PSET 2 Review

Aditya Dutt

CS231A

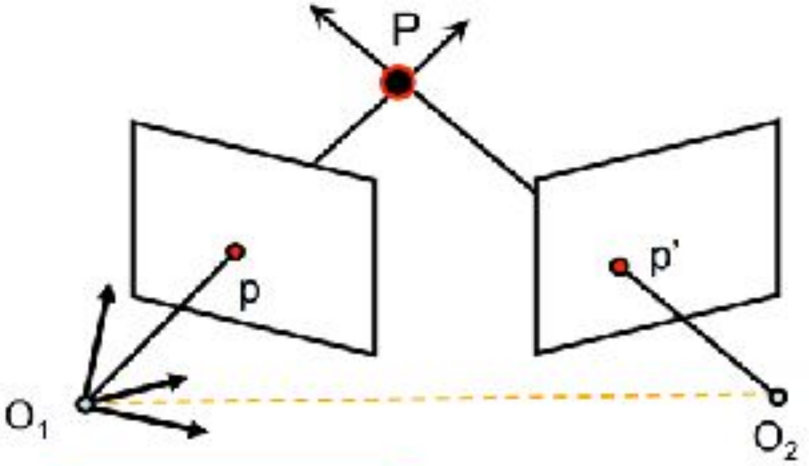
04/25/2025

Problem 1

Fundamental Matrix Estimation From Point Correspondences

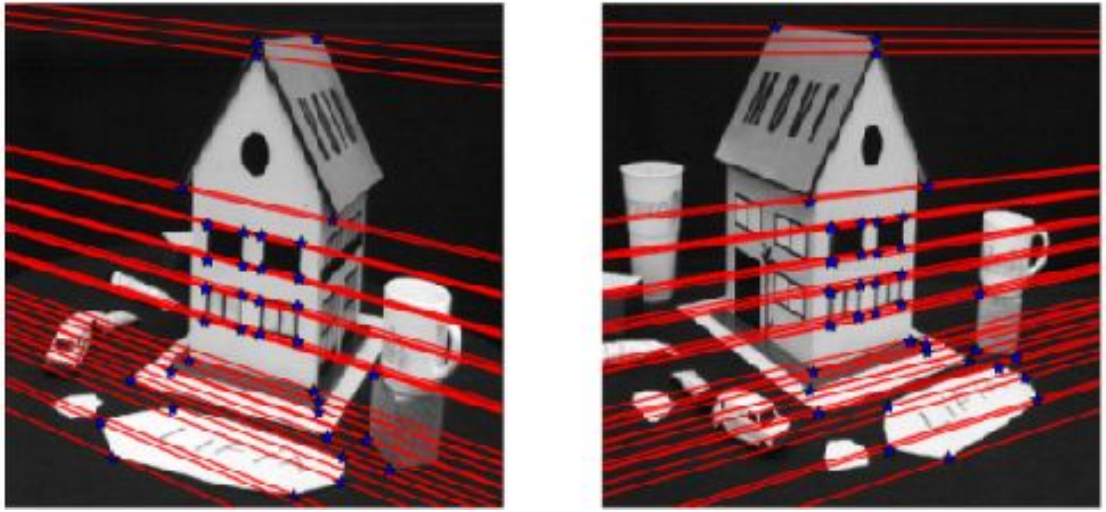
Fundamental Matrix

- A matrix which maps the relationship of correspondences between stereo images



[Eq. 13] $\boxed{p^T F p' = 0}$

$$F = K^{-T} \cdot [T_\times] \cdot R K'^{-1}$$



(a) Implement the linear least-squares eight point algorithm in `lls_eight_point_alg()`. Remember to enforce the rank-two constraint for the fundamental matrix via singular value decomposition. [**15 points**]

(b) **Include your resulting fundamental matrix with 4 decimal places minimum and briefly describe your implementation in your written report.** [**5 points**]

Computing F:
8-point algorithm

[Eq. 13] $p^T F p' = 0 \Rightarrow$

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad p' = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}$$

$$(u, v, 1)\begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix}\begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0$$

$$\Rightarrow \quad (uu', uv', u, vu', vv', v, u', v', 1)\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} = 0$$

[Eq. 14]

Let's take 8 corresponding points

Estimating F

Problem?

- Points can be very far off from each other
- W is highly unbalanced (not well conditioned)

$$\mathbf{W}\begin{pmatrix} u_1u'_1 & u_1v'_1 & u_1 & v_1u'_1 & v_1v'_1 & v_1 & u'_1 & v'_1 & 1 \\ u_2u'_2 & u_2v'_2 & u_2 & v_2u'_2 & v_2v'_2 & v_2 & u'_2 & v'_2 & 1 \\ u_3u'_3 & u_3v'_3 & u_3 & v_3u'_3 & v_3v'_3 & v_3 & u'_3 & v'_3 & 1 \\ u_4u'_4 & u_4v'_4 & u_4 & v_4u'_4 & v_4v'_4 & v_4 & u'_4 & v'_4 & 1 \\ u_5u'_5 & u_5v'_5 & u_5 & v_5u'_5 & v_5v'_5 & v_5 & u'_5 & v'_5 & 1 \\ u_6u'_6 & u_6v'_6 & u_6 & v_6u'_6 & v_6v'_6 & v_6 & u'_6 & v'_6 & 1 \\ u_7u'_7 & u_7v'_7 & u_7 & v_7u'_7 & v_7v'_7 & v_7 & u'_7 & v'_7 & 1 \\ u_8u'_8 & u_8v'_8 & u_8 & v_8u'_8 & v_8v'_8 & v_8 & u'_8 & v'_8 & 1 \end{pmatrix} \begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} \mathbf{f} = 0$$

[Eqs. 15]

- Homogeneous system $\mathbf{W}\,\mathbf{f} = 0$
- Rank 8 ⟶ A non-zero solution exists (unique)
- If N>8 ⟶ Lsq. solution by SVD! ⟶ $\hat{\mathrm{F}}$

$\|\mathbf{f}\| = 1$

Final step:

Reduce rank(F) to 2

$$F = U \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$$

Where:

$$U \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{bmatrix} V^T = SVD(\hat{F})$$

[HZ] pag 281, chapter 11, “Computation of F”

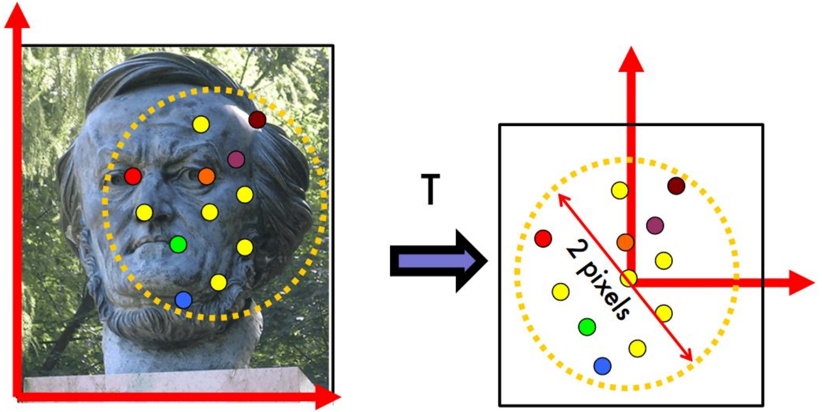
Possible improvement? Normalized Eight Point Algorithm

(c) Implement the normalized eight point algorithm in `normalized_eight_point_alg()`. Hint: Remember to enforce the rank-two constraint for the fundamental matrix via singular value decomposition. **[2.5 points]**

(d) **Report the returned fundamental matrix with 4 decimal places minimum. [2.5 points]**

Pre-condition our linear system to get more stable result

- origin = centroid of the points
- Set the mean square distance of the image points from origin to **~2px**



Coordinate system of the image before applying T

Coordinate system of the image after applying T

- Origin = centroid of image points
- Mean square distance of the image points from origin is ~2 pixels

The Normalized Eight-Point Algorithm

0. Compute T and T′ for image 1 and 2, respectively

1. Normalize coordinates in images 1 and 2:

$$q_i = T\ p_i \qquad q_i' = T'\ p_i'$$

2. Use the eight-point algorithm to compute $\hat{F}_q$ from the corresponding points q_i and q_i' .

1. Enforce the rank-2 constraint: $\rightarrow \mathrm{F_q}$ such that:

$$\begin{cases} \mathrm{q^T\ F_q\ q' = 0} \\ \mathrm{det(\ F_q\ ) = 0} \end{cases}$$

2. De-normalize F_q: $F = T^T F_q\ T'$

Computing Epipolar lines

(e) After implementing methods to determine the Fundamental matrix, we can now determine epipolar lines. Specifically to determine the accuracy of our Fundamental matrix, we will compute the average distance between a point and its corresponding epipolar line in `compute_distance_to_epipolar_lines()`. **[2.5 points]**

(f) **Briefly describe your implementation of** `compute_distance_to_epipolar_lines()` **in your written report. [2.5 points]**

Distance to epipolar lines

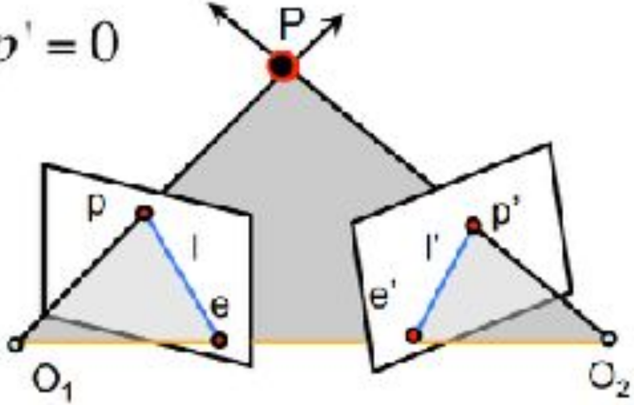
Computing epipolar lines from F

l = Fp’

l’ = F^Tp

Epipolar Constraint

$$p^T \cdot F\, p' = 0$$



$$\mathrm{distance}(ax + by + c = 0, (x_0, y_0)) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}.$$

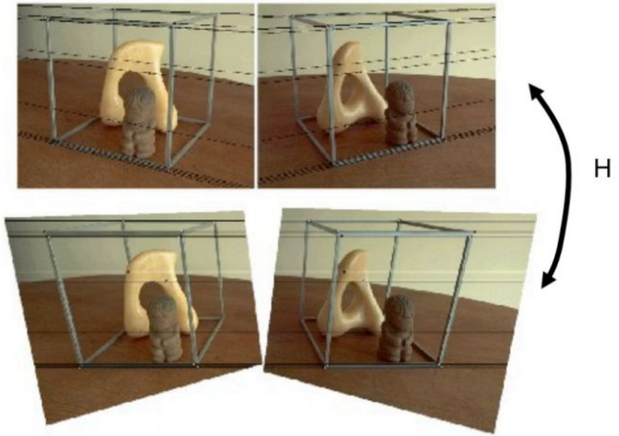
l = F p' is the epipolar line associated with p'

l'= F^Tp is the epipolar line associated with p

Problem 2

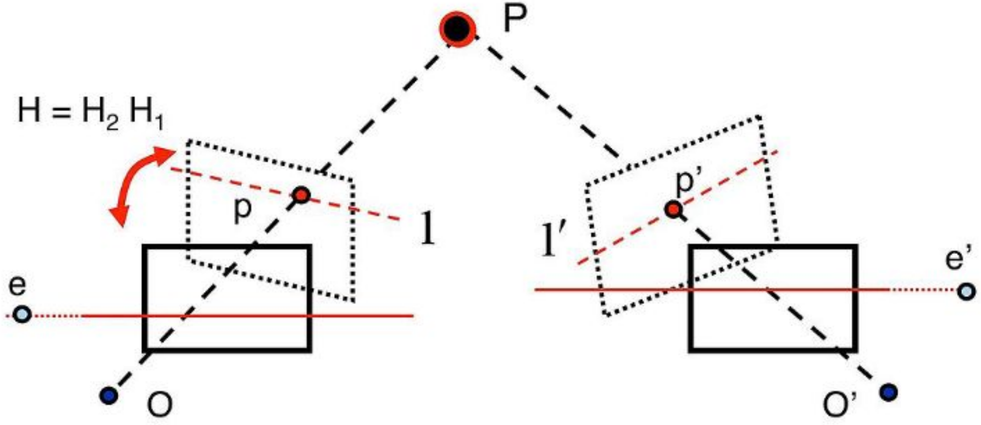
Matching Homographies for Image Rectification

Make two images parallel to each other
⇒ epipole at infinity along the horizontal axis



Make two images parallel to each other
⇒ epipole at infinity along the horizontal axis

1. Find epipoles
2. Solve for E



(a) Recall that:

- Epipolar line l = Fp'
- Epipole lies on epipolar lines
- Epipole is an intersection of all epipolar lines

$$\{x \mid \ell^T x = 0\}$$

(a) Compute epipole

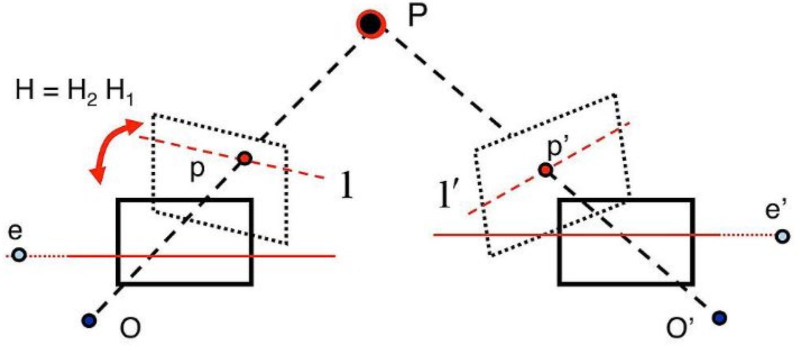
(b) Find an epipole of F using SVD

(c) Don’t forget to normalize so that e[2] = 1

$$\begin{bmatrix} \ell_1^T \\ \vdots \\ \ell_n^T \end{bmatrix} e = 0$$

(a) The first step in rectifying an image is to determine the epipoles. Complete the function `compute_epipole()`. Hint: Recall that $F^T e = 0$, and how you can use SVD to solve for e. **[2 points]**

(b) Solving for Homography matrix H that maps an epipole to infinity.



(b) Let's solve for the homography H that maps an epipole e to a point on the horizontal axis at infinity $(f, 0, 0)$. This may at first look complicated, but it is just a a sequence of several relatively simple steps. Complete the function `find_H()`. **[5 points]**

1. Find homography H_2 that maps the second epipole e’ to a horizontal axis at infinity (f, 0, 0)

i. Translate the second image s.t. the center is at (0, 0, 1) in homogeneous coord (***T***)

ii. Apply rotation to place the epipole on the horizontal axis (f, 0, 1) (***R***)

iii. Bring epipole at infinity on the horizontal axis (f, 0, 0) (***G***)

$$H_2 = T^{-1}GRT$$

i. Translate the second image s.t. the center is at (0, 0, 1) in homogeneous coord (***T***)

$$T = \begin{bmatrix} 1 & 0 & -\frac{\text{width}}{2} \\ 0 & 1 & -\frac{\text{height}}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

ii. Apply rotation to place the epipole on the horizontal axis (f, 0, 1) (***R***)

The translated epipole ***T***e’ = $(e'_1, e'_2, 1)$

$$R = \begin{bmatrix} \alpha\frac{e'_1}{\sqrt{e'^2_1+e'^2_2}} & \alpha\frac{e'_2}{\sqrt{e'^2_1+e'^2_2}} & 0 \\ -\alpha\frac{e'_2}{\sqrt{e'^2_1+e'^2_2}} & \alpha\frac{e'_1}{\sqrt{e'^2_1+e'^2_2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where $\alpha = 1$ if $e'_1 \geq 0$ and $\alpha = -1$ otherwise.

iii. Bring epipole (f, 0, 1) at infinity on the horizontal axis (f, 0, 0) (***G***)

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{1}{f} & 0 & 1 \end{bmatrix}$$

Find homography H_2 that maps the second epipole e’ to a horizontal axis at infinity (f, 0, 0)

i. Translate the second image s.t. the center is at (0, 0, 1) in homogeneous coord (***T***)

ii. Apply rotation to place the epipole on the horizontal axis (f, 0, 1) (***R***)

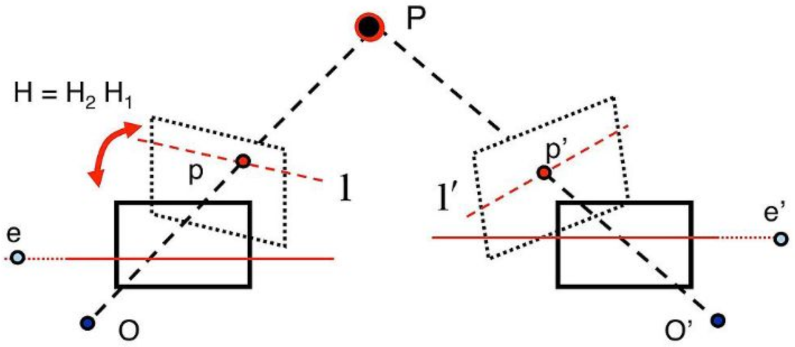
iii. Bring epipole at infinity on the horizontal axis (f, 0, 0) (***G***)

Bring together the three steps:

$$H_2 = T^{-1}GRT$$

2. Find two homographies that shift epipoles to infinity

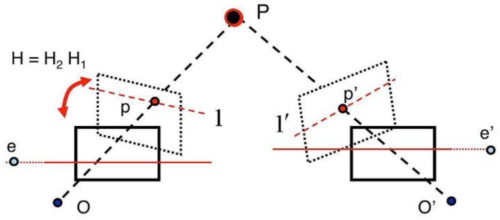
a. Find homography H_2 that maps the second epipole e’ to a horizontal axis at infinity (f, 0, 0)
b. Find the matching homography H_1 for the first image



(c)  Now for the trickier bit - finding a matching pair of homographies H_1 and H_2 by implementing `compute_matching_homographies()`. **[10 points]**

Find the matching homography H_1 for the first image

$$\arg\min_{H_1} \sum_i \|H_1 p_i - H_2 p_i'\|^2$$



Although the derivation is out of the scope of this class, we know that H_1 is of the form $H_1 = H_A H_2 M$

$$M = [e]_\times F + ev^T$$

$$v^T = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$H_A = \begin{bmatrix} a_1 & a_2 & a_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Find the matching homography H_1 for the first image

$$\arg\min_{H_1} \sum_i \|H_1 p_i - H_2 p'_i\|^2$$

Since we already know H_2 and M, we can write

$$\hat{p}_i = H_2 M p_i \qquad \hat{p}'_i = H_2 p'_i$$

and the minimization problem becomes

$$\arg\min_{H_A} \sum_i \|H_A \hat{p}_i - \hat{p}'_i\|^2$$

Solve for a = (a_1, a_2, a_3) in $H_A = \begin{bmatrix} a_1 & a_2 & a_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ with $\arg\min_{H_A} \sum_i \|H_A \hat{p}_i - \hat{p}'_i\|^2$

where $\hat{p}_i = H_2 M p_i \quad \hat{p}'_i = H_2 p'_i$

Let $\hat{p}_i = (\hat{x}_i, \hat{y}_i, 1)$ and $\hat{p}'_i = (\hat{x}'_i, \hat{y}'_i, 1)$, the problem becomes

$$\arg\min_{\mathbf{a}} \sum_i (a_1 \hat{x}_i + a_2 \hat{y}_i + a_3 - \hat{x}'_i)^2 + \cancel{(\hat{y}_i - \hat{y}'_i)^2}$$

constant value

Solve least-square $W\mathbf{a} = b$ with $W = \begin{bmatrix} \hat{x}_1 & \hat{y}_1 & 1 \\ & \vdots & \\ \hat{x}_n & \hat{y}_n & 1 \end{bmatrix} \quad b = \begin{bmatrix} \hat{x}'_1 \\ \vdots \\ \hat{x}'_n \end{bmatrix}$

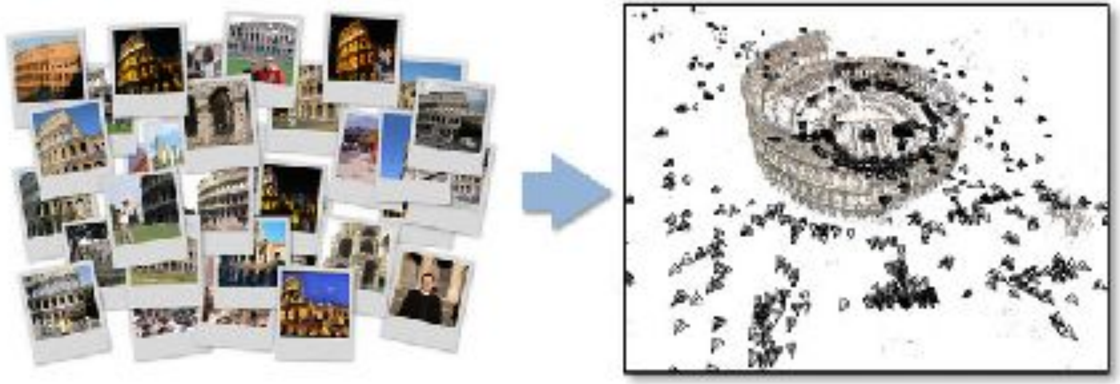
Problem 3+4: Structure from Motion

Structure from Motion (SfM)

Estimating 3D structure from 2D images that may be coupled with local motions

Input: 2D images

Output: 3D structure
(+ camera extrinsic)



Problem 3

The Factorization Method

Tomasi and Kanade Factorization Method

(a) Implement the factorization method as described in lecture and in the course notes. Complete the function `factorization_method()`. **[8 points]**

(b) **Briefly describe your implementation in your written report. [2 points]**

Data centering step: center the data at the origin

$$\hat{x}_{ij} = x_{ij} - \bar{x}_i = x_{ij} - \frac{1}{n}\sum_{j=1}^{n} x_{ij}$$

Affine SfM problem $x_{ij} = A_i X_j + b_i$

After centering

$$\begin{aligned}
\hat{x}_{ij} &= x_{ij} - \frac{1}{n}\sum_{k=1}^{n} x_{ik} \\
&= A_i X_j - \frac{1}{n}\sum_{k=1}^{n} A_i X_k \\
&= A_i (X_j - \frac{1}{n}\sum_{k=1}^{n} X_k) \\
&= A_i (X_j - \bar{X}) \\
&= A_i \hat{X}_j
\end{aligned}$$

Factorization: Factor out motion matrix A_i and structure X_j $\hat{x}_{ij} = A_i \hat{X}_j$

Build the measure matrix from all camera observations (2mxn)

$$D = \begin{bmatrix} \hat{x}_{11} & \hat{x}_{12} & \dots & \hat{x}_{1n} \\ \hat{x}_{21} & \hat{x}_{22} & \dots & \hat{x}_{2n} \\ & & \ddots & \\ \hat{x}_{m1} & \hat{x}_{m2} & \dots & \hat{x}_{mn} \end{bmatrix}$$

Factorization $D = U\Sigma V^T$

$$= \begin{bmatrix} u_1 & \dots & u_n \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ & & & & \ddots & \\ 0 & 0 & 0 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_n^T \end{bmatrix}$$

$$= \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ v_3^T \end{bmatrix}$$

$$= U_3 \Sigma_3 V_3^T$$

Structure matrix (3xn): $S = \sqrt{\Sigma_3} V_3^T$

Motion matrix (2mx3): $M = U_3 \sqrt{\Sigma_3}$

Problem 4

Triangulation in Structure From Motion

1. Compute essential matrix E from two views

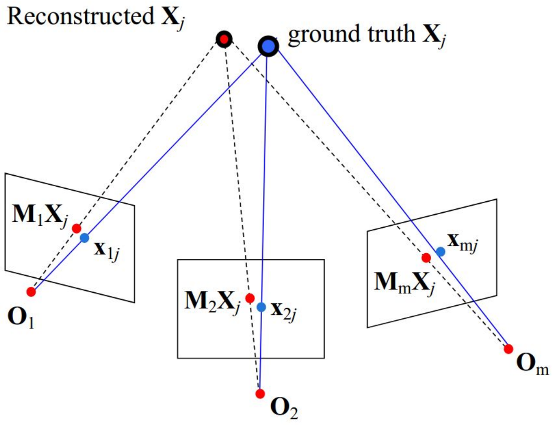
2. Use E to make initial estimate of relative rotation R and translation T

3. Estimate 3D reconstructed points given R,T

4. Optimize (<u>bundle adjustment</u>)
    - Jointly optimize all relative camera motions (R’s and T’s)
    - Minimize total reprojection error with respect to all 3D point and camera parameters

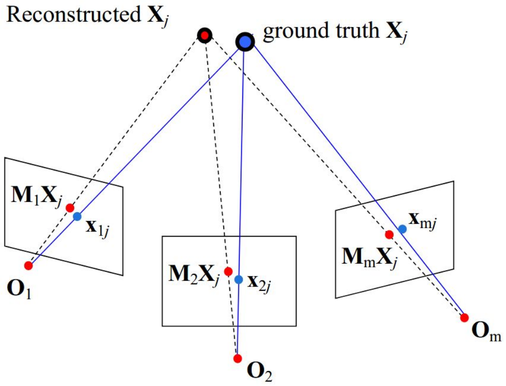
5. Repeat 3 and 4 for pairs of frames

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



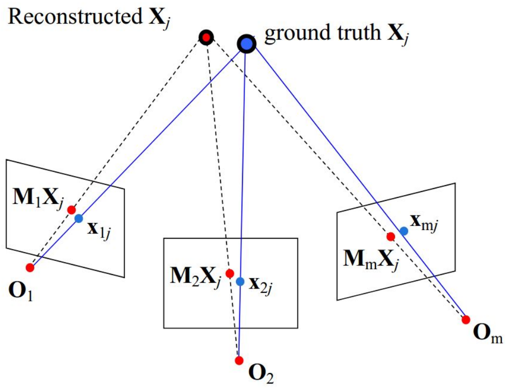
1. **Compute essential matrix E from two views**
2. Use E to make initial estimate of relative rotation R and translation T
3. Estimate 3D reconstructed points given R,T
4. Optimize (bundle adjustment)
   - Jointly optimize all relative camera motions (R's and T's)
   - Minimize total reprojection error with respect to all 3D point and camera parameters
5. Repeat 3 and 4 for pairs of frames

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



1. Compute essential matrix E from two views
2. **Use E to make initial estimate of relative rotation R and translation T**
3. Estimate 3D reconstructed points given R,T
4. Optimize (bundle adjustment)
   - Jointly optimize all relative camera motions (R's and T's)
   - Minimize total reprojection error with respect to all 3D point and camera parameters
5. Repeat 3 and 4 for pairs of frames

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



1. To compute R: Given the singular value decomposition $E = UDV^T$

$$Q = UWV^T \ or \ UW^TV^T, \text{ where}$$

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

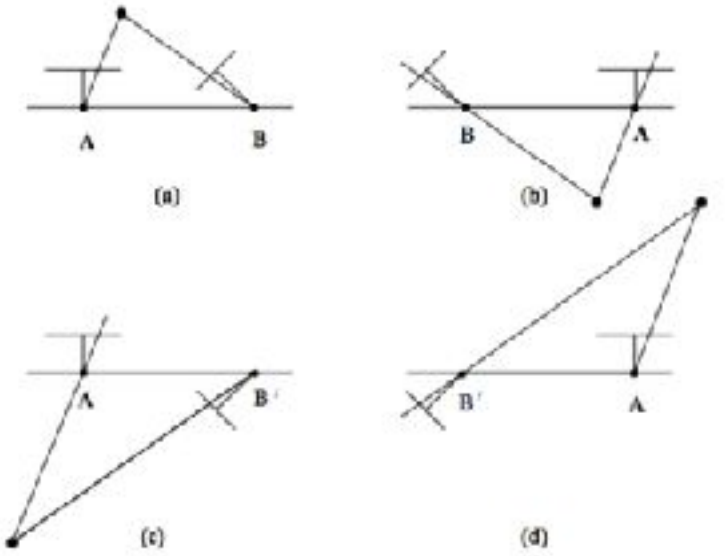
   Note that this factorization of E only guarantees that Q is orthogonal. To find a rotation, we simply compute $R = (\det Q)Q$.

2. To compute T: Given that $E = U\Sigma V^T$, T is simply either u_3 or $-u_3$, where u_3 is the third column vector of U.

Use E to make initial estimate of relative rotation R and translation T

However, this gives four pairs of rotation and translation, $(R_1, R_2) \times (T, -T)$

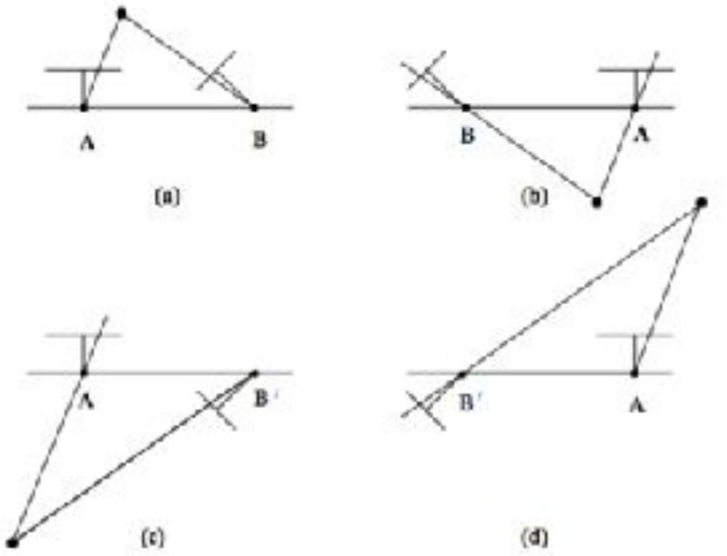
How do we find out which R and T is the correct one?



There exists only **one** solution that will consistently produce 3D points which are both in front of camera

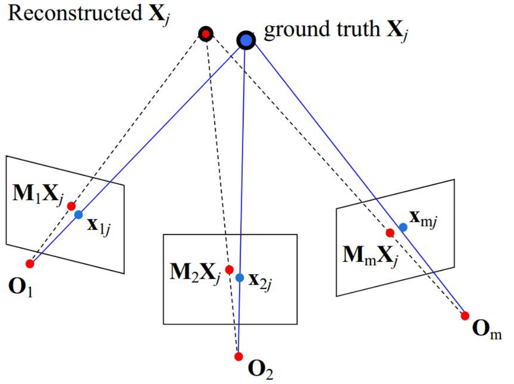
Compute 3D point's location in the R,T frame!

- Find 3D location of the image points given R,T frame
- Chose the one which has the most 3D points with positive depth (z-coordinate) with respect to both camera frame



1. Compute essential matrix E from two views
2. Use E to make initial estimate of relative rotation R and translation T
3. **Estimate 3D reconstructed points given R,T**
4. Optimize (bundle adjustment)
   - Jointly optimize all relative camera motions (R’s and T’s)
   - Minimize total reprojection error with respect to all 3D point and camera parameters
5. Repeat 3 and 4 for pairs of frames

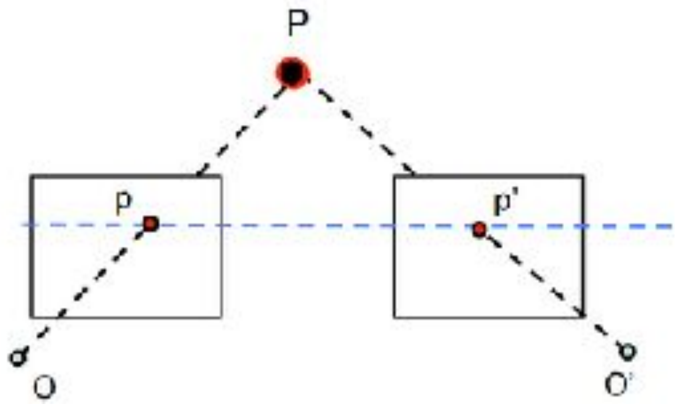
$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



Estimate 3D reconstructed points given
1. projective camera matrix
2. their image coordinates

Two different possible approaches:

1. **Formulating a linear equation to solve**
2. Nonlinear optimization to minimize reprojection error



1. For each image i, we have $p_i = M_i P$, where P is the 3D point, p_i is the homogenous image coordinate of that point, and M_i is the projective camera matrix.

2. Formulate matrix

$$A = \begin{bmatrix} p_{1,1} m^{3\top} - m^{1\top} \\ p_{1,2} m^{3\top} - m^{2\top} \\ \vdots \\ p_{n,1} m^{3\top} - m^{1\top} \\ p_{n,2} m^{3\top} - m^{2\top} \end{bmatrix}$$

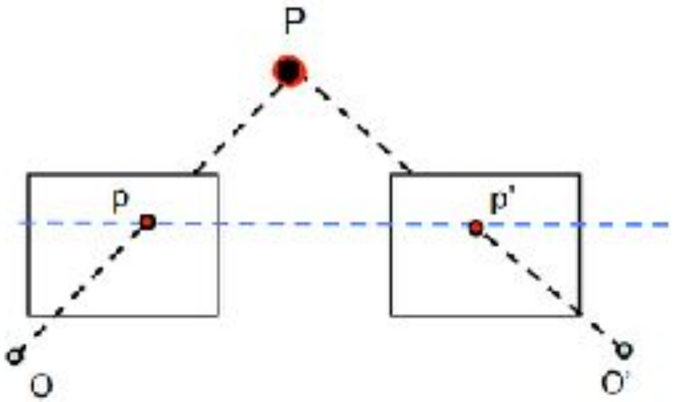
   where $p_{i,1}$ and $p_{i,2}$ are the xy coordinates in image i and $m^{k\top}$ is the k-th row of M.

3. The 3D point can be solved for by using the singular value decomposition.

Estimate 3D location of the reconstruction given 1. projective camera matrix 2. their image coordinates

Two different possible approaches:

1. Formulating a linear equation to solve
2. **Nonlinear optimization to minimize reprojection error**



Estimate 3D location of the reconstruction
given 1. projective camera matrix 2. their image coordinates

Nonlinear optimization to minimize reprojection error

Gauss-Newton algorithm

$$\hat{P} = \hat{P} - (J^T J)^{-1} J^T e$$

Begin from linear estimation for better initialization (reprojection) error: difference between the projected point ($M_i P$) and ground-truth image coordinate p_i

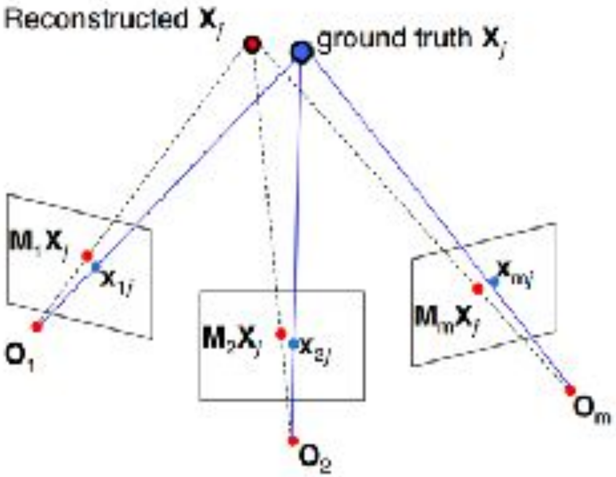
$$e = \begin{bmatrix} e_1 \\ \vdots \\ e_N \end{bmatrix} = \begin{bmatrix} p_1 - M_1 \hat{P} \\ \vdots \\ p_n - M_n \hat{P} \end{bmatrix}$$

Jacobian:

$$J = \begin{bmatrix} \frac{\partial e_1}{\partial \hat{P}_1} & \frac{\partial e_1}{\partial \hat{P}_2} & \frac{\partial e_1}{\partial \hat{P}_3} \\ \vdots & \vdots & \vdots \\ \frac{\partial e_N}{\partial \hat{P}_1} & \frac{\partial e_N}{\partial \hat{P}_2} & \frac{\partial e_N}{\partial \hat{P}_3} \end{bmatrix}$$

1. Compute essential matrix E from two views
2. Use E to make initial estimate of relative rotation R and translation T
3. Estimate 3D reconstructed points given R,T
4. Optimize (bundle adjustment)
    - Jointly optimize all relative camera motions (R’s and T’s)
    - Minimize total reprojection error with respect to all 3D point and camera parameters
5. Repeat 3 and 4 for pairs of frames

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



Thanks!