camera pose estimation, match move
- Object recognition

Readings

Applications of projective geometry

Reconstructions by Criminisi et al.

Image rectification

To unwarp (rectify) an image
- solve for homography \( \mathbf{H} \) given \( \mathbf{p} \) and \( \mathbf{p}' \)
- solve equations of the form: \( w \mathbf{p}' = \mathbf{H} \mathbf{p} \)
  - linear in unknowns: \( w \) and coefficients of \( \mathbf{H} \)
  - \( \mathbf{H} \) is defined up to an arbitrary scale factor
  - how many points are necessary to solve for \( \mathbf{H} \)?

Approach: unwarp then measure

What kind of warp is this?

Measurements on planes

Image rectification

To unwarp (rectify) an image

\[
\begin{bmatrix}
\frac{x'}{1} \\
\frac{y'}{1}
\end{bmatrix} =
\begin{bmatrix}
\frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}} \\
\frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}
\end{bmatrix}
\begin{bmatrix}
x_i \\
y_i
\end{bmatrix}
\]

\[
x'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{00}x_i + h_{01}y_i + h_{02}
y'_i(h_{20}x_i + h_{21}y_i + h_{22}) = h_{10}x_i + h_{11}y_i + h_{12}
\]

\[
\begin{bmatrix}
x_i & y_i & 1 & 0 & 0 & -x'_i & y'_i & -1 \\
0 & 0 & 0 & x_i & y_i & -x'_i & y'_i & -1
\end{bmatrix}
\begin{bmatrix}
h_{00} \\
h_{01} \\
h_{02} \\
h_{10} \\
h_{11} \\
h_{12} \\
h_{20} \\
h_{21} \\
h_{22}
\end{bmatrix} =
\begin{bmatrix}
0 \\
0
\end{bmatrix}
\]
Solving for homographies

\[
\begin{bmatrix}
 x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 & x'_2 & -x'_3 \\
 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 & y'_2 & -y'_3 \\
 x_n & y_n & 1 & 0 & 0 & 0 & -x'_n & x'_y & -x'_3 \\
 0 & 0 & 0 & x_n & y_n & 1 & -y'_n & y'_2 & -y'_3 \\
\end{bmatrix}
\begin{bmatrix}
 h_{00} \\
 h_{01} \\
 h_{02} \\
 h_{10} \\
 h_{11} \\
 h_{12} \\
 h_{20} \\
 h_{21} \\
 h_{22} \\
\end{bmatrix} = \begin{bmatrix}
 0 \\
 0 \\
 0 \\
 0 \\
\end{bmatrix}
\]

\(A \in 2n \times 9, \ h \in \mathbb{R}^9, \ \mathbf{0} \in 2n\)

Defines a least squares problem: minimize \(\|Ah - \mathbf{0}\|^2\)
- Since \(h\) is only defined up to scale, solve for unit vector \(\hat{h}\)
- Solution: \(\hat{h} = \text{eigenvector of } A^T A \text{ with smallest eigenvalue}\)
- Works with 4 or more points

---

The projective plane

Why do we need homogeneous coordinates?
- represent points at infinity, homographies, perspective projection, multi-view relationships

What is the geometric intuition?
- a point in the image is a ray in projective space

\[\begin{array}{c}
\text{(sx, sy, s)} \\
\text{image plane} \\
\end{array}\]

- Each point \((x, y)\) on the plane is represented by a ray \((sx, sy, s)\)
  - all points on the ray are equivalent: \((x, y, 1) \equiv (sx, sy, s)\)

---

Projective lines

What does a line in the image correspond to in projective space?

- A line is a plane of rays through origin
  - all rays \((x, y, z)\) satisfying: \(ax + by + cz = 0\)

  in vector notation: \(0 = [a, b, c] \begin{bmatrix} x \\ y \\ z \end{bmatrix}\)

- A line is also represented as a homogeneous 3-vector \(l\)

---

Point and line duality

- A line \(l\) is a homogeneous 3-vector
  - It is \(\perp\) to every point (ray) \(p\) on the line: \(l \cdot p = 0\)

What is the line \(l\) spanned by rays \(p_1\) and \(p_2\)?
- \(l\) is \(\perp\) to \(p_1\) and \(p_2\) \(\Rightarrow\) \(l = p_1 \times p_2\)
- \(l\) is the plane normal

What is the intersection of two lines \(l_1\) and \(l_2\)?
- \(p\) is \(\perp\) to \(l_1\) and \(l_2\) \(\Rightarrow\) \(p = l_1 \times l_2\)

Points and lines are dual in projective space
- given any formula, can switch the meanings of points and lines to get another formula
Ideal points and lines

Ideal point ("point at infinity")
- \( p \cong (x, y, 0) \) – parallel to image plane
- It has infinite image coordinates

Ideal line
- \( l \cong (a, b, 0) \) – parallel to image plane
- Corresponds to a line in the image (finite coordinates)
  - goes through image origin (principle point)

Homographies of points and lines

Computed by 3x3 matrix multiplication
- To transform a point: \( p' = Hp \)
- To transform a line: \( lp = 0 \rightarrow l'p' = 0 \)
  - \( 0 = lp = lH^{-1}Hp = lH^{-1}p' \Rightarrow l' = lH^{-1} \)
  - lines are transformed by postmultiplication of \( H^{-1} \)

3D projective geometry

These concepts generalize naturally to 3D
- Homogeneous coordinates
  - Projective 3D points have four coords: \( P = (X,Y,Z,W) \)
- Duality
  - A plane \( N \) is also represented by a 4-vector
  - Points and planes are dual in 3D: \( N P = 0 \)
- Projective transformations
  - Represented by 4x4 matrices \( T \): \( P' = TP \), \( N' = N T^{-1} \)

3D to 2D: “perspective” projection

Matrix Projection: \[
\begin{bmatrix}
wx \\
wy \\
wz \\
w
\end{bmatrix} \rightarrow \begin{bmatrix}X \\Y \\Z \\1 \end{bmatrix} \text{ or } \mathbf{P} \]

What is not preserved under perspective projection?

What IS preserved?
Vanishing points

- projection of a point at infinity

Properties

- Any two parallel lines have the same vanishing point \( v \)
- The ray from \( C \) through \( v \) is parallel to the lines
- An image may have more than one vanishing point
  - in fact every pixel is a potential vanishing point

Vanishing lines

Multiple Vanishing Points

- Any set of parallel lines on the plane define a vanishing point
- The union of all of vanishing points from lines on the same plane is the vanishing line
  - For the ground plane, this is called the horizon

Multiple Vanishing Points

- Different planes define different vanishing lines
Computing vanishing points

\[ P = P_0 + tD \]

Properties
- \( P_\infty \) is a point at infinity, \( v \) is its projection
- They depend only on line direction
- Parallel lines \( P_0 + tD, P_1 + tD \) intersect at \( P_\infty \)

Computing the horizon

Properties
- \( l \) is intersection of horizontal plane through \( C \) with image plane
- Compute \( l \) from two sets of parallel lines on ground plane
- All points at same height as \( C \) project to \( l \)
  - points higher than \( C \) project above \( l \)
- Provides way of comparing height of objects in the scene
Fun with vanishing points

Perspective cues
Comparing heights

What is the height of the camera?

Measuring height

Measuring height without a ruler

Comparing heights

Vanishing Point

Measuring height

What is the height of the camera?

Computing vanishing points (from lines)

Intersect $p_1q_1$ with $p_2q_2$

$$v = (p_1 \times q_1) \times (p_2 \times q_2)$$

Least squares version

- Better to use more than two lines and compute the "closest" point of intersection
- See notes by Bob Collins for one good way of doing this:

Computing vanishing points (from lines)

Intersect $p_1q_1$ with $p_2q_2$

$$v = (p_1 \times q_1) \times (p_2 \times q_2)$$

Least squares version

- Better to use more than two lines and compute the "closest" point of intersection
- See notes by Bob Collins for one good way of doing this:

Measuring height without a ruler

Compute $Z$ from image measurements

- Need more than vanishing points to do this
The cross ratio

A Projective Invariant

- Something that does not change under projective transformations (including perspective projection)

The cross-ratio of 4 collinear points

\[ \frac{P_3 - P_1}{P_3 - P_2} \cdot \frac{P_4 - P_2}{P_4 - P_1} \]

Can permute the point ordering

\[ \frac{P_1 - P_3}{P_1 - P_2} \cdot \frac{P_4 - P_2}{P_4 - P_1} \]

- 4! = 24 different orders (but only 6 distinct values)

This is the fundamental invariant of projective geometry

Measuring height

\[ \frac{||T-B||}{||R-B||} = \frac{H}{R} \]

Scene cross ratio

\[ \frac{||t-b||}{||r-b||} = \frac{H}{R} \]

Image cross ratio

ground plane

scene points represented as \( P = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \)

image points as \( p = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \)

Measuring height

vanishing line (horizon)

\[ v \cong (b \times t_0) \times (v_x \times v_y) \]

What if the point on the ground plane \( b_0 \) is not known?

- Here the guy is standing on the box, height of box is known
- Use one side of the box to help find \( b_0 \) as shown above
Computing \((X,Y,Z)\) coordinates

Camera calibration

Goal: estimate the camera parameters
- Version 1: solve for projection matrix
  \[
  x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \\ \end{bmatrix} = \Pi X
  \]

- Version 2: solve for camera parameters separately
  - intrinsics (focal length, principle point, pixel size)
  - extrinsics (rotation angles, translation)
  - radial distortion

Vanishing points and projection matrix

\[
\Pi = \begin{bmatrix}
\pi_1 & \pi_2 & \pi_3 & \pi_4 \\
\pi_5 & \pi_6 & \pi_7 & \pi_8 \\
\pi_9 & \pi_{10} & \pi_{11} & \pi_{12} \\
\pi_{13} & \pi_{14} & \pi_{15} & \pi_{16} \\
\end{bmatrix} = \begin{bmatrix}
\pi_1 & \pi_2 & \pi_3 & \pi_4 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

- \(\pi_1 = \Pi [1 \ 0 \ 0 \ 0]^T = v_x\) (X vanishing point)
- similarly, \(\pi_2 = v_y, \pi_3 = v_z\)
- \(\pi_4 = \Pi [0 \ 0 \ 0 \ 1]^T = \) projection of world origin

\[
\Pi = \begin{bmatrix}
v_x & v_y & v_z & 0 \\
\end{bmatrix}
\]

Not So Fast! We only know \(v\)'s and \(o\) up to a scale factor

\[
\Pi = \begin{bmatrix}
a v_x & b v_y & c v_z & d o \\
\end{bmatrix}
\]

- Need a bit more work to get these scale factors…
Finding the scale factors…

Let's assume that the camera is reasonable

- Square pixels
- Image plane parallel to sensor plane
- Principal point in the center of the image

\[
\Pi = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\frac{r_1}{f} & \frac{r_2}{f} & \frac{r_3}{f} & t_1' \\
\frac{r_2}{f} & \frac{r_3}{f} & t_2' \\
r_3 & t_3' \\
\end{bmatrix}
= \begin{bmatrix}
a & v_x & b & v_y & c & v_z & d & \dot{d}
\end{bmatrix}
\]

Solving for f

\[
\begin{bmatrix}
v_{x1} & v_{y1} & v_{z1} & 0 \\
v_{x2} & v_{y2} & v_{z2} & 0 \\
v_{x3} & v_{y3} & v_{z3} & 0
\end{bmatrix}
= \begin{bmatrix}
r_{x1}/f & r_{x2}/f & r_{x3}/f & t_1' \\
r_{y1}/f & r_{y2}/f & r_{y3}/f & t_2' \\
r_{z1}/f & r_{z2}/f & r_{z3}/f & t_3'
\end{bmatrix}
\]

Orthogonal vectors

\[
\begin{bmatrix}
v_{x1} & v_{y1} & v_{z1} \\
v_{x2} & v_{y2} & v_{z2} \\
v_{x3} & v_{y3} & v_{z3}
\end{bmatrix}
= \begin{bmatrix}
r_{x1}/a & r_{y1}/a & r_{z1}/a \\
r_{x2}/a & r_{y2}/a & r_{z2}/a \\
r_{x3}/a & r_{y3}/a & r_{z3}/a
\end{bmatrix}
\]

Orthogonal vectors

\[
\begin{bmatrix}
v_{x1}v_{y1} + v_{x2}v_{y2} + v_{x3}v_{z3} = 0
\end{bmatrix}
\]

\[
f = \sqrt{\frac{v_{x1}v_{y1} + v_{x2}v_{y2}}{-v_{x3}v_{z3}}}
\]

Solving for d

\[
\begin{bmatrix}
w & w & w & w
\end{bmatrix}
= \begin{bmatrix}
r_{x1} & r_{x2} & r_{x3} & t_1'/d \\
r_{y1} & r_{y2} & r_{y3} & t_2'/d \\
r_{z1} & r_{z2} & r_{z3} & t_3'/d
\end{bmatrix}
\]

Suppose we have one reference height H

- E.g., we know that (0, 0, H) gets mapped to (u, v)

\[
u = \frac{r_{x3}H + t_1'/d}{r_{x3}H + t_3'/d}
\]

\[
d = \frac{t_1' - u' t_1'}{u_r r_{33} H - r_{13} H}
\]

Finally, we can solve for t

\[
\begin{bmatrix}
t_1 \\
t_2 \\
t_3
\end{bmatrix}
= \begin{bmatrix}
r_{x1} & r_{x2} & r_{x3} & t_1' \\
r_{y1} & r_{y2} & r_{y3} & t_2' \\
r_{z1} & r_{z2} & r_{z3} & t_3'
\end{bmatrix}
\]
Calibration using a reference object

Place a known object in the scene

- identify correspondence between image and scene
- compute mapping from scene to image

Issues

- must know geometry very accurately
- must know 3D->2D correspondence

Chromaglyphs

Courtesy of Bruce Culbertson, HP Labs
http://www.hpl.hp.com/personal/Bruce_Culbertson/ibr98/chromagl.htm

Estimating the projection matrix

Place a known object in the scene

- identify correspondence between image and scene
- compute mapping from scene to image

\[
\begin{bmatrix}
u_i \\ v_i \\ 1
\end{bmatrix} =
\begin{bmatrix}
m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23}
\end{bmatrix}
\begin{bmatrix}
X_i \\ Y_i \\ Z_i \\ 1
\end{bmatrix}
\]

Direct linear calibration

\[
u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}
\]

\[
v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}
\]

\[u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}\]

\[v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}\]
Direct linear calibration

Can solve for \( m_{ij} \) by linear least squares
- use eigenvector trick that we used for homographies

\[
\begin{bmatrix}
X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & -w_1 X_1 & -w_1 Y_1 & -w_1 Z_1 & -u_1 \\
0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -v_1 X_1 & -v_1 Y_1 & -v_1 Z_1 & -v_1 \\
X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & -w_n X_n & -w_n Y_n & -w_n Z_n & -u_n \\
0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -v_n X_n & -v_n Y_n & -v_n Z_n & -v_n \\
\end{bmatrix}
= \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \end{bmatrix}
\]

Advantage:
- Very simple to formulate and solve

Disadvantages:
- Doesn’t tell you the camera parameters
- Doesn’t model radial distortion
- Hard to impose constraints (e.g., known focal length)
- Doesn’t minimize the right error function

For these reasons, **nonlinear methods** are preferred
- Define error function \( E \) between projected 3D points and image positions
  - \( E \) is nonlinear function of intrinsics, extrinsics, radial distortion
- Minimize \( E \) using nonlinear optimization techniques
  - e.g., variants of Newton’s method (e.g., Levenberg Marquardt)

Alternative: multi-plane calibration

Advantage
- Only requires a plane
- Don’t have to know positions/orientations
- Good code available online!

Some Related Techniques

**Image-Based Modeling and Photo Editing**
- Mok et al., SIGGRAPH 2001

**Single View Modeling of Free-Form Scenes**
- Zhang et al., CVPR 2001

**Tour Into The Picture**
- Anjyo et al., SIGGRAPH 1997
- [http://koigakubo.hitachi.co.jp/little/DL_TipE.html](http://koigakubo.hitachi.co.jp/little/DL_TipE.html)