CS231A PS1 Review

Computer Vision: From 3D Reconstruction to Recognition

Spring 2017
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
Cross Products

- If lines $k$ and $l$ are parallel, and $k_1$ and $k_2$ are any 2 points on $k$, and $l_1$ and $l_2$ are any 2 points on $l$, by definition of parallel lines:
  \[(k_1 - k_2) \times (l_1 - l_2) = 0\]

- Given a square $pqrs$,
  \[- \text{Area} = \left\| (q - p) \times (s - p) \right\|\]
Sample Problem

Prove that the angle between intersecting lines in the world reference system is the same as in the camera reference system.
Problem Outline

• Q1: Projective Geometry

• Q2: Affine Camera Calibration

• Q3: Single View Geometry
Least Squares

\[ Ax = b \]

\[ x = (A^TA)^{-1}A^Tb \]

Useful: numpy.linalg.lstsq
Problem Outline

• Q1: Projective Geometry

• Q2: Affine Camera Calibration

• Q3: Single View Geometry
Vanishing Points

*Courtesy of last year’s slides*
Vanishing Points

• Under perspective projection, lines that are parallel in the world frame are no longer parallel in the image frame
  – Exception: Lines parallel to the image plane remain parallel

• In the image plane, parallel lines meet at the vanishing point

*Courtesy of last year’s slides*
Calculating Vanishing Point

(a) Image 1 (1.jpg) with marked pixels
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 $L_2$: $(x_3, y_3), (x_4, y_4) \rightarrow m_2 = (y_4 - y_3)/(x_4 - x_3)$

• Equation of a line: $y = mx + b$
  - $b_1 = y_2 - m_1x_2$; $b_2 = y_4 - m_2x_4$
  - $L_1$: $y = m_1x + b_1$; $L_2$: $y = m_2x + b_2$

• Intersection of $L_1$ and $L_2$: $(x, y)$
  - $m_1x + b_1 = m_2x + b_2$
    - $x = (b_2 - b_1)/(m_1 - m_2)$
    - $y = m_1x + b_1 = m_1[(b_2 - b_1)/(m_1 - m_2)] + b_1$
Vanishing Points to Compute K

- Lecture 4 slides are very useful!
- Only need 3 vanishing points
- \( \omega = (K K^T)^{-1} \)
  - Matrix \( \omega \) is the projective transformation in the image plane of an absolute conic in 3D

\[
\omega = \begin{bmatrix}
\omega_1 & \omega_2 & \omega_4 \\
\omega_2 & \omega_3 & \omega_5 \\
\omega_4 & \omega_5 & \omega_6
\end{bmatrix}
\]
Vanishing Points to Compute $K$

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

$$\omega = \begin{bmatrix} \omega_1 & 0 & \omega_4 \\ 0 & \omega_1 & \omega_5 \\ \omega_4 & \omega_5 & \omega_6 \end{bmatrix}$$
Single view calibration - example

\[ \omega = \begin{bmatrix} \omega_1 & 0 & \omega_4 \\ 0 & \omega_1 & \omega_5 \\ \omega_4 & \omega_5 & \omega_6 \end{bmatrix} \]

- Square pixels \( \Rightarrow \omega_2 = 0 \)
- No skew \( \Rightarrow \omega_1 = \omega_3 \)

\[ \begin{align*}
V_1^T \omega \ V_2 &= 0 \\
V_1^T \omega \ V_3 &= 0 \\
V_2^T \omega \ V_3 &= 0
\end{align*} \]

Once \( \omega \) is calculated, we get \( K \):

\[ \omega = \left( K \ K^T \right)^{-1} \]

(Cholesky factorization; HZ pag 582)
Angle between 2 vanishing points

\[
\cos \theta = \frac{V_1^T \omega V_2}{\sqrt{V_1^T \omega V_1} \sqrt{V_2^T \omega V_2}}
\]

[Eq. 28]

If \( \theta = 90 \) \( \rightarrow \)

\[
V_1^T \omega V_2 = 0
\]

Scalar equation
Compute Angle Between Planes

• Similar to the previous slide!
• 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

$$\cos \theta = \frac{\ell_1^T \omega^{-1} \ell_2}{\sqrt{\ell_1 \omega^{-1} \ell_1} \sqrt{\ell_2 \omega^{-1} \ell_2}}$$
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:
  \[
  v = Kd \Rightarrow d = \frac{K^{-1} v}{\|K^{-1} v\|} \quad \text{where } d = \text{direction of line}
  \]

• \(d_i' = R d_i\) , where
  – \(d_i' = \text{direction of the } i^{th} \text{ vanishing point in second image}\)
  – \(d_i = \text{direction of the } i^{th} \text{ vanishing point in first image}\)