Computer Vision
and
Image Understanding
(Multiview geometry)

Bhabatosh Chanda
chanda@isical.ac.in

Epipolar lines and Fundamental matrix

- From $(P_h{}^{I_2})^T (K_2^{-1})^T E K_1^{-1} P_h{}^{I_1} = 0$

we simplify to $(P_h{}^{I_2})^T F P_h{}^{I_1} = 0$

- The matrix $F = (K_2^{-1})^T E K_1^{-1}$ is *Fundamental matrix*.
- Epipolar lines:

$$l_1 \equiv (P_h{}^{I_2})^T F \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \qquad l_2 \equiv (P_h{}^{I_1})^T F^T \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0$$

Fundamental matrix

- Fundamental matrix: $F = (K_2^{-1})^T E K_1^{-1}$
- Each pair of points (one from camera-1 and other from camera-2) produces one equation:

$$(P_h{}^{I_2})^T F P_h{}^{I_1} = 0$$

- So 8-pairs of correspondence points are sufficient to find fundamental matrix *F* between two cameras.
- This is known as *8-point algorithm*.
- If K_1 and K_2 are known *Essential matrix* E can be estimated from Fundamental matrix as $E = (K_2)^T F K_1$

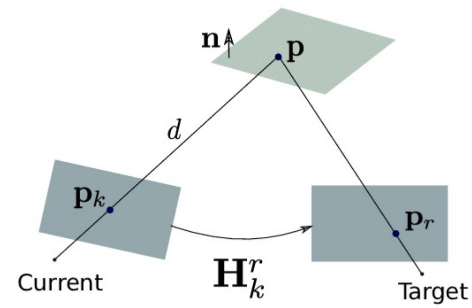
Binocular stereo reconstruction

1. Compute image features.
2. Compute feature descriptors.
3. Find initial matches.
4. Compute fundamental matrix.
5. Refine matches.
6. Estimate essential matrix.
7. Decompose essential matrix.
8. Estimate 3D points.

Binocular stereo reconstruction

1. Compute image features.
2. Compute feature descriptors.
3. Find initial matches.
4. Compute fundamental matrix.
5. Refine matches.
6. Estimate essential matrix.
7. Decompose essential matrix.
8. Estimate 3D points.

Homography



Projection matrix for image formation

- Combining the matrices of

$$\begin{bmatrix} u^c\varphi_x + w^c d_x/f \\ v^c\varphi_y + w^c d_y/f \\ w^c/f \end{bmatrix} = \begin{bmatrix} \varphi_x & 0 & d_x \\ 0 & \varphi_y & d_y \\ 0 & 0 & 1/f \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_u \\ r_{21} & r_{22} & r_{23} & t_v \\ r_{31} & r_{32} & r_{33} & t_w \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$$

- We obtain $\begin{bmatrix} u^c\varphi_x + w^c d_x/f \\ v^c\varphi_y + w^c d_y/f \\ w^c/f \end{bmatrix} = K[R \quad t] \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} = A \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$
- Finally, $\begin{bmatrix} x' \\ y' \\ w^c/f \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$

Formation of image in two cameras

- With respect to camera-1

$$\begin{bmatrix} x_h^{I_1} \\ y_h^{I_1} \\ \lambda_1 \end{bmatrix} = K_1[R_1 \quad t_1] \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$$

- With respect to camera-2

$$\begin{bmatrix} x_h^{I_2} \\ y_h^{I_2} \\ \lambda_2 \end{bmatrix} = K_2[R_2 \quad t_2] \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix}$$

Homography

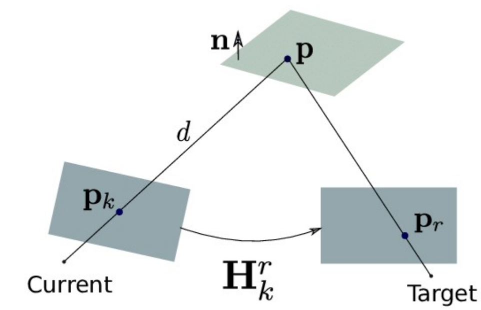
- <u>For mapping from camera-2 to camera-1</u>

$$\begin{bmatrix} x_h^{I_1} \\ y_h^{I_1} \\ \lambda_1 \end{bmatrix} = K_1 [R_1 \quad t_1][R_2 \quad t_2]^{-1} K_2^{-1} \begin{bmatrix} x_h^{I_2} \\ y_h^{I_2} \\ \lambda_2 \end{bmatrix}$$

- <u>OR</u>

$$\begin{bmatrix} x_h^{I_1} \\ y_h^{I_1} \\ \lambda_1 \end{bmatrix} = K_1 H_2^1 K_2^{-1} \begin{bmatrix} x_h^{I_2} \\ y_h^{I_2} \\ \lambda_2 \end{bmatrix}$$

Homography



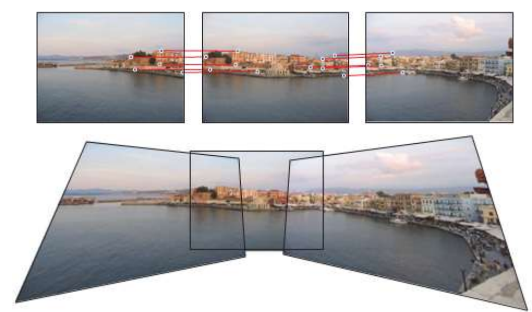
Homography due to 3D plane

- Suppose the points in 3D space are lying on a plane with surface normal $\vec{n}$ and lying at distance d.
- If, with respect to camera-1, camera-2 is
  Rotated by R and translated by t, then

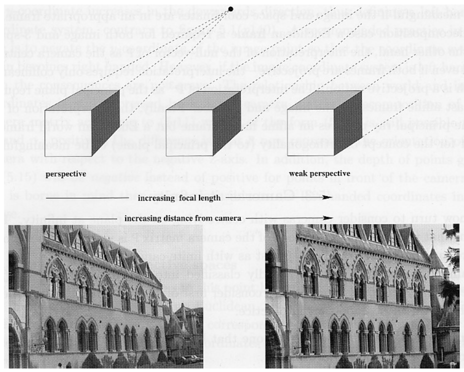
$$H_2^1 = R - \frac{t\vec{n}^T}{d}$$

- *Two important applications are: Image registration (for panoramic view generation) and image rectification (for dense 3D reconstruction).*

Panorama

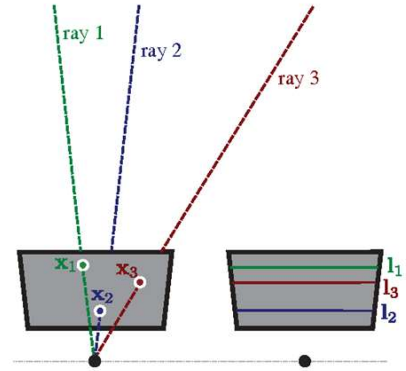


Approximating an affine camera



Source: Hartley & Zisserman

Rectification



Rectification

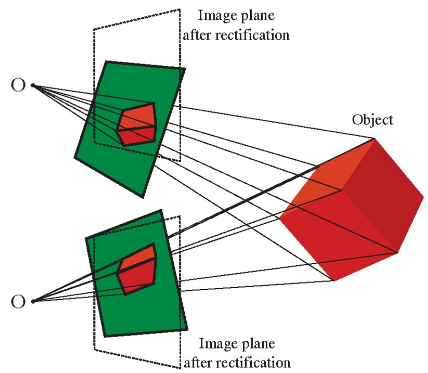
- Sparse matching is enough for navigation.
- However, for generation 3D model of the scene maximum possible points matched to find depth at that point.
  - This is known as *dense reconstruction*.
- Dense stereo reconstruction algorithms generally assume that corresponding points lie on the same horizontal scanline in the other image.
  - This is achieved by *rectification*.

Rectification

- There are two types of rectification methods
  - Planer rectification
  - Polar rectification

- Planer rectification orients image planes in Cartesian domain
  - Decomposition of fundamental matrix *F*

Rectification

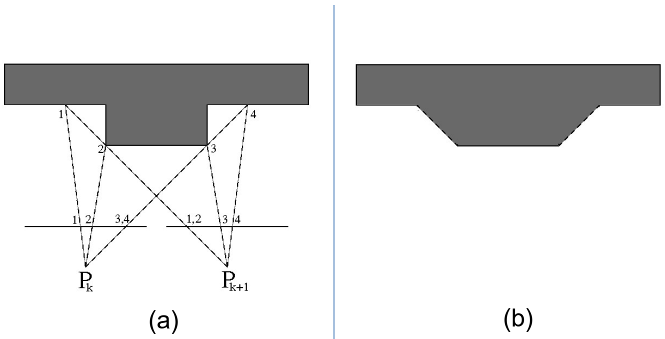


Rectification

Original image

Rectified image

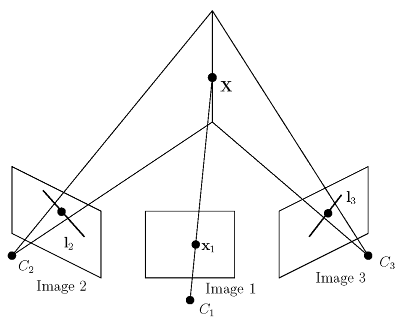
Self-occlusion



Multi-view reconstruction

- Often number of cameras is more than two.
  - This is referred to as ***structure from motion*** or ***multi-view reconstruction***.
- Suppose we have a number of frames all taken with the same camera
  - sufficient constraints to estimate intrinsic parameters well.
- Consider a point that is matched in three frames:
  - The point in the third frame will lie on an epipolar line due to first frame and epipolar line due to second frame.
  - its position has to be at intersection of two epipolar lines.

Multi-view reconstruction



Bundle adjustment

- Due to quanitization and other measurement errors, pair of same corresponding points in various pairs of images may not compute identical 3D coordinate of the object point.
- The distance between the re-projected 3D point and the measured image points should be minimized.
- This is done by minimizing the cost function (e.g., Euclidean distance) between said coordinates.
  - Various optimization tools may used.

Binocular stereo reconstruction

1. Compute image features.
2. Compute feature descriptors.
3. Find initial matches.
4. Compute fundamental matrix.
5. Refine matches.
6. Estimate essential matrix.
7. Decompose essential matrix.
8. Estimate 3D points.

Assignment – 1

1. Download and install any 3D reconstruction software (***e.g., meshroom, 3df zephyr***). Also download and install 3D display software ***(e.g., meshlab, 3D viewer***).
2. Choose a ***small object*** of size, say, within 1'x1'x1'.
3. Take enough photographs of the object from all sides.
4. Feed the photos to the ***3D reconstruction software***.
5. Display the reconstructed 3D object using appropriate software or send to me the object file so that I can display it on Meshlab app.

Thank you !

Any question?