Geometry of a single camera
Our goal: Recovery of 3D structure

Things aren’t always as they appear…

http://en.wikipedia.org/wiki/Ames_room
Single-view ambiguity
Single-view ambiguity
Single-view ambiguity

Rashad Alakbarov shadow sculptures
Anamorphic perspective
Our goal: Recovery of 3D structure

- When certain assumptions hold, we can recover structure from a single view
- In general, we need multi-view geometry

- But first, we need to understand the geometry of a single camera…
Camera calibration

- **Normalized (camera) coordinate system**: camera center is at the origin, the *principal axis* is the z-axis, x and y axes of the image plane are parallel to x and y axes of the world

- **Camera calibration**: figuring out transformation from *world* coordinate system to *image* coordinate system
Review: Pinhole camera model

\[(X, Y, Z) \mapsto \left(\frac{fX}{Z}, \frac{fY}{Z}\right)\]

\[
\begin{pmatrix}
X \\
Y \\
Z \\
1
\end{pmatrix}
\mapsto
\begin{pmatrix}
fX \\
fY \\
Z
\end{pmatrix} =
\begin{bmatrix}
f & 0 & \frac{X}{Z} \\
f & 0 & \frac{Y}{Z} \\
1 & 0 & \frac{1}{Z}
\end{bmatrix}
\begin{pmatrix}
X \\
Y \\
Z \\
1
\end{pmatrix}
\]

\[\lambda x = PX\]
**Principal point**

- **Principal point (p):** point where principal axis intersects the image plane
- Normalized coordinate system: origin of the image is at the principal point
- Image coordinate system: origin is in the corner
Principal point offset

We want the principal point to map to \((p_x, p_y)\) instead of \((0,0)\)

\[
(X, Y, Z) \mapsto (f \frac{X}{Z} + p_x, f \frac{Y}{Z} + p_y)
\]

\[
\begin{pmatrix}
X \\
Y \\
Z \\
1
\end{pmatrix} \mapsto \begin{pmatrix}
\frac{fX}{Z} + Zp_x \\
\frac{fY}{Z} + Zp_y \\
Z
\end{pmatrix} = \begin{bmatrix}
f & p_x & 0 \\
f & p_y & 0 \\
1 & 0 & 0
\end{bmatrix} \begin{pmatrix}
X \\
Y \\
Z \\
1
\end{pmatrix}
\]
Principal point offset

principal point: \( (p_x, p_y) \)

\[
\begin{bmatrix}
  f & p_x & 0 \\
  f & p_y & 0 \\
  1 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  X \\
  Y \\
  Z
\end{bmatrix}
\]
**Principal point offset**

![Diagram](image)

**principal point:** \((p_x, p_y)\)

\[
\begin{bmatrix}
 f & p_x \\
 f & p_y \\
 1 & 1 \\
\end{bmatrix}
\begin{bmatrix}
 1 & 0 \\
 1 & 0 \\
 1 & 0 \\
\end{bmatrix}
\begin{bmatrix}
 X \\
 Y \\
 Z \\
 1 \\
\end{bmatrix}
= \begin{bmatrix}
 f & p_x & 0 \\
 f & p_y & 0 \\
 1 & 0 \\
\end{bmatrix}
\begin{bmatrix}
 X \\
 Y \\
 Z \\
 1 \\
\end{bmatrix}
\]

**calibration matrix**  
**projection matrix**  
\(K, [I \mid 0]\)

\[
P = K[I \mid 0]
\]
Pixel coordinates

Pixel size: \( \frac{1}{m_x} \times \frac{1}{m_y} \)

\( m_x \) pixels per meter in horizontal direction, 
\( m_y \) pixels per meter in vertical direction

\[
K = \begin{bmatrix} m_x & m_y \\ 1 & 1 \end{bmatrix} \begin{bmatrix} f & p_x \\ f & p_y \end{bmatrix} = \begin{bmatrix} \alpha_x & \beta_x \\ \alpha_y & \beta_y \end{bmatrix} \begin{bmatrix} \text{pixels/m} \\ \text{m} \end{bmatrix} \text{ pixels}
\]
Camera rotation and translation

- In general, the camera coordinate frame will be related to the world coordinate frame by a rotation and a translation.

- Conversion from world to camera coordinate system (in non-homogeneous coordinates):

  \[
  \tilde{X}_{\text{cam}} = R(\tilde{X} - \tilde{C})
  \]

  - coords. of point in camera frame
  - coords. of a point in world frame
  - coords. of camera center in world frame
Camera rotation and translation

\[ \tilde{X}_{\text{cam}} = R \left( \tilde{X} - \tilde{C} \right) \]

\[
\begin{pmatrix}
\tilde{X}_{\text{cam}} \\
1
\end{pmatrix} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \tilde{X} \\
1
\end{pmatrix}
\]

3D transformation matrix (4 x 4)
Camera rotation and translation

\[ \tilde{X}_{\text{cam}} = R(\tilde{X} - \tilde{C}) \]

\[ X_{\text{cam}} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \]

3D transformation matrix (4 x 4)
Camera rotation and translation

\[ x = \text{K}[I | 0] \begin{bmatrix} R & -R\hat{C} \\ 0 & 1 \end{bmatrix} X \]

2D transformation matrix (3 x 3)
3D transformation matrix (4 x 4)
Perspective projection matrix (3 x 4)
Camera rotation and translation

\[ x = K [R \mid -R\tilde{C}] X \]
Camera rotation and translation

\[ x = K[R \mid t]X \quad t = -R\hat{C} \]
Camera parameters

\[ P = K[R \ t] \]

- **Intrinsic parameters**
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - **Skew** (*non-rectangular pixels*)
  - **Radial distortion**

\[
K = \begin{bmatrix}
    m_x & f & p_x \\
    m_y & f & p_y \\
    1 & 1 & 1
\end{bmatrix}
= \begin{bmatrix}
    \alpha_x & \beta_x \\
    \alpha_y & \beta_y \\
    1 & 1
\end{bmatrix}
\]
Camera parameters \[ P = K [ R \quad t ] \]

- **Intrinsic parameters**
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*

- **Extrinsic parameters**
  - Rotation and translation relative to world coordinate system

What is the projection of the camera center?

\[ \text{PC} = K \begin{bmatrix} R & - R \hat{C} \end{bmatrix} \begin{bmatrix} \hat{C} \\ 1 \end{bmatrix} = 0 \]

The camera center is the *null space* of the projection matrix!
Camera calibration

\[ \lambda \mathbf{x} = K[R \ t] \mathbf{X} \]

\[
\begin{bmatrix}
\lambda x \\
\lambda y \\
\lambda \\
1
\end{bmatrix}
= \begin{bmatrix}
* & * & * & * & * \\
* & * & * & * & * \\
* & * & * & * & * \\
1 & & & & \\
\end{bmatrix}
\begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]

Source: D. Hoiem
Camera calibration

- Given \( n \) points with known 3D coordinates \( X_i \) and known image projections \( x_i \), estimate the camera parameters.
Camera calibration: Linear method

$$\lambda x_i = PX_i \quad x_i \times PX_i = 0$$

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \times \begin{bmatrix} \textbf{P}_1^T X_i \\ \textbf{P}_2^T X_i \\ \textbf{P}_3^T X_i \end{bmatrix} = 0$$

$$\begin{bmatrix} 0 & -X_i^T & y_iX_i^T \\ X_i^T & 0 & -x_iX_i^T \\ -y_iX_i^T & x_iX_i^T & 0 \end{bmatrix} \begin{bmatrix} \textbf{P}_1 \\ \textbf{P}_2 \\ \textbf{P}_3 \end{bmatrix} = 0$$

Two linearly independent equations
Camera calibration: Linear method

\[
\begin{bmatrix}
0^T & X_1^T & -y_1X_1^T \\
X_1^T & 0^T & -x_1X_1^T \\
\vdots & \vdots & \vdots \\
0^T & X_n^T & -y_nX_n^T \\
X_n^T & 0^T & -x_nX_n^T
\end{bmatrix}
\begin{bmatrix}
P_1 \\
P_2 \\
P_3
\end{bmatrix} = 0 \quad \text{Ap} = 0
\]

- P has 11 degrees of freedom
- One 2D/3D correspondence gives us two linearly independent equations
  - 6 correspondences needed for a minimal solution
- Homogeneous least squares: find p minimizing \( \|Ap\|^2 \)
  - Solution given by eigenvector of \( A^TA \) with smallest eigenvalue
Camera calibration: Linear method

\[
\begin{bmatrix}
0^T & X_1^T & -y_1X_1^T \\
X_1^T & 0^T & -x_1X_1^T \\
\cdots & \cdots & \cdots \\
0^T & X_n^T & -y_nX_n^T \\
X_n^T & 0^T & -x_nX_n^T
\end{bmatrix}
\begin{pmatrix}
P_1 \\
P_2 \\
P_3
\end{pmatrix} = 0 \quad \text{Ap} = 0
\]

• Note: for coplanar points that satisfy \( \Pi^TX = 0 \), we will get degenerate solutions \((\Pi,0,0), (0,\Pi,0), \text{ or } (0,0,\Pi)\)
Camera calibration: Linear vs. nonlinear

- Linear calibration is easy to formulate and solve, but it doesn’t directly tell us the camera parameters

\[
\begin{bmatrix}
\lambda x \\
\lambda y \\
\lambda
\end{bmatrix} = \begin{bmatrix}
* & * & * & * \\
* & * & * & * \\
* & * & * & *
\end{bmatrix} \begin{bmatrix}
X \\
Y \\
Z \\
1
\end{bmatrix}
\]

vs.

\[x = K[R \ t] X\]

- In practice, non-linear methods are preferred
  - Write down objective function in terms of intrinsic and extrinsic parameters
  - Define error as sum of squared distances between measured 2D points and estimated projections of 3D points
  - Minimize error using Newton’s method or other non-linear optimization
  - Can model radial distortion and impose constraints such as known focal length and orthogonality
Homography Example

Slide from A. Efros, S. Seitz, D. Hoiem
Problem set-up

\[ x = K \begin{bmatrix} R & t \end{bmatrix} X \]

\[ x' = K' \begin{bmatrix} R' & t' \end{bmatrix} X \]

\( t = t' = 0 \)

\[ x' = Hx \quad \text{where} \quad H = K' R' R^{-1} K^{-1} \]

Typically only \( R \) and \( f \) will change (4 parameters),
but, in general, \( H \) has 8 parameters.
A taste of multi-view geometry: Triangulation

- Given projections of a 3D point in two or more images (with known camera matrices), find the coordinates of the point
Triangulation

• Given projections of a 3D point in two or more images (with known camera matrices), find the coordinates of the point
Triangulation

- We want to intersect the two visual rays corresponding to $x_1$ and $x_2$, but because of noise and numerical errors, they don’t meet exactly.
Triangulation: Geometric approach

- Find shortest segment connecting the two viewing rays and let $X$ be the midpoint of that segment.
Triangulation: Nonlinear approach

Find $X$ that minimizes

$$d^2(x_1, P_1X) + d^2(x_2, P_2X)$$
Triangulation: Linear approach

\[ \lambda_1 \mathbf{x}_1 = \mathbf{P}_1 \mathbf{X} \quad \mathbf{x}_1 \times \mathbf{P}_1 \mathbf{X} = 0 \quad [\mathbf{x}_1 \times \mathbf{P}_1 \mathbf{X} = 0 \]

\[ \lambda_2 \mathbf{x}_2 = \mathbf{P}_2 \mathbf{X} \quad \mathbf{x}_2 \times \mathbf{P}_2 \mathbf{X} = 0 \quad [\mathbf{x}_2 \times \mathbf{P}_2 \mathbf{X} = 0 \]

Cross product as matrix multiplication:

\[
\mathbf{a} \times \mathbf{b} = \begin{bmatrix}
0 & -a_z & a_y \\
 a_z & 0 & -a_x \\
-a_y & a_x & 0
\end{bmatrix} \begin{bmatrix}
b_x \\
b_y \\
b_z
\end{bmatrix} = [\mathbf{a} \times \mathbf{b}]
\]
Triangulation: Linear approach

\[ \lambda_1 x_1 = P_1 X \quad x_1 \times P_1 X = 0 \quad [x_1 \times] P_1 X = 0 \]

\[ \lambda_2 x_2 = P_2 X \quad x_2 \times P_2 X = 0 \quad [x_2 \times] P_2 X = 0 \]

Two independent equations each in terms of three unknown entries of \( X \)
Camera calibration revisited

- What if world coordinates of reference 3D points are not known?
- We can use scene features such as vanishing points
Camera calibration revisited

- What if world coordinates of reference 3D points are not known?
- We can use scene features such as vanishing points
Recall: Homogenous Coordinates

Points

Points at infinity

Lines

Lines passing through 2 points

Intersection of 2 lines

Intersection of 2 parallel lines?
Recall: Homogenous Coordinates

Points

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a} \times] \mathbf{b}$$

Points at infinity

Lines

Lines passing through 2 points

Intersection of 2 lines

Intersection of 2 parallel lines?
Recall: Vanishing points

• All lines having the same direction share the same vanishing point
Computing vanishing points

- \( \mathbf{X}_\infty \) is a point at infinity, \( \mathbf{v} \) is its projection: \( \mathbf{v} = \mathbf{P} \mathbf{X}_\infty \)
- The vanishing point depends only on line direction
- All lines having direction \( \mathbf{d} \) intersect at \( \mathbf{X}_\infty \)
Calibration from vanishing points

Consider a scene with three orthogonal vanishing directions:

Note: $v_1, v_2$ are finite vanishing points and $v_3$ is an infinite vanishing point.
Calibration from vanishing points

• Consider a scene with three orthogonal vanishing directions:

• We can align the world coordinate system with these directions
Calibration from vanishing points

\[
P = \begin{bmatrix}
* & * & * & * \\
* & * & * & * \\
* & * & * & *
\end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \end{bmatrix}
\]

- \( p_1 = P(1,0,0,0)^T \) – the vanishing point in the x direction
- Similarly, \( p_2 \) and \( p_3 \) are the vanishing points in the y and z directions
- \( p_4 = P(0,0,0,1)^T \) – projection of the origin of the world coordinate system
- Problem: we can only know the four columns up to independent scale factors, additional constraints needed to solve for them
Calibration from vanishing points

Let us align the world coordinate system with three orthogonal vanishing directions in the scene:

\[
\begin{align*}
    e_1 &= \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, & e_2 &= \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, & e_3 &= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\end{align*}
\]

\[
\lambda_i v_i = K[R | t] \begin{bmatrix} e_i \\ 0 \end{bmatrix}
\]
Calibration from vanishing points

- Let us align the world coordinate system with three orthogonal vanishing directions in the scene:

\[
\begin{align*}
e_1 &= \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, & e_2 &= \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, & e_3 &= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\end{align*}
\]

\[
\lambda_i v_i = K R e_i
\]

\[
e_i = \lambda_i R^T K^{-1} v_i
\]

- Orthogonality constraint: \( e_i^T e_j = 0 \)

\[
\underbrace{v_i^T K^{-T} RR^T K^{-1}}_{i} v_j = 0
\]

\[
\underbrace{e_i^T}_{i} \underbrace{e_j}_{j}
\]
Calibration from vanishing points

• Let us align the world coordinate system with three orthogonal vanishing directions in the scene:

\[ e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad e_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \]

\[ \lambda_i v_i = KRe_i \]

\[ e_i = \lambda_i R^T K^{-1} v_i \]

• Orthogonality constraint: \[ e_i^T e_j = 0 \]

\[ v_i^T K^{-T} K^{-1} v_j = 0 \]

• Rotation disappears, each pair of vanishing points gives constraint on focal length and principal point
Calibration from vanishing points

- 1 finite vanishing point, 2 infinite vanishing points
- 2 finite vanishing points, 1 infinite vanishing point
- 3 finite vanishing points

Cannot recover focal length, principal point is the third vanishing point

Can solve for focal length, principal point
Rotation from vanishing points

- Constraints on vanishing points: $\lambda_i v_i = KRe_i$
- After solving for the calibration matrix:

$$\lambda_i K^{-1}v_i = Re_i$$

- Notice: $Re_1 = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = r_1$
- Thus, $r_i = \lambda_i K^{-1}v_i$

- Get $\lambda_i$ by using the constraint $||r_i||^2 = 1$. 
Calibration from vanishing points: Summary

- Solve for K (focal length, principal point) using three orthogonal vanishing points
- Get rotation directly from vanishing points once calibration matrix is known

**Advantages**
- No need for calibration chart, 2D-3D correspondences
- Could be completely automatic

**Disadvantages**
- Only applies to certain kinds of scenes
- Inaccuracies in computation of vanishing points
- Problems due to infinite vanishing points
Application: Single View Reconstruction

Piero della Francesca, *Flagellation*, ca. 1455

- Find heights (Hint: estimate horizon)
- Find location on ground
- Find pattern of the ground (Hint: homography)

Application: Single View Reconstruction

• Are the heights of the two groups of people consistent with one another?
  • Measure heights using Christ as reference

Piero della Francesca, *Flagellation*, ca. 1455

Application: 3D modeling from a single image

Measurements on planes

Approach: unwarp then measure
What kind of warp is this?
To unwarp (rectify) an image

- solve for homography $H$ given $p$ and $p'$
- how many points are necessary to solve for $H$?
Image rectification: example

Piero della Francesca, *Flagellation*, ca. 1455
Application: 3D modeling from a single image

Application: Fully automatic modeling


http://dhoiem.cs.illinois.edu/projects/popup/popup_movie_450_250.mp4
Application: Object detection

D. Hoiem, A.A. Efros, and M. Hebert, Putting Objects in Perspective, CVPR 2006
Application: Image editing

Inserting synthetic objects into images:

http://vimeo.com/28962540

K. Karsch and V. Hedau and D. Forsyth and D. Hoiem, Rendering Synthetic Objects into Legacy Photographs, SIGGRAPH Asia 2011
• Given 2D point correspondences between multiple images, compute the camera parameters and the 3D points
• **Structure:** Given *known cameras* and projections of the same 3D point in two or more images, compute the 3D coordinates of that point

• Triangulation!
• **Motion:** Given a set of *known* 3D points seen by a camera, compute the camera parameters
  • Calibration!