CS 231A PS1 Review

CS231A

Computer Vision: From 3D Reconstruction to Recognition

Winter 2018

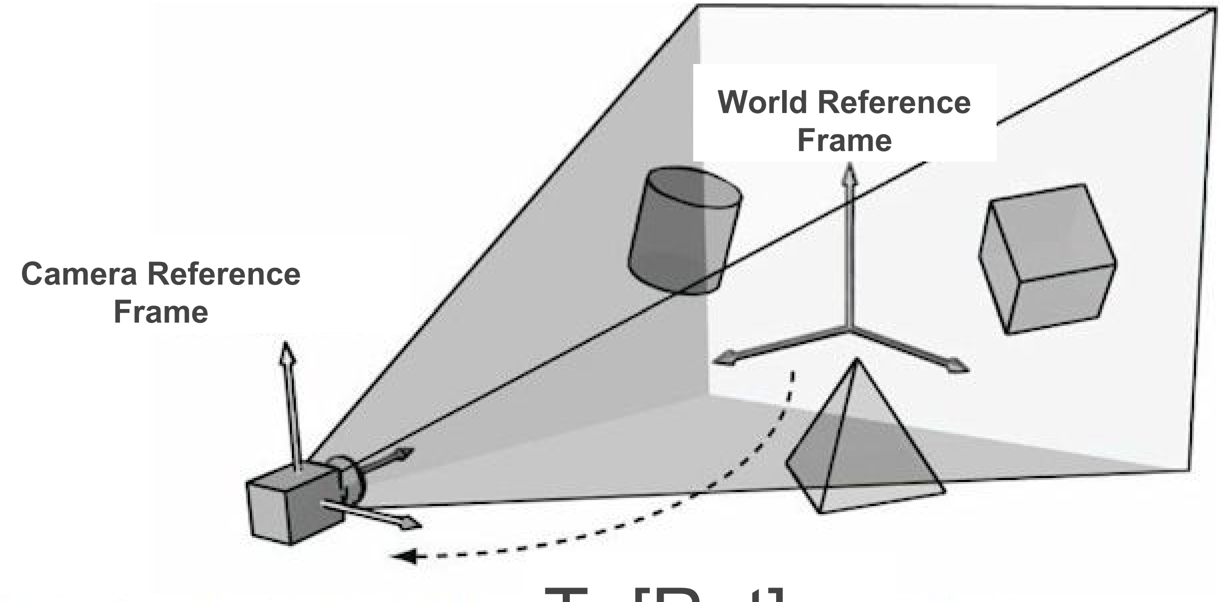
Problem Outline

- Q1: Projective Geometry
- Q2: Affine Camera Calibration
- Q3: Single View Geometry

Problem Outline

- Q1: Projective Geometry
- Q2: Affine Camera Calibration
- Q3: Single View Geometry

P1: Reference Frames



T: [R, t]

Source:http://ycpcs.github.io/cs470-fall2014/

P1: Cross Products

- Lines *k* and *l* are parallel
  - k_1 and k_2 are any two points on *k*
  - l_1 and l_2 are any two points on *l*
  - by definition of parallel lines:

$$\boxed{(k_1 - k_2) \times (l_1 - l_2) = 0}$$

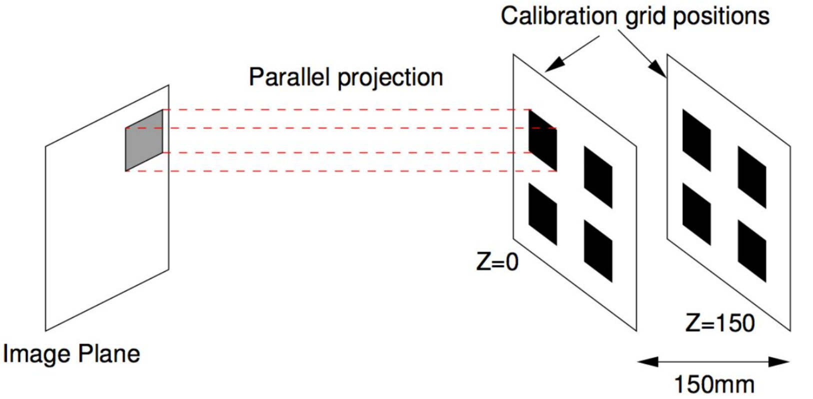
- Given a square *pqrs*,
  - Area = $\boxed{\|(q-p) \times (s-p)\|}$

*Courtesy of last year's slides

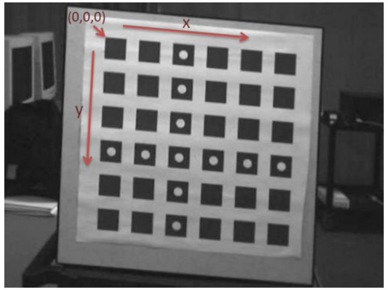
Problem Outline

- Q1: Projective Geometry
- Q2: Affine Camera Calibration
- Q3: Single View Geometry

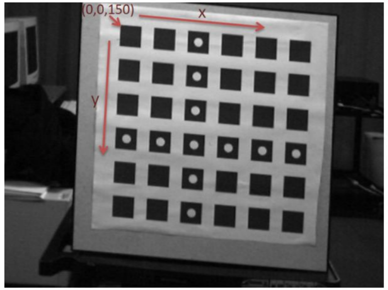
P2: Setup



(a) Image formation in an affine camera. Points are projected via parallel rays onto the image plane



(b) Image of calibration grid at Z=0



(c) Image of calibration grid at Z=150

P2: Perspective Camera Model

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} \mathrm{u_i} \\ \mathrm{v_i} \end{bmatrix} = \begin{bmatrix} \dfrac{\mathbf{m}_1 \, \mathrm{P_i}}{\mathbf{m}_3 \, \mathrm{P_i}} \\ \dfrac{\mathbf{m}_2 \, \mathrm{P_i}}{\mathbf{m}_3 \, \mathrm{P_i}} \end{bmatrix}$$

M **P**

P2: Affine Camera Model

$$\begin{bmatrix} x \\ y \\ \boxed{1} \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ \boxed{0 \quad 0 \quad 0 \quad 1} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- **Linear**
- **8 Unknowns**

P2: Solve for P

$$Ax = b$$

$$x = (A^T A)^{-1} A^T b$$

Hint: numpy.linalg.pinv / numpy.linalg.lstsq

Problem Outline

- Q1: Projective Geometry
- Q2: Affine Camera Calibration
- Q3: Single View Metrology

Vanishing Points

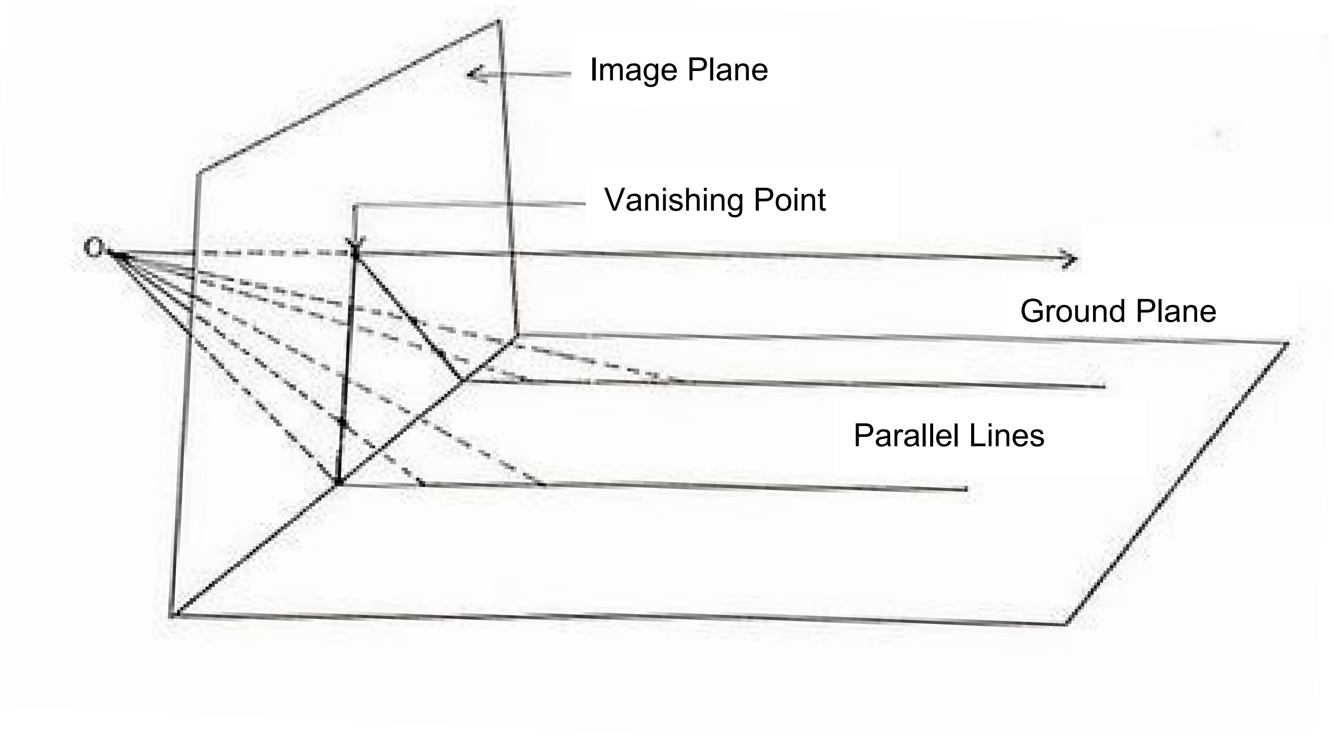
*Courtesy of last year's slides

Vanishing Points

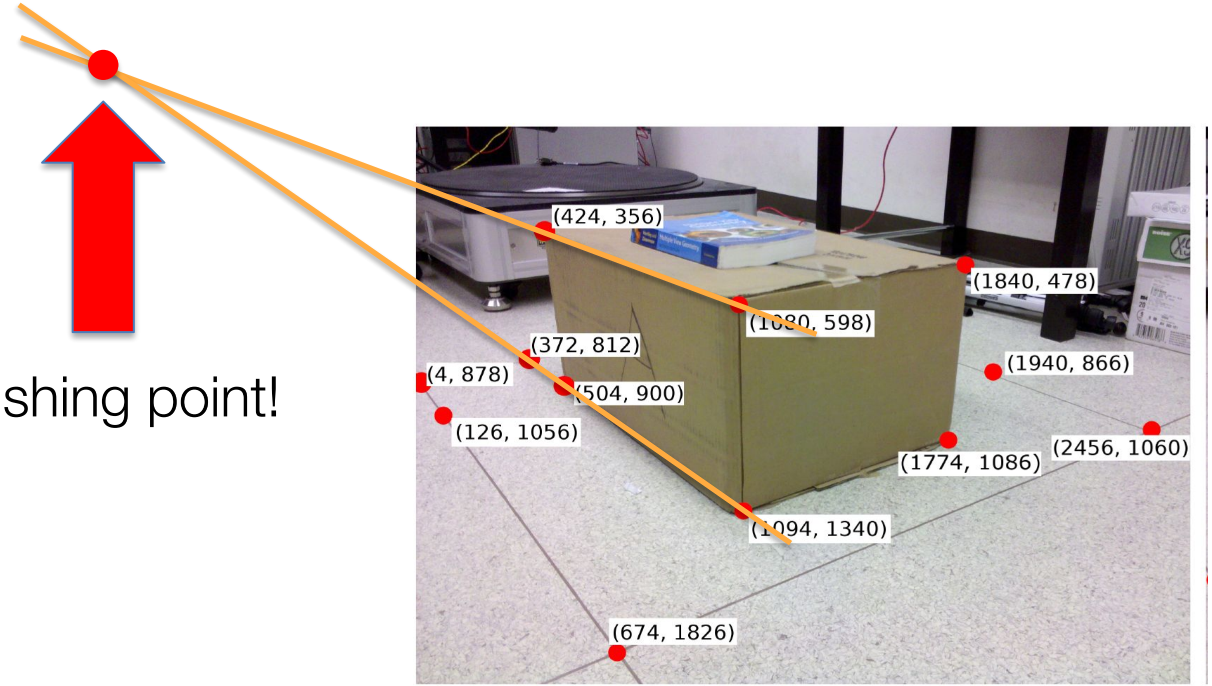
- In 3D space, points at infinity are defined as the intersection of parallel lines, which have direction *d*
- In the image plane, parallel lines meet at the vanishing point *v*.
- With camera intrinsic matrix as *K*, we have

$$v = Kd$$

Vanishing Points



Calculating Vanishing Point



Vanishing point!

(a) Image 1 (1.jpg) with marked pixels

*Courtesy of last year's slides

Calculating Vanishing Point

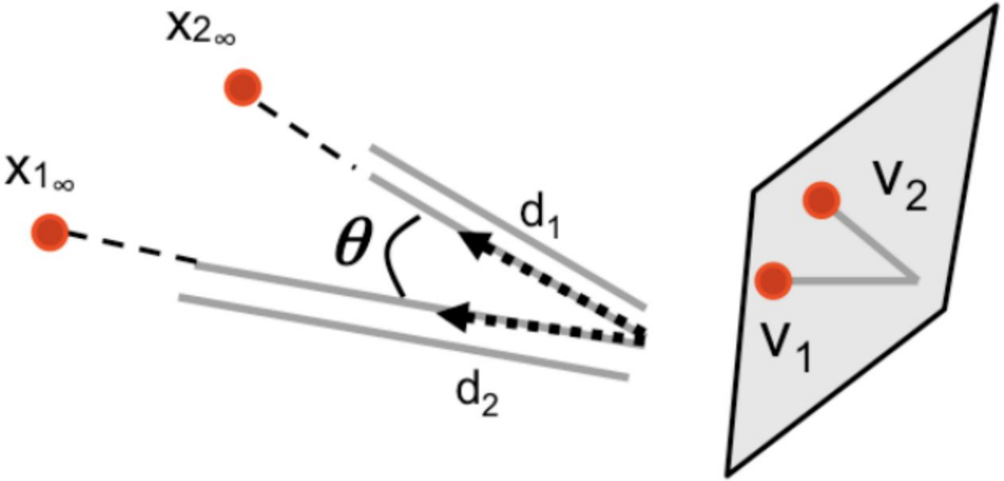
- Points in L_1: (x_1, y_1), (x_2, y_2) $\rightarrow$ $m_1 = (y_2 - y_1)/(x_2 - x_1)$
- Points in L_2: (x_3, y_3), (x_4, y_4) $\rightarrow$ $m_2 = (y_4 - y_3)/(x_4 - x_3)$
- Intersection of L_1 and L_2: Vanishing Point

*Courtesy of last year's slides

Vanishing Points to Compute K

- Course notes “Single View Metrology”
- HZ (2nd edition) Page 223-226

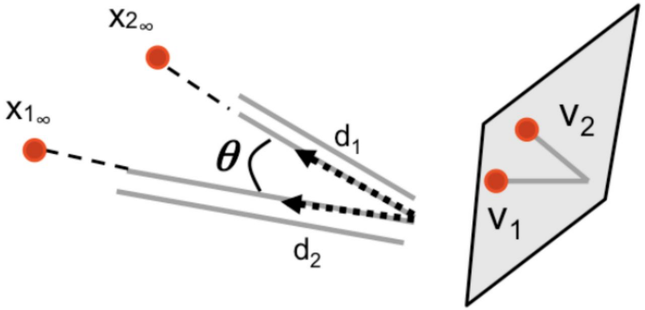
Vanishing Points to Compute K



Vanishing Points to Compute K

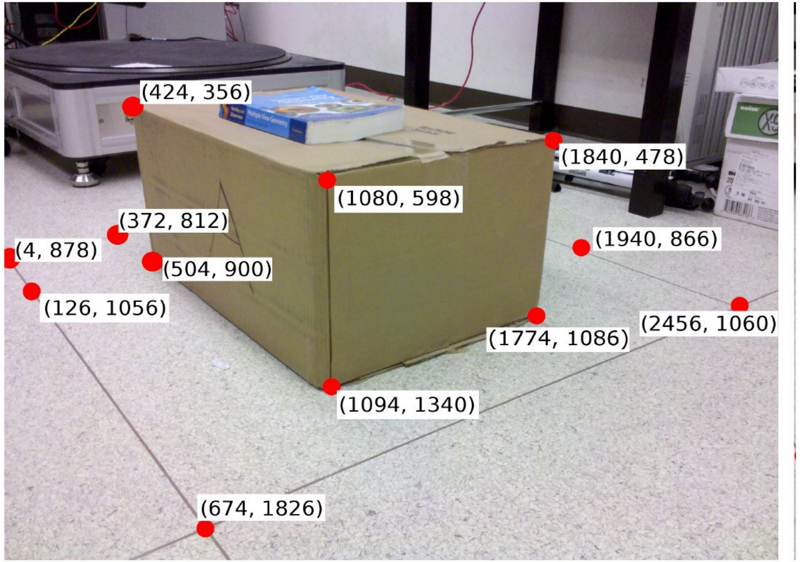
- Define vanishing line d_1 and d_2, vanishing points v_1 and v_2. We have:

$$\cos\theta = \frac{d_1 \cdot d_2}{\|d_1\|\|d_2\|} = \frac{v_1^T \omega v_2}{\sqrt{v_1^T \omega v_1}\sqrt{v_2^T \omega v_2}}$$



, where $\omega = (KK^T)^{-1}$

Vanishing Points to Compute K

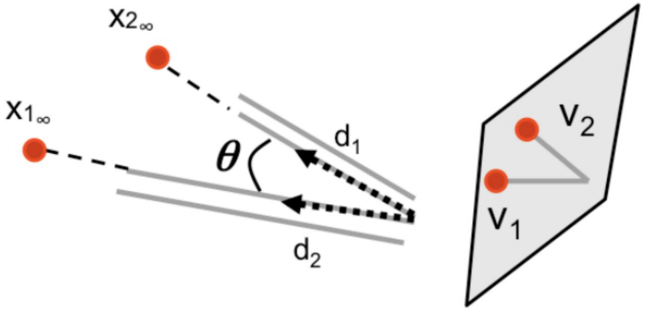


(a) Image 1 (1.jpg) with marked pixels

Vanishing Points to Compute K

- Define vanishing line d_1 and d_2, vanishing points v_1 and v_2. We have:

$$\cos\theta = \frac{d_1 \cdot d_2}{\|d_1\|\|d_2\|} = \frac{v_1^T \omega v_2}{\sqrt{v_1^T \omega v_1}\sqrt{v_2^T \omega v_2}}$$



, where $\omega = (KK^T)^{-1}$

$\boxed{\text{If } d_1 \text{ and } d_2 \text{ are orthogonal, } v_1^T \omega v_2 = 0}$

Vanishing Points to Compute K

- $\omega = (K K^T)^{-1}$
  - Matrix ω is the projective transformation in the image plane of an absolute conic in 3D

$$\omega = \boxed{\begin{bmatrix} \omega_1 & \omega_2 & \omega_4 \\ \omega_2 & \omega_3 & \omega_5 \\ \omega_4 & \omega_5 & \omega_6 \end{bmatrix}}$$

*Courtesy of last year's slides

Vanishing Points to Compute K

- We assume the camera has zero skew and square pixels
  - Zero skew: $\omega_2 = 0$
  - Square pixels: $\omega_1 = \omega_3$

$$\boxed{\omega = \begin{bmatrix} \omega_1 & 0 & \omega_4 \\ 0 & \omega_1 & \omega_5 \\ \omega_4 & \omega_5 & \omega_6 \end{bmatrix}}$$

*Courtesy of last year's slides

Compute Angle Between Planes

- Vanishing lines L_1 and L_2
- $L_1 = v_1 \times v_2$; $L_2 = v_3 \times v_4$
  - v_1 *and* v_2 = vanishing points corresponding to one plane
  - v_3 *and* v_4 for the other plane

$$\boxed{cos\theta = \frac{\mathbf{l}_1^T \omega^* \mathbf{l}_2}{\sqrt{\mathbf{l}_1^T \omega^* \mathbf{l}_1}\sqrt{\mathbf{l}_2^T \omega^* \mathbf{l}_2}}}$$

, where $\omega^* = \omega^{-1} = KK^T$

*Courtesy of last year's slides

Rotation Matrix using Vanishing Points

- Find corresponding vanishing points from both images (v_1, v_2, v_3) and (v_1', v_2', v_3')
- Calculate directions of vanishing points:
  - $v = K\,d \rightarrow$ $\boxed{\mathbf{d} = \dfrac{K^{-1}\,\mathbf{v}}{\left\|K^{-1}\,\mathbf{v}\right\|}}$
- $d_i' = R\,d_i$, where
  - d_i' = direction of the i^{th} vanishing point in second image
  - d_i = direction of the i^{th} vanishing point in first image

*Courtesy of last year's slides