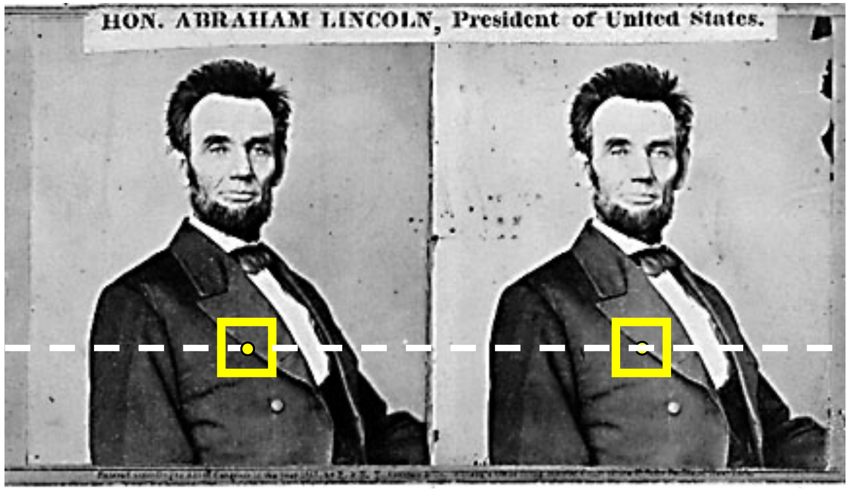
Folien zur Vorlesung am 13.05.2025

3D Computer Vision

# EPIPOLARGEOMETRIE

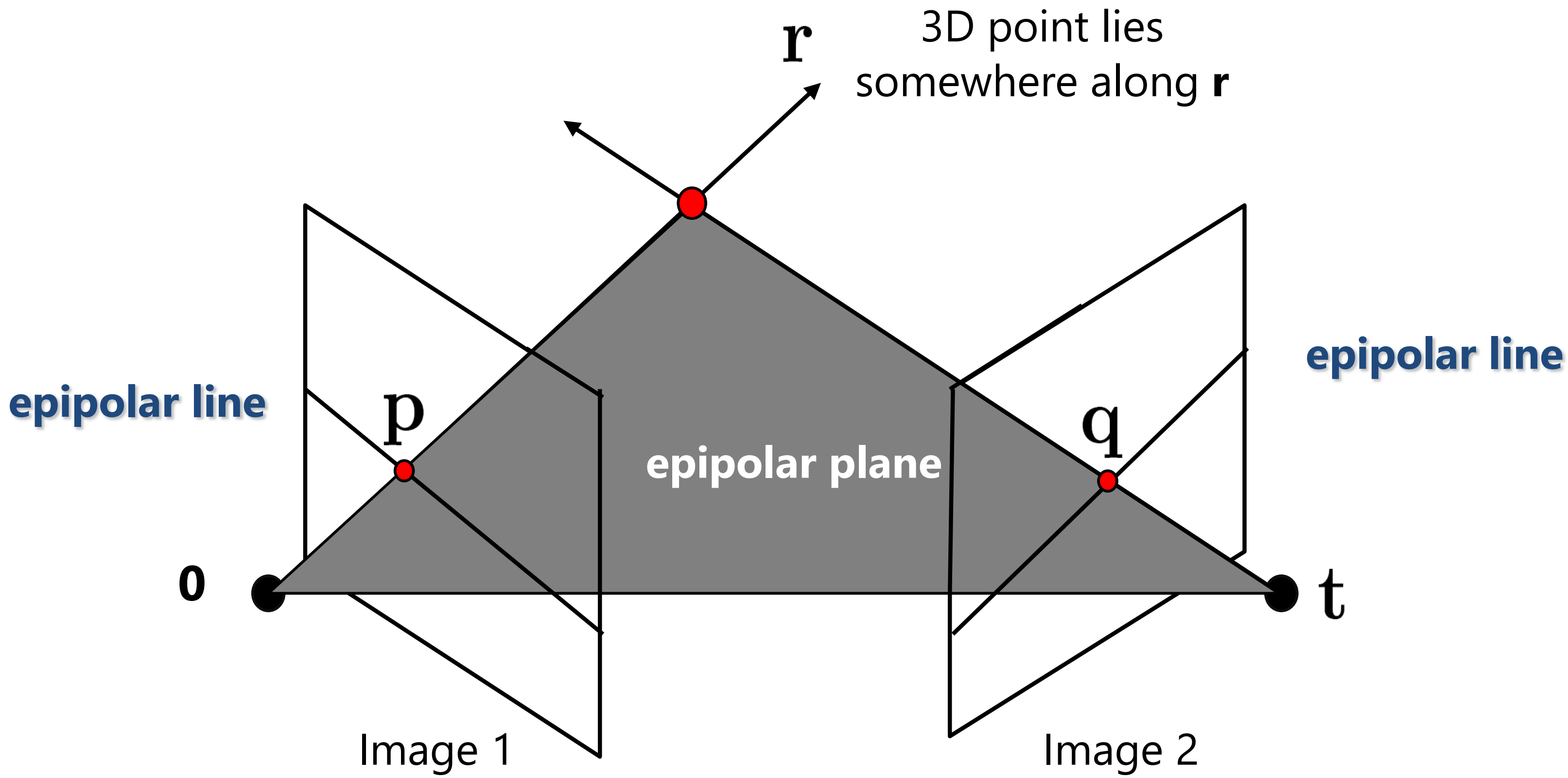
# Back to stereo



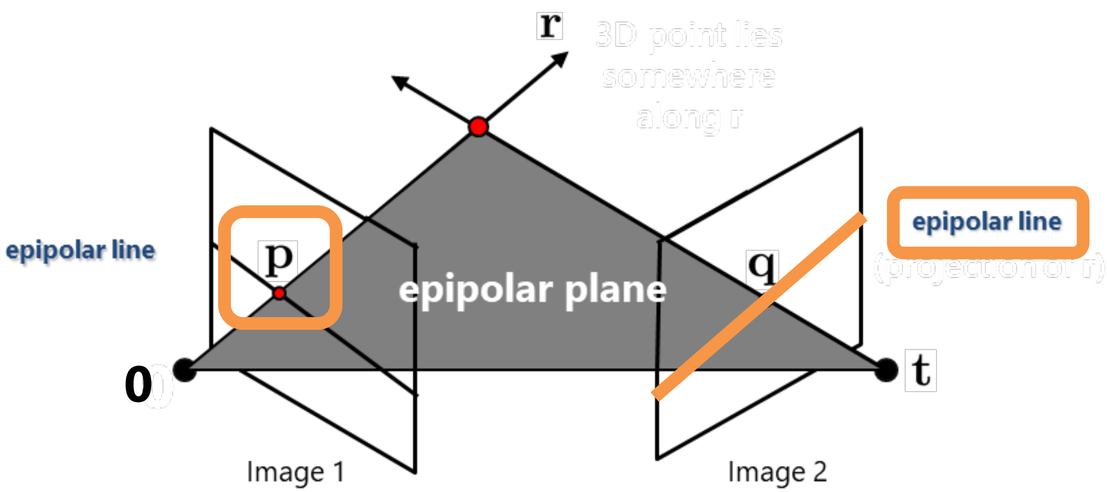

- Where do epipolar lines come from?

# Two-view geometry

- Where do epipolar lines come from?



# Fundamental matrix




- This *epipolar geometry* of two views is described by a very special 3x3 matrix $\mathbf{F}$ , called the *fundamental matrix*
- $\mathbf{F}$ maps (homogeneous) *points* in image 1 to *lines* in image 2!
- The epipolar line (in image 2) of point **p** is: $\mathbf{Fp}$
- *Epipolar constraint* on corresponding points: $\mathbf{q}^T\mathbf{F}\mathbf{p} = 0$

# Epipolar constraint

- Valid for all corresponding point pairs (p, q)

$$q^T \quad F \quad p \quad = \quad 0$$

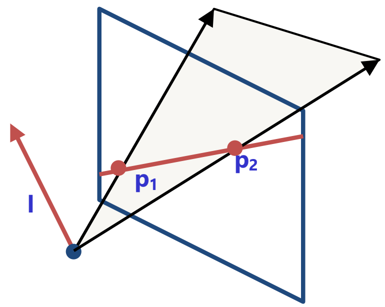
homogeneous image coordinates

3x3 matrix

scalar value

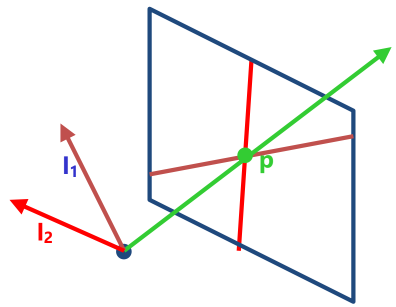
# Reminder: Point and line duality

- A line **l** is a homogeneous 3-vector
- It is $\perp$ to every point (ray) **p** on the line: $\mathbf{l} \cdot \mathbf{p} = 0$




What is the line **l** spanned by points $\mathbf{p_1}$ and $\mathbf{p_2}$?

- **l** is $\perp$ to $\mathbf{p_1}$ and $\mathbf{p_2}$ $\Rightarrow$ $\mathbf{l} = \mathbf{p_1} \times \mathbf{p_2}$
- **l** can be interpreted as a *plane normal*



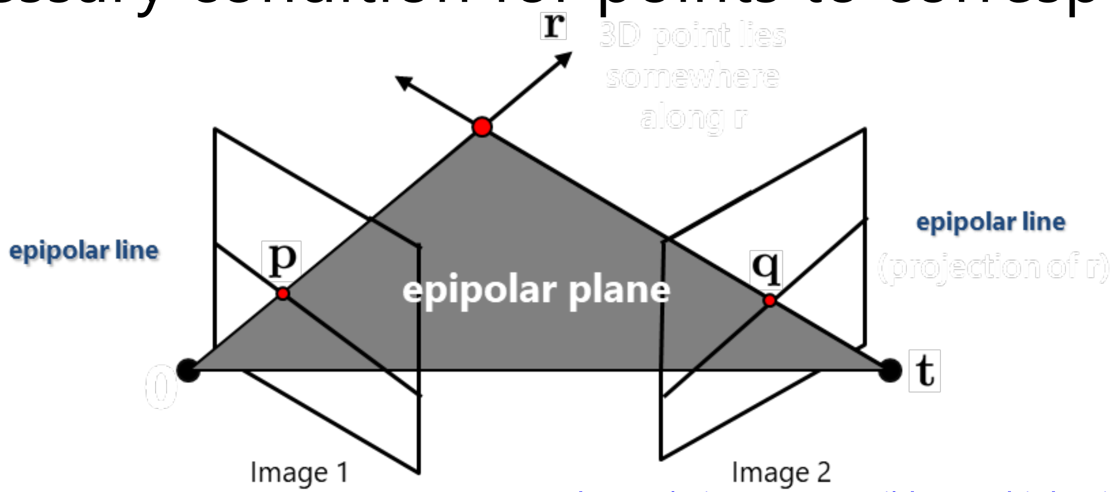

What is the intersection of two lines $\mathbf{l_1}$ and $\mathbf{l_2}$?

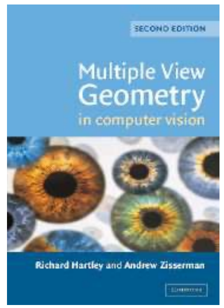
- **p** is $\perp$ to $\mathbf{l_1}$ and $\mathbf{l_2}$ $\Rightarrow$ $\mathbf{p} = \mathbf{l_1} \times \mathbf{l_2}$

Points and lines are *dual* in projective space

# Epipolar constraint

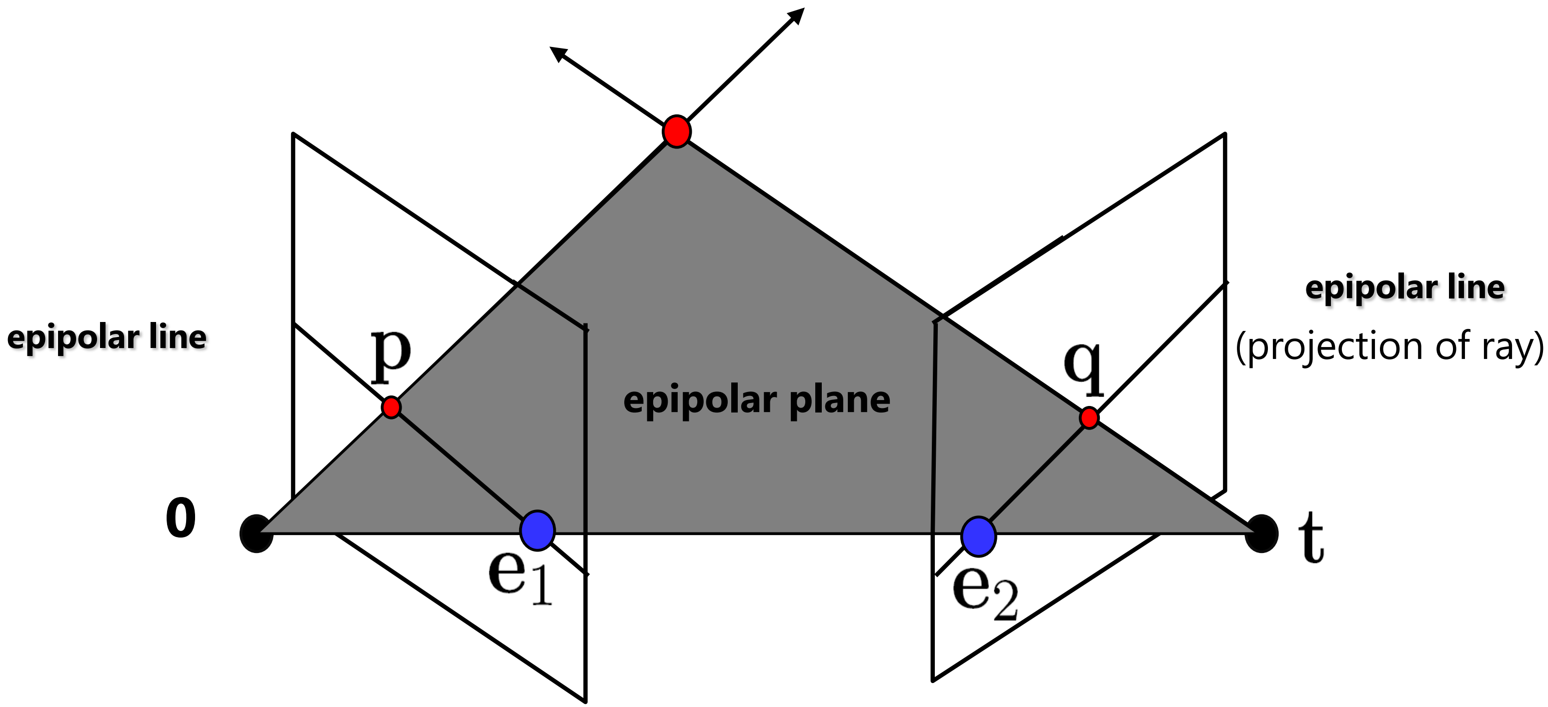
- If points p and q correspond, then q lies on the epipolar line l′ = Fp corresponding to the point q.
- In other words $0 = q^T l' = q'^T F p$.
- Conversely, if image points satisfy the relation $q'^T F p = 0$ then the rays defined by these points are coplanar, which is a necessary condition for points to correspond.



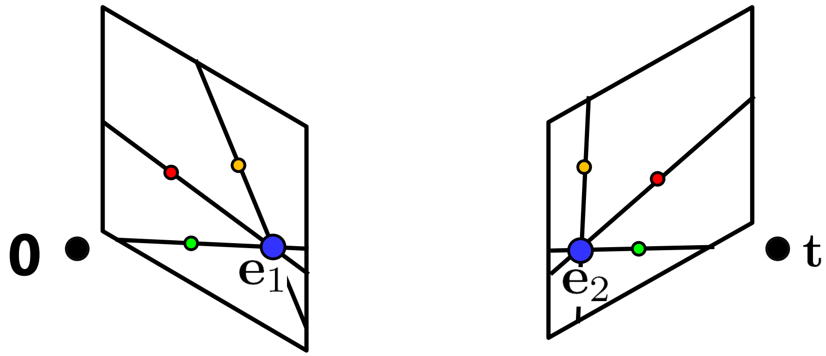

From [Hartley and Zisserman's "Bible": Multiple View Geometry in Computer Vision]

# Fundamental matrix




- Two special points: $\mathbf{e}_1$ and $\mathbf{e}_2$ (the *epipoles*): projection of one camera into the other

# Fundamental matrix



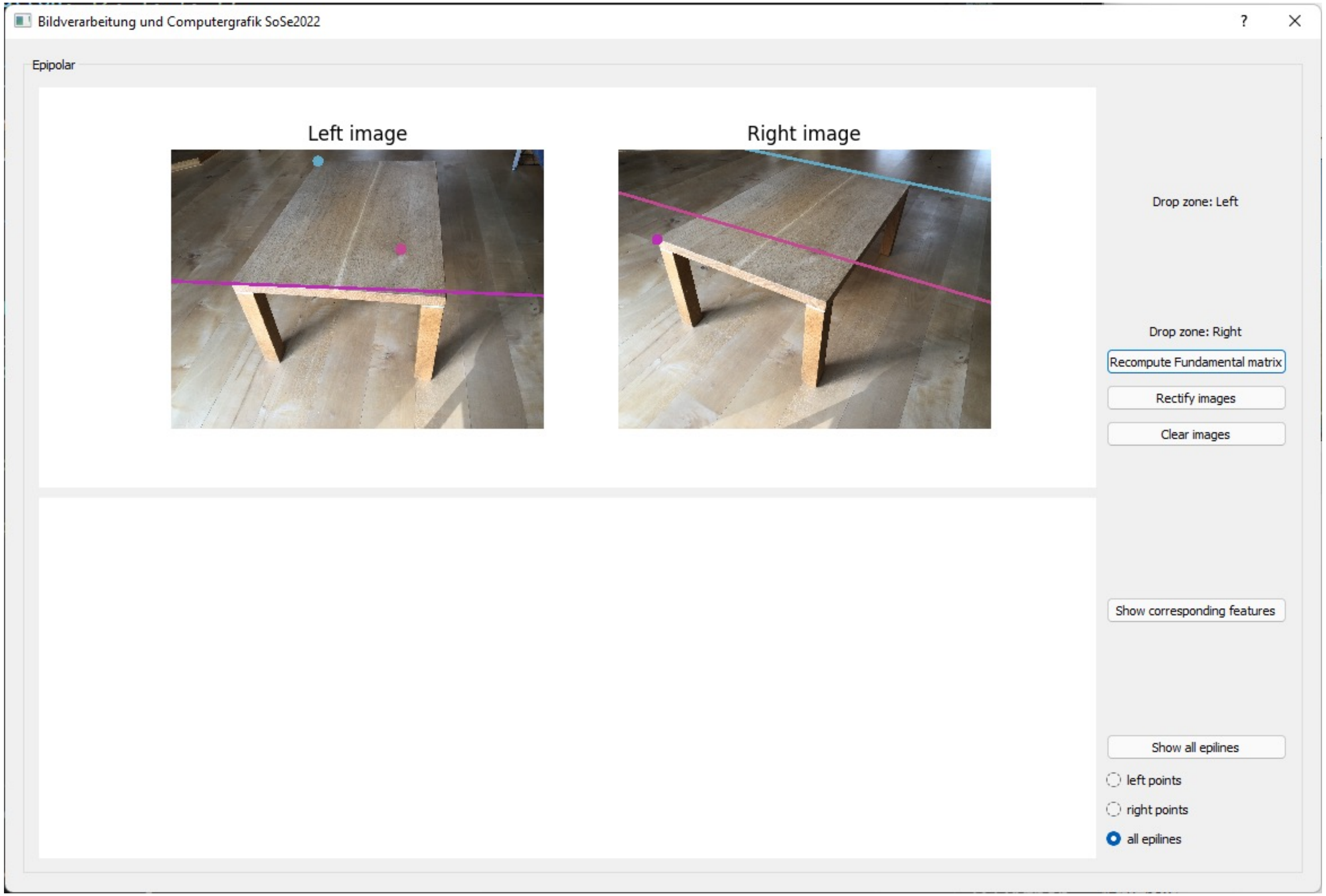

- Two special points: $\mathbf{e}_1$ and $\mathbf{e}_2$ (the *epipoles*): projection of one camera into the other
- All of the epipolar lines in an image pass through the epipole
- Epipoles may or may not be inside the image

# Epipoles

- If the epipoles are inside the image – what is visible in both images if they are taken at the same moment?

# Example

# Demo

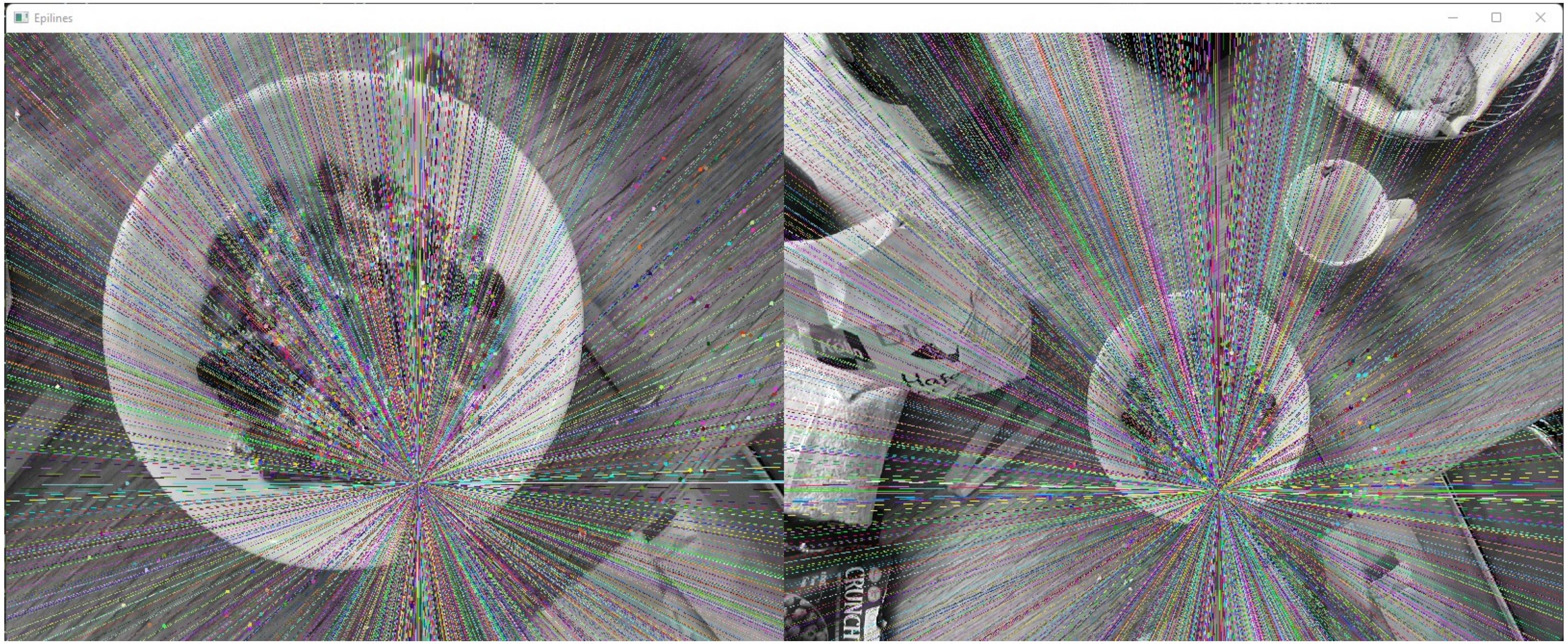

Bildverarbeitung und Computergrafik SoSe2022
Epipolar
Left image
Right image
Drop zone: Left
Drop zone: Right
Recompute Fundamental matrix
Rectify images
Clear images
Show corresponding features
Show all epilines
left points
right points
all epilines

# Epipoles

- Given those two images:

Where are the epipolar lines?

# Epipoles



# Epipoles

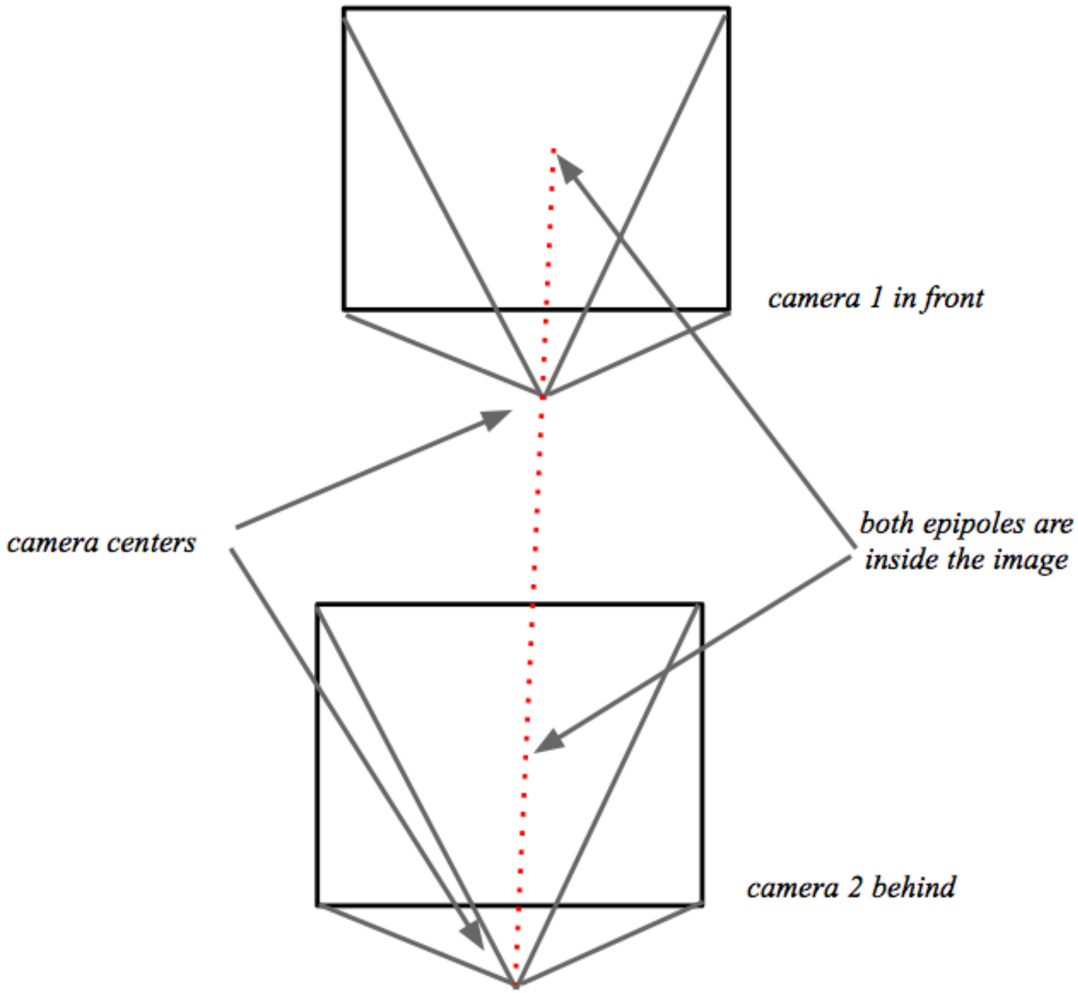
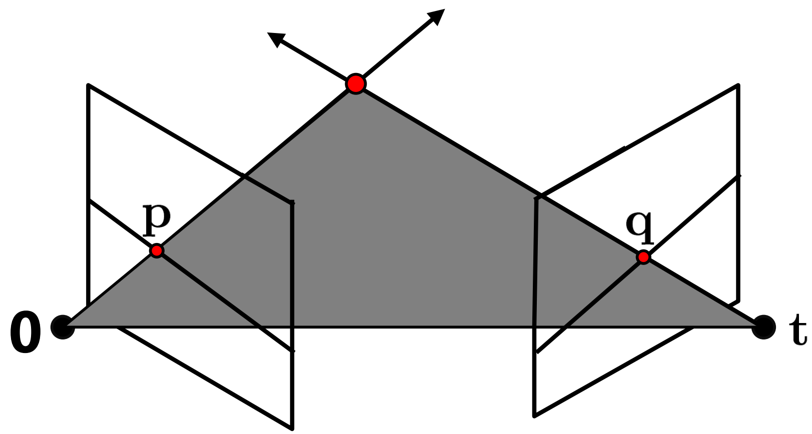



Image from koubiak posted at https://answers.opencv.org/question/17912/location-of-epipole/

# Properties of the Fundamental Matrix

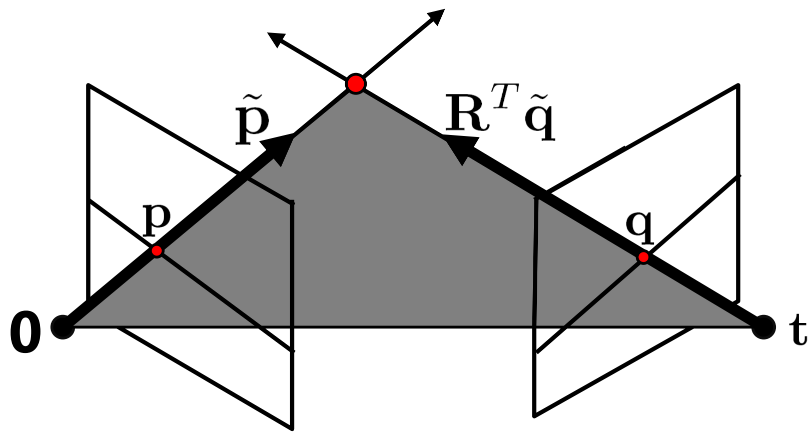
- $\mathbf{F}\mathbf{p}$ is the epipolar line associated with $\mathbf{p}$
- $\mathbf{F}^T\mathbf{q}$ is the epipolar line associated with $\mathbf{q}$
- $\mathbf{F}\mathbf{e}_1 = \mathbf{0}$ and $\mathbf{F}^T\mathbf{e}_2 = \mathbf{0}$
- $\mathbf{F}$ is rank **2**
- $\mathbf{F}$ has 9 values, but only defined up to a scale factor* and as the determinant is zero, 7 degrees of freedom remain.
- * $q^TFp = 0 = q^TkFp$ ← k as a scale factor

# Fundamental matrix




- Why does **F** exist?
- Let’s derive it…

# Fundamental matrix – calibrated case




$\mathbf{K}_1$ : intrinsics of camera 1

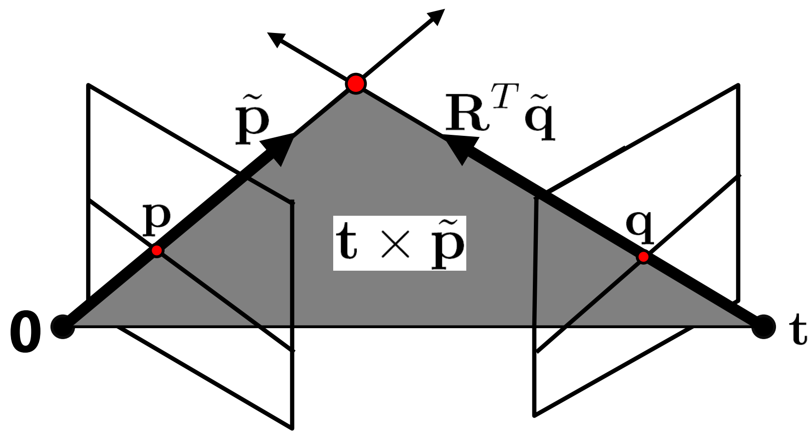
$\mathbf{K}_2$ : intrinsics of camera 2

$\mathbf{R}$ : rotation of image 2 w.r.t. camera 1

$\tilde{\mathbf{p}} = \mathbf{K}_1^{-1}\mathbf{p}$ : ray through **p** in camera 1's (and world) coordinate system

$\tilde{\mathbf{q}} = \mathbf{K}_2^{-1}\mathbf{q}$ : ray through **q** in camera 2's coordinate system

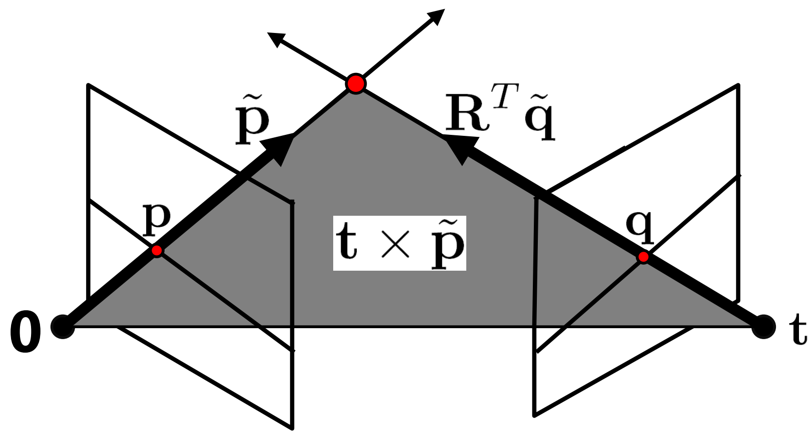
# Fundamental matrix – calibrated case




- $\tilde{\mathbf{p}}$ , $\mathbf{R}^T\tilde{\mathbf{q}}$ , and $\mathbf{t}$ are coplanar
- epipolar plane can be represented as with its normal $\mathbf{t} \times \tilde{\mathbf{p}}$

$$(\mathbf{R}^T\tilde{\mathbf{q}})^T(\mathbf{t} \times \tilde{\mathbf{p}}) = 0$$

# Fundamental matrix – calibrated case




$$(\mathbf{R}^T\tilde{\mathbf{q}})^T(\mathbf{t} \times \tilde{\mathbf{p}}) = 0$$

$$\tilde{\mathbf{q}}^T\mathbf{R}(\mathbf{t} \times \tilde{\mathbf{p}}) = 0$$

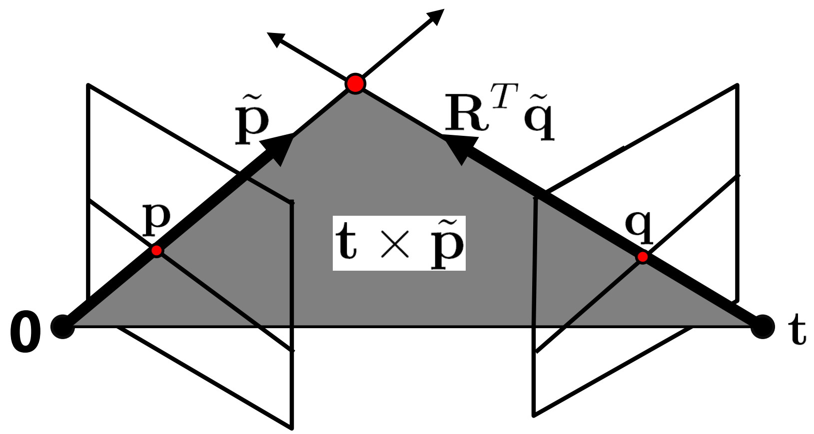
# Cross-product as linear operator

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

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

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

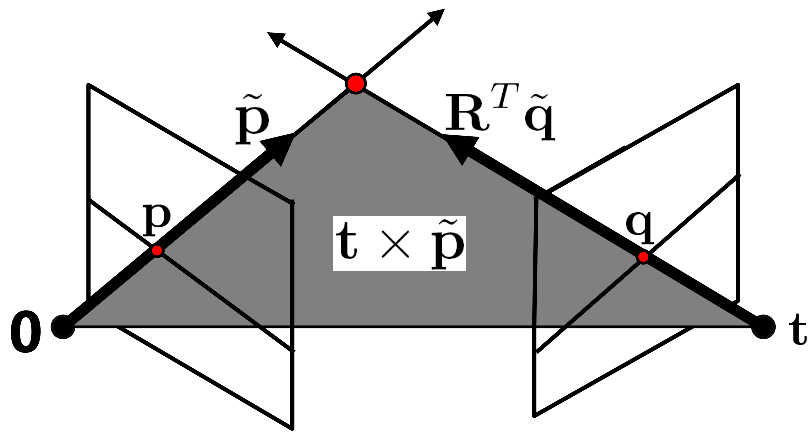
# Fundamental matrix – calibrated case




- One more substitution:
  – Cross product with **t** (on left) can be represented as a 3x3 matrix

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

# Fundamental matrix – calibrated case

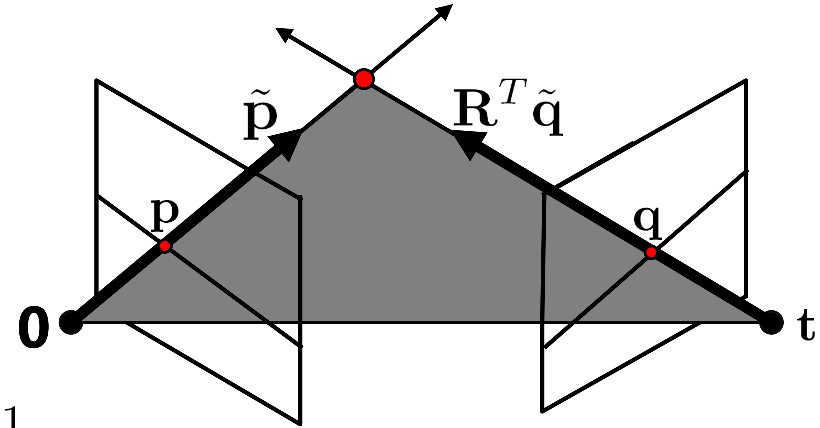



$$\tilde{\mathbf{q}}^T \mathbf{R}(\mathbf{t} \times \tilde{\mathbf{p}}) = 0$$

$$\Downarrow$$

$$\tilde{\mathbf{q}}^T \mathbf{R} \left[\mathbf{t}\right]_{\times} \tilde{\mathbf{p}} = 0$$

# Fundamental matrix – calibrated case



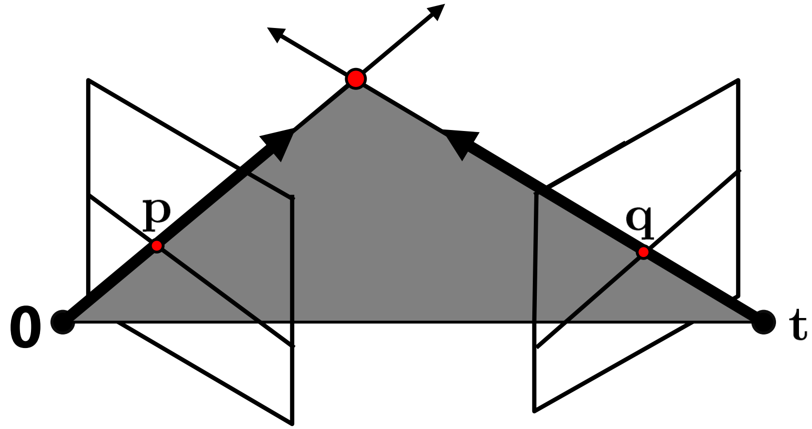

$\tilde{\mathbf{p}} = \mathbf{K}_1^{-1}\mathbf{p}$ : ray through **p** in camera 1's (and world) coordinate system

$\tilde{\mathbf{q}} = \mathbf{K}_2^{-1}\mathbf{q}$ : ray through **q** in camera 2's coordinate system

$$\tilde{\mathbf{q}}^T \underbrace{\mathbf{R}\,[\mathbf{t}]_\times}_{\mathbf{E}} \tilde{\mathbf{p}} = 0 \qquad \tilde{\mathbf{q}}^T\mathbf{E}\tilde{\mathbf{p}} = 0$$

$\mathbf{E}$ ← the Essential matrix

# Fundamental matrix – uncalibrated case




$\mathbf{K}_1$ : intrinsics of camera 1

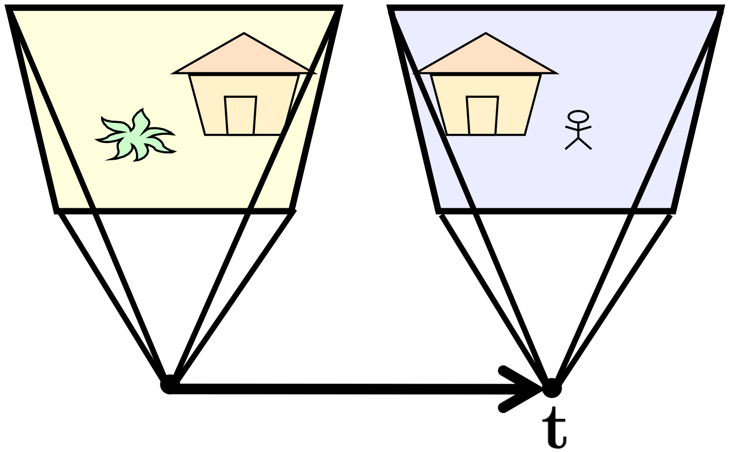
$\mathbf{K}_2$ : intrinsics of camera 2

$\mathbf{R}$ : rotation of image 2 w.r.t. camera 1

$$\mathbf{q}^T \underbrace{\mathbf{K}_2^{-T} \mathbf{R} \left[\mathbf{t}\right]_\times \mathbf{K}_1^{-1}}_{\mathbf{F}} \mathbf{p} = 0$$

$\mathbf{F}$ ← the Fundamental matrix

# Rectified case

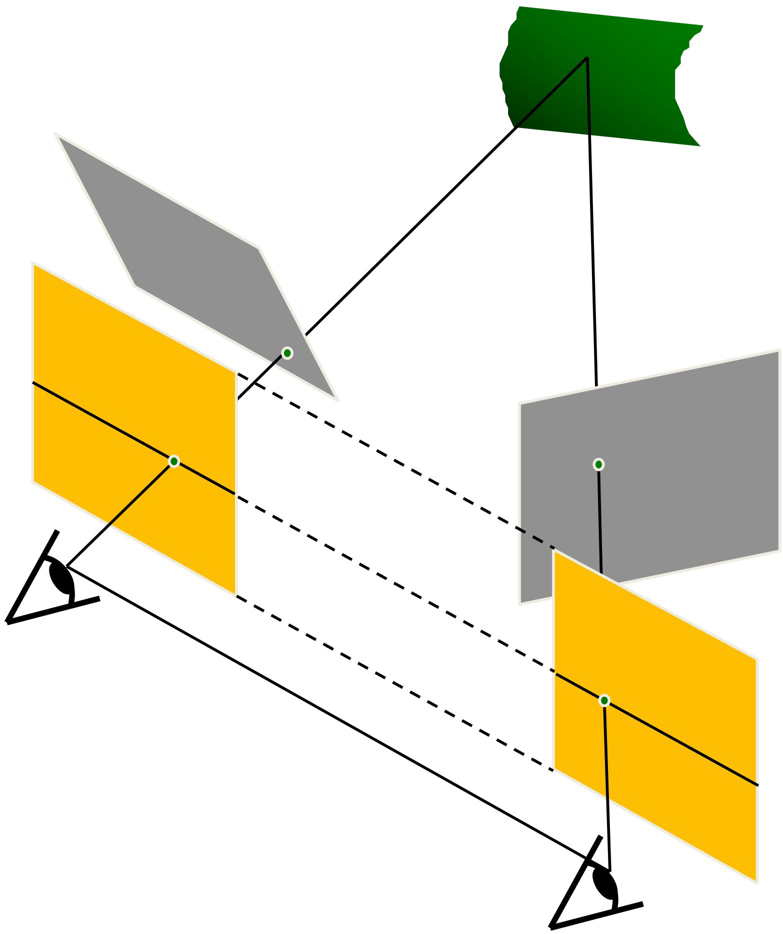



$$\mathbf{R} = \mathbf{I}_{3\times 3}$$
$$\mathbf{t} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$$

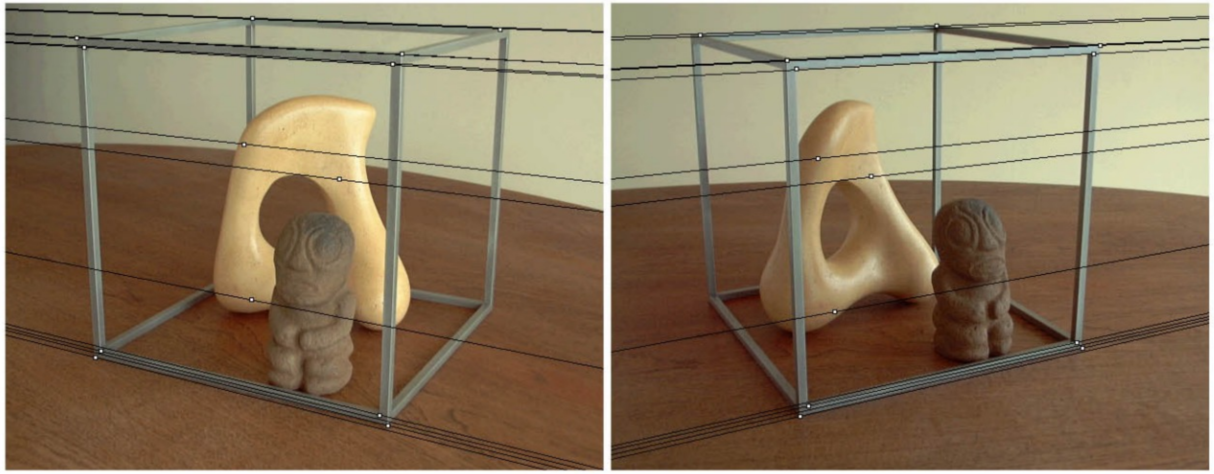
$$\mathbf{E} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

# Stereo image rectification

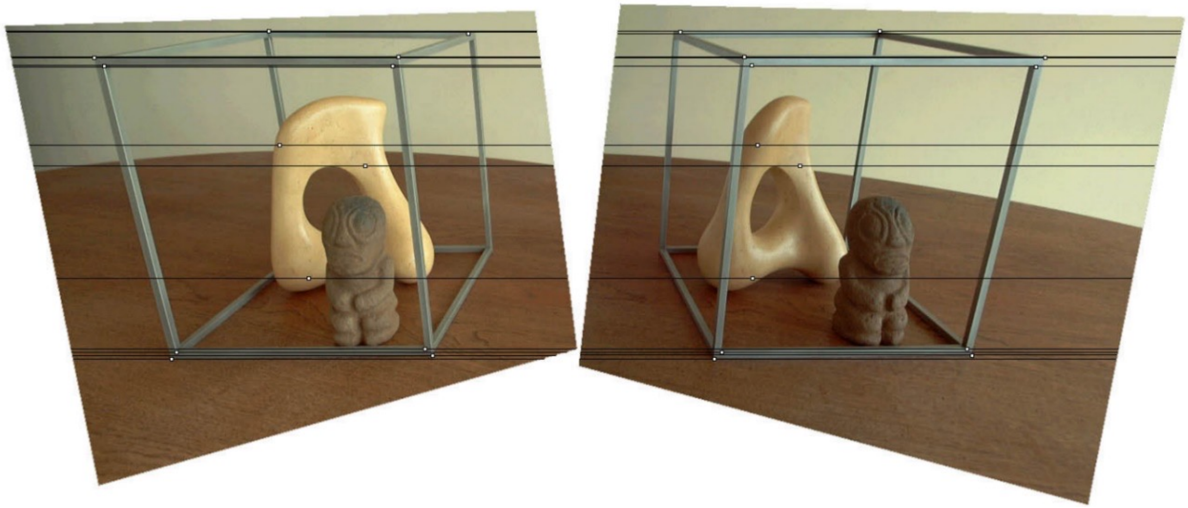
- reproject image planes onto a common plane
  - plane parallel to the line between optical centers
- pixel motion is horizontal after this transformation
- two homographies, one for each input image reprojection
- Various realizations:
  - C. Loop and Z. Zhang. [Computing Rectifying Homographies for Stereo Vision](). CVPR 1999.
  - OpenCV implements: Richard I Hartley. [Theory and practice of projective rectification](). *International Journal of Computer Vision*, 35(2):115–127, 1999.



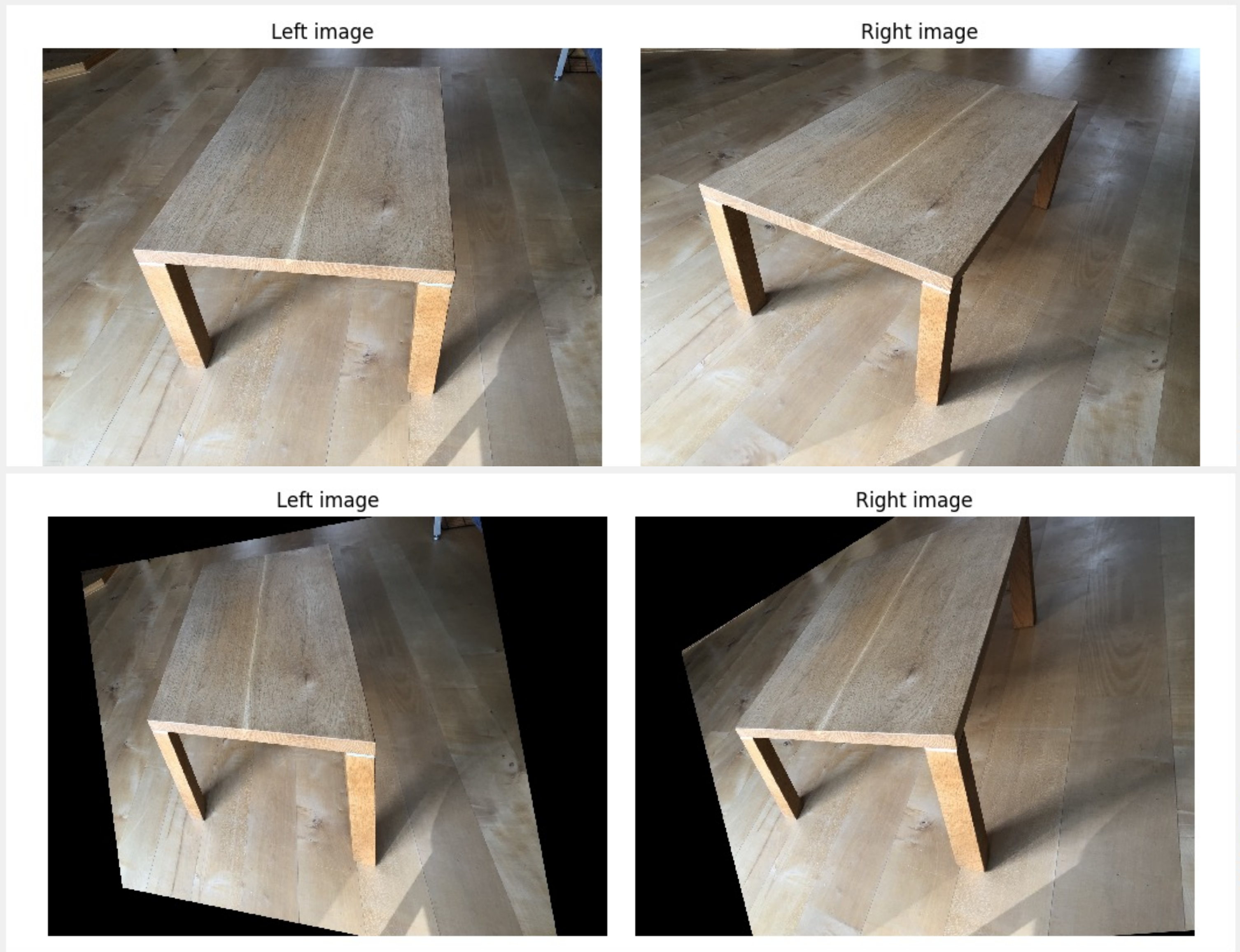
# Rectification example



Original stereo pair



# Rectification demo



# Estimating F


- If we don't know $\mathbf{K}_1$, $\mathbf{K}_2$, $\mathbf{R}$, or $\mathbf{t}$, can we estimate $\mathbf{F}$ for two images?
- Yes, given enough correspondences

# Estimating F

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

each match gives a linear equation

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

# Estimating F using 8-points

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

Like with homographies, instead of solving $\mathbf{Af} = 0$, we seek **f** to minimize $||\mathbf{Af}||$, least eigenvector of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ .

# Reminder: Least squares

$$\mathbf{A}\mathbf{t} = \mathbf{b}$$

Find **t** so that $||\mathbf{A}\mathbf{t} - \mathbf{b}||^2$ is minimized.

Define the normal equations

$$\mathbf{A}^\mathrm{T}\mathbf{A}\mathbf{t} = \mathbf{A}^\mathrm{T}\mathbf{b}$$

$$\mathbf{t} = \left(\mathbf{A}^\mathrm{T}\mathbf{A}\right)^{-1}\mathbf{A}^\mathrm{T}\mathbf{b}$$

https://de.wikipedia.org/wiki/Methode_der_kleinsten_Quadrate#L%C3%B6sung_des_Minimierungsproblems

# Reminder: Eigenvector

We want $Af$ as close to 0 as possible and $\|\mathbf{f}\|^2 = 1$:

$$\boxed{\min_{\mathbf{f}} \|A\mathbf{f}\|^2 \quad \text{such that } \|\mathbf{f}\|^2 = 1}$$

Constrained linear least squares problem

Known from homographies (GDV):

We know that:

$$\|A\mathbf{h}\|^2 = (A\mathbf{h})^T(A\mathbf{h}) = \mathbf{h}^T A^T A\mathbf{h} \quad \text{and} \quad \|\mathbf{h}\|^2 = \mathbf{h}^T\mathbf{h} = 1$$

$$\min_{\mathbf{h}}(\mathbf{h}^T A^T A\mathbf{h}) \quad \text{such that } \mathbf{h}^T\mathbf{h} = 1$$

Define Loss function $L(\mathbf{h}, \lambda)$:

$$L(\mathbf{h}, \lambda) = \mathbf{h}^T A^T A\mathbf{h} - \lambda(\mathbf{h}^T\mathbf{h} - 1)$$

Taking derivatives of $L(\mathbf{h}, \lambda)$ w.r.t $\mathbf{h}$: $\quad 2A^T A\mathbf{h} - 2\lambda\mathbf{h} = \mathbf{0}$

$$\boxed{A^T A\mathbf{h} = \lambda\mathbf{h}} \quad \text{Eigenvalue Problem}$$

Eigenvector $\mathbf{h}$ with smallest eigenvalue $\lambda$ of matrix $A^T A$ minimizes the loss function $L(\mathbf{h})$.

From https://fpcv.cs.columbia.edu/

# 8-point algorithm

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

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

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix}, \quad \text{let} \quad \Sigma' = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

then $\mathbf{F}' = \mathbf{U}\Sigma'\mathbf{V}^{\mathrm{T}}$ is the solution (closest rank-2 matrix to **F**)

# 8-point algorithm

```
% 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 rank2 constraint
[U,D,V] = svd(F);
F = U*diag([D(1,1) D(2,2) 0])*V';
```

# 8-point algorithm

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

# Problem with 8-point algorithm

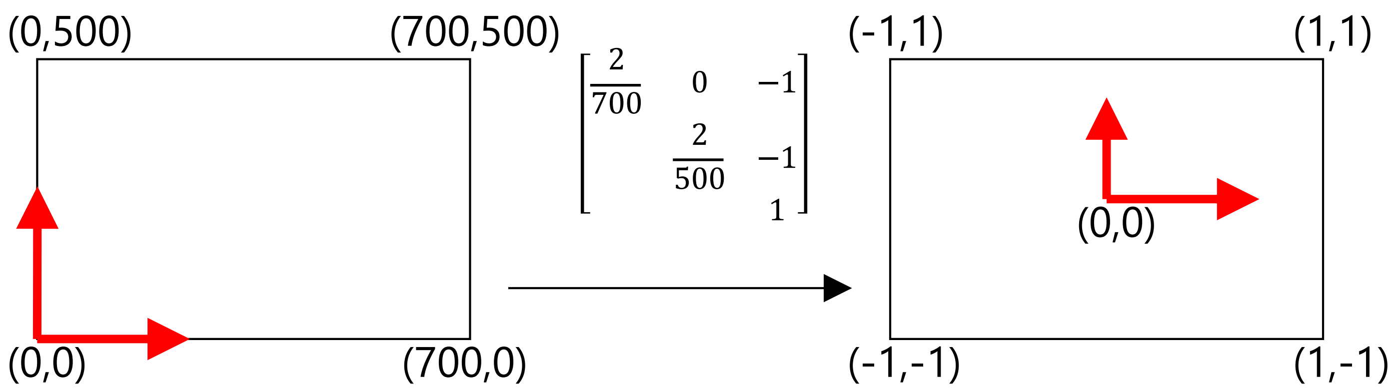
$$\begin{array}{c} \begin{bmatrix} u_1u_1' & v_1u_1' & u_1' & u_1v_1' & v_1v_1' & v_1' & u_1 & v_1 & 1 \\ u_2u_2' & v_2u_2' & u_2' & u_2v_2' & v_2v_2' & v_2' & u_2 & v_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_n' & v_nu_n' & u_n' & u_nv_n' & v_nv_n' & v_n' & u_n & v_n & 1 \end{bmatrix} \\ \begin{array}{ccccccccc} \sim 10000 & \sim 10000 & \sim 100 & \sim 10000 & \sim 10000 & \sim 100 & \sim 100 & \sim 100 & 1 \end{array} \end{array} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0$$

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

# Normalized 8-point algorithm

normalized least squares yields good results

Transform image to ~[-1,1]x[-1,1]




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

# Normalized 8-point algorithm

- Transform input by $\hat{\mathbf{x}}_\mathbf{i} = \mathbf{T}\mathbf{x}_\mathbf{i}$ , $\hat{\mathbf{x}}'_\mathbf{i} = \mathbf{T}\mathbf{x}'_\mathbf{i}$
- Call 8-point on $\hat{\mathbf{x}}_\mathbf{i}, \hat{\mathbf{x}}'_\mathbf{i}$ to obtain $\hat{\mathbf{F}}$
- $\mathbf{F} = \mathbf{T}'^{\mathrm{T}}\hat{\mathbf{F}}\mathbf{T}$

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

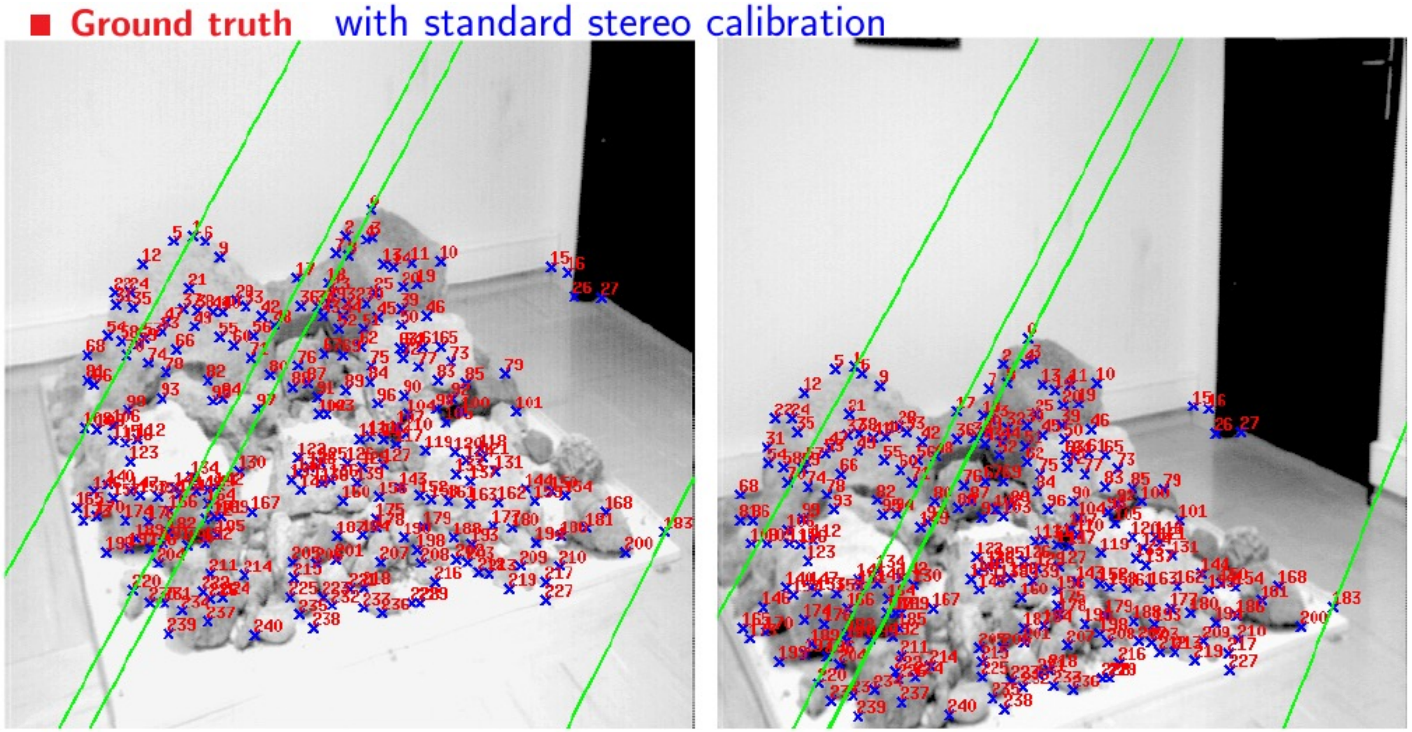
$$\boxed{\hat{\mathbf{x}}'^{\mathrm{T}}\mathbf{T}'^{-\mathrm{T}}}\,\mathbf{F}\,\boxed{\mathbf{T}^{-1}\hat{\mathbf{x}}} = 0$$

$$\hat{\mathbf{F}}$$

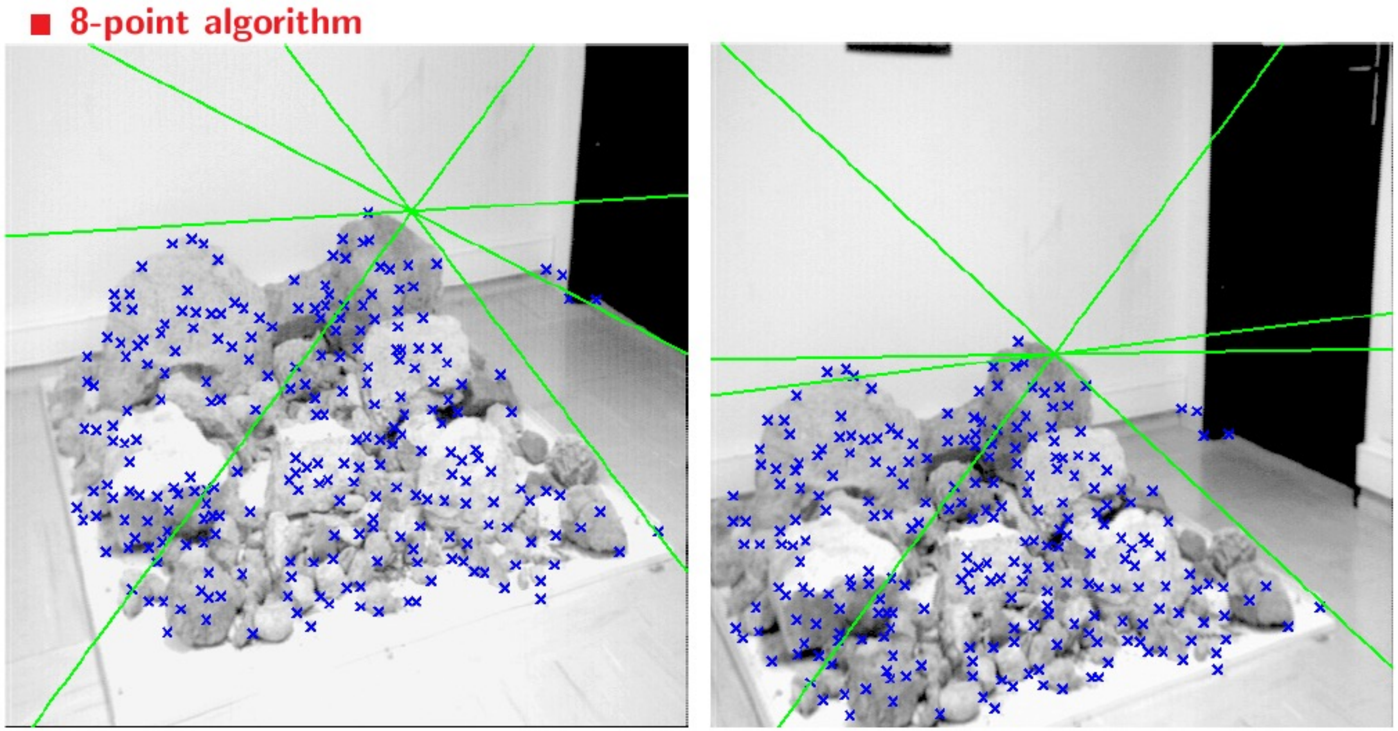
# Normalized 8-point algorithm

```
[x1, T1] = normalise2dpts(x1);
[x2, T2] = normalise2dpts(x2);

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);

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

[U,D,V] = svd(F);
F = U*diag([D(1,1) D(2,2) 0])*V';

% Denormalise
F = T2'*F*T1;
```

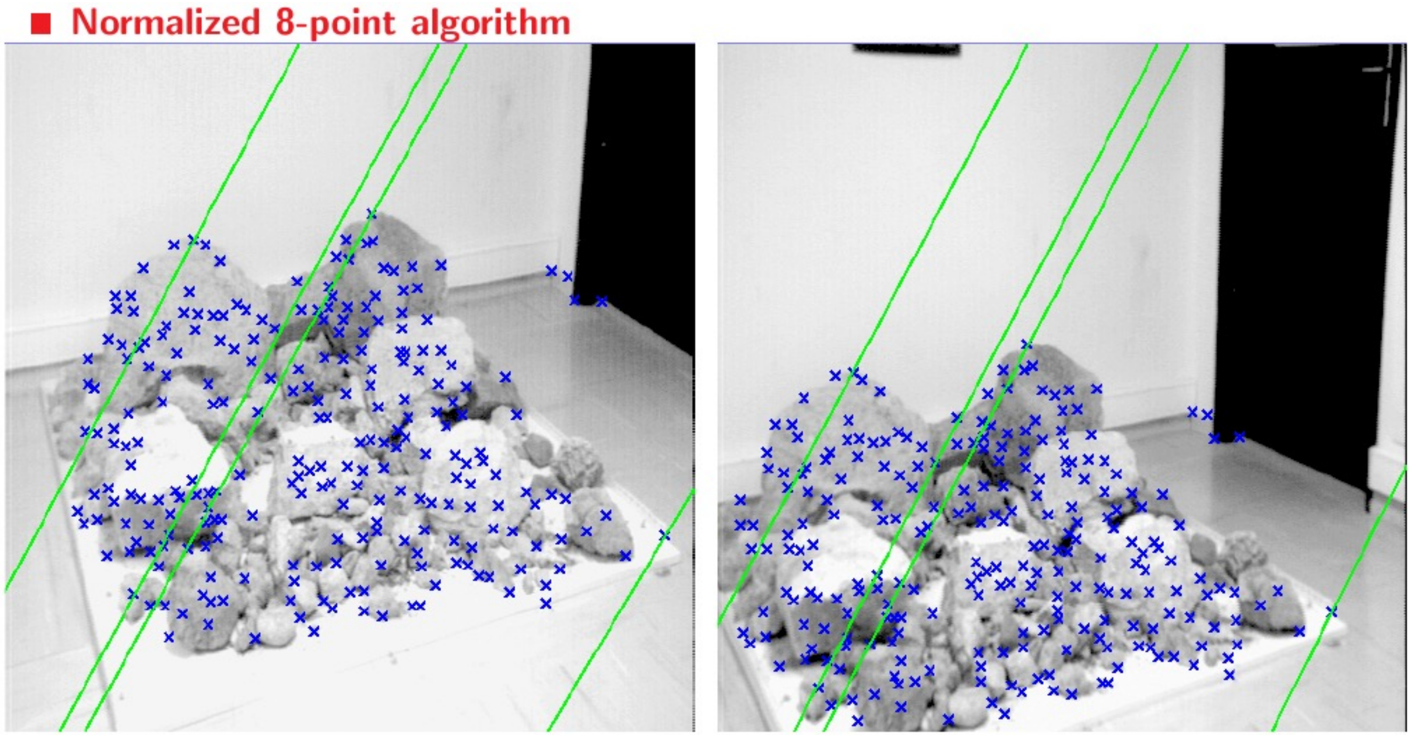
# Results (ground truth)



# Results (ground truth)



# Results (normalized 8-point algorithm)



# What about more than two views?

- The geometry of three views is described by a 3 x 3 x 3 tensor called the *trifocal tensor*
- The geometry of four views is described by a 3 x 3 x 3 x 3 tensor called the *quadrifocal tensor*
- After this it starts to get complicated…so usually numerical solutions are implemented.