Vision Algorithms for Mobile Robotics

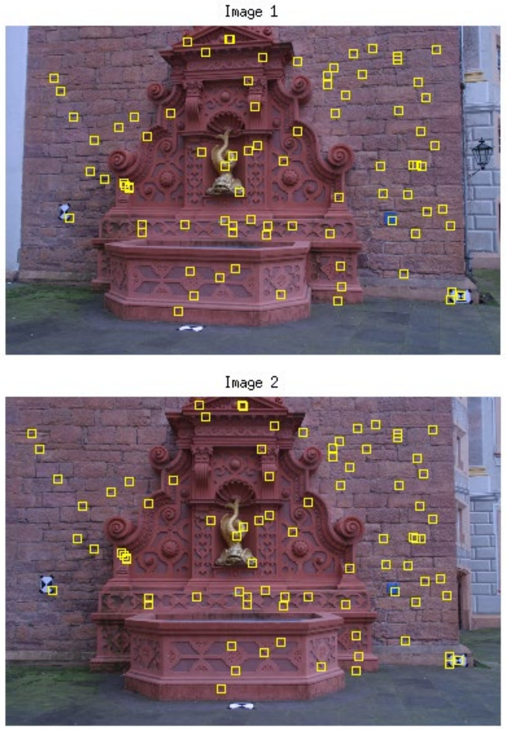
Lecture 08
Multiple View Geometry 2

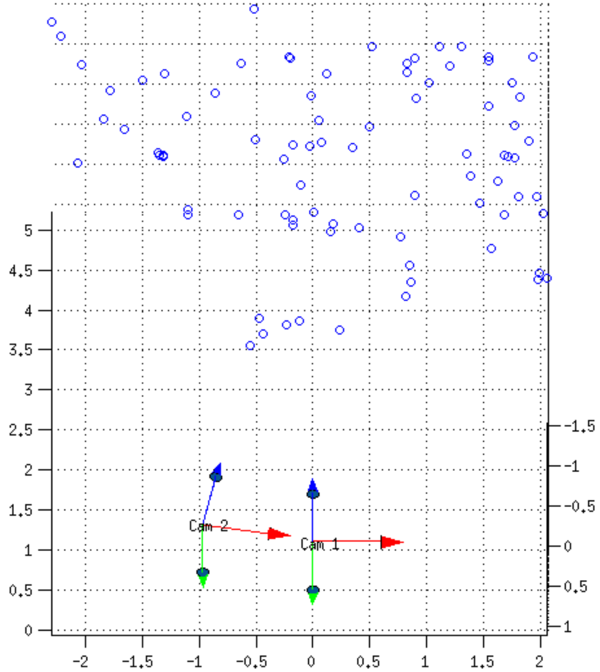
Davide Scaramuzza

http://rpg.ifi.uzh.ch

Lab Exercise 6 - Today

Implement the 8-point algorithm





Estimated poses and 3D structure

2-View Geometry: recap

Depth from stereo (i.e., stereo vision):

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

2-view Structure From Motion:

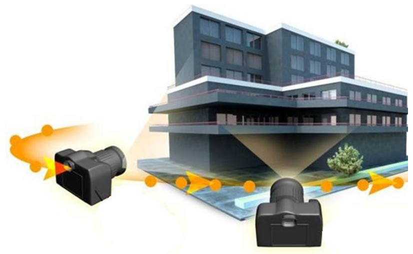
- **Assumptions**: none (K, T, and R are unknown).
- **Goal**: Recover simultaneously 3D scene structure and camera poses (up to scale) from two images

P_i =?

K_1, R_1, T_1

K_2, R_2, T_2

P_i =?



K_1, R_1, T_1 =?

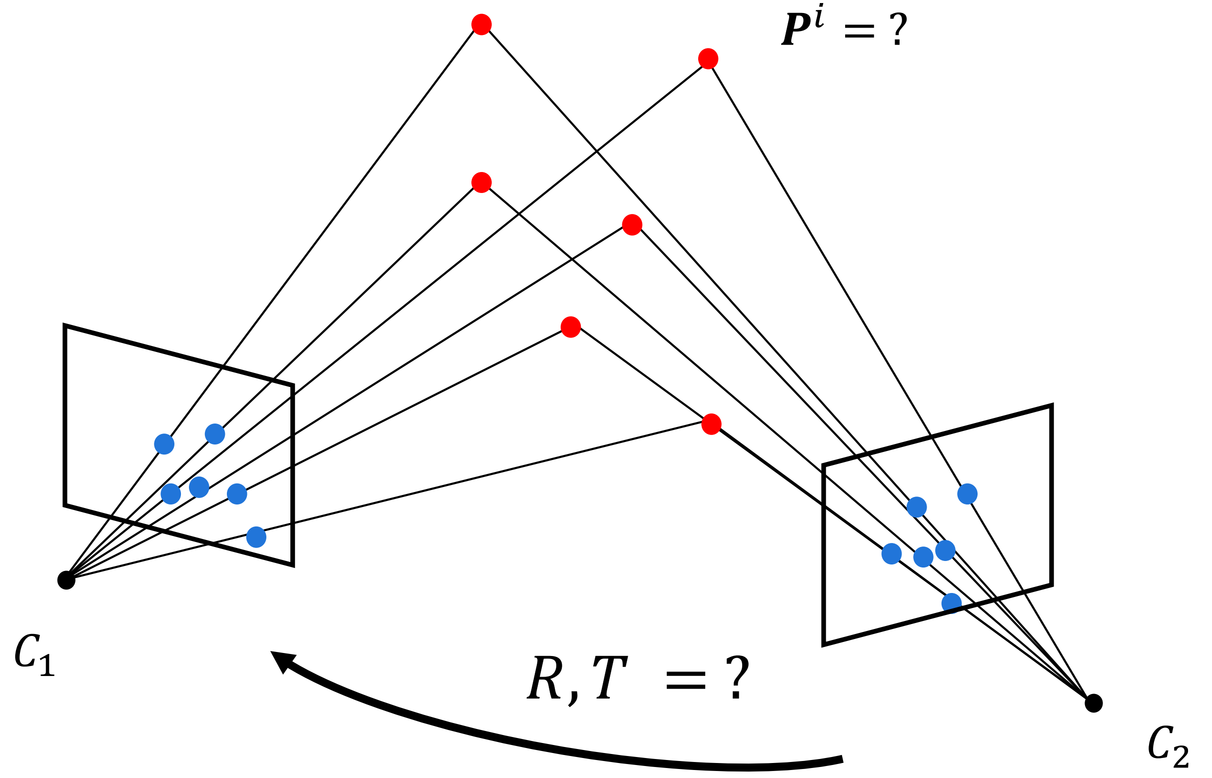
K_2, R_2, T_2 =?

Structure from Motion (SFM)

Problem formulation: Given a set of n point *correspondences* between two images, $\{p^i{}_1 = (u^i{}_1, v^i{}_1),\ p^i{}_2 = (u^i{}_2, v^i{}_2)\}$, where $i = 1 \dots n$, the goal is to simultaneously

- estimate the 3D points $\boldsymbol{P}^i$,
- the camera relative-motion parameters $(\boldsymbol{R}, \boldsymbol{T})$,
- and the camera intrinsics $\boldsymbol{K}_1, \boldsymbol{K}_2$ that satisfy:

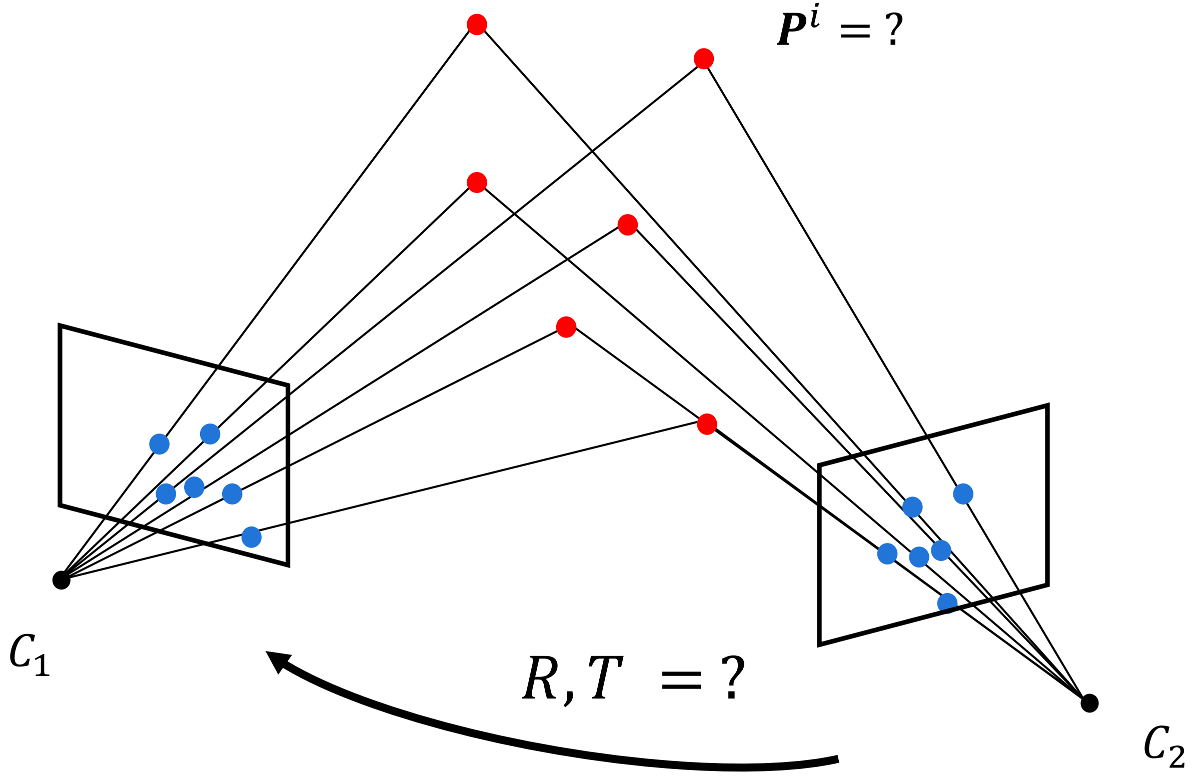
$$
\begin{cases}
\lambda_1^i \begin{bmatrix} u^i{}_1 \\ v^i{}_1 \\ 1 \end{bmatrix} = K_1 [I|0] \cdot \begin{bmatrix} X^i{}_w \\ Y^i{}_w \\ Z^i{}_w \\ 1 \end{bmatrix} \\
\lambda_2^i \begin{bmatrix} u^i{}_2 \\ v^i{}_2 \\ 1 \end{bmatrix} = K_2 [R|T] \cdot \begin{bmatrix} X^i{}_w \\ Y^i{}_w \\ Z^i{}_w \\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



Structure from Motion (SFM)

Two variants exist:

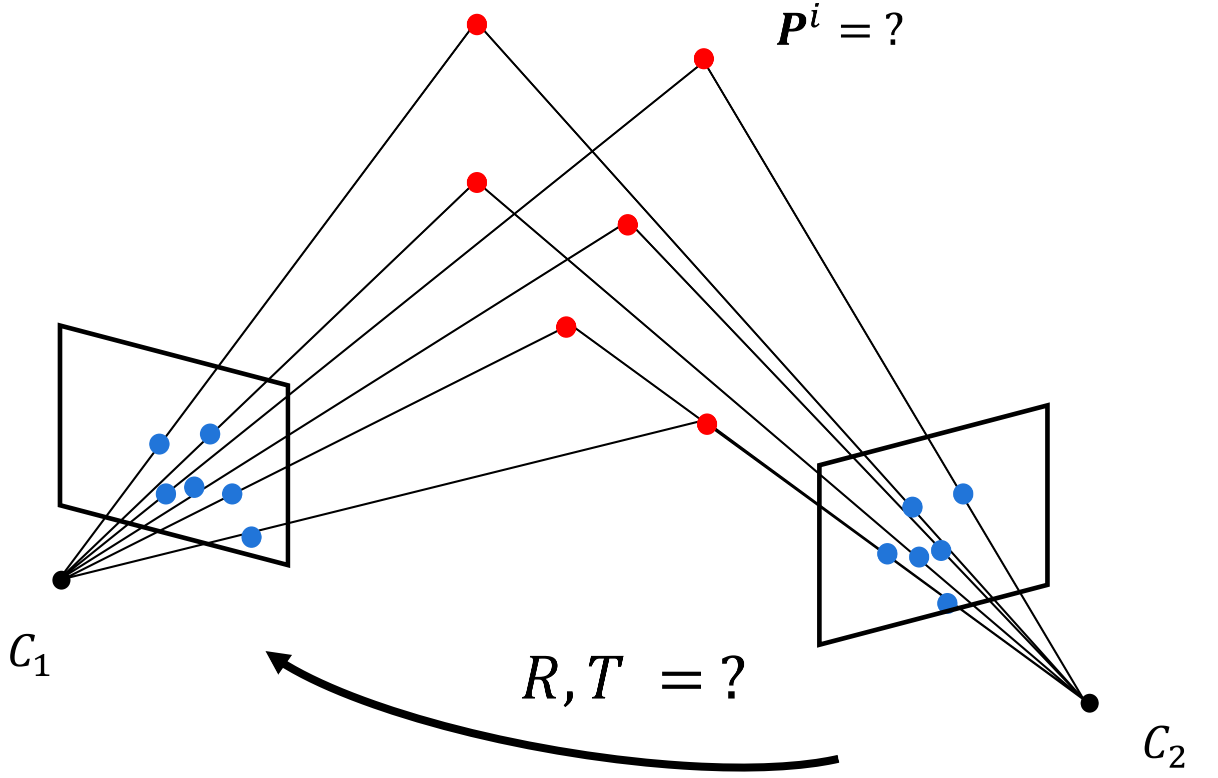
- **Calibrated** camera(s) ⇒ $\boldsymbol{K_1}, \boldsymbol{K_2}$ **are known**
- **Uncalibrated** camera(s) ⇒ $\boldsymbol{K_1}, \boldsymbol{K_2}$ **are unknown**



Structure from Motion (SFM)

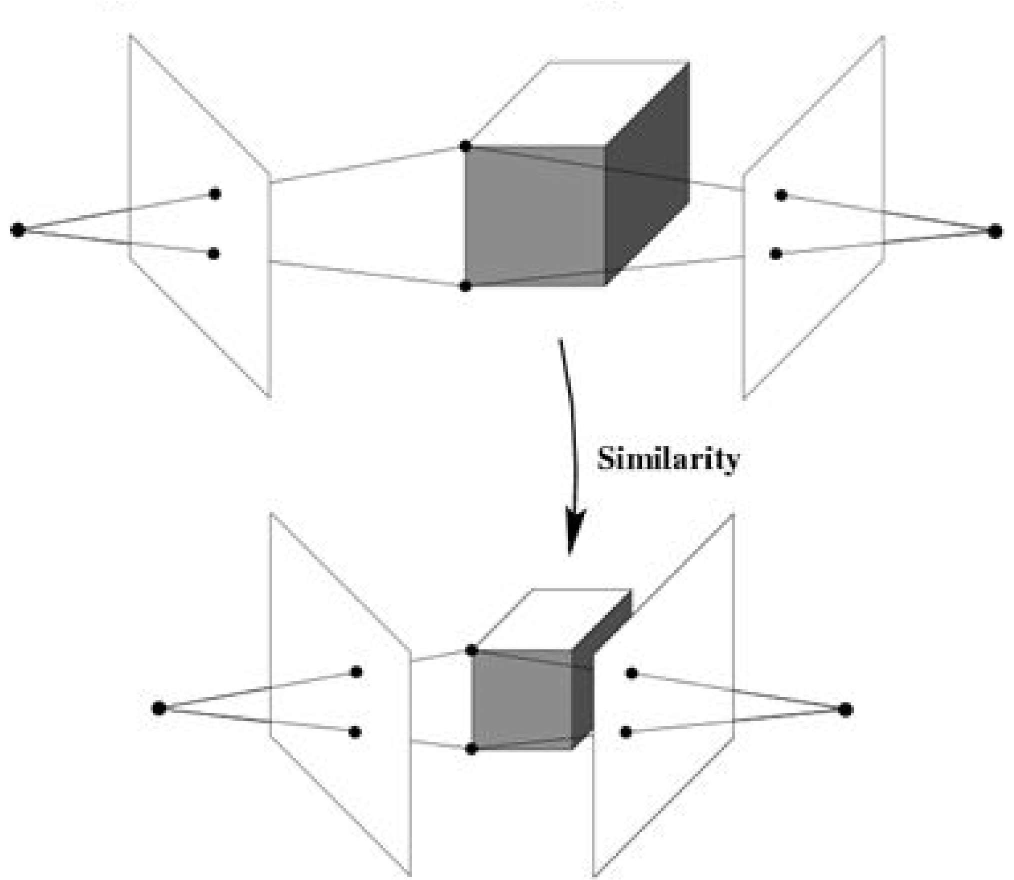
- Let's study the case in which the cameras are **calibrated**
- For convenience, let's use *normalized image coordinates* $\rightarrow$ $\begin{bmatrix} \bar{u} \\ \bar{v} \\ 1 \end{bmatrix} = K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$
- Thus, we want to find $\boldsymbol{R}, \boldsymbol{T}, \boldsymbol{P}^i$ that satisfy:

$$\begin{cases} \lambda^i_1 \begin{bmatrix} \bar{u}^i_1 \\ \bar{v}^i_1 \\ 1 \end{bmatrix} = [I|0] \cdot \begin{bmatrix} X^i_w \\ Y^i_w \\ Z^i_w \\ 1 \end{bmatrix} \\ \lambda^i_2 \begin{bmatrix} \bar{u}^i_2 \\ \bar{v}^i_2 \\ 1 \end{bmatrix} = [R|T] \cdot \begin{bmatrix} X^i_w \\ Y^i_w \\ Z^i_w \\ 1 \end{bmatrix} \end{cases}$$



Scale Ambiguity

If we rescale the entire scene and camera views by a constant factor (i.e., similarity transformation), the projections (in pixels) of the scene points in both images remain exactly the same:



Scale Ambiguity

- In Structure from Motion, it is therefore **not possible** to recover the absolute scale of the scene!
  - What about stereo vision? Is it possible? Why?
- 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?
  - **$\boldsymbol{4n}$ knowns:**
    - n correspondences; each one $(u^i{}_1, v^i{}_1)$ and $(u^i{}_2, v^i{}_2)$, $\ i = 1 \dots n$
  - **$\boldsymbol{5 + 3n}$ unknowns**
    - 5 for the motion up to a scale (3 for rotation, 2 for translation)
    - $3n$ = number of coordinates of the n 3D points

- Does a solution exist?
  - If and only if the *number of independent equations* $\geq$ *number of unknowns*
    $\Rightarrow 4n \geq 5 + 3n \Rightarrow \boxed{\mathbf{n \geq 5}}$
  - First attempt to identify the solutions by Kruppa in 1913 (see historical note on slide 16).

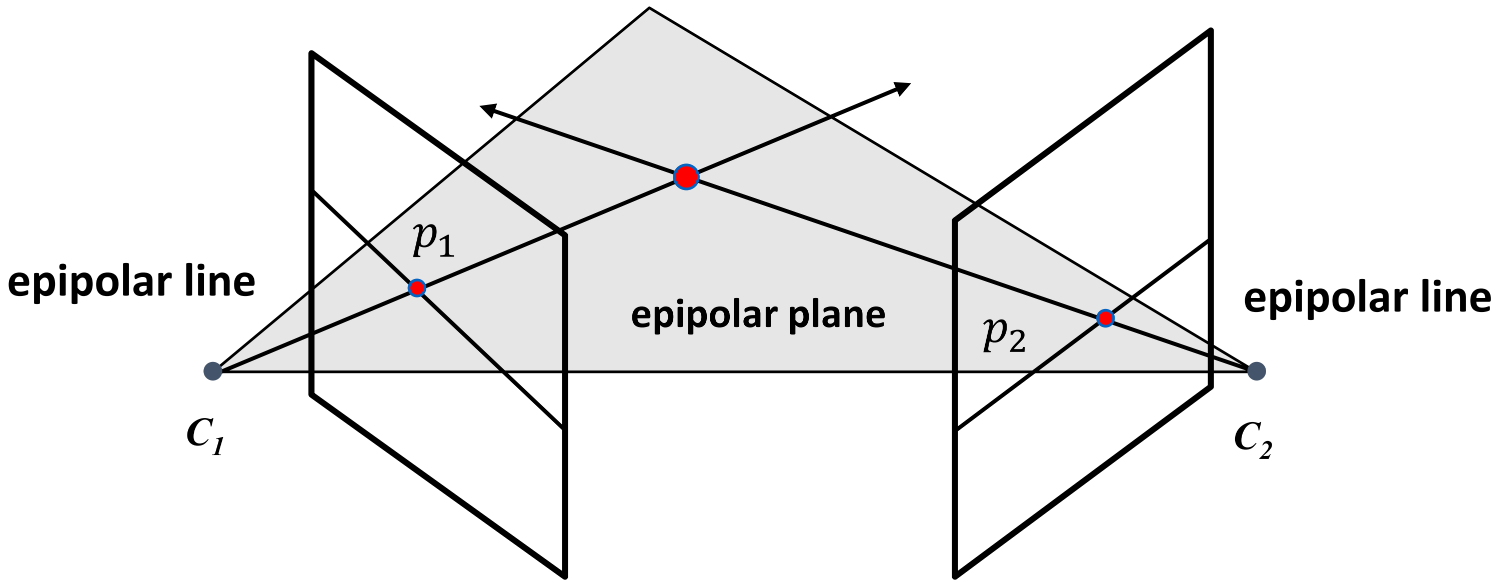
E. Kruppa, Zur Ermittlung eines Objektes aus zwei Perspektiven mit Innerer Orientierung, *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa.*, 1913. – English Translation plus original paper by Guillermo Gallego, Arxiv, 2017

Structure From Motion (SFM)

- Can we solve the estimation of relative motion (R, T) independently of the estimation of the 3D points? Yes! The next couple of slides prove that this is possible.
- Once (R, T) are known, the 3D points can be triangulated using the triangulation algorithm from Lecture 7 (i.e., least square approximation plus reprojection error minimization)

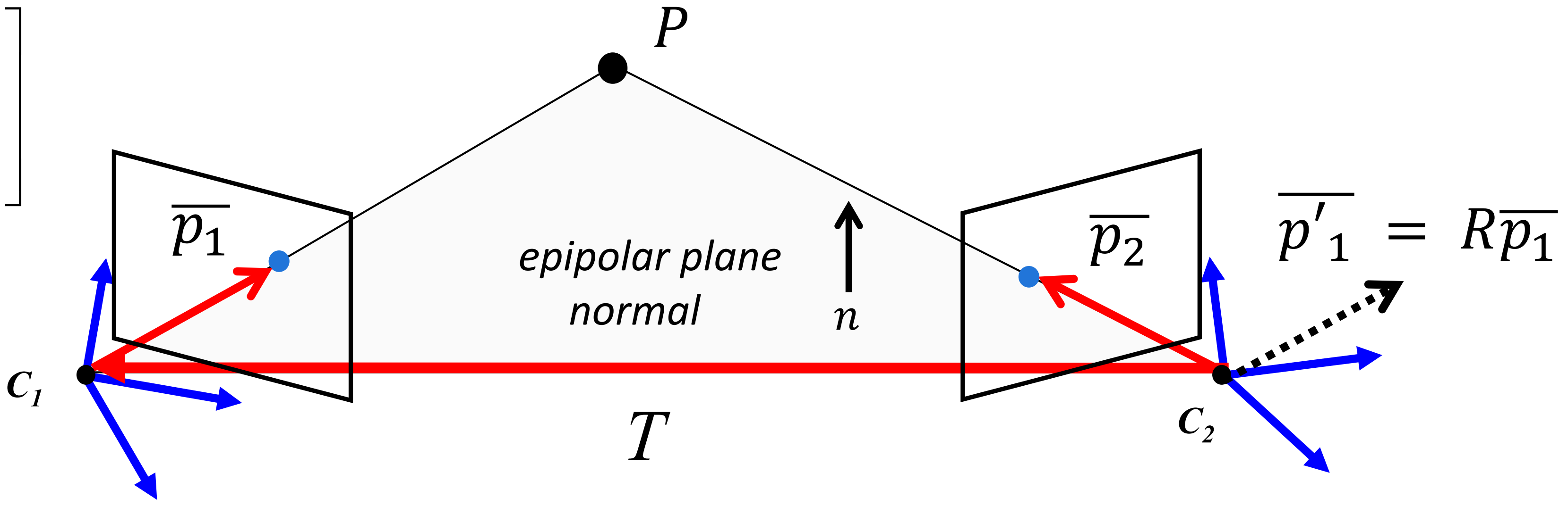
The Epipolar Constraint: Recap from Lecture 07

- The camera centers C_1, C_2 and one image point p_1 (or p_2) determine the so called **epipolar plane**
- The intersections of the epipolar plane with the two image planes are called **epipolar lines**
- **Corresponding points must therefore lie along the epipolar lines**: this constraint is called **epipolar constraint**
- An alternative way to formulate the epipolar constraint is to notice that **two corresponding image vectors plus the baseline must be coplanar**



Epipolar Geometry

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



$\bar{p}_1, \bar{p}_2, T$ **are coplanar**:

$$\bar{p}_2^T \cdot n = 0 \Rightarrow \qquad \Rightarrow \bar{p}_2^T\,[T_\times]R\,\bar{p}_1 = 0 \qquad \Rightarrow \boxed{\bar{p}_2^T\ E\ \bar{p}_1 = 0}$$

epipolar constraint

$$\boxed{\mathrm{E} = [T_\times]R \qquad \textit{essential matrix}}$$

Epipolar Geometry

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

$$\bar{p}_2^T \, E \, \bar{p}_1 = 0$$ *Epipolar constraint or Longuet-Higgins equation (1981)*

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

R and T can be computed from E recalling that:

$$\mathrm{E} = [T_\times]R$$

H. Christopher Longuet-Higgins, A computer algorithm for reconstructing a scene from two projections, Nature, 1981, PDF.

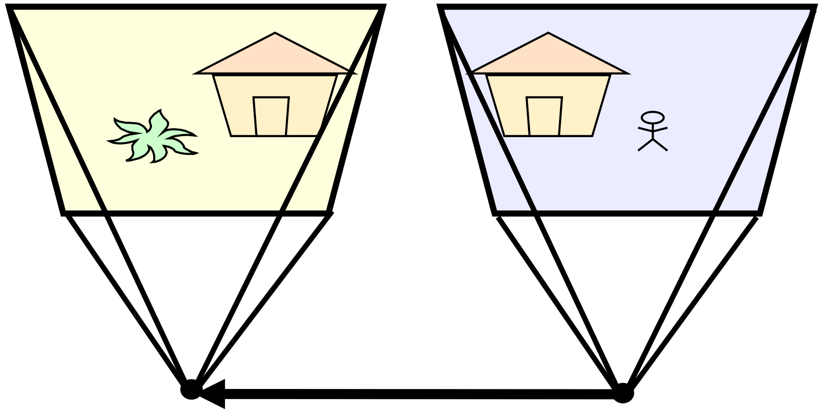
Example: Essential Matrix of a Camera Translating along x

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

$$[\mathrm{T}_\times] = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & b \\ 0 & -b & 0 \end{bmatrix}$$

$$\mathrm{R} = \mathrm{I}_{3\times 3}$$

$$\rightarrow E = [\mathrm{T}_\times]R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & b \\ 0 & -b & 0 \end{bmatrix}$$



$$\mathrm{T} = \begin{bmatrix} -b \\ 0 \\ 0 \end{bmatrix}$$

How to compute the Essential Matrix?

- If we don't know (R, T) can we estimate E from two images?
- Yes, given at least 5 correspondences

Image 1

Image 2

A Note of History

- **Kruppa showed in 1913 that 5 image correspondences is the minimal case** and that there can be at up to 11 solutions
- However, in **1988, Demazure** showed that there are actually at most **10 distinct solutions**.
- In **1996**, Philipp proposed an **iterative algorithm to find these solutions**.
- In **2004**, Nister proposed the **first efficient and non iterative solution**. It uses Groebner basis decomposition.
- The first popular solution uses 8 points and is called **the 8-point algorithm** or **Longuet-Higgins algorithm** (1981). Because of its ease of implementation, it is still used today (e.g., NASA rovers).

[1] E. Kruppa, Zur Ermittlung eines Objektes aus zwei Perspektiven mit Innerer Orientierung, *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa.*, 1913. – English Translation plus original paper by Guillermo Gallego, Arxiv, 2017

[2] H. Christopher Longuet-Higgins, A computer algorithm for reconstructing a scene from two projections, Nature, 1981, PDF.

[3] D. Nister, An Efficient Solution to the Five-Point Relative Pose Problem, PAMI, 2004, PDF

The 8-point algorithm

- Each pair of point correspondences $\bar{\boldsymbol{p}}_1 = (\bar{u}_1, \bar{v}_1, 1)^T, \quad \bar{\boldsymbol{p}}_2 = (\bar{u}_2, \bar{v}_2, 1)^T$ provides a linear equation:

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

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

$$\bar{u}_2\bar{u}_1 e_{11} + \quad \bar{u}_2\bar{v}_1 e_{12} + \quad \bar{u}_2 e_{13} + \quad \bar{v}_2\bar{u}_1 e_{21} + \quad \bar{v}_2\bar{v}_1 e_{22} + \quad \bar{v}_2 e_{23} + \quad \bar{u}_1 e_{31} + \quad \bar{v}_1 e_{32} + \quad e_{33} = 0$$

H. Christopher Longuet-Higgins, A computer algorithm for reconstructing a scene from two projections, **Nature**, 1981, PDF.

The 8-point algorithm

- For n points, we can write

$$\underbrace{\begin{bmatrix} \bar{u}_2^{\,1}\bar{u}_1^{\,1} & \bar{u}_2^{\,1}\bar{v}_1^{\,1} & \bar{u}_2^{\,1} & \bar{v}_2^{\,1}\bar{u}_1^{\,1} & \bar{v}_2^{\,1}\bar{v}_1^{\,1} & \bar{v}_2^{\,1} & \bar{u}_1^{\,1} & \bar{v}_1^{\,1} & 1 \\ \bar{u}_2^{\,2}\bar{u}_1^{\,2} & \bar{u}_2^{\,2}\bar{v}_1^{\,2} & \bar{u}_2^{\,2} & \bar{v}_2^{\,2}\bar{u}_1^{\,2} & \bar{v}_2^{\,2}\bar{v}_1^{\,2} & \bar{v}_2^{\,2} & \bar{u}_1^{\,2} & \bar{v}_1^{\,2} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \bar{u}_2^{\,n}\bar{u}_1^{\,n} & \bar{u}_2^{\,n}\bar{v}_1^{\,n} & \bar{u}_2^{\,n} & \bar{v}_2^{\,n}\bar{u}_1^{\,n} & \bar{v}_2^{\,n}\bar{v}_1^{\,n} & \bar{v}_2^{\,n} & \bar{u}_1^{\,n} & \bar{v}_1^{\,n} & 1 \end{bmatrix}}_{\text{Q (this matrix is } \textbf{known}\text{)}} \underbrace{\begin{bmatrix} e_{11} \\ e_{12} \\ e_{13} \\ e_{21} \\ e_{22} \\ e_{23} \\ e_{31} \\ e_{32} \\ e_{33} \end{bmatrix}}_{\bar{E} \text{ (this matrix is } \textbf{unknown}\text{)}} = 0$$

The 8-point algorithm

$$Q \cdot \bar{E} = 0$$

Minimal solution
- $Q_{(n\times 9)}$ should have rank 8 to have a unique (up to a scale) non-trivial solution $\bar{E}$
- Each point correspondence provides 1 independent equation
- Thus, 8 point correspondences are needed

Over-determined solution
- *n* > 8 points
- A solution is to minimize $||Q\bar{E}||^2$ subject to the constraint $||\bar{E}||^2 = 1$.
  The solution is the eigenvector corresponding to the smallest eigenvalue of the matrix $Q^T Q$ (because it is the unit vector x that minimizes $||Qx||^2 = x^T Q^T Q x$).
- It can be solved through Singular Value Decomposition (SVD). Matlab instructions:

```
[U,S,V] = svd(Q);
Ev = V(:,9);
E = reshape(Ev,3,3)';
```

Degenerate Configurations
- The solution of the 8-point algorithm is **degenerate when the 3D points are coplanar**.
- Conversely, the 5-point algorithm works also for coplanar points

8-point algorithm: Matlab code

A few lines of code. In today's exercise you will learn how to implement it

```
function E = 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);
Eh = V(:,9);

E = reshape(Eh,3,3)';
```

Extract R and T from E

Won't be asked at the exam ☺

- Singular Value Decomposition: $E = U\Sigma V^T$
- Enforcing rank-2 constraint: set smallest singular value of Σ to 0:

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \cancel{\sigma_3} \end{bmatrix} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

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

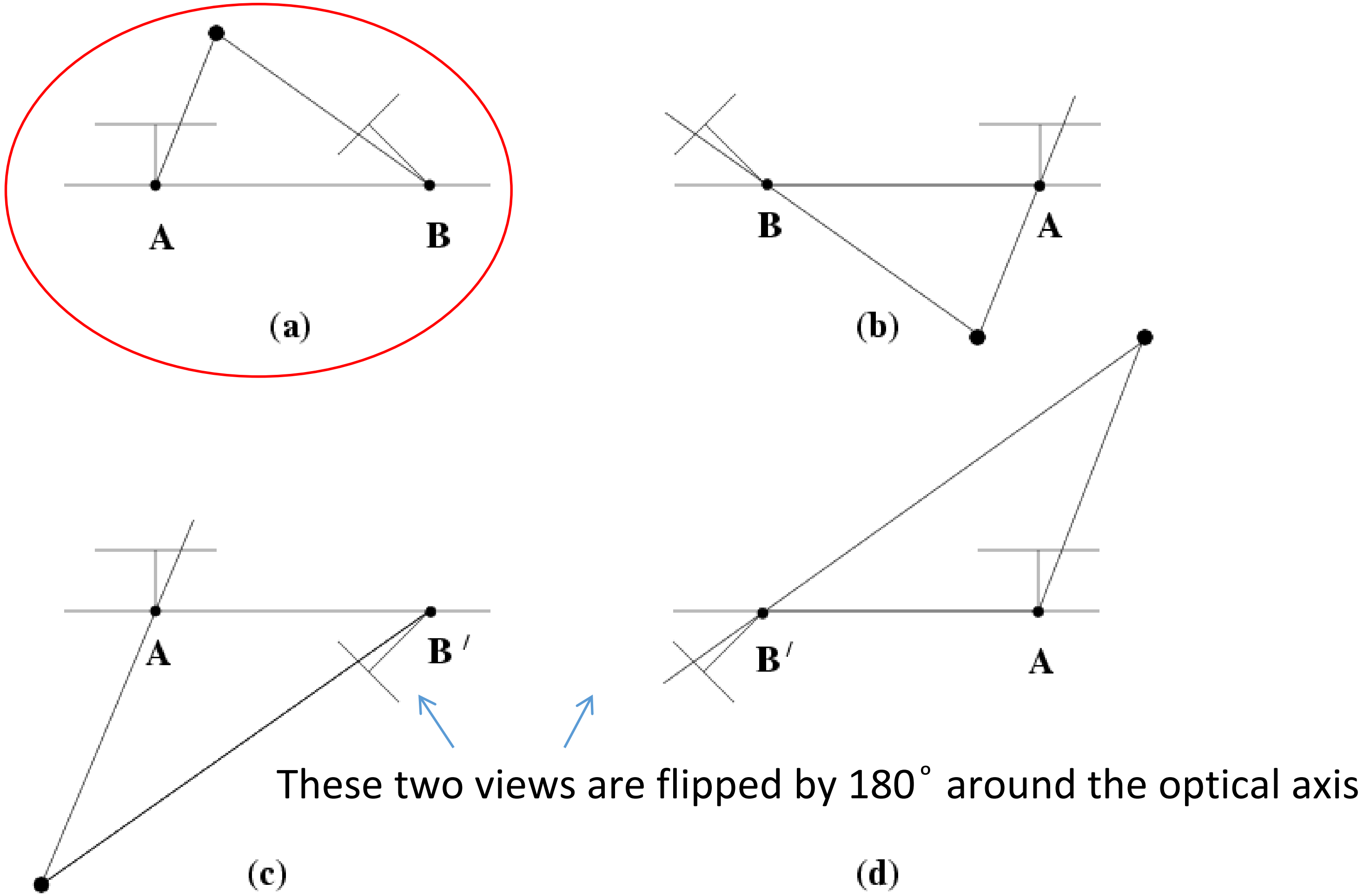
$$\hat{T} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & t_x \\ -t_y & t_x & 0 \end{bmatrix} \Rightarrow \hat{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

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

$$\boxed{\begin{aligned} T &= K_2 \hat{t} \\ R &= K_2 \hat{R} K_1^{-1} \end{aligned}}$$

4 possible solutions of R and T

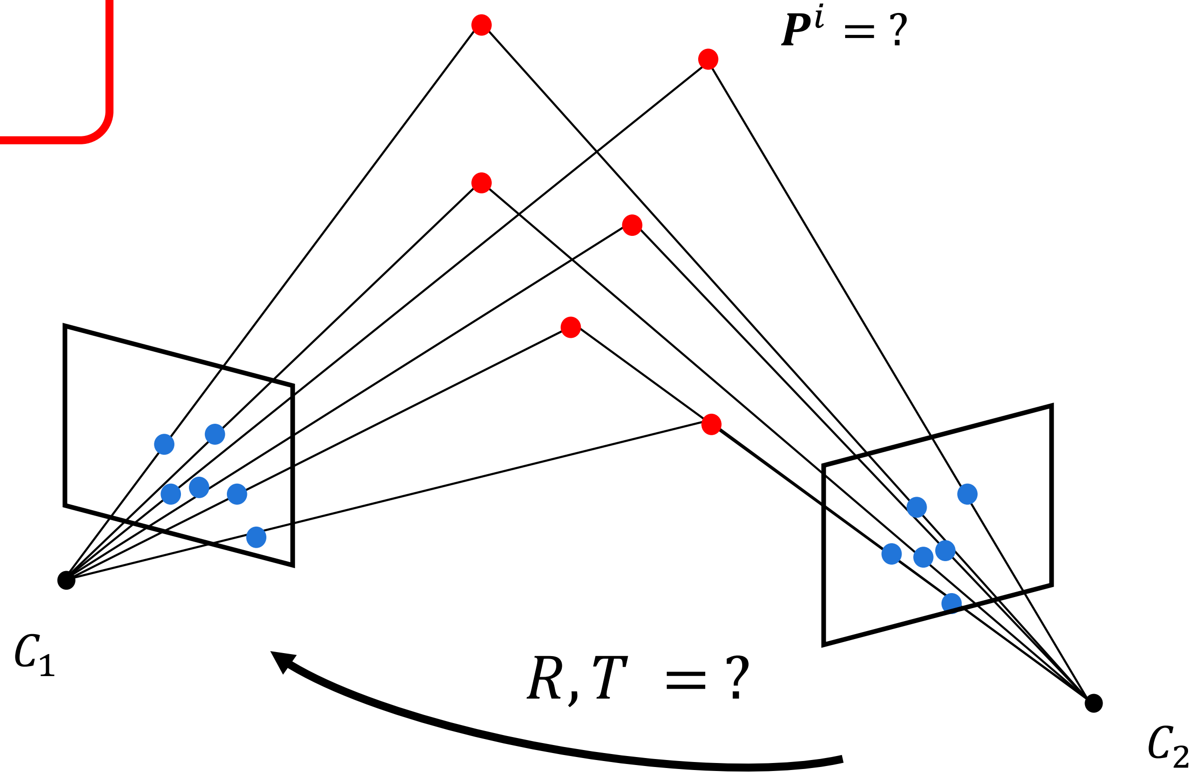
There exists **only one solution** where points are **in front of both cameras**



Structure from Motion (SFM)

Two variants exist:

- **Calibrated** camera(s) $\Rightarrow$ $\boldsymbol{K_1, K_2}$ **are known**
  - Uses the Essential matrix
- **Uncalibrated** camera(s) $\Rightarrow$ $\boldsymbol{K_1, K_2}$ **are unknown**
  - Uses the Fundamental matrix



The Fundamental Matrix

So far, we have assumed to know the camera intrinsic parameters and we have used normalized image coordinates to get the epipolar constraint for **calibrated cameras**:

$$\begin{bmatrix} \overline{u}_1^i \\ \overline{v}_1^i \\ 1 \end{bmatrix} = \mathrm{K}_1^{-1} \begin{bmatrix} u_1^i \\ v_1^i \\ 1 \end{bmatrix} \qquad \begin{bmatrix} \overline{u}_2^i \\ \overline{v}_2^i \\ 1 \end{bmatrix} = \mathrm{K}_2^{-1} \begin{bmatrix} u_2^i \\ v_2^i \\ 1 \end{bmatrix}$$

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

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

The Fundamental Matrix

So far, we have assumed to know the camera intrinsic parameters and we have used normalized image coordinates to get the epipolar constraint for **calibrated cameras:**

$$\begin{bmatrix} \overline{u}_1^i \\ \overline{v}_1^i \\ 1 \end{bmatrix} = \mathrm{K}_1^{-1} \begin{bmatrix} u_1^i \\ v_1^i \\ 1 \end{bmatrix} \qquad \begin{bmatrix} \overline{u}_2^i \\ \overline{v}_2^i \\ 1 \end{bmatrix} = \mathrm{K}_2^{-1} \begin{bmatrix} u_2^i \\ v_2^i \\ 1 \end{bmatrix}$$

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

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

The Fundamental Matrix

So far, we have assumed to know the camera intrinsic parameters and we have used normalized image coordinates to get the epipolar constraint for **calibrated cameras:**

$$\begin{bmatrix} \bar{u}_1^i \\ \bar{v}_1^i \\ 1 \end{bmatrix} = \mathrm{K}_1^{-1} \begin{bmatrix} u_1^i \\ v_1^i \\ 1 \end{bmatrix} \qquad \begin{bmatrix} \bar{u}_2^i \\ \bar{v}_2^i \\ 1 \end{bmatrix} = \mathrm{K}_2^{-1} \begin{bmatrix} u_2^i \\ v_2^i \\ 1 \end{bmatrix}$$

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

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

Fundamental Matrix $F = \mathrm{K}_2^{-\mathrm{T}} \ \mathrm{E} \ \mathrm{K}_1^{-1}$

Fun thing: check out the Fundamental Matrix song, https://youtu.be/DgGV3l82NTk :-)

The 8-point Algorithm for the Fundamental Matrix

- The same 8-point algorithm to compute the essential matrix from a set of normalized image coordinates can also be used to determine the Fundamental matrix:

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

- However, now the key advantage is that we work **directly in pixel coordinates**

Problem with 8-point algorithm

$$\begin{bmatrix} u_2^1 u_1^1 & u_2^1 v_1^1 & u_2^1 & v_2^1 u_1^1 & v_2^1 v_1^1 & v_2^1 & u_1^1 & v_1^1 & 1 \\ u_2^2 u_1^2 & u_2^2 v_1^2 & u_2^2 & v_2^2 u_1^2 & v_2^2 v_1^2 & v_2^2 & u_1^2 & v_1^2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_2^n u_1^n & u_2^n v_1^n & u_2^n & v_2^n u_1^n & v_2^n v_1^n & v_2^n & u_1^n & v_1^n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

Problem with 8-point algorithm

- **Poor numerical conditioning**, which makes results **very sensitive to noise**
- Can be fixed by rescaling the data: **Normalized 8-point algorithm**

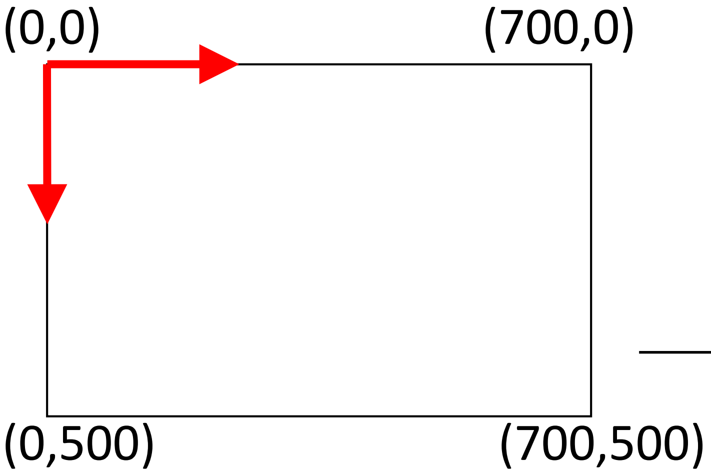
250906.36	183269.57	921.81	200931.10	146766.13	738.21	272.19	198.81	1.00
2692.28	131633.03	176.27	6196.73	302975.59	405.71	15.27	746.79	1.00
416374.23	871684.30	935.47	408110.89	854384.92	916.90	445.10	931.81	1.00
191183.60	171759.40	410.27	416435.62	374125.90	893.65	465.99	418.65	1.00
48988.86	30401.76	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
~10000	~10000	~100	~10000	~10000	~100	~100	~100	1

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

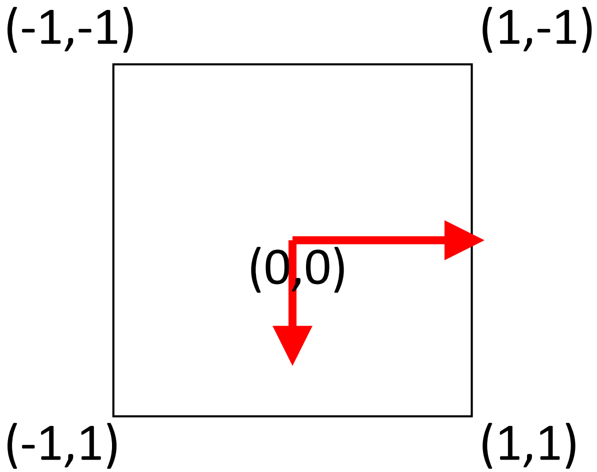
Orders of magnitude difference
between column of data matrix
$\rightarrow$ least-squares yields poor results

Normalized 8-point algorithm (1/3)

- This can be fixed using a normalized 8-point algorithm [Hartley, 1997], which estimates the Fundamental matrix on a set of **Normalized correspondences** (with better numerical properties) and **then unnormalizes** the result to obtain the fundamental matrix for the **given (unnormalized) correspondences**
- **Idea:** Transform image coordinates so that they are in the range $\sim[-1,1] \times [-1,1]$
- One way is to apply the following rescaling and shift



$$\begin{bmatrix} \frac{2}{700} & 0 & -1 \\ 0 & \frac{2}{500} & -1 \\ 0 & 0 & 1 \end{bmatrix}$$



Hartley, In defense of the eight-point algorithm, IEEE Transactions of Pattern Analysis and Machine Intelligence, PDF

Normalized 8-point algorithm (3/3)

The Normalized 8-point algorithm can be summarized in three steps:

1. **Normalize** the point correspondences: $\widehat{p_1} = B_1 p_1$, $\widehat{p_2} = B_2 p_2$
2. Estimate **normalized** $\hat{F}$ with 8-point algorithm using normalized coordinates $\widehat{p_1}, \widehat{p_2}$
3. Compute **unnormalized** F from $\hat{F}$:

$$\widehat{p_2}^T \hat{\mathrm{F}} \widehat{p_1} = 0$$

$$\underbrace{\boxed{p_2^\top B_2{}^\top} \quad \hat{\mathrm{F}} \quad \boxed{B_1 p_1}}_{\mathrm{F} = \mathrm{B}_2^\top \hat{\mathrm{F}}\, \mathrm{B}_1}$$

Normalized 8-point algorithm (2/3)

- In the original 1997 paper, Hartley proposed to rescale the two point sets such that the centroid of each set is 0 and the mean standard deviation $\sqrt{2}$ (equivalent to having the points distributed around a circled passing through the four corners of the $[-1,1] \times [-1,1]$ square).

- This can be done for every point as follows: $\widehat{p^i} = \frac{\sqrt{2}}{\sigma}(p^i - \mu)$

  where $\mu = (\mu_x, \mu_y) = \frac{1}{N}\sum_{i=1}^{n} p^i$ is the centroid and $\sigma = \frac{1}{N}\sum_{i=1}^{n}\left\|p^i - \mu\right\|^2$ is the mean standard deviation of the point set

- This transformation can be expressed in matrix form using homogeneous coordinates:

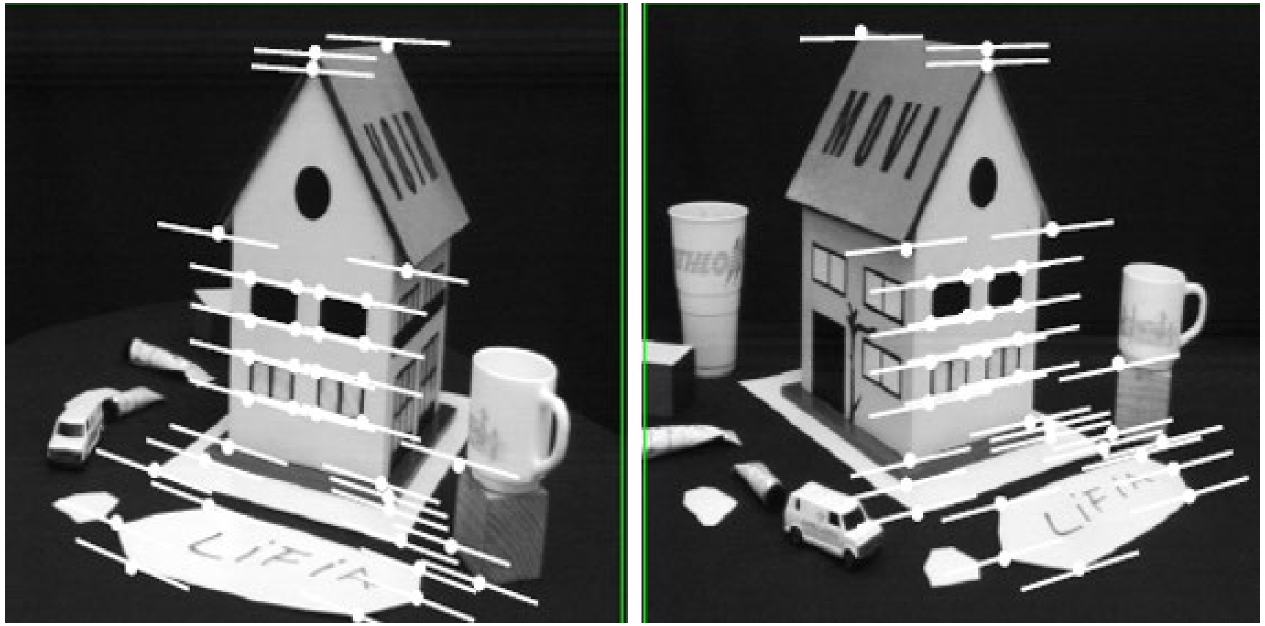
$$\widehat{p^i} = \begin{bmatrix} \frac{\sqrt{2}}{\sigma} & 0 & -\frac{\sqrt{2}}{\sigma}\mu_x \\ 0 & \frac{\sqrt{2}}{\sigma} & -\frac{\sqrt{2}}{\sigma}\mu_y \\ 0 & 0 & 1 \end{bmatrix} p^i$$

Hartley, In defense of the eight-point algorithm, IEEE Transactions of Pattern Analysis and Machine Intelligence, PDF

Can R, T, K_1, K_2 be extracted from F?

- In general **no**: infinite solutions exist
- However, if the coordinates of the principal points of each camera are known and the two cameras have the same focal length f in pixels, then R, T, f can determined uniquely

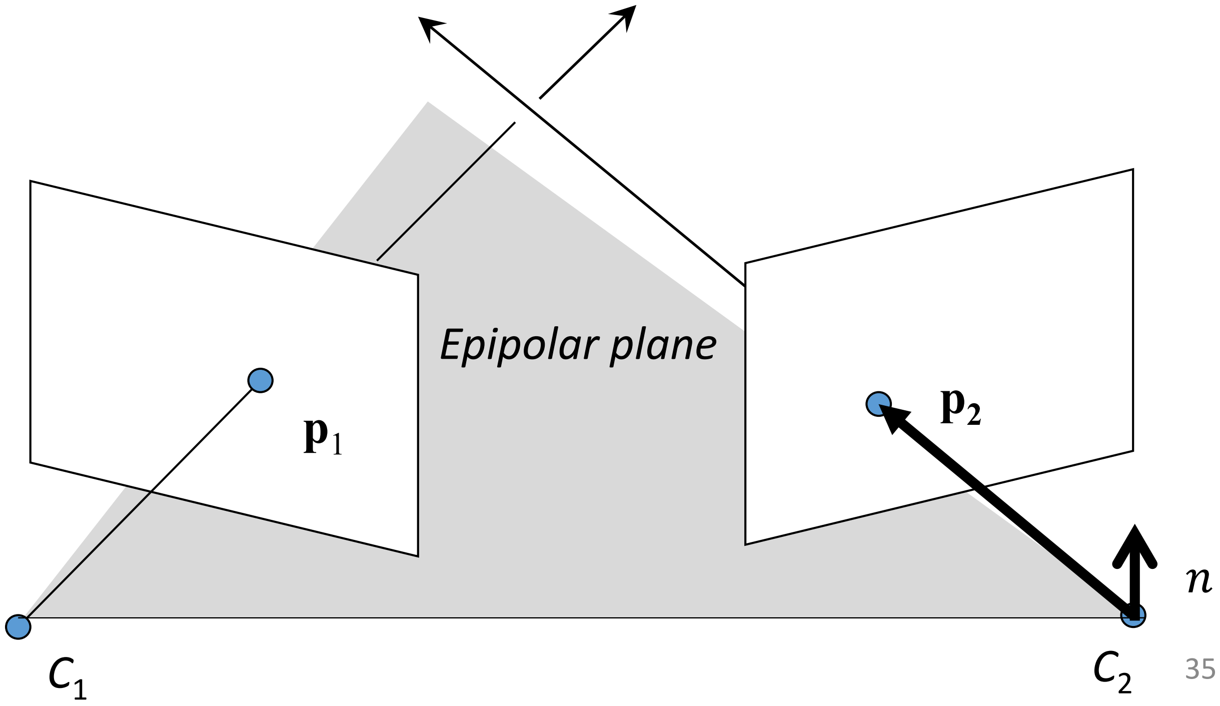
Comparison between Normalized and non-normalized algorithm



	8-point	Normalized 8-point	Nonlinear refinement
Avg. Ep. Line Distance	2.33 pixels	0.92 pixel	0.86 pixel

Error Measures

- The **quality of the estimated Essential or Fundamental matrix** can be measured using different error metrics:
  - **Algebraic error**
  - **Directional Error**
  - **Epipolar Line Distance**
  - **Reprojection Error**
- When is the error 0?
- **These errors will be exactly 0 only if E (or F) is computed from just 8 points** (because in this case a **non-overdetermined solution** exists).
- **For more than 8 points**, **it will only be 0 if there is no noise or outliers in the data**
  (if there is image noise or outliers then it the system becomes overdetermined)



Algebraic Error

- It follows directly from the 8-point algorithm, which seeks to minimize the **algebraic error:**

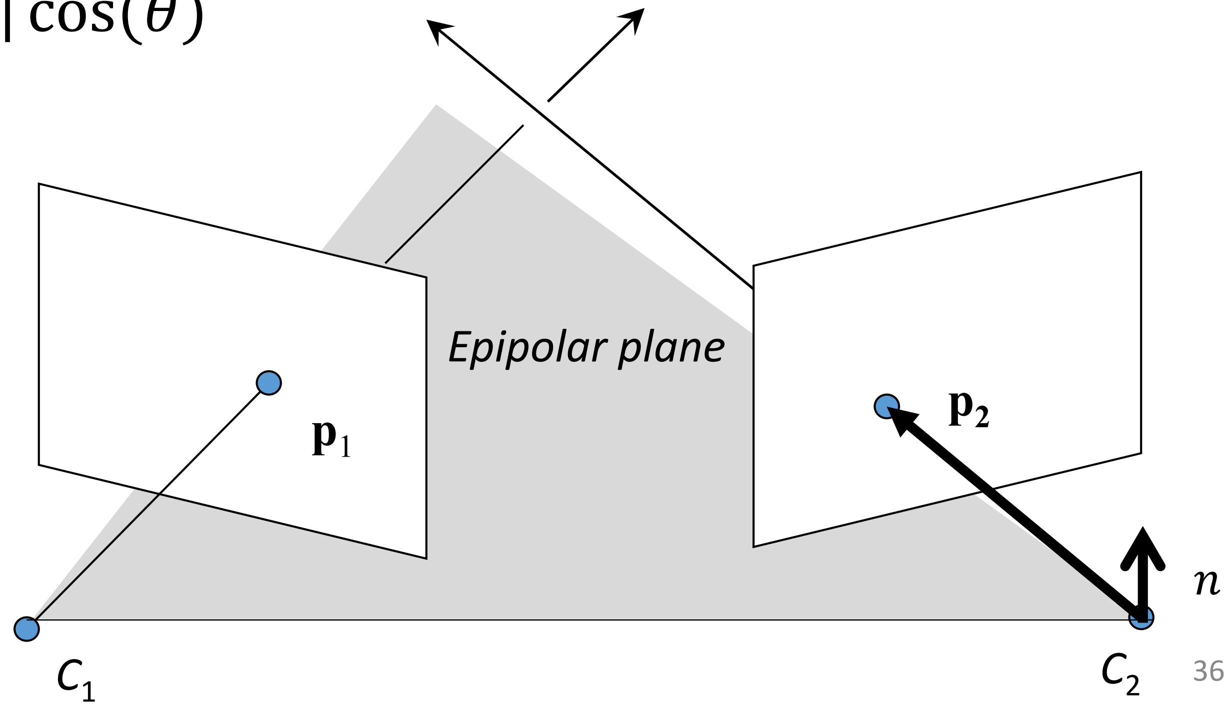
$$err = \left\| QE \right\|^2 = \sum_{i=1}^{N} \left(\bar{p}'^{i\,\mathrm{T}}_{2} \boldsymbol{E}\, \bar{p}'^{i}_{1} \right)^2$$

- From the proof of the epipolar constraint and using the definition of dot product, it can be observed that:

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

$$= \| \overline{\boldsymbol{p}}_2 \| \| [\mathrm{T}_\times] R\, \overline{\boldsymbol{p}}_1 \| \cos(\theta)$$

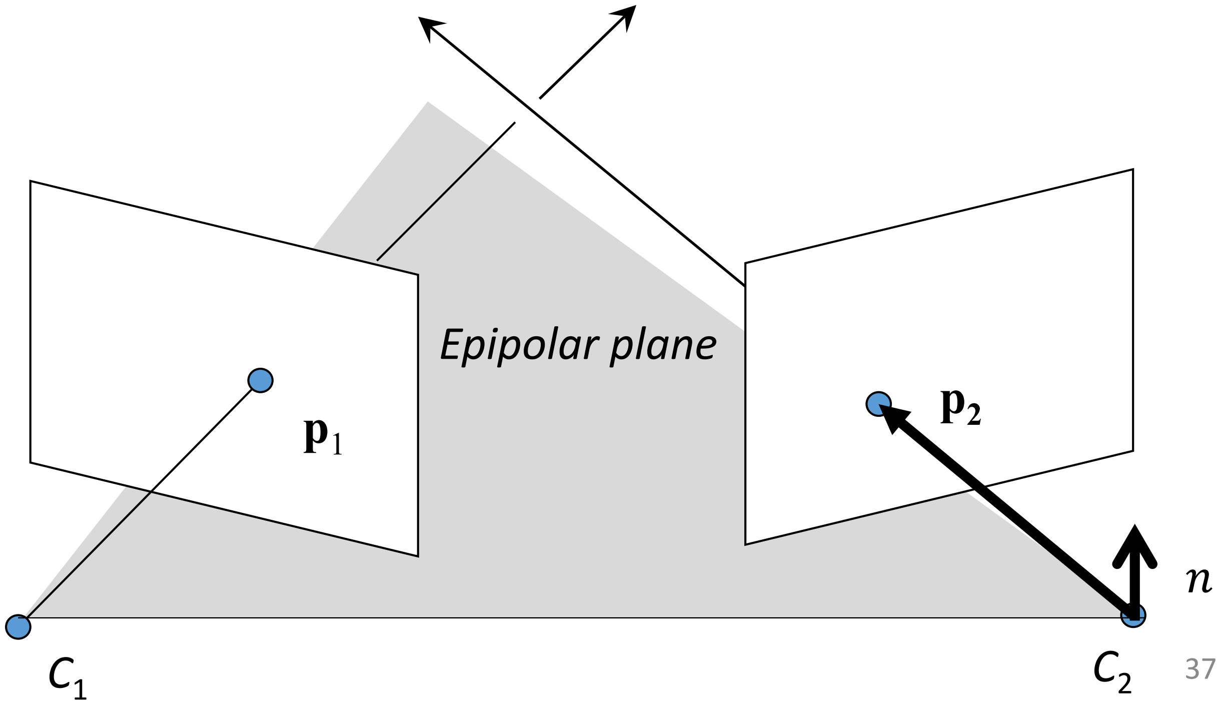
- We can see that this product depends on the angle θ between $\overline{\boldsymbol{p}}_2$ and the normal $\boldsymbol{n} = \boldsymbol{E}\boldsymbol{p}_1$ to the epipolar plane.
  It is non zero when $\overline{\boldsymbol{p}}_1$,$\overline{\boldsymbol{p}}_2$, and $\boldsymbol{T}$ are not coplanar
- What is the drawback of this error measure?



Directional Error

- Sum of squared **cosines of the angle from the epipolar plane**: $\text{err} = \sum_{i=1}^{N} (\cos(\theta_i))^2$

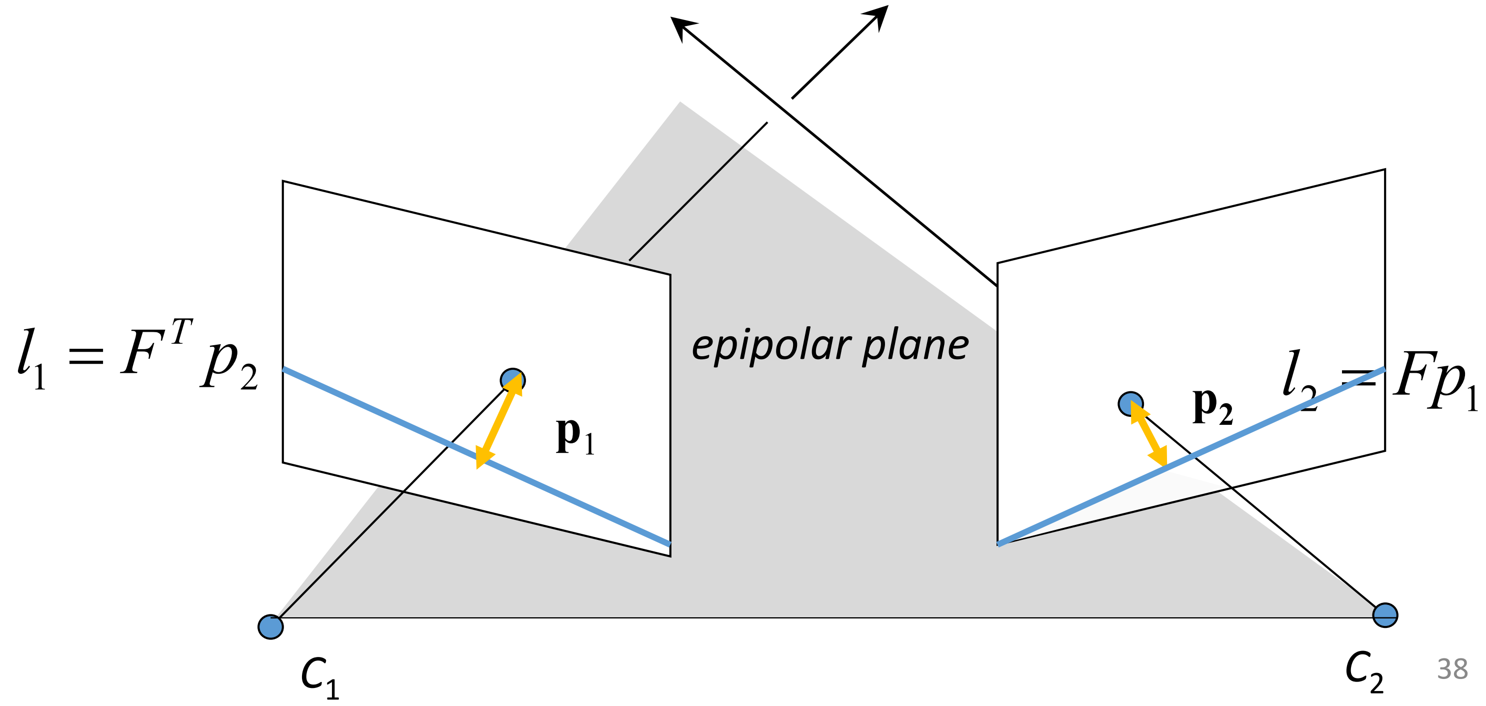
- It is obtained by **normalizing the algebraic error**: $\cos(\theta) = \frac{\overline{\boldsymbol{p}}_2^\top \boldsymbol{E} \overline{\boldsymbol{p}}_1}{\|\boldsymbol{p}_2\| \|\boldsymbol{E}\boldsymbol{p}_1\|}$



Epipolar Line Distance

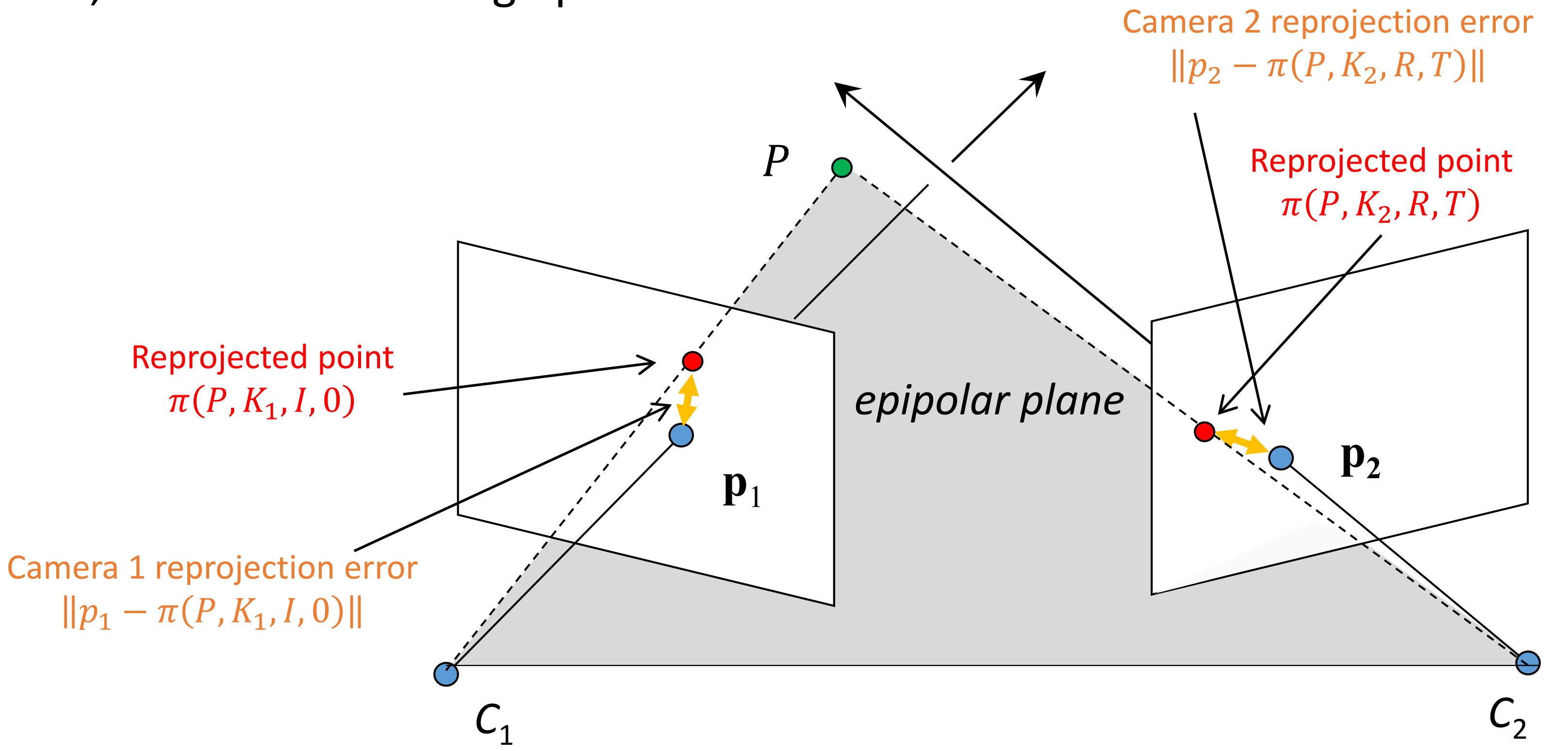
- **Sum of Squared Epipolar-Line-to-point Distances**: $err = \sum_{i=1}^{N} \left(d\left(p_1^i, l_1^i\right)\right)^2 + \left(d\left(p_2^i, l_2^i\right)\right)^2$

- Cheaper than reprojection error because does not require point triangulation



Reprojection Error

- Sum of the **Squared Reprojection Errors**: $err = \sum_{i=1}^{N} \|p_1^i - \pi(P^i, K_1, I, 0)\|^2 + \|p_2^i - \pi(P^i, K_2, R, T)\|^2$
- More **expensive** than the previous three errors because it **requires to first triangulate the 3D points!**
- **However it is the most popular because more accurate.** The reason is that the error is computed directly with the respect the raw input data, which are the image points



Things to remember

- SFM from 2 view
    - Calibrated and uncalibrated case
    - Proof of Epipolar Constraint
    - 8-point algorithm and algebraic error
    - Normalized 8-point algorithm
    - Algebraic, directional, Epipolar line distance, Reprojection error

Readings

- CH. 11.3 of Szeliski book, 2nd edition
- Ch. 14.2 of Corke book

Understanding Check

Are you able to answer the following questions?

- What's the minimum number of correspondences required for calibrated SFM and why?
- Are you able to derive the epipolar constraint?
- Are you able to define the essential matrix?
- Are you able to derive the 8-point algorithm?
- How many rotation-translation combinations can the essential matrix be decomposed into?
- Are you able to provide a geometrical interpretation of the epipolar constraint?
- Are you able to describe the relation between the essential and the fundamental matrix?
- Why is it important to normalize the point coordinates in the 8-point algorithm?
- Describe one or more possible ways to achieve this normalization.
- Are you able to describe the normalized 8-point algorithm?
- Are you able to provide quality metrics and their interpretation for the essential and fundamental matrix estimation?