CSE 152A: Computer Vision

Manmohan Chandraker

Lecture 8: Two-View Reconstruction

Overall goals for the course

- Introduce fundamental concepts in computer vision
- Enable one or all of several such outcomes
  - Pursue higher studies in computer vision
  - Join industry to do cutting-edge work in computer vision
  - Gain appreciation of modern computer vision technologies
- Engage in discussions and interaction
- This is a great time to study computer vision!

Course Details

Course details

- Class webpage:
  - https://cseweb.ucsd.edu/~mkchandraker/classes/CSE152A/Winter2024/
- Instructor email:
  - mkchandraker@ucsd.edu
- Grading
  - 35% final exam
  - 40% homework assignments
  - 20% mid-term
  - 5% self-study exercise
  - Ungraded quizzes
- Aim is to learn together, discuss and have fun!

Course details

- TAs
  - Nicholas Chua: nchua@ucsd.edu
  - Tarun Kalluri: sskallur@ucsd.edu
  - Sreyas Ravichandran: srravichandran@ucsd.edu
- Tutors
  - Kun Wang, Kevin Chan, Zixian Wang: {kuw010, tsc003, ziw081}@ucsd.edu
- Discussion section: M 3-3:50pm
- TA office hours and tutor hours to be posted on webpage
- Piazza for questions and discussions:
  - https://piazza.com/ucsd/winter2024/cse152a

Self-Study Assignment

- Pick a technology area primarily driven by computer vision
  - Can pick one of these suggestions, or use anything else that you like

- **Virtual Reality**
  - Meta Quest Pro
  - Oculus Rift
- **Augmented Reality**
  - Microsoft Hololens
  - Magic Leap 2
- **Self-Driving**
  - Waymo
  - Tesla
- **Content Creation**
  - Adobe Photoshop
  - OpenAI Dall-E
- **Cloud Services**
  - Amazon Rekognition
  - Microsoft Azure Cognitive Services

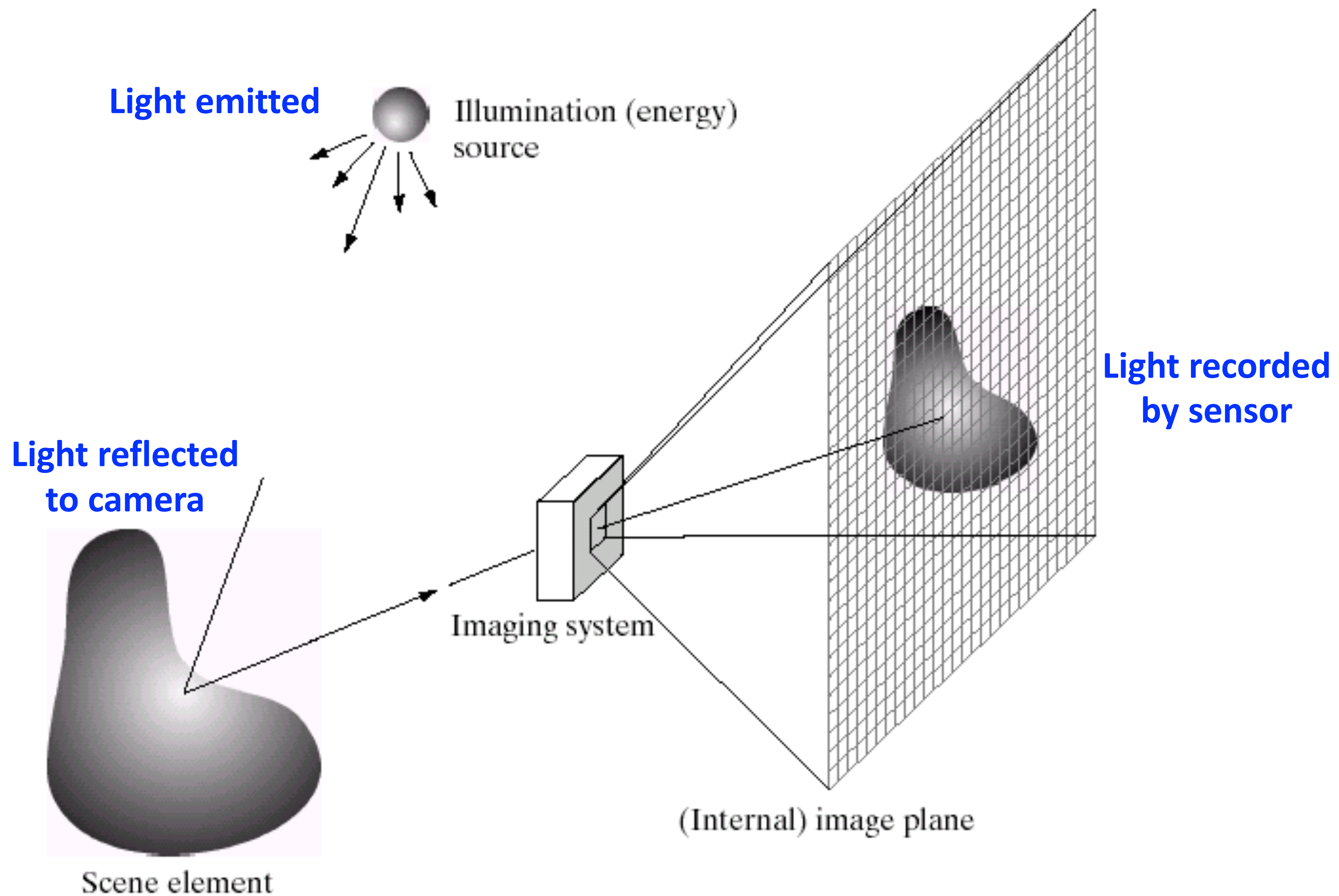
- **Sports**
  - Hawk-Eye
  - Gameface.ai
- **Face Recognition**
  - Face++
  - Apple FaceID
- **Robotics**
  - Boston Dynamics
  - iRobot Roomba
- **Space Exploration**
  - James Webb Telescope
  - Mars Rover
- **Social Media**
  - Snap
  - Instagram

Self-Study Assignment

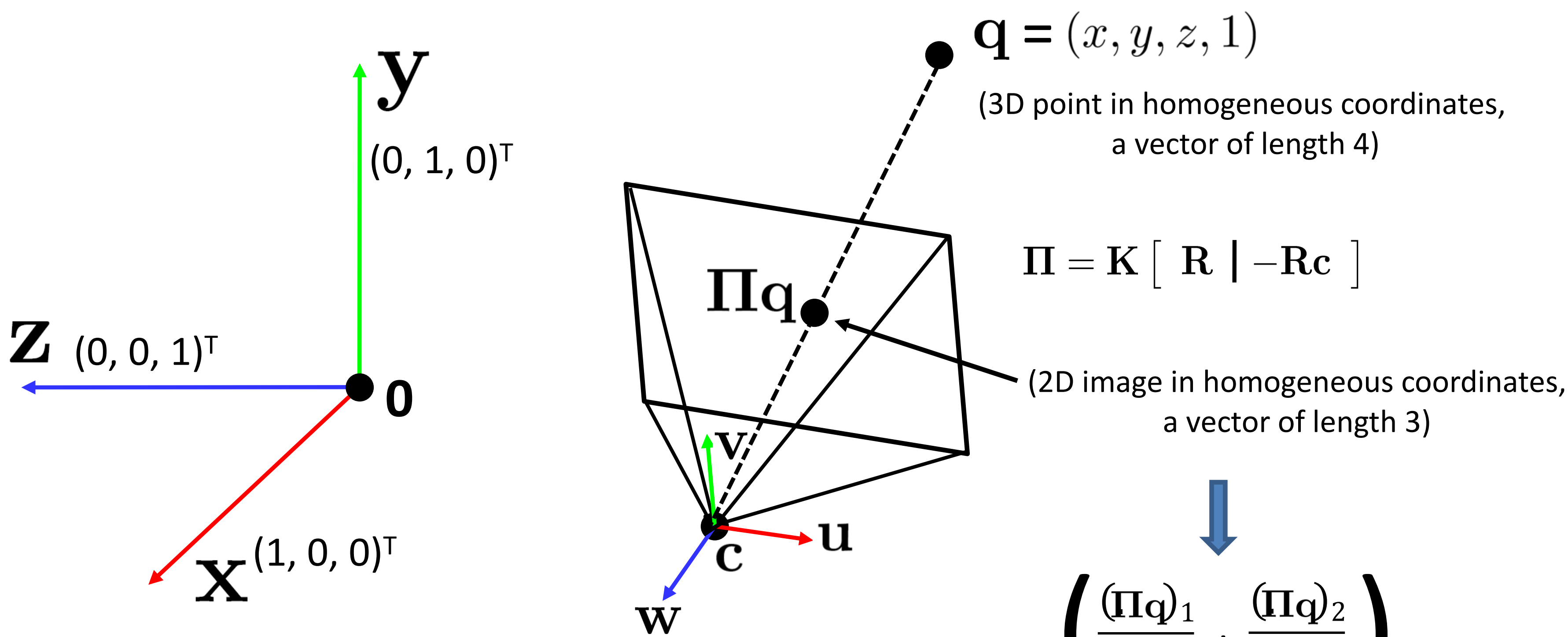
- Form teams of 4 students (mandatory, cannot be less than 4)
- Pick a technology area primarily driven by computer vision
  - Can pick one of these suggestions, or use anything else that you like
- Make a 5-slide PPT report
  - Include pictures (with citations), brief text bullet points or captions
- Prompts for each slide
  - Slide 1: Title and team members
  - Slide 2: Describe the technology and the abilities it enables
  - Slide 3: How does computer vision overcome barriers or solve needs in this technology?
  - Slide 4: How do you anticipate technology in this area will advance in the next 10 years?
  - Slide 5: What are the potential benefits and dangers from this technology in the future?
- Due date: Mar 4, 2023
- Students and instructors will vote for the top-5 studies by Mar 9
  - Top-5 studies may be presented in-class by the teams during Mar 15 lecture

Recap

Photometric: Modeling appearance



Geometric: Modeling projection



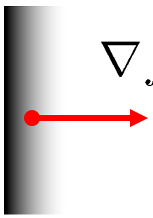
$$\Pi = \mathbf{K} \left[\ \mathbf{R} \ | -\mathbf{R}\mathbf{c} \ \right]$$

$$\left(\frac{(\Pi\mathbf{q})_1}{(\Pi\mathbf{q})_3}, \frac{(\Pi\mathbf{q})_2}{(\Pi\mathbf{q})_3} \right)$$

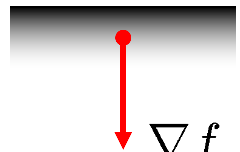
(Usual Cartesian coordinates,
a vector of length 2)

Edge Detection with Image Gradients

- Gradient represents direction of most rapid change in intensity

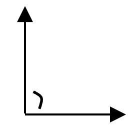


$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$



- The gradient encodes *edge strength* and *edge direction* as

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \qquad \theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$$

- Can efficiently compute gradient using convolutions

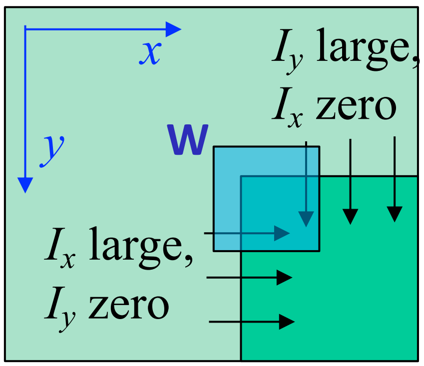
$$\boldsymbol{K}_x = \frac{1}{2}\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{K}_y = \frac{1}{2}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

- Sobel operator is often used in practice

$$\boldsymbol{K}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \qquad \boldsymbol{K}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Harris Corner Detector

First, consider the second moment matrix for a simpler case:



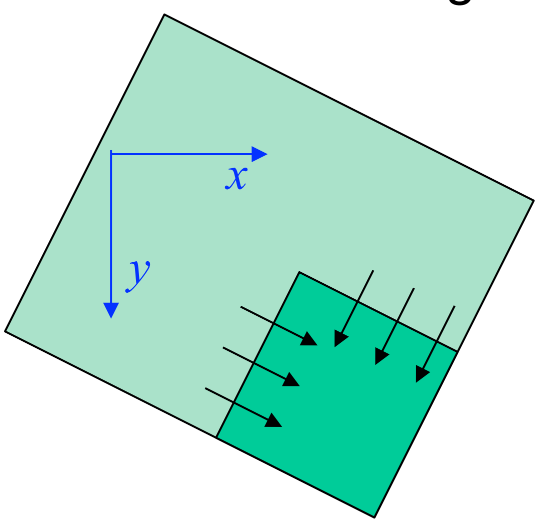
Sum over a small window W around hypothetical corner

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means dominant gradient directions align with x or y axis.

Q

In the general case, since C is symmetric, it can be shown:



$$C = Q^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} Q$$

Eigenvalues

Rotation

If either λ close to 0, then **not** a corner, so seek locations where both large.

Simple matching methods

- SSD (Sum of Squared Differences)

$$\sum_{x,y} |W_1(x,y) - W_2(x,y)|^2$$

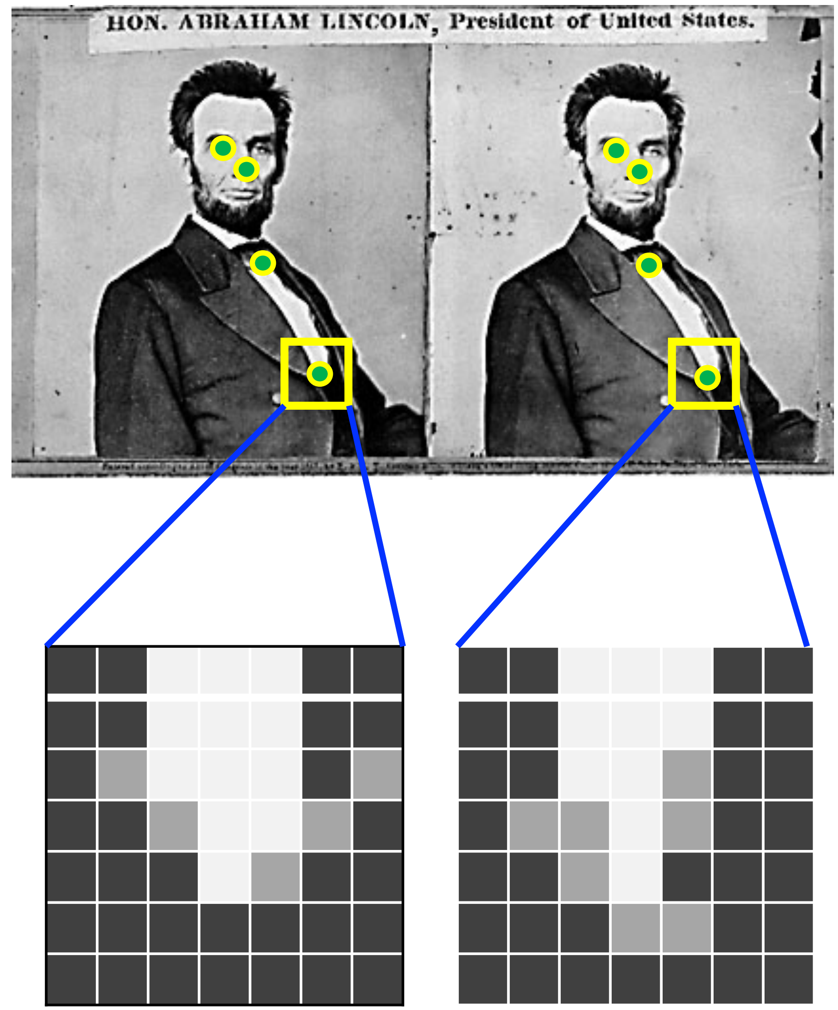
- NCC (Normalized Cross Correlation)

$$\sum_{x,y} \frac{(W_1(x,y) - \overline{W_1})(W_2(x,y) - \overline{W_2})}{\sigma_{W_1}\sigma_{W_2}}$$

$$\overline{W_i} = \frac{1}{n}\sum_{x,y} W_i \,, \quad \sigma_{W_i} = \sqrt{\frac{1}{n}\sum_{x,y}(W_i - \overline{W_i})^2}$$

(Mean) (Standard deviation)

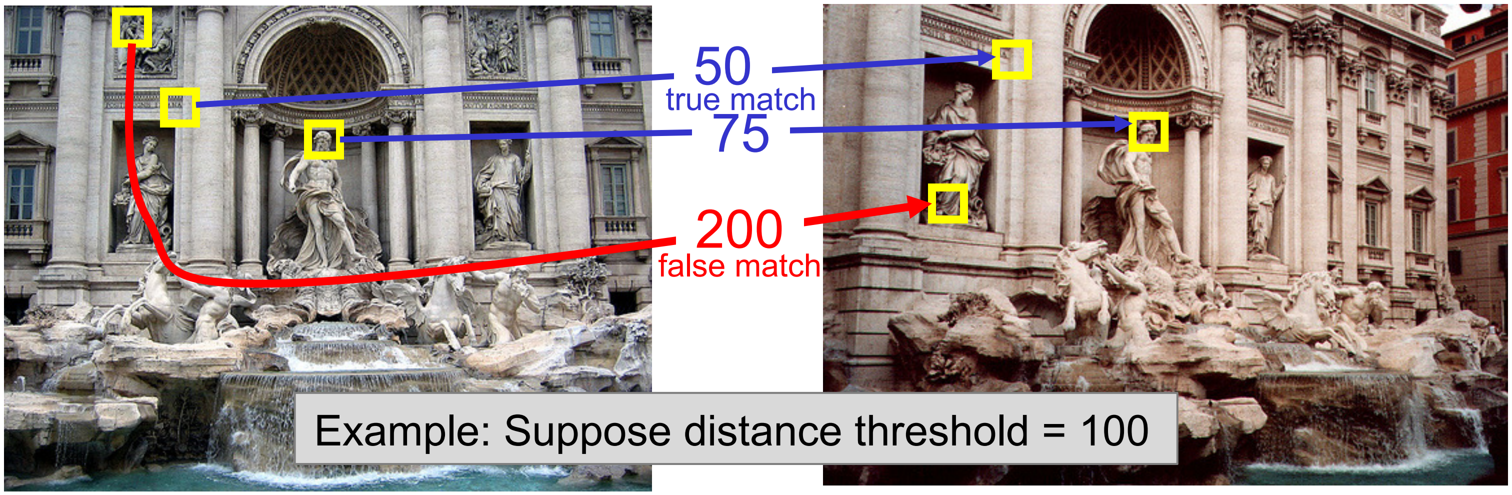
- What advantages might NCC have over SSD?



$W_1(x, y)$: k x k pixel patch in image 1

$W_2(x, y)$: k x k pixel patch in image 2

True or false positives



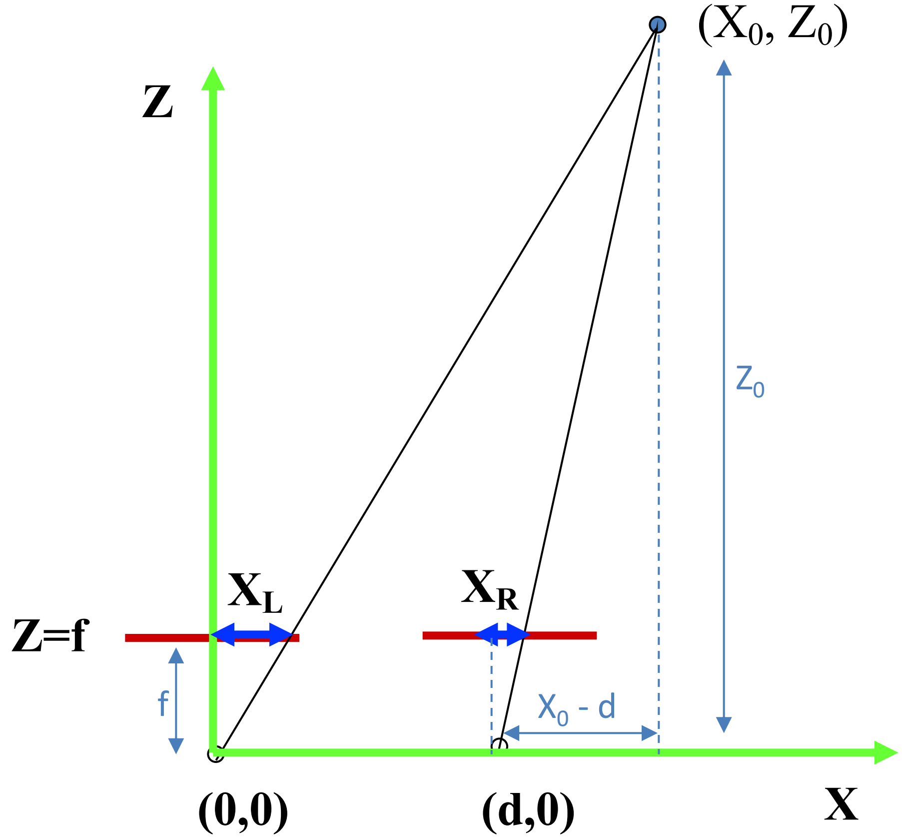
Positive match if SSD < distance threshold

The distance threshold affects performance

- True positives = number of detected matches that are correct
  - Suppose we want to maximize these—how to choose threshold?
  - Increase threshold (uncertain matches are also allowed)

- False positives = number of detected matches that are incorrect
  - Suppose we want to minimize these—how to choose threshold?
  - Decrease threshold (matches discarded unless they are very certain)

Depth from correspondence



Using similar triangles:

$$\frac{X_L}{f} = \frac{X_0}{Z_0} \qquad \frac{X_R}{f} = \frac{X_0 - d}{Z_0}$$

Two measurements: X_L, X_R
Two unknowns: X_0, Z_0

Constants:
Baseline: d
Focal length: f

$$X_0 = \frac{d\, X_L}{(X_L - X_R)}$$

$$Z_0 = \frac{d\, f}{(X_L - X_R)}$$

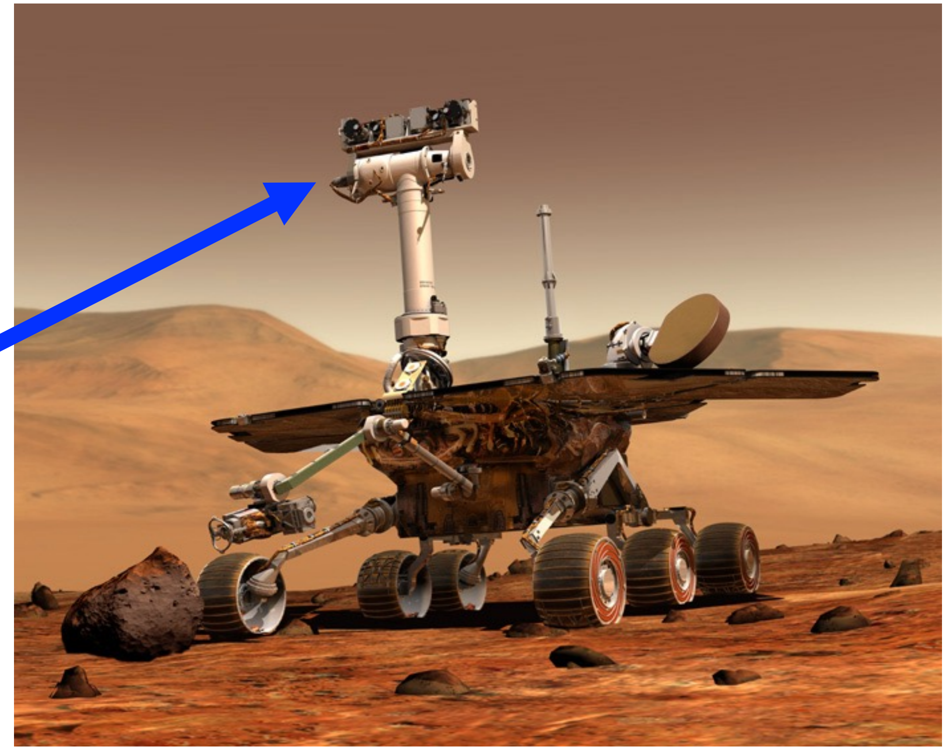
Disparity: $(X_L - X_R)$

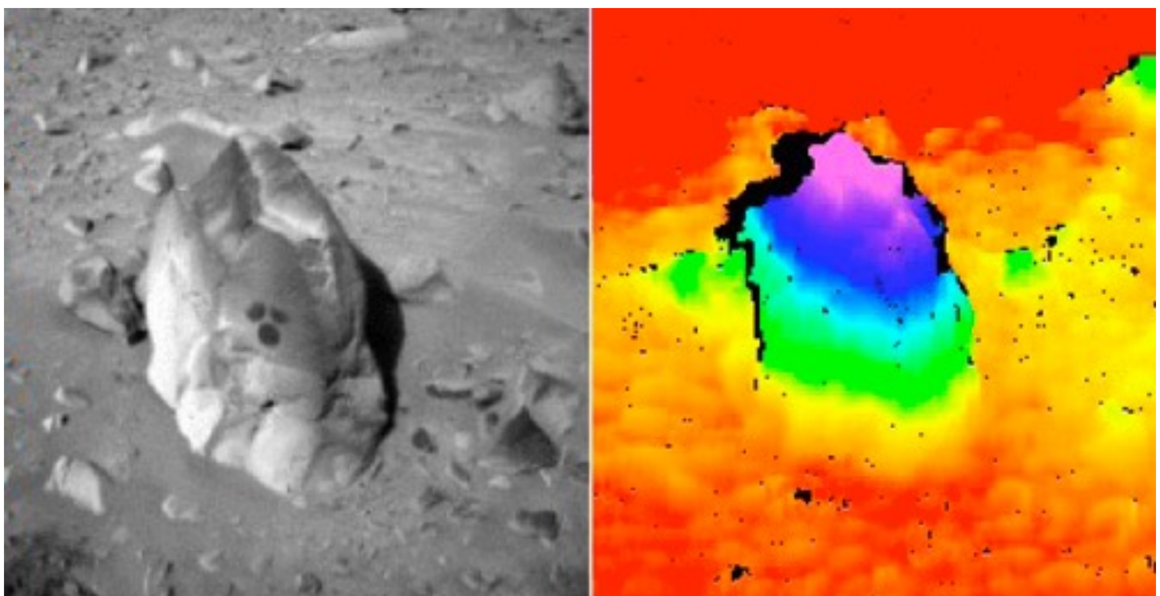
Depth is inversely proportional to disparity

(Adapted from Hager)

Mars Exploratory Rovers:
Spirit and Opportunity,
2004

Stereo camera



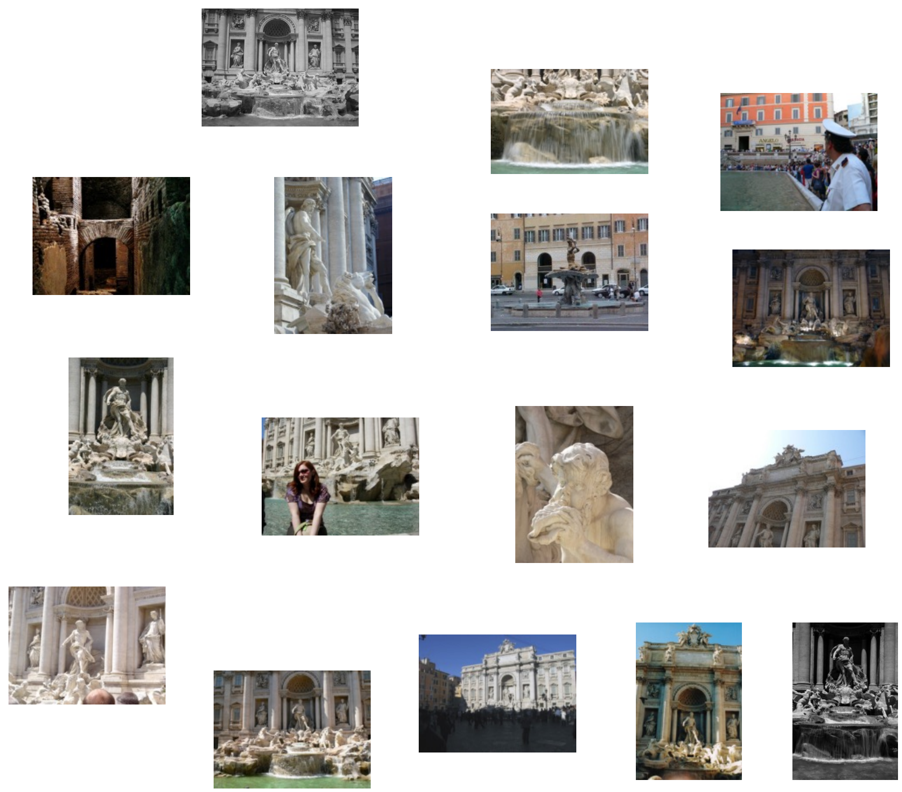


Structure from Motion (SFM)
Visual SLAM

Structure from Motion

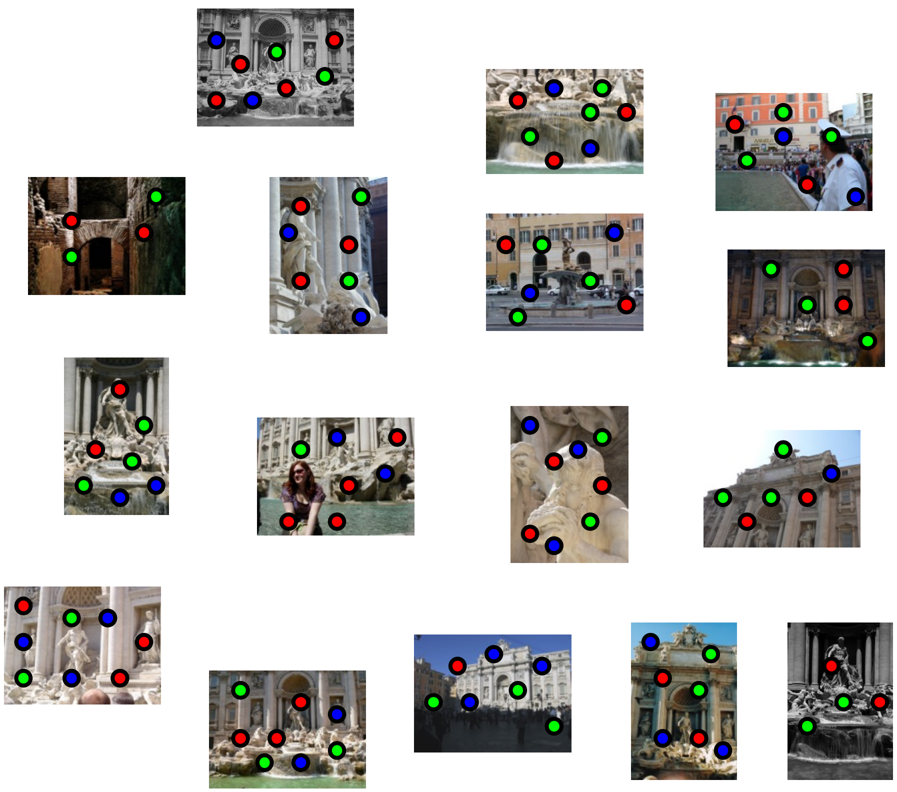
Feature detection

Several images observe a scene from different viewpoints



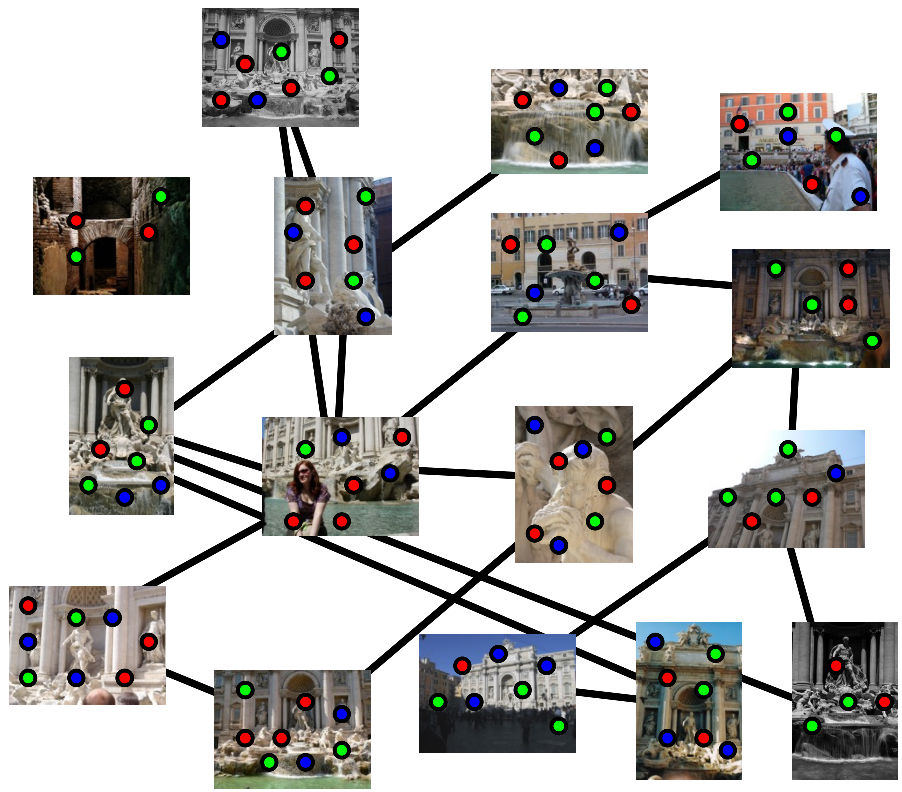
Feature detection

Detect features using, for example, corners or SIFT [Lowe, IJCV 2004]



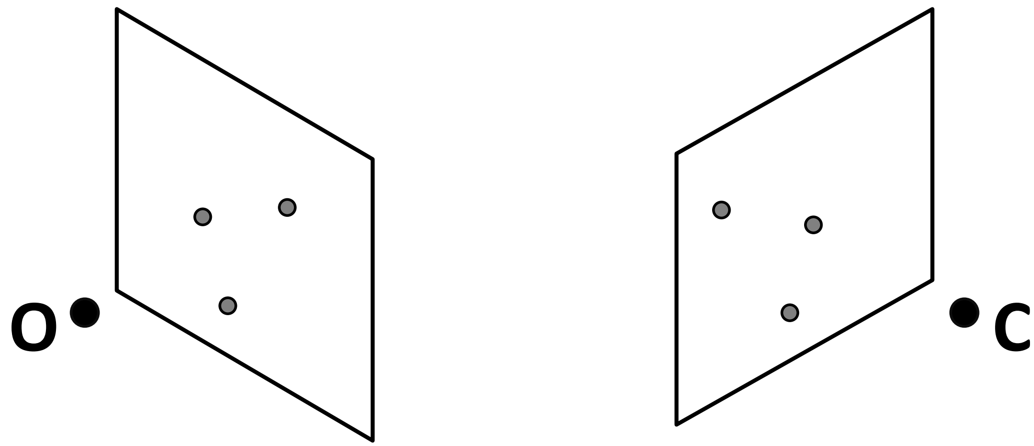
Feature matching

Match features between each pair of images



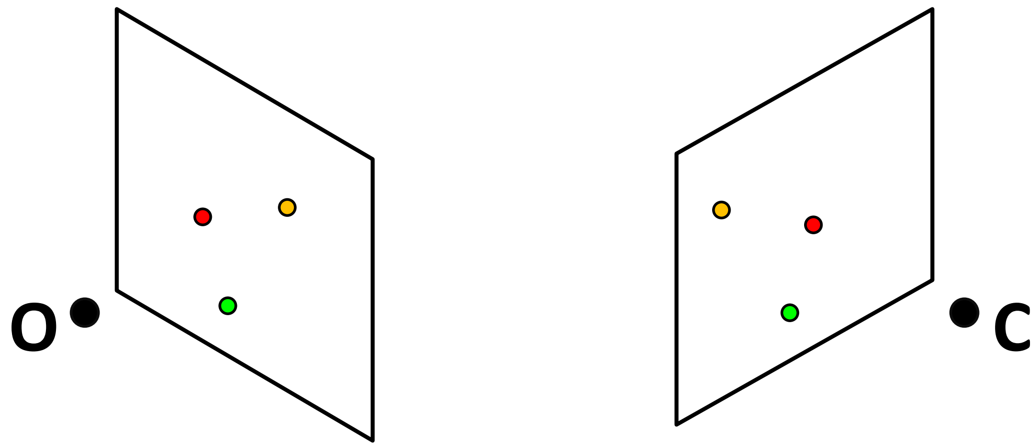
Two-View Reconstruction

Two-View Reconstruction: Overall Idea



Step 1: Detect features in each view

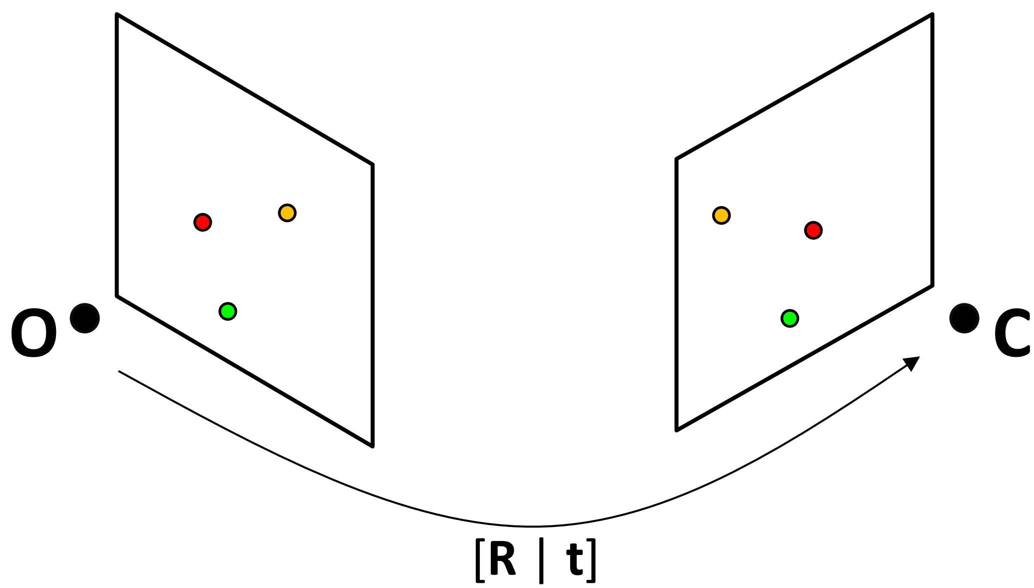
Two-View Reconstruction: Overall Idea



Step 1: Detect features in each view

Step 2: Match features across two views

Two-View Reconstruction: Overall Idea

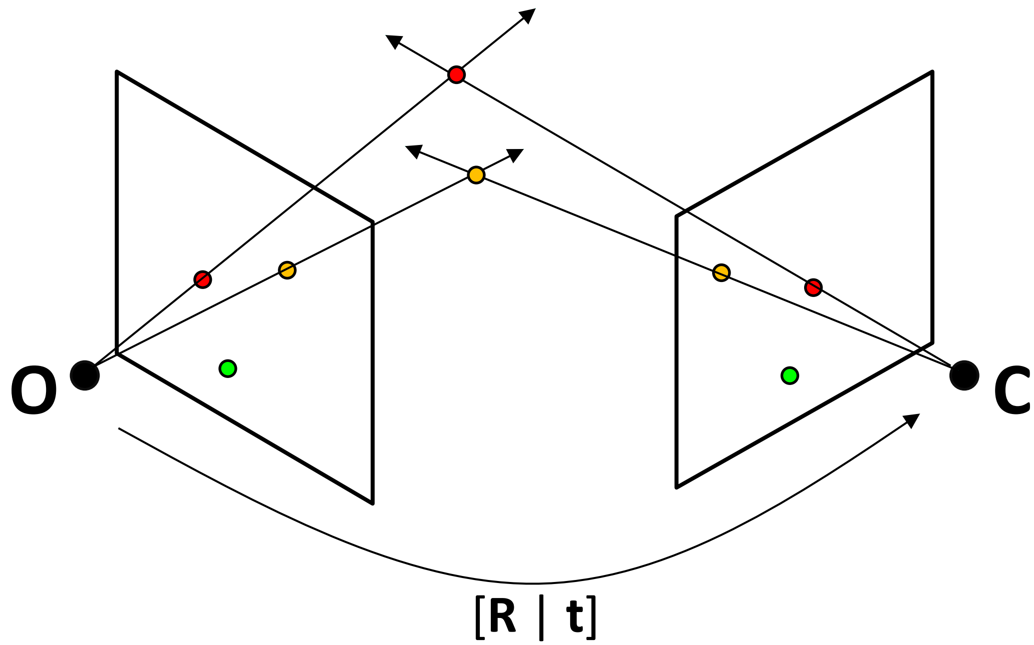


Step 1: Detect features in each view

Step 2: Match features across two views

Step 3: Estimate camera rotation and translation across views

Two-View Reconstruction: Overall Idea



Step 1: Detect features in each view

Step 2: Match features across two views

Step 3: Estimate camera rotation and translation across views

Step 4: Backproject rays from camera centers to triangulate 3D point

Cross-product as linear operator

Useful fact: Cross product with a vector **t** can be represented as multiplication with a (*skew-symmetric*) 3x3 matrix

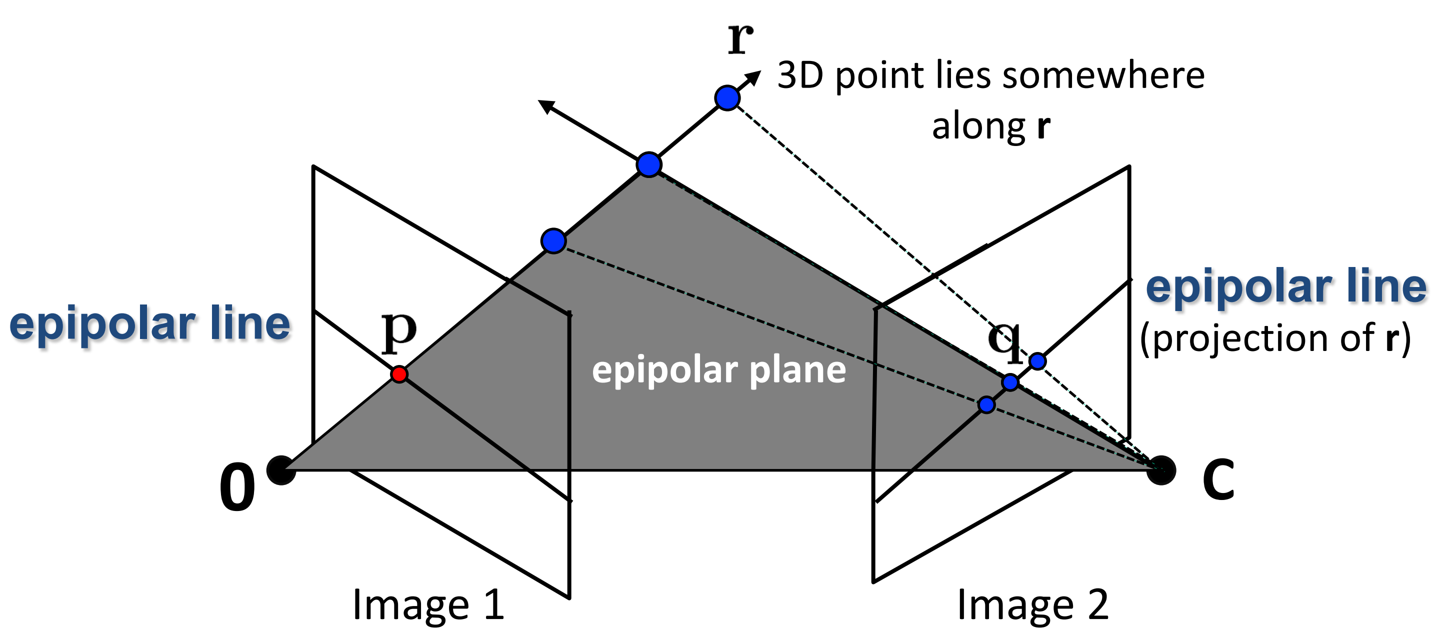
$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

$$\mathbf{t} \times \tilde{\mathbf{p}} = [\mathbf{t}]_\times \tilde{\mathbf{p}}$$

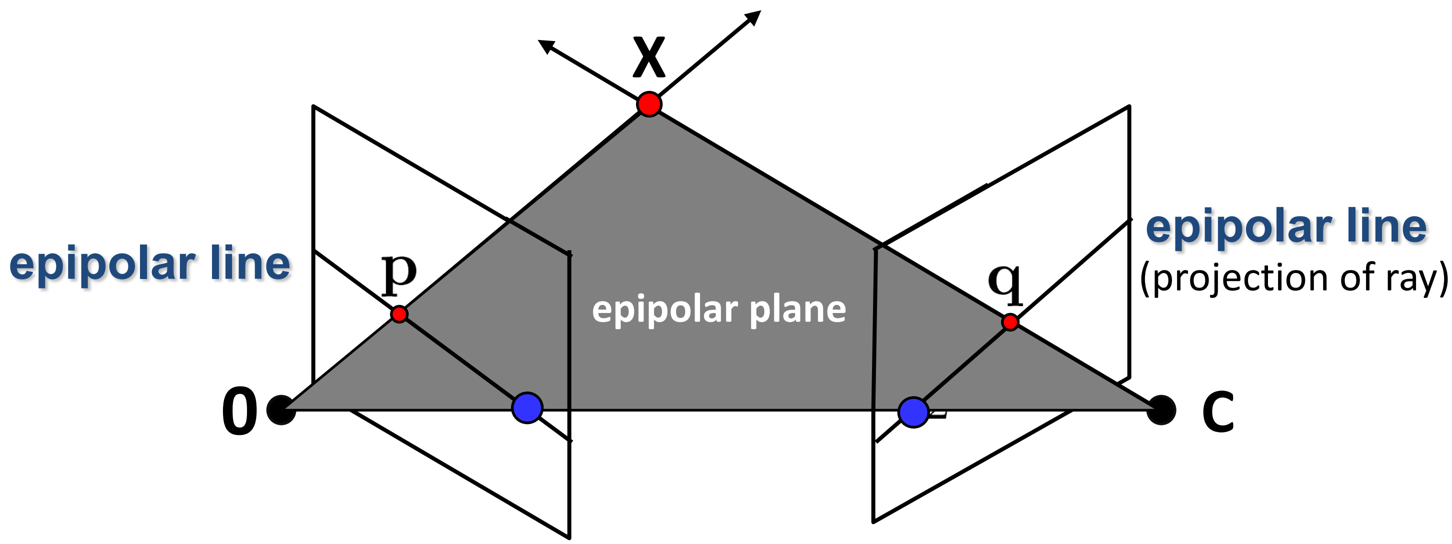
What is the rank of $[\mathbf{t}]_x$?

Two-view geometry

Corresponding point in other image is constrained to lie on a line, called the *epipolar line*.



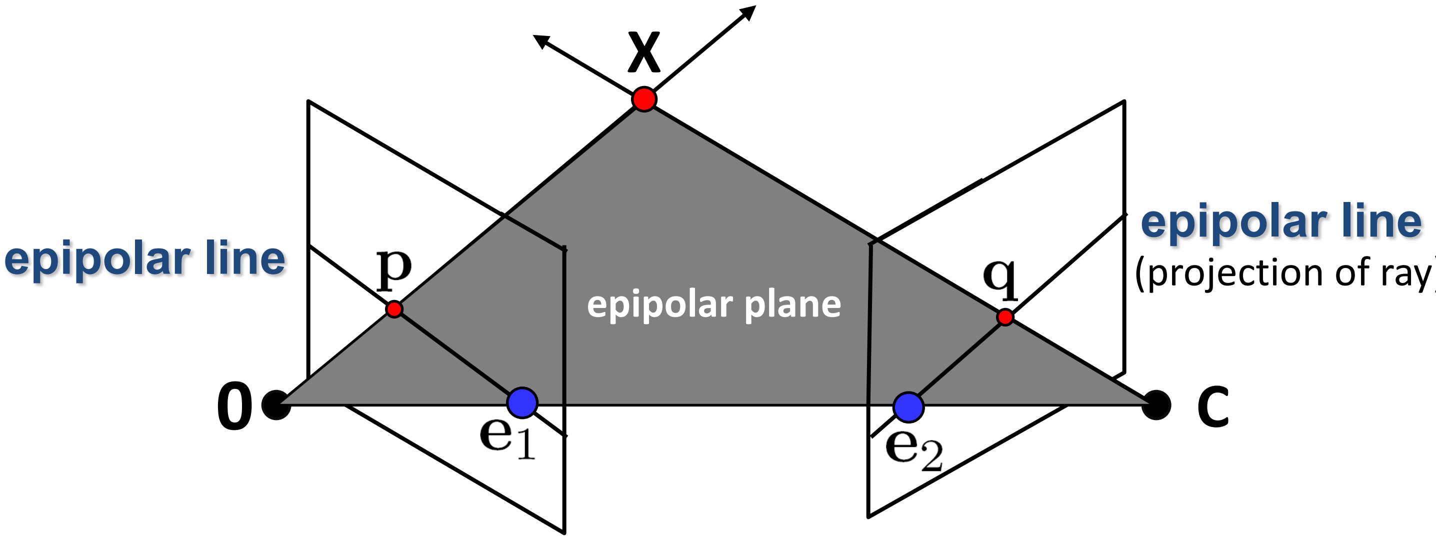
Essential matrix



- Assume calibrated cameras with $\mathbf{K}_1 = \mathbf{K}_2 = \mathbf{I}_{3x3}$.
- Let camera 1 be [**I**, **0**] and camera 2 be [**R**, **t**].
- In camera 1 coordinates, 3D point **X** is given by $\mathbf{X}_1 = \lambda_1 \mathbf{p}$.
- In camera 2 coordinates, 3D point **X** is given by $\mathbf{X}_2 = \lambda_2 \mathbf{q}$.
- Since camera 2 is related to camera 1 by rigid-body motion [**R**, **t**]

$$\mathbf{X}_2 = \mathbf{R}\mathbf{X}_1 + \mathbf{t}$$
$$\lambda_2 \mathbf{q} = \lambda_1 \mathbf{R}\mathbf{p} + \mathbf{t}$$

Essential matrix



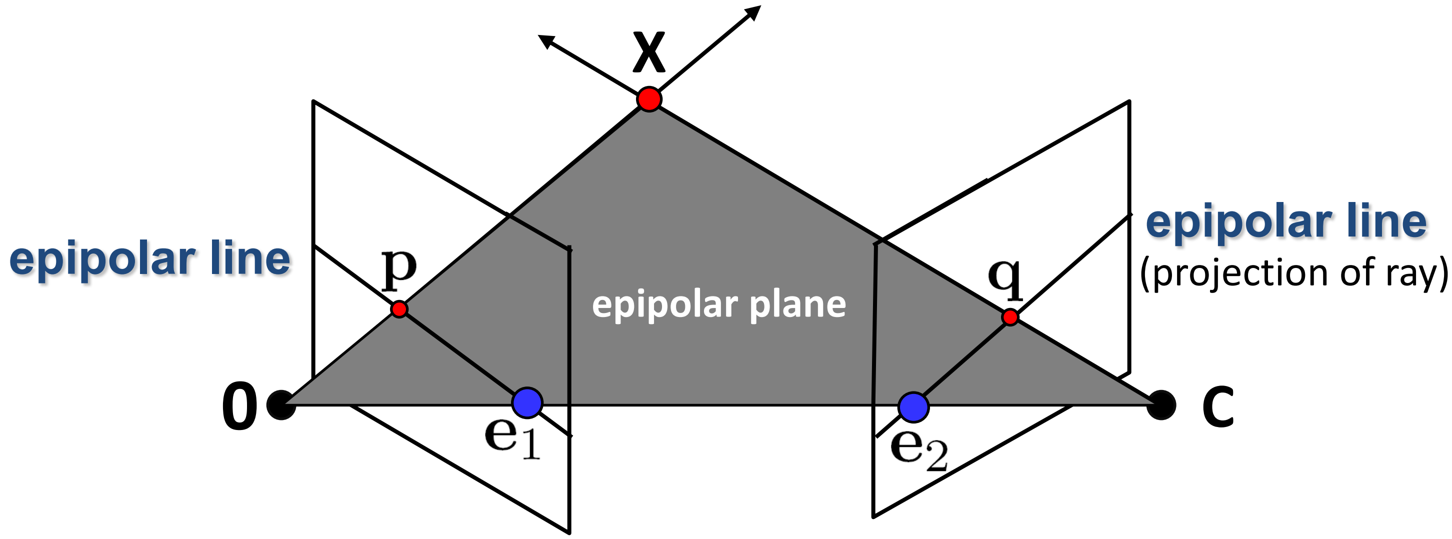
- We have: $\lambda_2 \mathbf{q} = \lambda_1 \mathbf{R}\mathbf{p} + \mathbf{t}$
- Take cross-product with respect to **t**:

$$\lambda_2 [\mathbf{t}]_\times \mathbf{q} = \lambda_1 [\mathbf{t}]_\times \mathbf{R}\mathbf{p}$$

- Take dot-product with respect to **q**:

$$0 = \lambda_1 \mathbf{q}^\top [\mathbf{t}]_\times \mathbf{R}\mathbf{p}$$

Essential matrix



- We have: $\mathbf{q}^\top [\mathbf{t}]_\times \mathbf{R}\mathbf{p} = 0$
- Define:

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$$

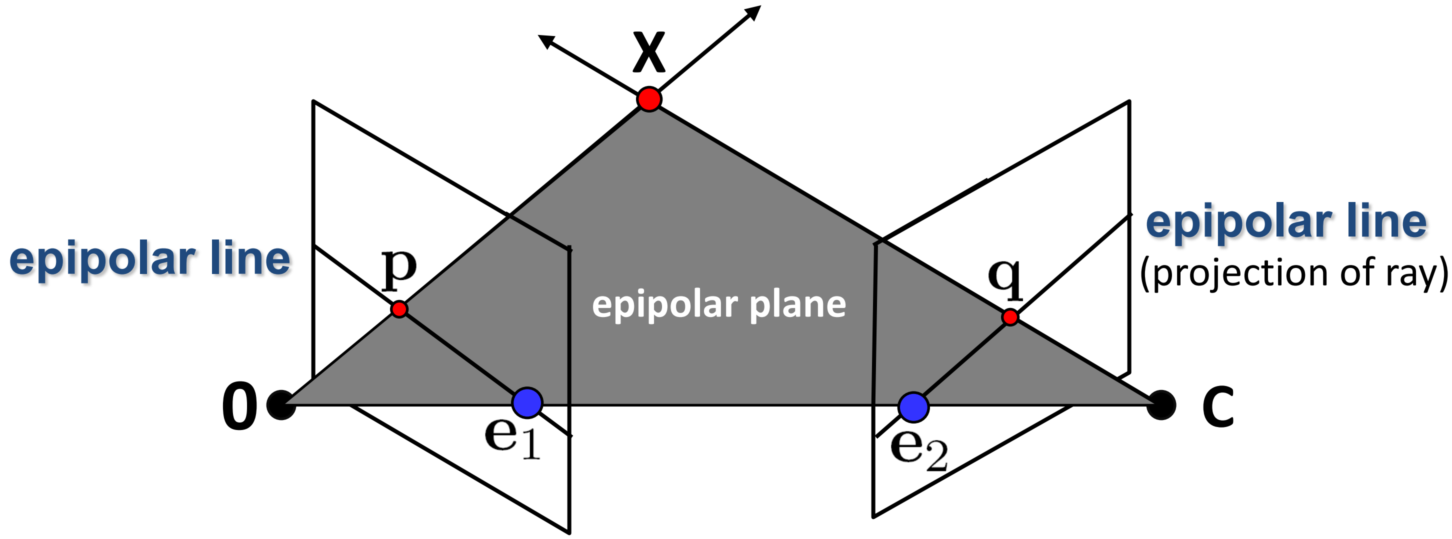
Essential matrix

- Then, we have:

$$\boxed{\mathbf{q}^\top \mathbf{E}\mathbf{p} = 0}$$

How many degrees of freedom does **E** have?

Fundamental matrix



- Relax the assumption of calibrated cameras.
- Then, **p** and **q** are in metric coordinates and pixel counterparts are:

$$\mathbf{p}' = \mathbf{K}_1 \mathbf{p} \qquad \mathbf{q}' = \mathbf{K}_2 \mathbf{q}$$

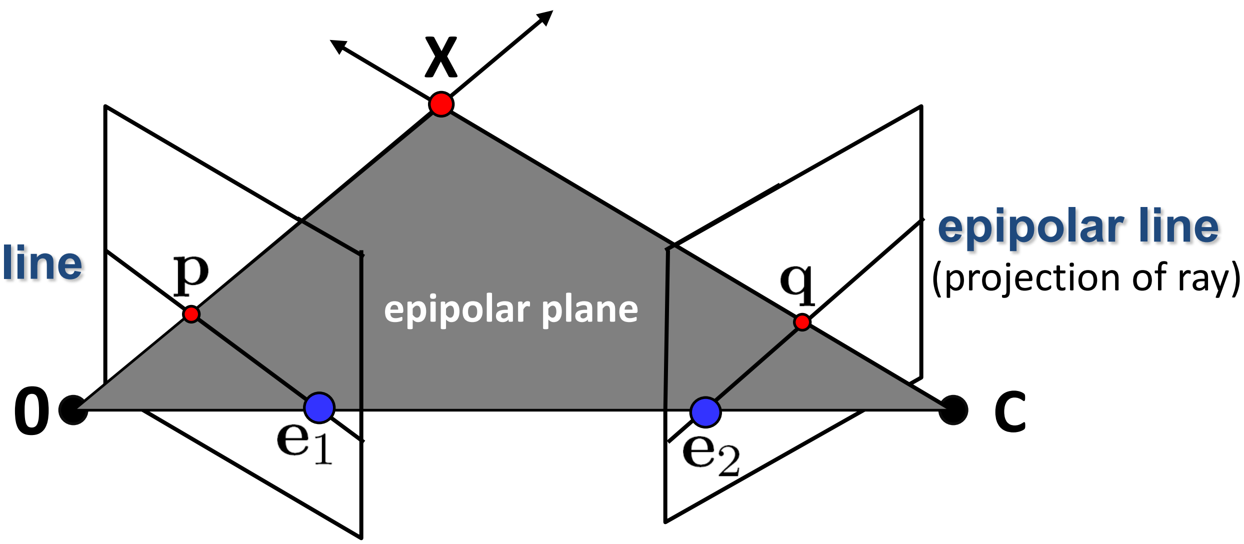
- Recall essential matrix constraint:

$$\mathbf{q}^\top \mathbf{E} \mathbf{p} = 0$$

- Substituting, we have:

$$(\mathbf{K}_2^{-1} \mathbf{q}')^\top \mathbf{E} (\mathbf{K}_1^{-1} \mathbf{p}') = 0$$

Fundamental matrix



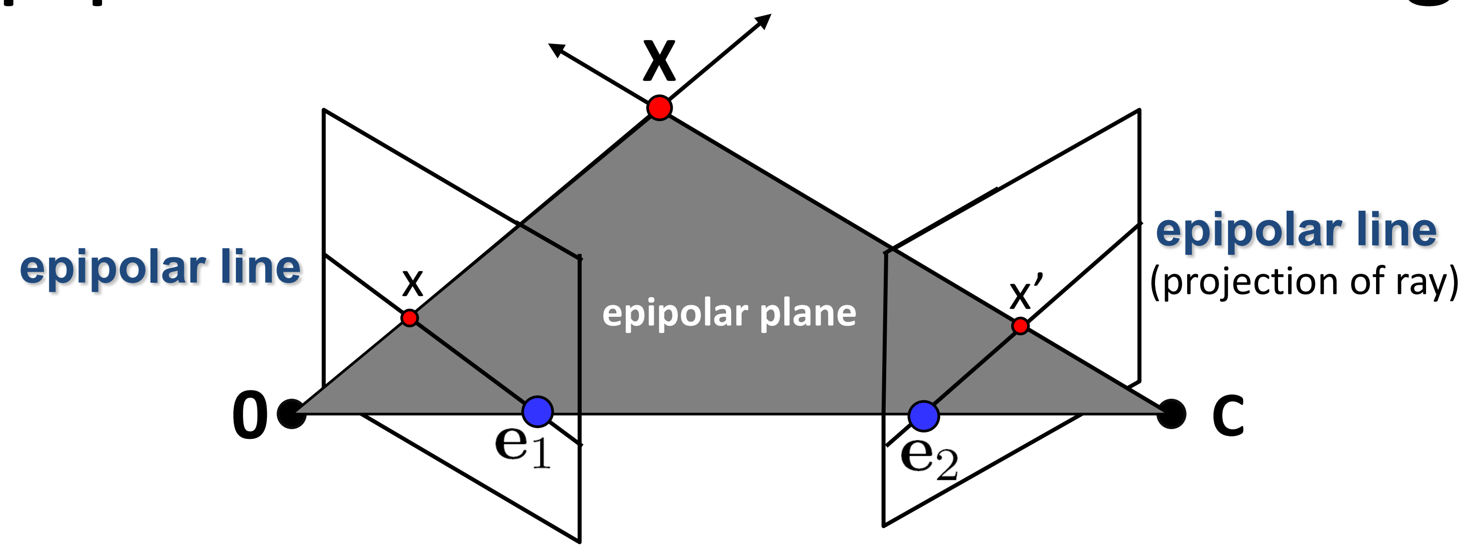
- Essential matrix constraint in pixel space: $(\mathbf{K}_2^{-1}\mathbf{q}')^\top \mathbf{E}(\mathbf{K}_1^{-1}\mathbf{p}') = 0$.
- Rearranging: $$\mathbf{q}'^\top(\mathbf{K}_2^{-\top}\mathbf{E}\mathbf{K}_1^{-1})\mathbf{p}' = 0$$
- Define: $\mathbf{F} = \mathbf{K}_2^{-\top}\mathbf{E}\mathbf{K}_1^{-1}$

Fundamental matrix

- Then, we have: $$\boxed{\mathbf{q}'^\top\mathbf{F}\mathbf{p}' = 0}$$

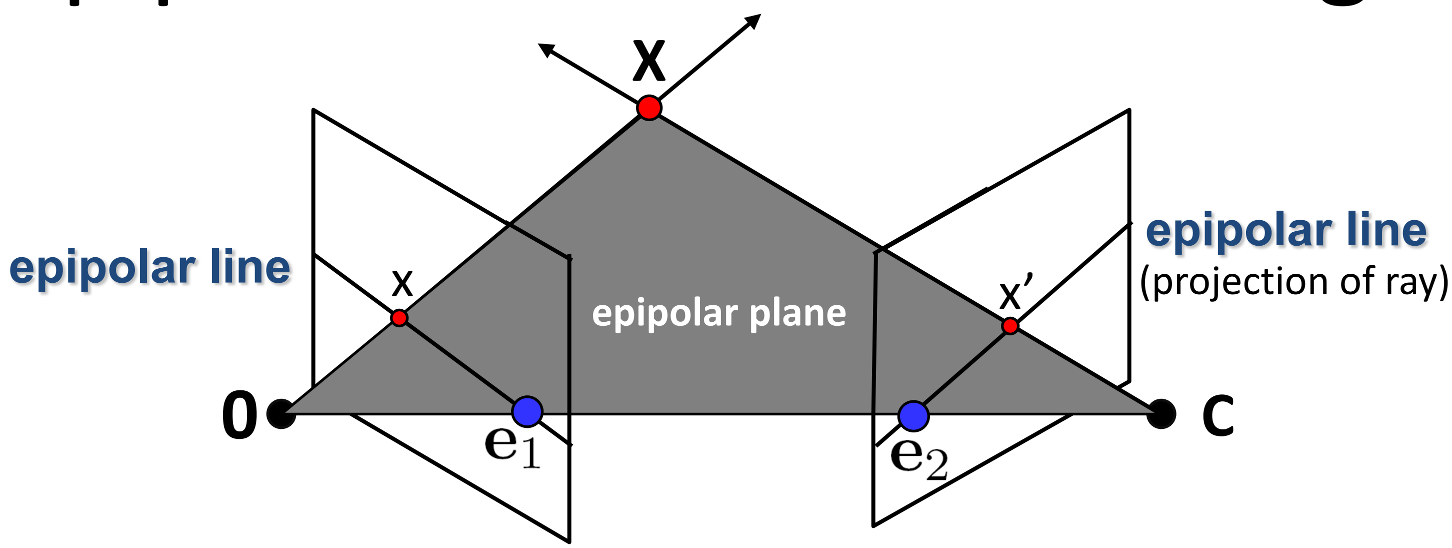
How many degrees of freedom does **F** have?

Epipolar line in the second image



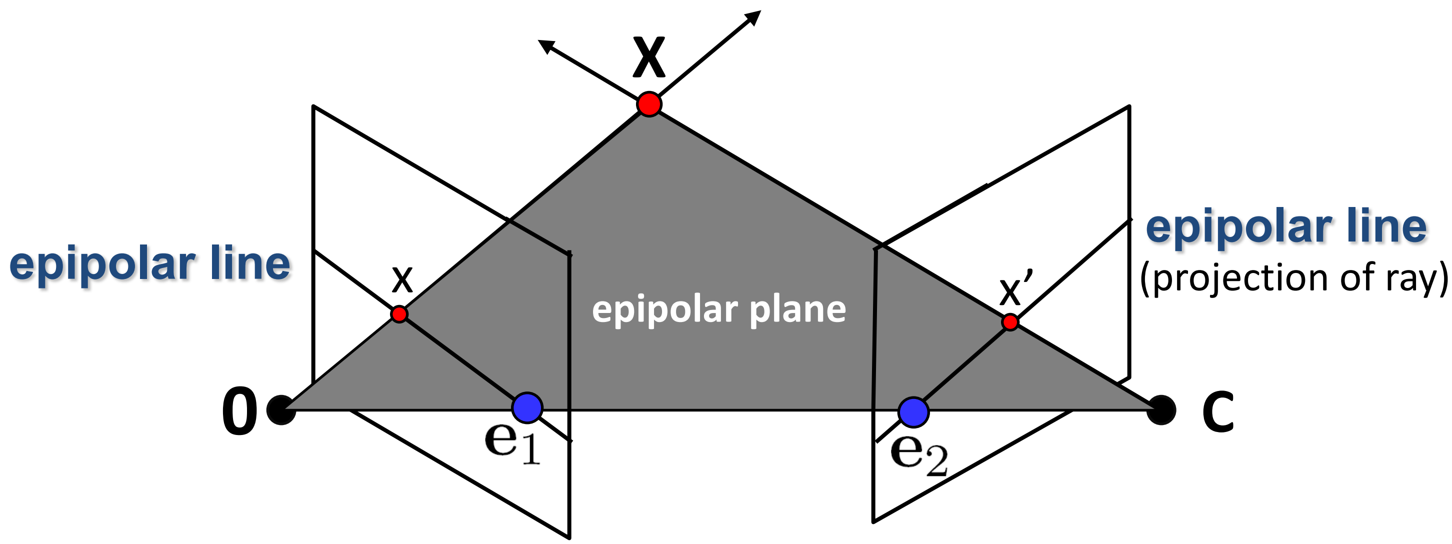
- For corresponding points **x** and **x'**, we have $\mathbf{x'}^T\mathbf{F}\mathbf{x} = 0$
- Define $\mathbf{l'} = \mathbf{F}\mathbf{x}$, then we have $\mathbf{x'}^T\mathbf{l'} = 0$
- Then, for point **x**, the line **Fx** contains corresponding point **x'**
- So, $\mathbf{l'} = \mathbf{F}\mathbf{x}$ is the epipolar line in the second image

Epipolar line in the first image



- For corresponding points **x** and **x'**, we have $\mathbf{x'}^T\mathbf{F}\mathbf{x} = 0$
- Taking transpose, it is the same as $\mathbf{x}^T\mathbf{F}^T\mathbf{x'} = 0$
- Define $\mathbf{l} = \mathbf{F}^T\mathbf{x'}$, then we have $\mathbf{x}^T\mathbf{l} = 0$
- Then, for point **x'**, the line $\mathbf{F}^T\mathbf{x'}$ contains corresponding point **x**
- So, $\mathbf{l} = \mathbf{F}^T\mathbf{x'}$ is the epipolar line in the first image

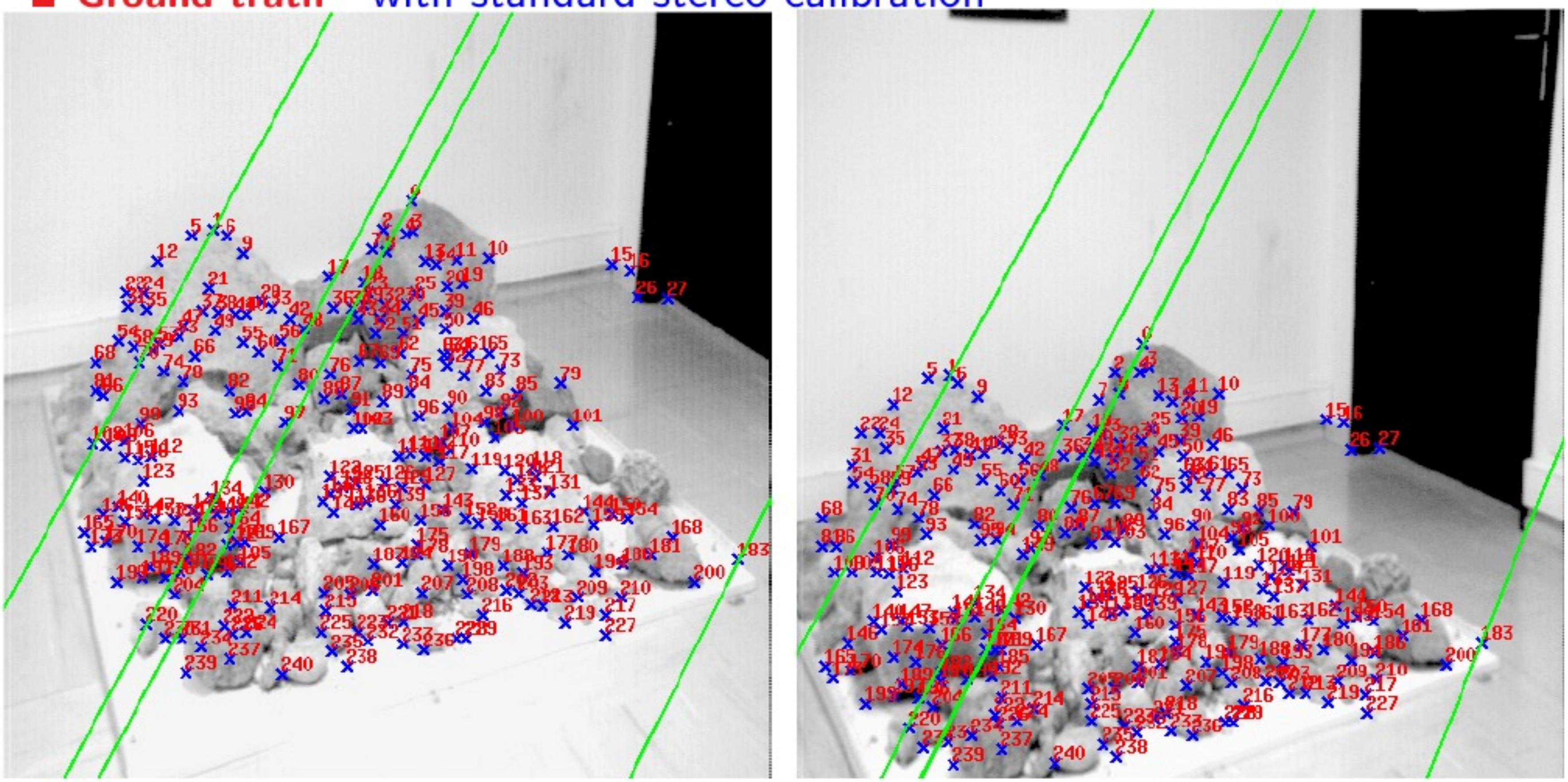
Properties of the fundamental matrix



- **Fx** is the epipolar line associated with **x**
- $\mathbf{F}^T\mathbf{x'}$ is the epipolar line associated with **x'**
- **F** is rank 2.

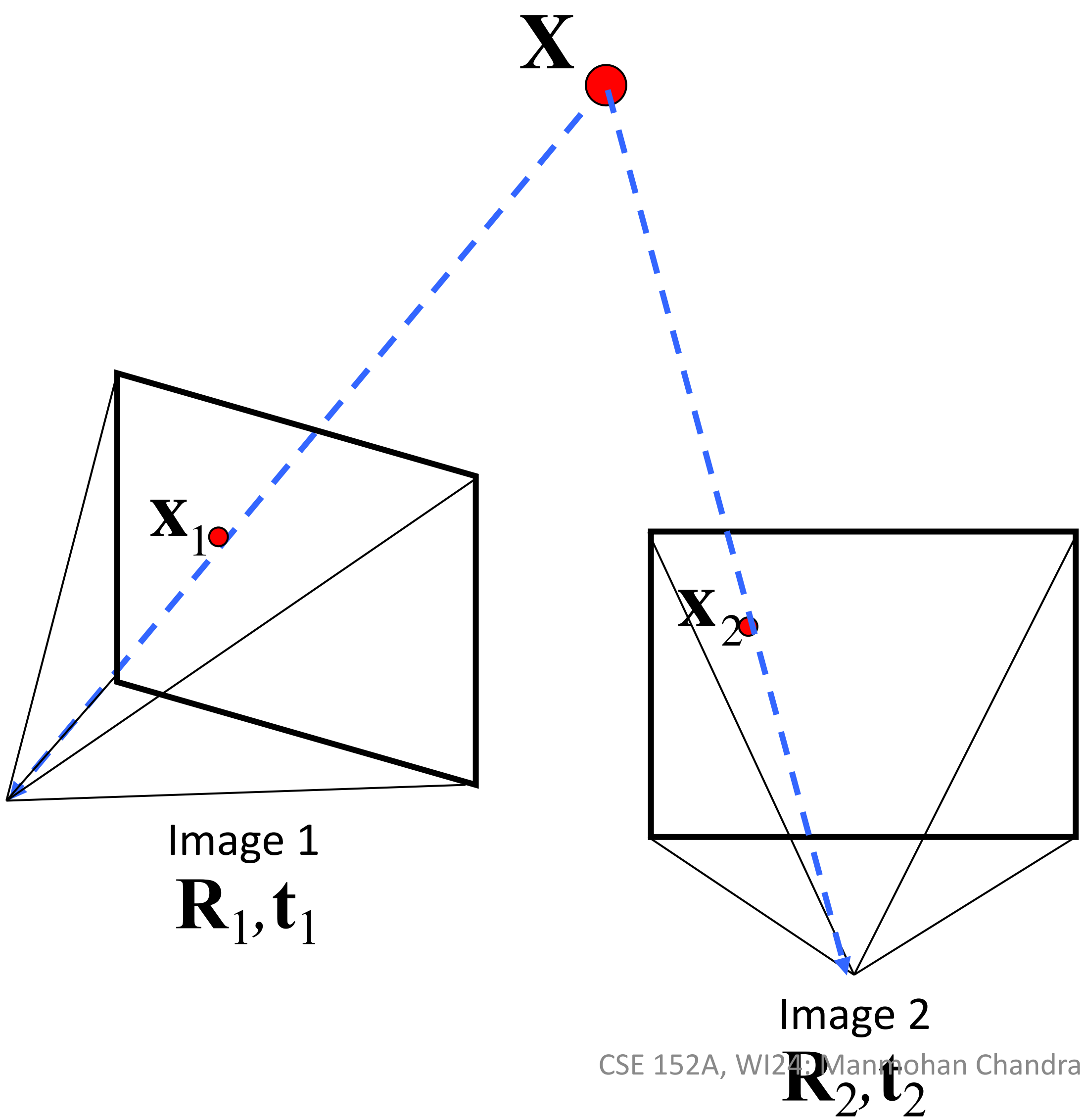
Results (ground truth)

- **Ground truth** with standard stereo calibration



Fundamental Matrix

Fundamental Matrix



$$\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$$

$$\mathbf{x}_1^T \mathbf{F} \mathbf{x}_2 = 0$$

Estimating **F**

- Given just the two images, can we estimate **F**?
- Yes, with enough correspondences.

Estimating F: 8-point algorithm

- The fundamental matrix F is defined by

$$\mathbf{x}'^{\mathrm{T}}\mathbf{F}\mathbf{x} = 0$$

for any pair of matches x and x' in two images.

- Let **x** = $(u,v,1)^{\mathrm{T}}$ and **x'** = $(u',v',1)^{\mathrm{T}}$, $\mathbf{F} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}$

- Each match gives a linear equation:

$$uu'f_{11} + vu'f_{12} + u'f_{13} + uv'f_{21} + vv'f_{22} + v'f_{23} + uf_{31} + vf_{32} + f_{33} = 0$$

8-point algorithm

Given *n* point correspondences, set up a system of equations:

$$\begin{bmatrix} u_1u_1' & v_1u_1' & u_1' & u_1v_1' & v_1v_1' & v_1' & u_1 & v_1 & 1 \\ u_2u_2' & v_2u_2' & u_2' & u_2v_2' & v_2v_2' & v_2' & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_n' & v_nu_n' & u_n' & u_nv_n' & v_nv_n' & v_n' & u_n & v_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

- In reality, instead of solving $\mathbf{Af} = 0$, we seek $\mathbf{f}$ to minimize $\|\mathbf{Af}\|$.

Solving homogeneous systems

- In reality, instead of solving $\mathbf{Af} = 0$, we seek $\mathbf{f}$ to minimize $\|\mathbf{Af}\|$.
- Singular value decomposition:

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

$$\mathbf{U}, \mathbf{V} \text{ are rotation matrices}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} s_1 & & \\ & \ddots & \\ & & s_n \end{bmatrix}$$

- Solution **f** given by the last column of **V**.

8-point algorithm: Problem?

- **F** should have rank 2
- To enforce that **F** is of rank 2, **F** is replaced by **F'** that minimizes $\|\mathbf{F}^\top \mathbf{F}'\|$ subject to the rank constraint.

- This is achieved by SVD. Let $\mathbf{F} = \mathbf{U\Sigma V}^{\mathrm{T}}$, where

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix}. \text{ Let } \Sigma' = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

then $\mathbf{F}' = \mathbf{U}\Sigma'\mathbf{V}^{\mathrm{T}}$ is the solution.

8-point algorithm

```
% Normalization on 2D points (advanced concept, implemented for you)

% Build the constraint matrix
   A = [x2(1,:)’.*x1(1,:)'   x2(1,:)'.*x1(2,:)'  x2(1,:)' ...
        x2(2,:)'.*x1(1,:)'   x2(2,:)'.*x1(2,:)'  x2(2,:)' ...
        x1(1,:)'             x1(2,:)'            ones(npts,1) ];

   [U,D,V] = svd(A);

% Extract fundamental matrix from the column of V
% corresponding to the smallest singular value.
   F = reshape(V(:,9),3,3)';

% Enforce rank 2 constraint
   [U,D,V] = svd(F);
   F = U * diag([D(1,1) D(2,2) 0]) * V’;

% Do the reverse normalization on 2D points
```

8-point algorithm

- Pros: it is linear, easy to implement and fast
- Cons: susceptible to noise

Motion from correspondences

- Use 8-point algorithm to estimate F
- Get E from F:

$$\mathbf{F} = \mathbf{K}_2^{-\top} \mathbf{E} \mathbf{K}_1^{-1}$$

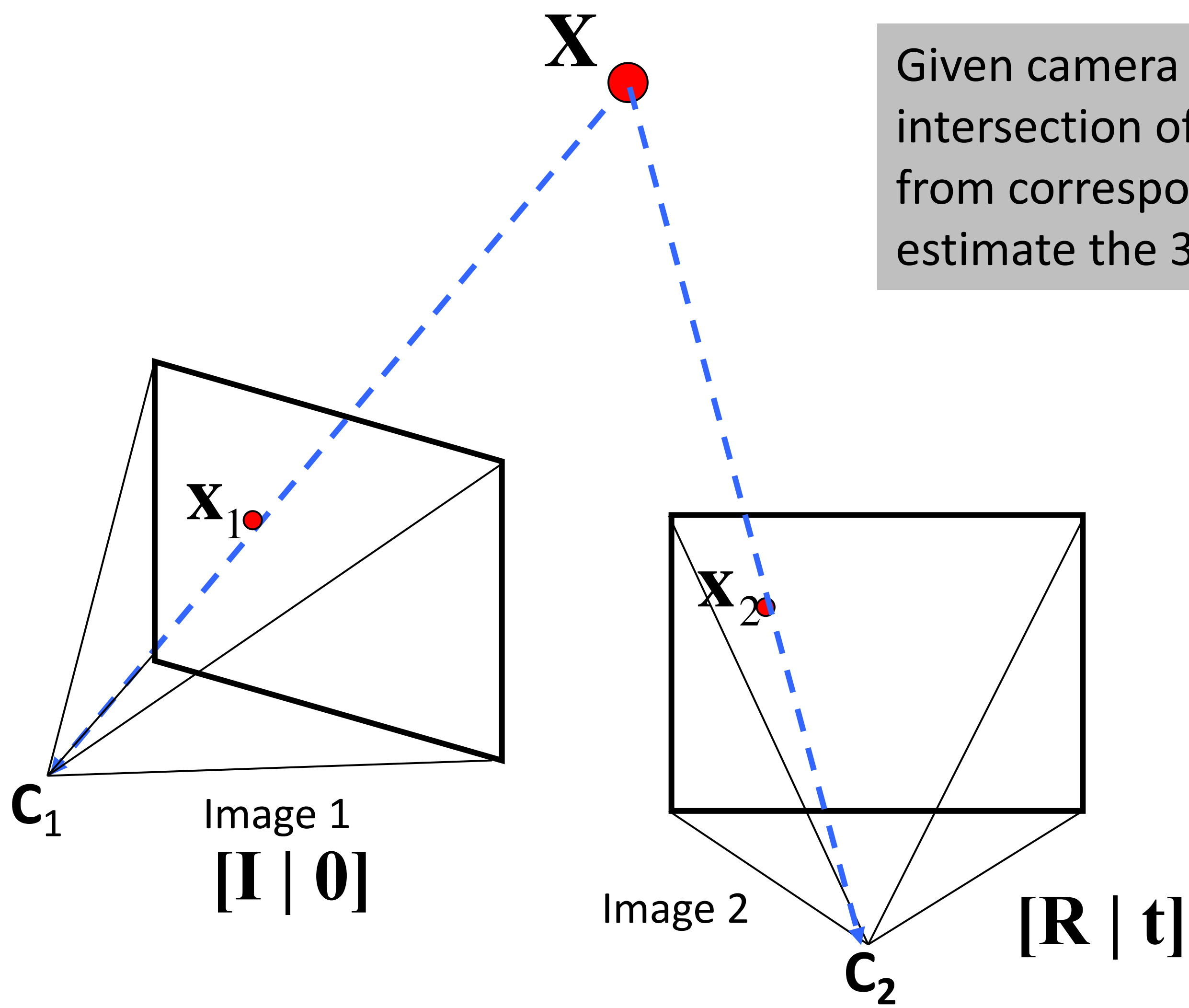
$$\mathbf{E} = \mathbf{K}_2^{\top} \mathbf{F} \mathbf{K}_1$$

- Decompose E into skew-symmetric and rotation matrices:

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$$

Can estimate rotation and translation from **E**

Triangulation



Given camera motion [R | t], can find intersection of back-projected rays from corresponding 2D points to estimate the 3D points

First ray: $\mathbf{C}_1 + k_1\mathbf{x}_1^w$
Second ray: $\mathbf{C}_2 + k_2\mathbf{x}_2^w$