Lecture 08
Multiple View Geometry 2

Prof. Dr. Davide Scaramuzza

sdavide@ifi.uzh.ch

Course Topics

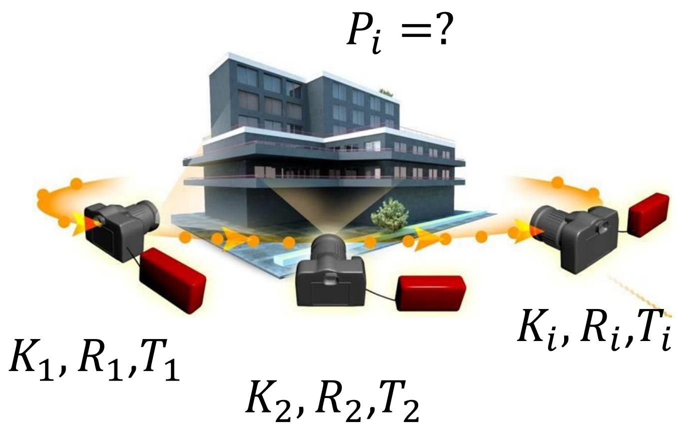
- Principles of image formation
- Image filtering
- Feature detection
- Multi-view geometry
- 3D Reconstruction
- Recognition

Multiple View Geometry

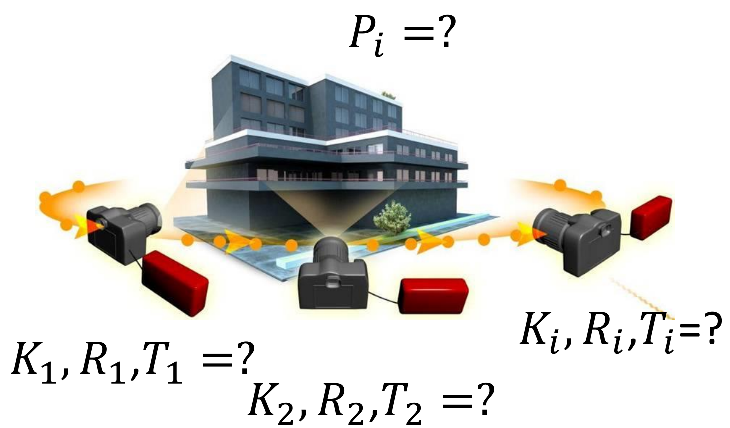
San Marco square, Venice
14,079 images, 4,515,157 points

Multiple View Geometry

- **3D reconstruction from multiple views:**
  - **Assumptions:** K, T and R are known.
  - **Goal**: Recover the 3D structure from images



- **Structure From Motion:**
  - **Assumptions**: none (K, T, and R are unknown).
  - **Goal**: Recover simultaneously 3D scene structure and camera poses (up to scale) from multiple images



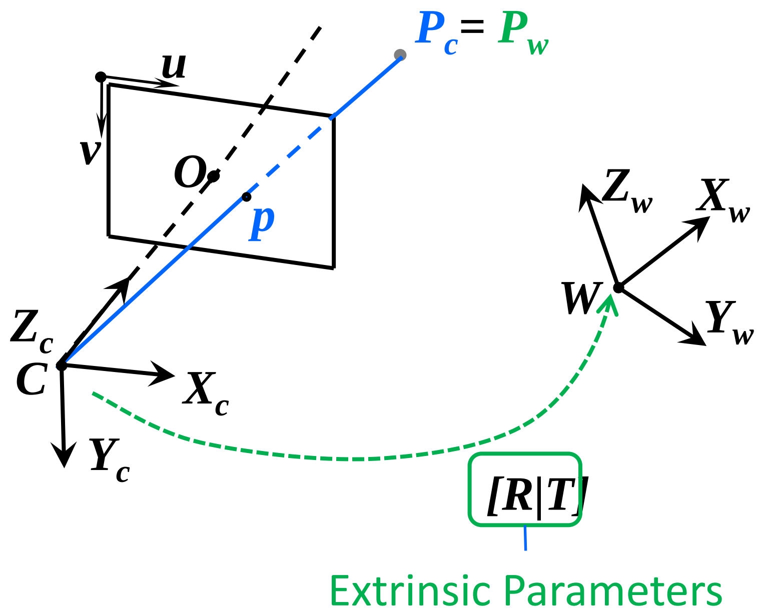
Review: Perspective Projection

Perspective Projection Equation

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|T] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \Rightarrow \lambda p = MP$$

Normalized image coordinates

$$\begin{bmatrix} \bar{u} \\ \bar{v} \\ 1 \end{bmatrix} = K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$



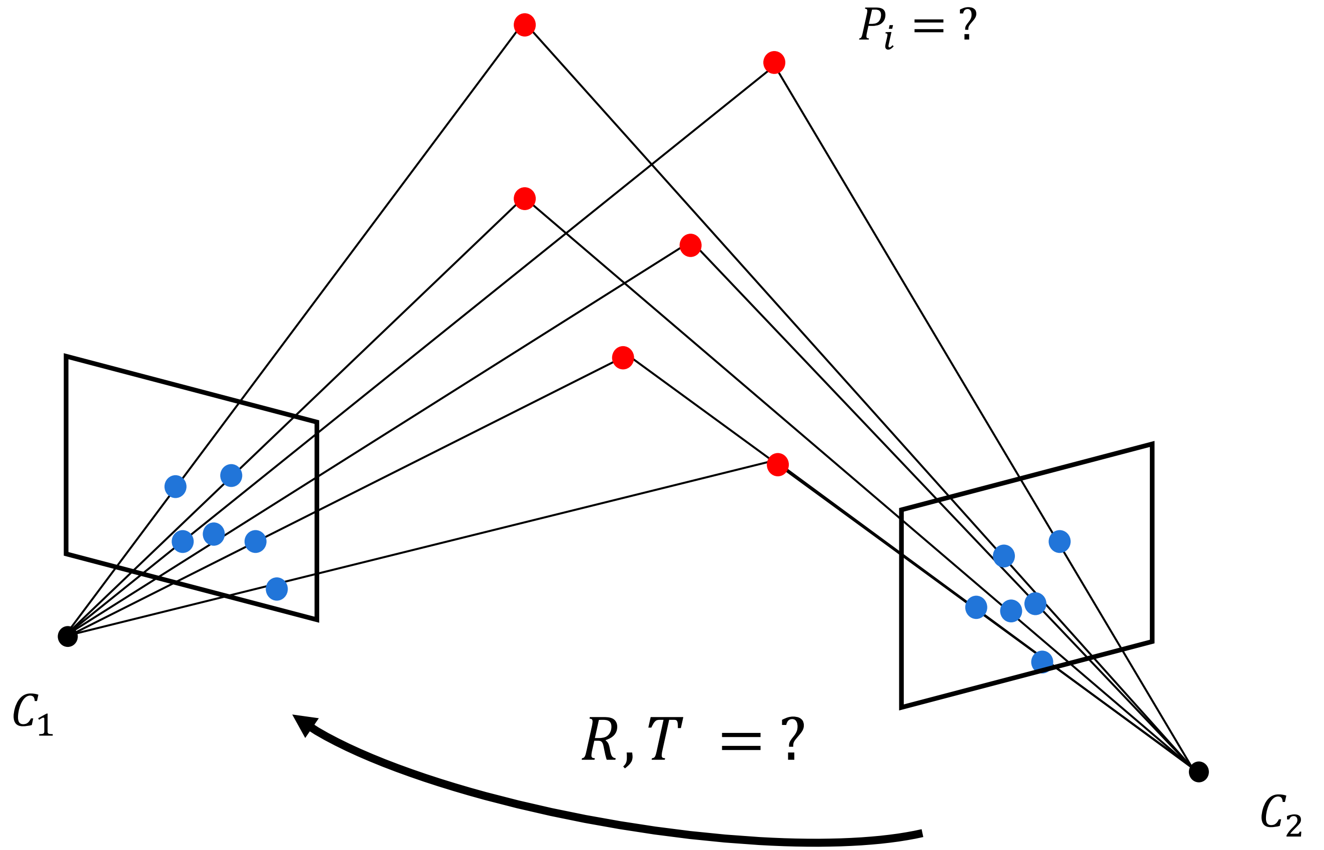
Today's outline

- Structure from Motion

Structure from Motion (SFM)

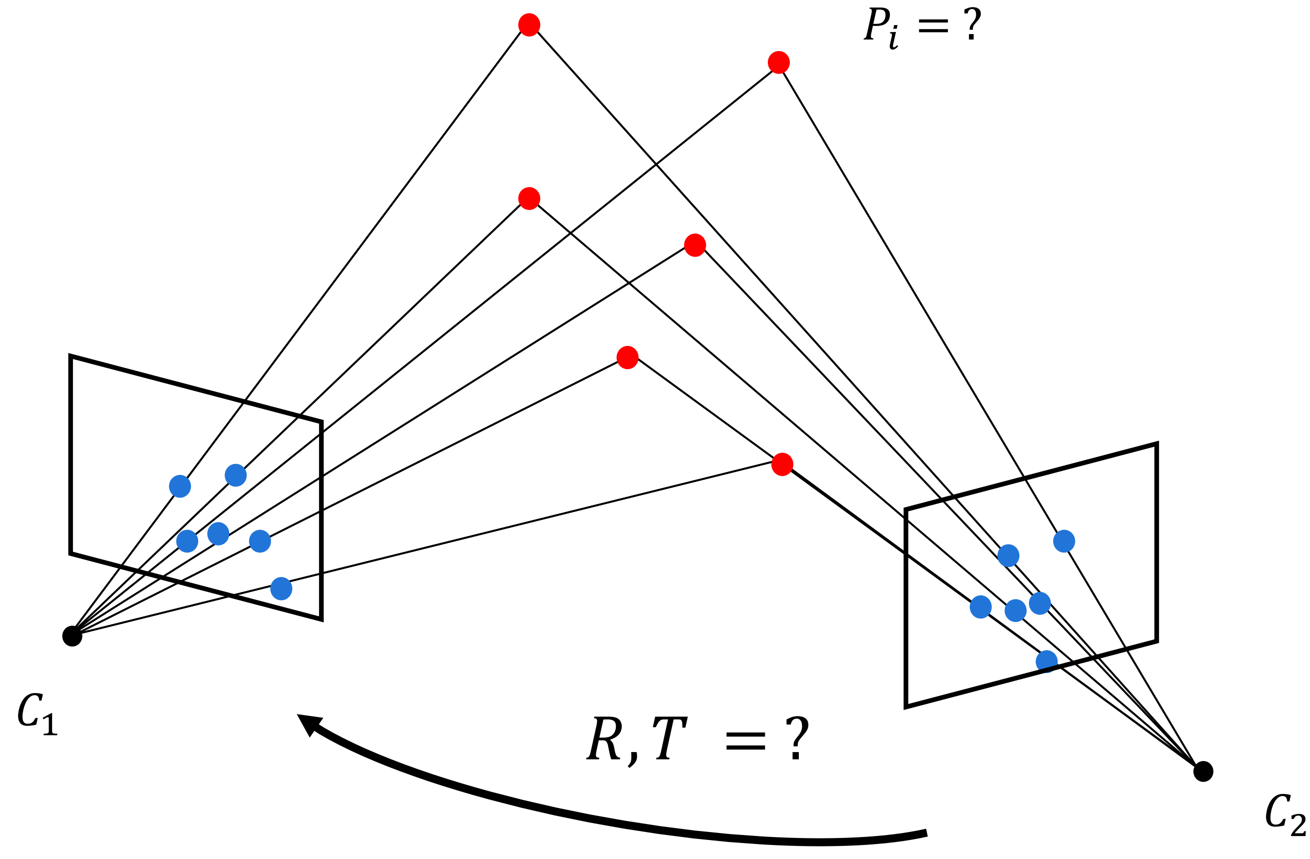
- **Problem formulation:** Given n points in *correspondence* across two images, $\{(u^i{}_1, v^i{}_1), (u^i{}_2, v^i{}_2)\}$, simultaneously compute the 3D location $\boldsymbol{P}_i$, the camera relative-motion parameters $(\boldsymbol{R}, \boldsymbol{t})$, and camera intrinsic $\boldsymbol{K}_{1,2}$ that satisfy

$$\begin{cases} \lambda_1 \begin{bmatrix} u^i{}_1 \\ v^i{}_1 \\ 1 \end{bmatrix} = K_1 [I|0] \cdot \begin{bmatrix} X^i{}_w \\ Y^i{}_w \\ Z^i{}_w \\ 1 \end{bmatrix} \\ \lambda_2 \begin{bmatrix} u^i{}_2 \\ v^i{}_2 \\ 1 \end{bmatrix} = K_2 [R|T] \cdot \begin{bmatrix} X^i{}_w \\ Y^i{}_w \\ Z^i{}_w \\ 1 \end{bmatrix} \end{cases}$$



Structure from Motion (SFM)

- Two variants exist:
  - **Uncalibrated** camera(s) -> K is unknown
  - **Calibrated** camera(s) -> K is known

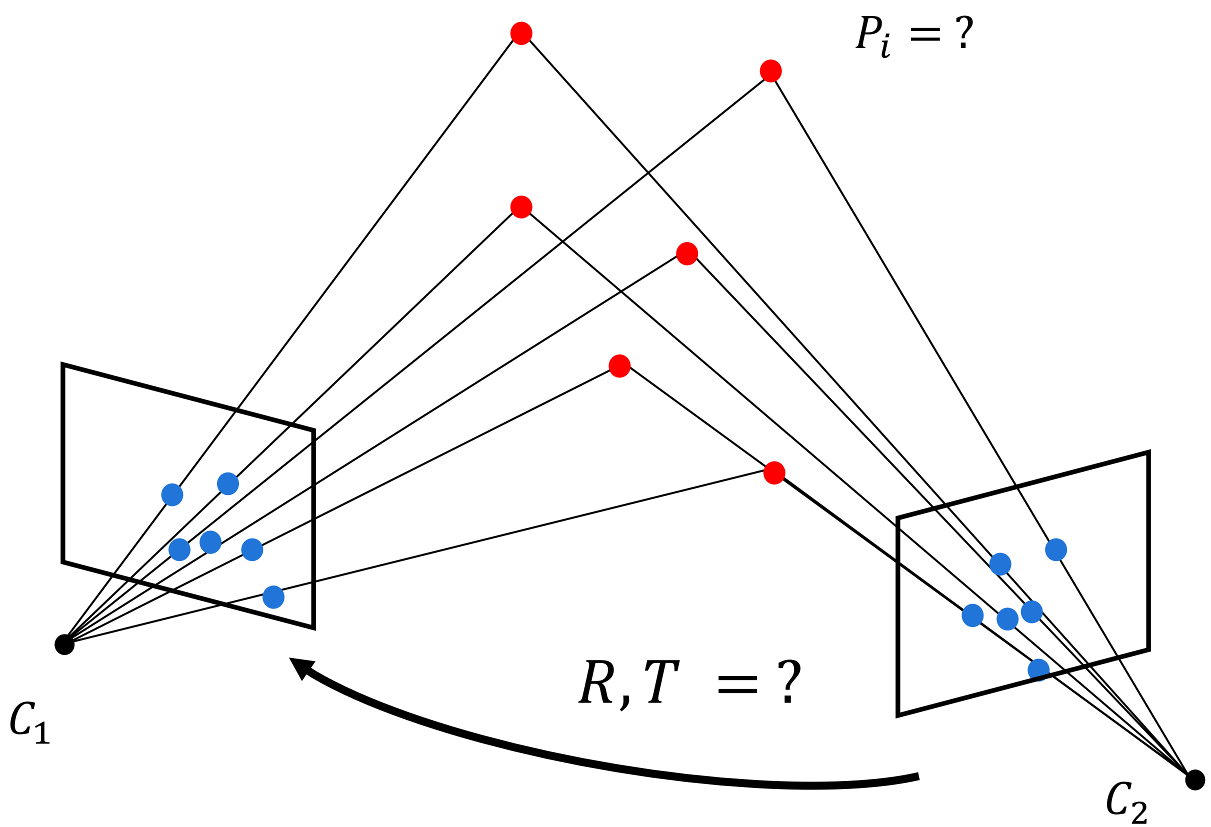


Structure from Motion (SFM)

- Let's study the case in which the camera(s) is «calibrated»
- For convenience, let's use *normalized image coordinates*
- Thus, we want to find R, T, P_i that satisfy

$$\begin{bmatrix} \bar{u} \\ \bar{v} \\ 1 \end{bmatrix} = K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$\begin{cases} \lambda_1 \begin{bmatrix} \bar{u}^i{}_1 \\ \bar{v}^i{}_1 \\ 1 \end{bmatrix} = [I|0] \cdot \begin{bmatrix} X^i{}_w \\ Y^i{}_w \\ Z^i{}_w \\ 1 \end{bmatrix} \\ \lambda_2 \begin{bmatrix} \bar{u}^i{}_2 \\ \bar{v}^i{}_2 \\ 1 \end{bmatrix} = [R|T] \cdot \begin{bmatrix} X^i{}_w \\ Y^i{}_w \\ Z^i{}_w \\ 1 \end{bmatrix} \end{cases}$$

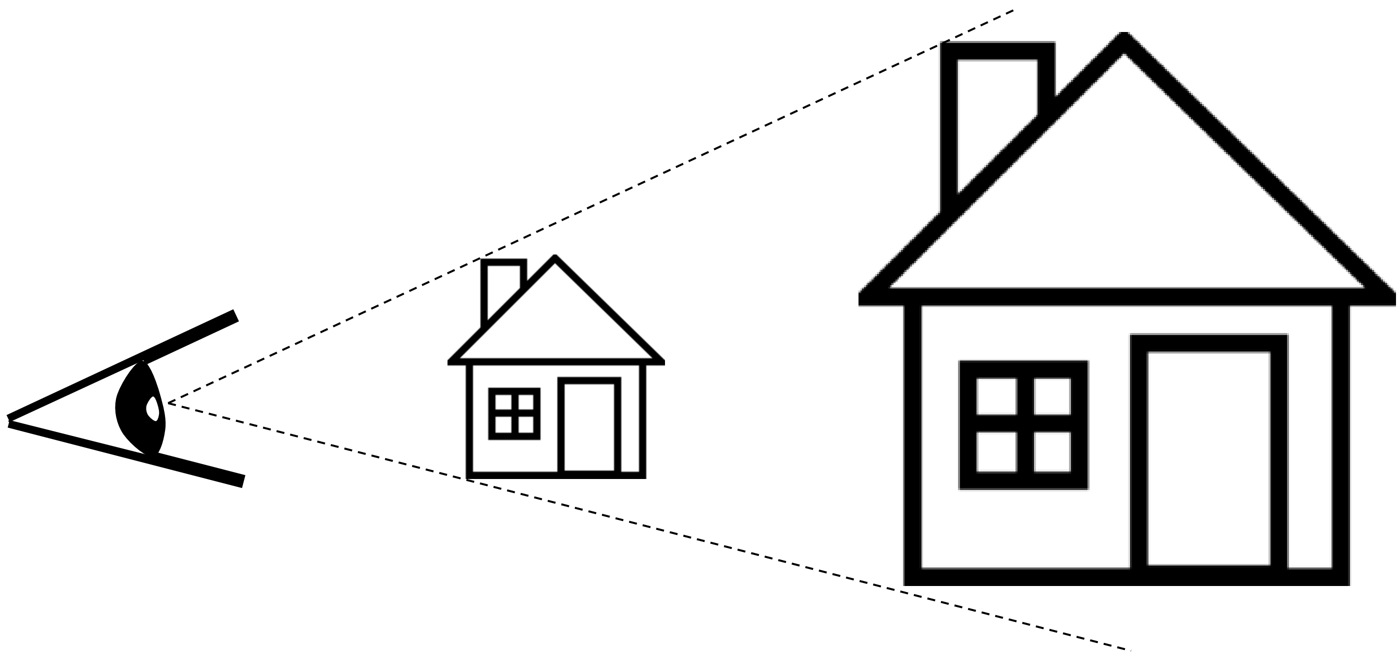


Scale Ambiguity

- With a single camera, we only know the relative scale
- No information about the *metric scale*

Scale Ambiguity

- With a single camera, we only know the relative scale
- No information about the *metric scale*
- If we scale the entire scene by some factor s, the projections of the scene points in the image remain exactly the same:



Scale Ambiguity

- In monocular vision, it is **impossible** to recover the absolute scale of the scene!
  - Stereo vision?
- Thus, only **5 degrees of freedom** are measurable:
  - **3** parameters to describe the **rotation**
  - **2** parameters for the **translation up to a scale** (we can only compute the direction of translation but not its length)

Structure from Motion (SfM)

- How many knowns and unknowns?
  - $\mathbf{4n}$ **knowns:**
    - n correspondences; each one $(u^i{}_1, v^i{}_1)$ and $(u^i{}_2, v^i{}_2)$, $i = 1 \dots n$
  - $\mathbf{5 + 3n}$ **unknowns**
    - 5 for the motion up to a scale (rotation-> 3, translation->2)
    - $3n =$ number of coordinates of the n 3D points

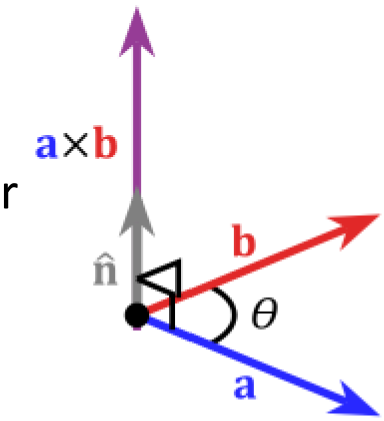
- Does a solution exist?
  - If and only if

    number of independent equations $\geq$ number of unknowns

    $\Rightarrow 4n \geq 5 + 3n \Rightarrow \mathbf{n \geq 5}$

Cross Product (or Vector Product)

$$\vec{a} \times \vec{b} = \vec{c}$$



- Vector cross product takes two vectors and returns a third vector that is perpendicular to both inputs

$$\vec{a} \cdot \vec{c} = 0$$

$$\vec{b} \cdot \vec{c} = 0$$

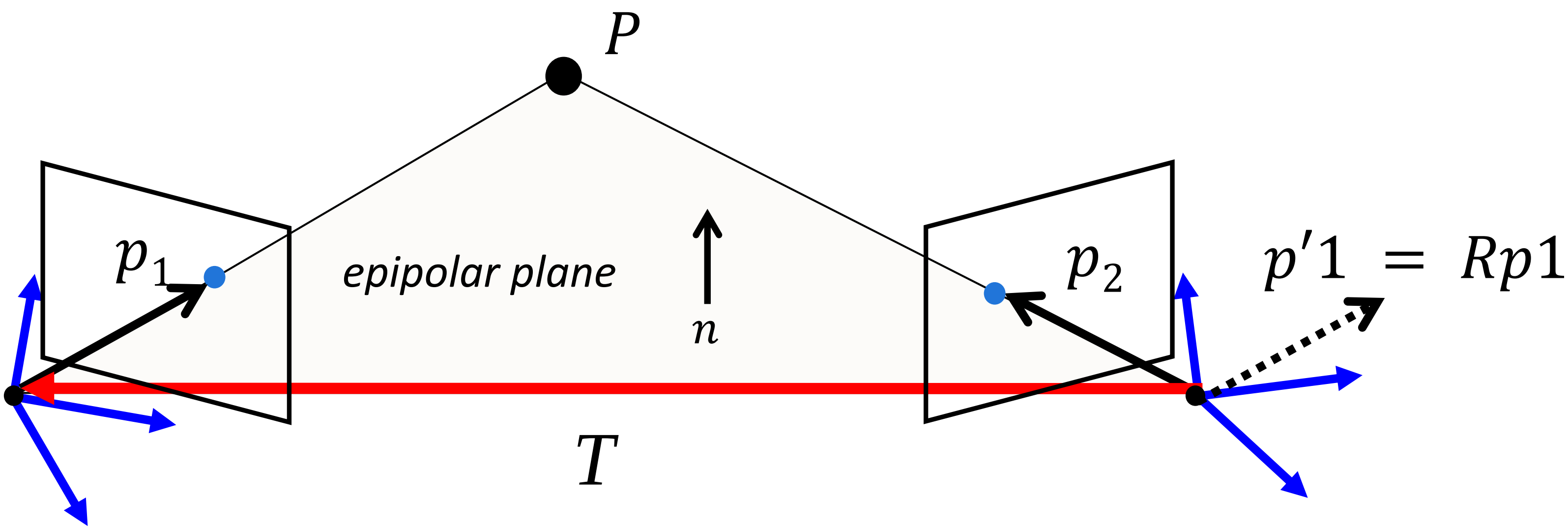
- So here, $\boldsymbol{c}$ is perpendicular to both $\boldsymbol{a}$ and $\boldsymbol{b}$, which means the dot product = 0
- Also, recall that the cross product of two parallel vectors = 0

- The **cross product** between **a** and **b** can also be expressed in matrix form as the product between the **skew-symmetric matrix** of **a** and a vector **b**

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a}_\times]\mathbf{b}$$

Epipolar Geometry

$$p_1 = \begin{bmatrix} \bar{u}_1 \\ \bar{v}_1 \\ 1 \end{bmatrix} \qquad p_2 = \begin{bmatrix} \bar{u}_2 \\ \bar{v}_2 \\ 1 \end{bmatrix}$$



p_1, p_2, T are coplanar:

$$p_2^T \cdot n = 0 \Rightarrow p_2^T \cdot (T \times p_1') = 0 \Rightarrow p_2^T \cdot (T \times (Rp_1)) = 0$$

$$\Rightarrow p_2^T \, [T]_\times R \, p_1 = 0 \Rightarrow p_2^T \, E \, p_1 = 0 \quad \textit{epipolar constraint}$$

$$\mathrm{E} = [T]_\times R \quad \textit{essential matrix}$$

Epipolar Geometry

$$p_1 = \begin{bmatrix} \bar{u}_1 \\ \bar{v}_1 \\ 1 \end{bmatrix} \quad p_2 = \begin{bmatrix} \bar{u}_2 \\ \bar{v}_2 \\ 1 \end{bmatrix}$$ *Normalized image coordinates*

$$p_2^T \, E \, p_1 = 0$$ *Epipolar constraint or Longuet-Higgins equation*

$$\mathrm{E} = [T]_\times R$$ *Essential matrix*

- The Essential Matrix can be computed from 5 image correspondences **[Kruppa, 1913]**. The more points, the higher accuracy in *presence of noise*
- The Essential Matrix can be decomposed into R and T recalling that $\mathrm{E} = [T]_\times R$ Four distinct solutions for R and T are possible.

H. Christopher Longuet-Higgins (September 1981). "A computer algorithm for reconstructing a scene from two projections". Nature **293** (5828): 133–135. PDF.

How to compute the Essential Matrix?

- The Essential Matrix can be computed from 5 image correspondences **[Kruppa, 1913]**. However, this solution is not simple. It took almost one century until an efficient solution was found! [Nister, CVPR'2004]

- The first popular solution uses 8 points and is called 8-point algorithm
  Longuet Higgins. *A computer algorithm for reconstructing a scene from two projections*. Nature (1981)

The eight-point algorithm

$$\boldsymbol{p_1} = (\bar{u}_1, \bar{v}_1, 1)^T, \quad \boldsymbol{p_2} = (\bar{u}_2, \bar{v}_2, 1) \qquad p_2^T \, E \, p_1 = 0$$

$$\begin{bmatrix} \bar{u}_2 & \bar{v}_2 & 1 \end{bmatrix} \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \begin{bmatrix} \bar{u}_1 \\ \bar{v}_1 \\ 1 \end{bmatrix} = 0 \;\Rightarrow\; \underbrace{\begin{bmatrix} u_2 u_1 & u_2 v_1 & u_2 & v_2 u_1 & v_2 v_1 & v_2 & u_1 & v_1 & 1 \end{bmatrix}}_{Q} \underbrace{\begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{bmatrix}}_{E} = 0$$

Q (this matrix is **known**)

E (this matrix is **unknown**)

Minimize:

$$\left| Q \cdot E \right|^2$$

under the constraint

$\|\boldsymbol{E}\|^2 = 1$

For $n = 8$ points, a unique solution exists if the points are not coplanar. For $n > 8$ non-coplanar points, a linear least-square solution is given by the eigenvector of Q corresponding to its smallest eigenvalue (which is the unit vector that minimizes $\left| Q \cdot E \right|^2$). It can be done using Singular Value Decomposition.

8-point algorithm: Matlab code

```
function F = calibrated_eightpoint( p1, p2)

p1 = p1'; % 3xN vector; each column = [u;v;1]
p2 = p2'; % 3xN vector; each column = [u;v;1]

Q = [p1(:,1).*p2(:,1) , ...
    p1(:,2).*p2(:,1) , ...
    p1(:,3).*p2(:,1) , ...
    p1(:,1).*p2(:,2) , ...
    p1(:,2).*p2(:,2) , ...
    p1(:,3).*p2(:,2) , ...
    p1(:,1).*p2(:,3) , ...
    p1(:,2).*p2(:,3) , ...
    p1(:,3).*p2(:,3) ] ;

[U,S,V] = svd(Q);
F = V(:,9);

F = reshape(V(:,9),3,3)';
```

The eight-point algorithm

Meaning of the linear least-square error $\sum_{i=1}^{N} (p^{i\,T}_{2} \boldsymbol{E}\, p^{i}_{1})^2$:

Using the definition of dot product, it can be observed that

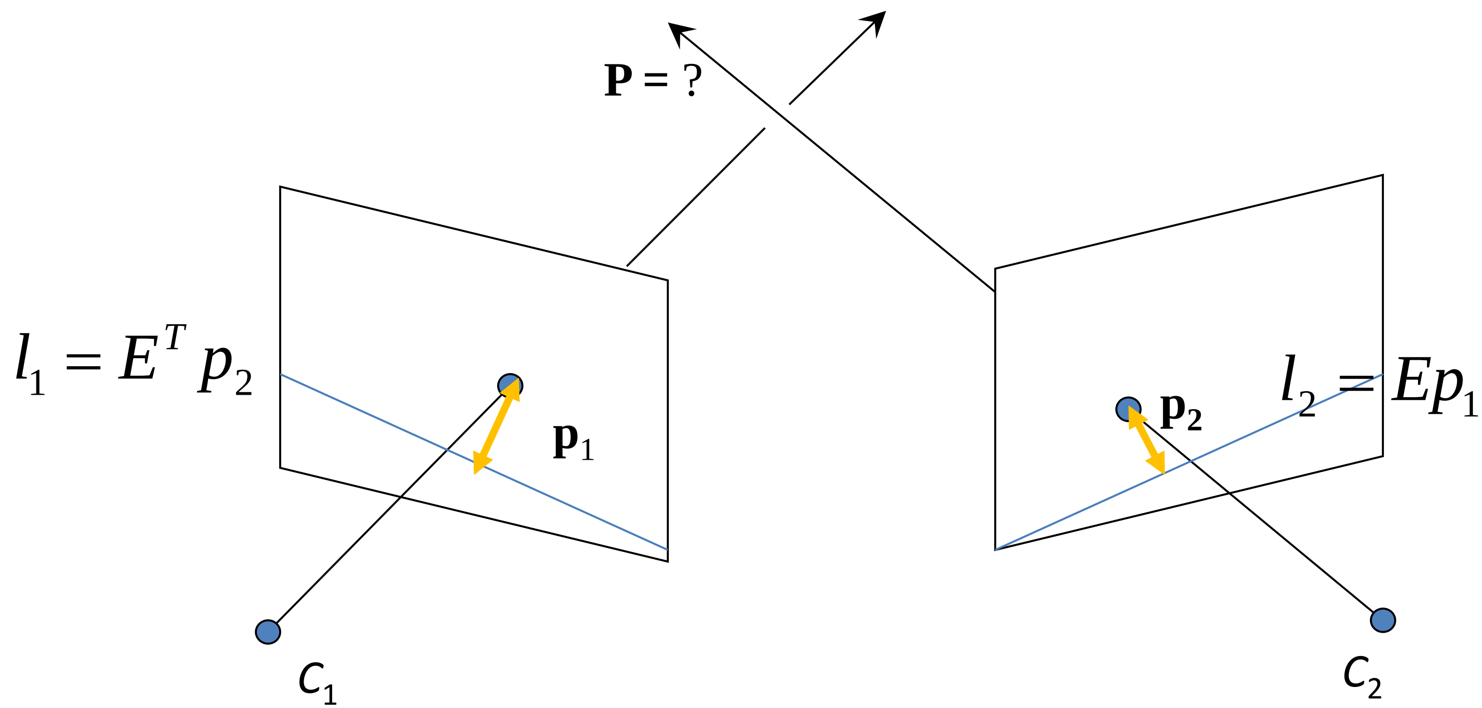
$$\boldsymbol{p}^T{}_1 \cdot \boldsymbol{E}\boldsymbol{p}_2 = \|\boldsymbol{p}^T{}_1\| \|\boldsymbol{E}\boldsymbol{p}_2\| \cos(\theta)$$

It can be observed that this product is non zero when, $\boldsymbol{p}^T{}_1$, $\boldsymbol{p}_2$, and $\boldsymbol{T}$ are not coplanar.

The eight-point algorithm

Nonlinear approach: minimize sum of squared *epipolar* **distances**

$$\sum_{i=1}^{N}\left[\mathrm{d}^2(p^i{}_2, l_2) + \mathrm{d}^2(p^i{}_1, l_1)\right]$$



Problem with eight-point algorithm

$$\begin{bmatrix} u_2u_1 & u_2v_1 & u_2 & v_2u_1 & v_2v_1 & v_2 & u_1 & v_1 & 1 \end{bmatrix} \begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{bmatrix} = 0$$

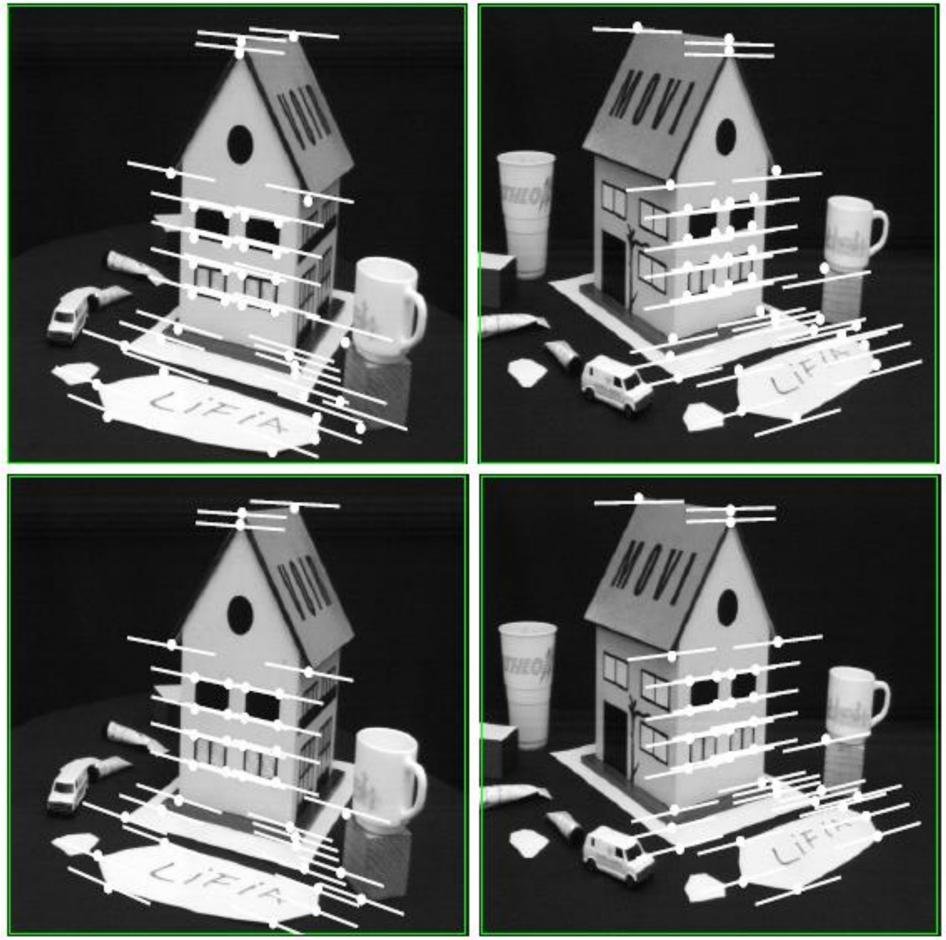
Problem with eight-point algorithm

250906.36	183269.57	921.81	200931.10	146766.13	738.21	272.19	198.81	1.00
2692.28	131633.03	176.27	6196.73	302975.59	405.71	15.27	746.79	1.00
416374.23	871684.30	935.47	408110.89	854384.92	916.90	445.10	931.81	1.00
191183.60	171759.40	410.27	416435.62	374125.90	893.65	465.99	418.65	1.00
48988.86	30401.76	57.89	298604.57	185309.58	352.87	846.22	525.15	1.00
164786.04	546559.67	813.17	1998.37	6628.15	9.86	202.65	672.14	1.00
116407.01	2727.75	138.89	169941.27	3982.21	202.77	838.12	19.64	1.00
135384.58	75411.13	198.72	411350.03	229127.78	603.79	681.28	379.48	1.00

$$\begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{bmatrix} = 0$$

- Poor numerical conditioning
- Can be fixed by rescaling the data: *Normalized 8-point algorithm* [Hartley, 1995]

Comparison of estimation algorithms



	8-point	Normalized 8-point	Nonlinear least squares
Reprojection error 1	2.33 pixels	0.92 pixel	0.86 pixel
Reprojection error 2	2.18 pixels	0.85 pixel	0.80 pixel

Extract R and T from E
(this slide will not be asked at the exam)

- Singular Value Decomposition

$$E = U \Sigma V^T$$

Enforcing rank-2 constraint

$$\overline{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\hat{T} = U \begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \overline{\Sigma} V^T$$

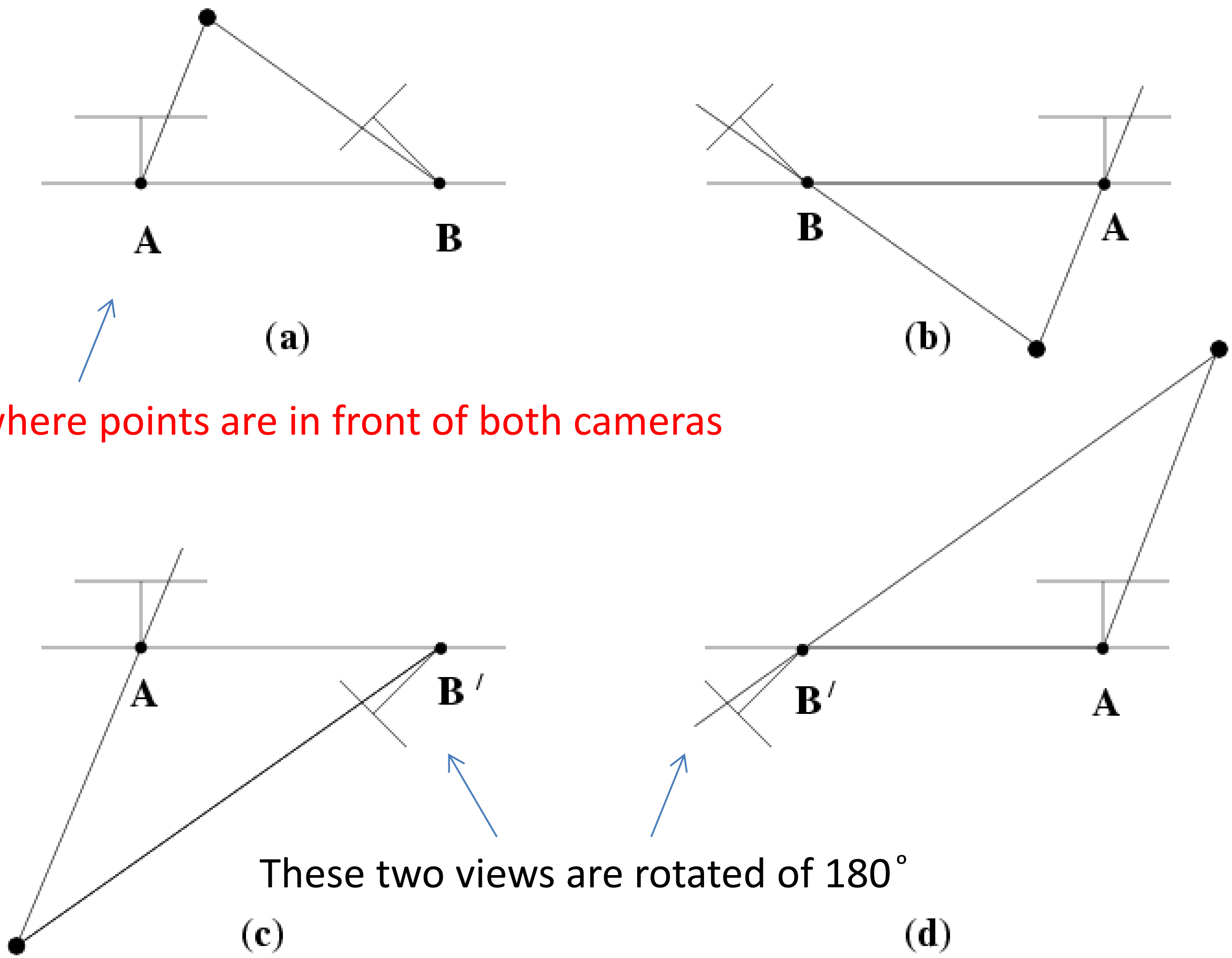
$$\hat{T} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & t_x \\ -t_y & t_x & 0 \end{bmatrix} \Rightarrow \hat{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

$$\hat{R} = U \begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} V^T$$

$$t = K_2 \hat{t}$$

$$R = K_2 \hat{R} K_1^{-1}$$

4 possible solutions of R and T

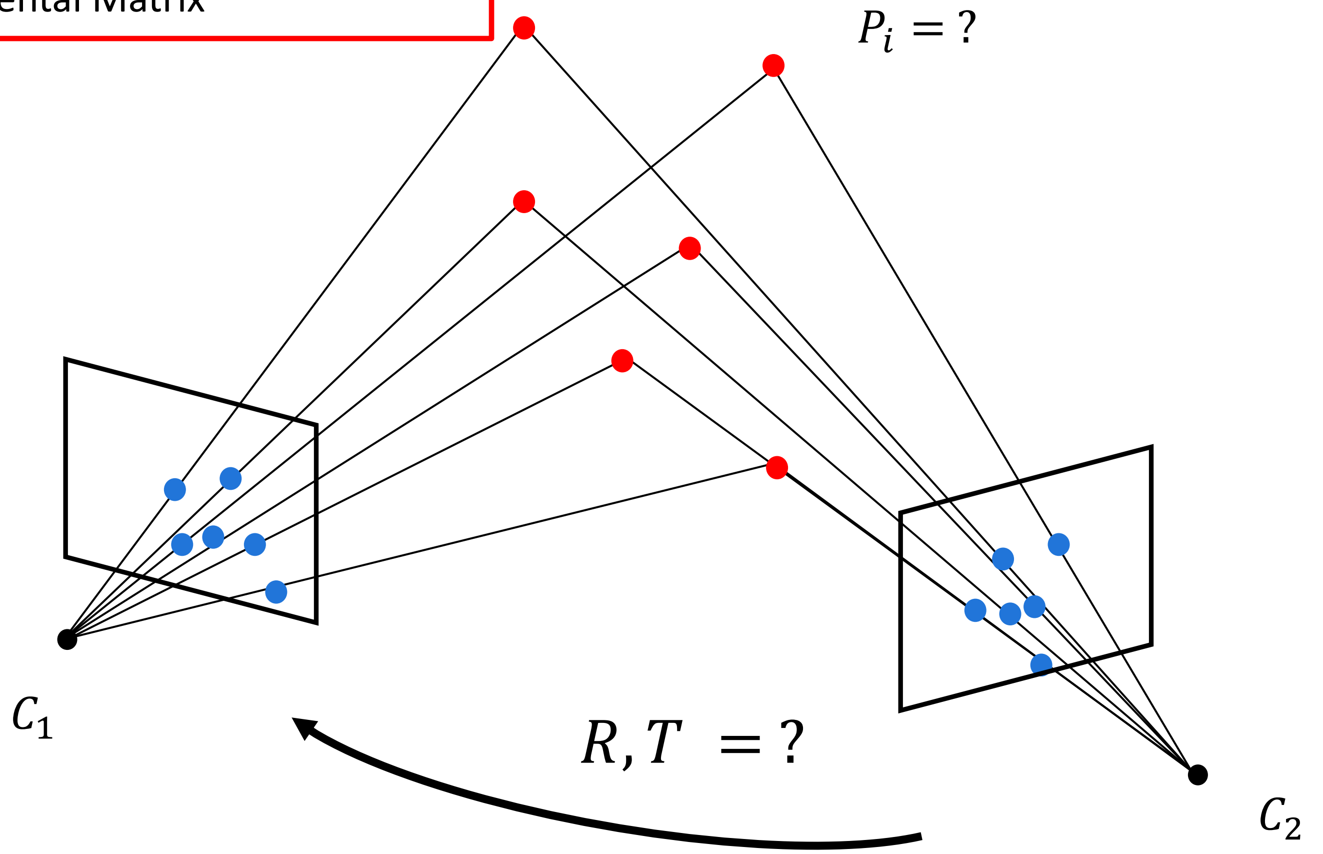


Only one solution where points are in front of both cameras

These two views are rotated of 180°

Structure from Motion (SFM)

- Two variants exist:
  - **Calibrated** camera(s) -> K is known
    - Uses the Essential Matrix
  - **Uncalibrated** camera(s) -> K is unknown
    - Uses the Fundamental Matrix



The Fundamental Matrix

- Before, we assumed to know the camera intrinsic parameters and we used normalized image coordinates

$$\mathrm{p}_2^T \ \mathrm{E} \ \mathrm{p}_1 = 0$$

$$\begin{bmatrix} \bar{u}_2^i \\ \bar{v}_2^i \\ 1 \end{bmatrix}^{\mathrm{T}} \mathrm{E} \begin{bmatrix} \bar{u}_1^i \\ \bar{v}_1^i \\ 1 \end{bmatrix} = 0$$

$$\begin{bmatrix} \bar{u}_1^i \\ \bar{v}_1^i \\ 1 \end{bmatrix} = \mathrm{K}_1^{-1} \begin{bmatrix} u_1^i \\ v_1^i \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \bar{u}_2^i \\ \bar{v}_2^i \\ 1 \end{bmatrix} = \mathrm{K}_2^{-1} \begin{bmatrix} u_2^i \\ v_2^i \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u_2^i \\ v_2^i \\ 1 \end{bmatrix}^{\mathrm{T}} \mathrm{K}_2^{-\mathrm{T}} \ \mathrm{E} \ \mathrm{K}_1^{-1} \begin{bmatrix} u_1^i \\ v_1^i \\ 1 \end{bmatrix} = 0$$

$$\begin{bmatrix} u_2^i \\ v_2^i \\ 1 \end{bmatrix}^{\mathrm{T}} \boxed{\mathrm{F}} \begin{bmatrix} u_1^i \\ v_1^i \\ 1 \end{bmatrix} = 0$$

Fundamental Matrix

$$\left. \begin{aligned} F &= \mathrm{K}_2^{-\mathrm{T}} \ \mathrm{E} \ \mathrm{K}_1^{-1} \\ E &= [T]_\times R \end{aligned} \right\} \Rightarrow \boxed{F = \mathrm{K}_2^{-\mathrm{T}} \, [T]_\times R \ \mathrm{K}_1^{-1}}$$