Lecture 19

Structure from motion

Administrative

A5 is out

- Due May 14th

Exam

- Mon, Mar 17 10:30-12:20
- Same room as lecture: G20
- List of topics posted on EdStem

Makeup exam

- Friday 14th (emails have been sent out with location and time)

So far: camera transformation

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I} | -\mathbf{C}]$$

3x3
intrinsics

3x3
3D rotation

3x3
identity

3x1
3D translation

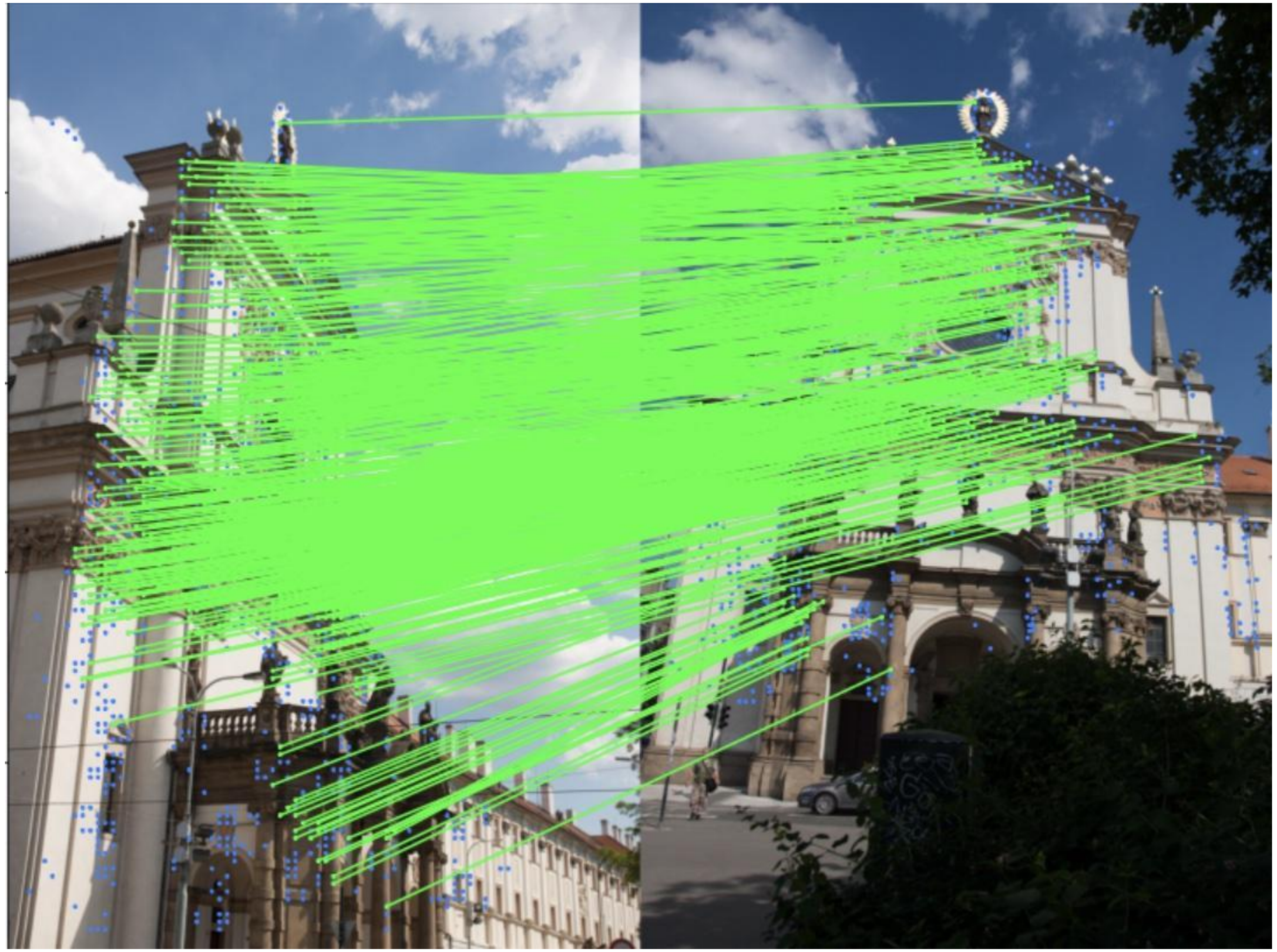
https://www.cs.cmu.edu/~16385/s17/Slides/11.1_Camera_matrix.pdf

So far: camera calibration

Estimate camera parameters (K[R|t) by measuring

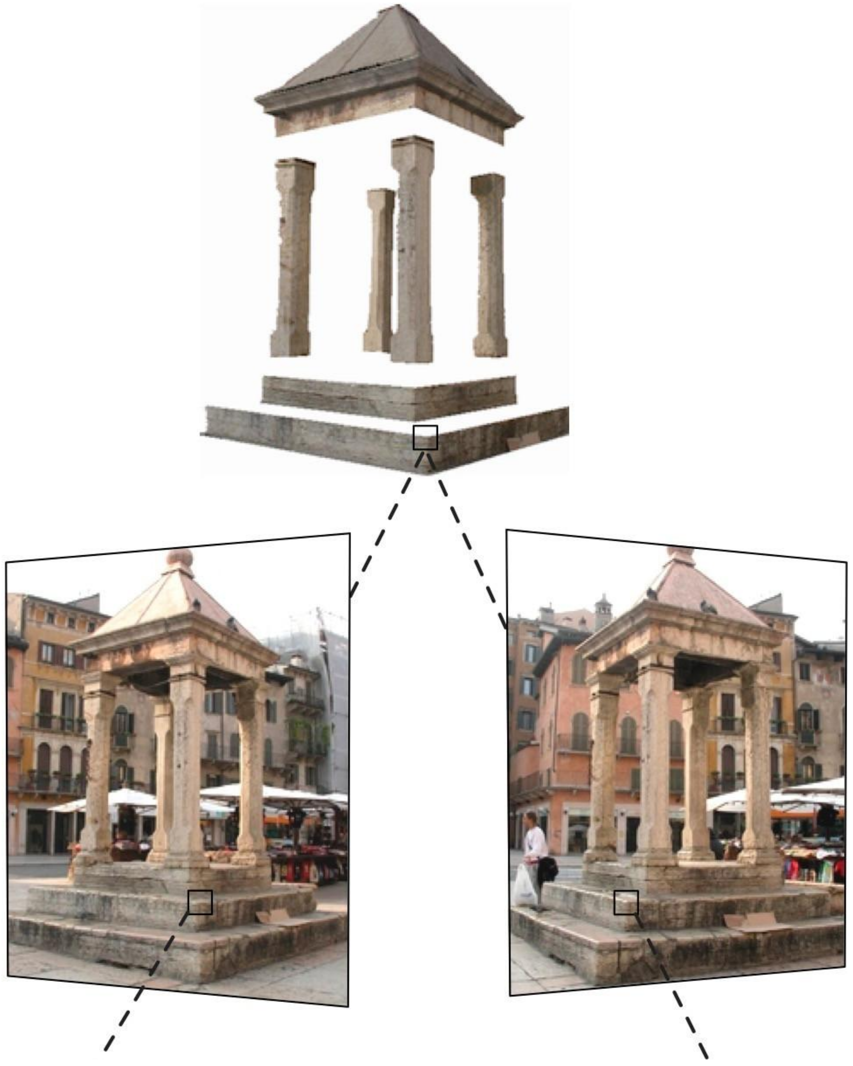
- real world points $\mathbf{X}_i$ in world space
- the same points in pixel space $\mathbf{x}_i$
- Solve for K[R | t] using SVD after posing the problem as $\mathbf{Ap} = 0$
    - where $\mathbf{p}$ are camera parameters and $\mathbf{A}$ is obtained from mapping $\mathbf{X}_i$ to $\mathbf{x}_i$

So far: in other words, we can estimate camera motion given multiple images…



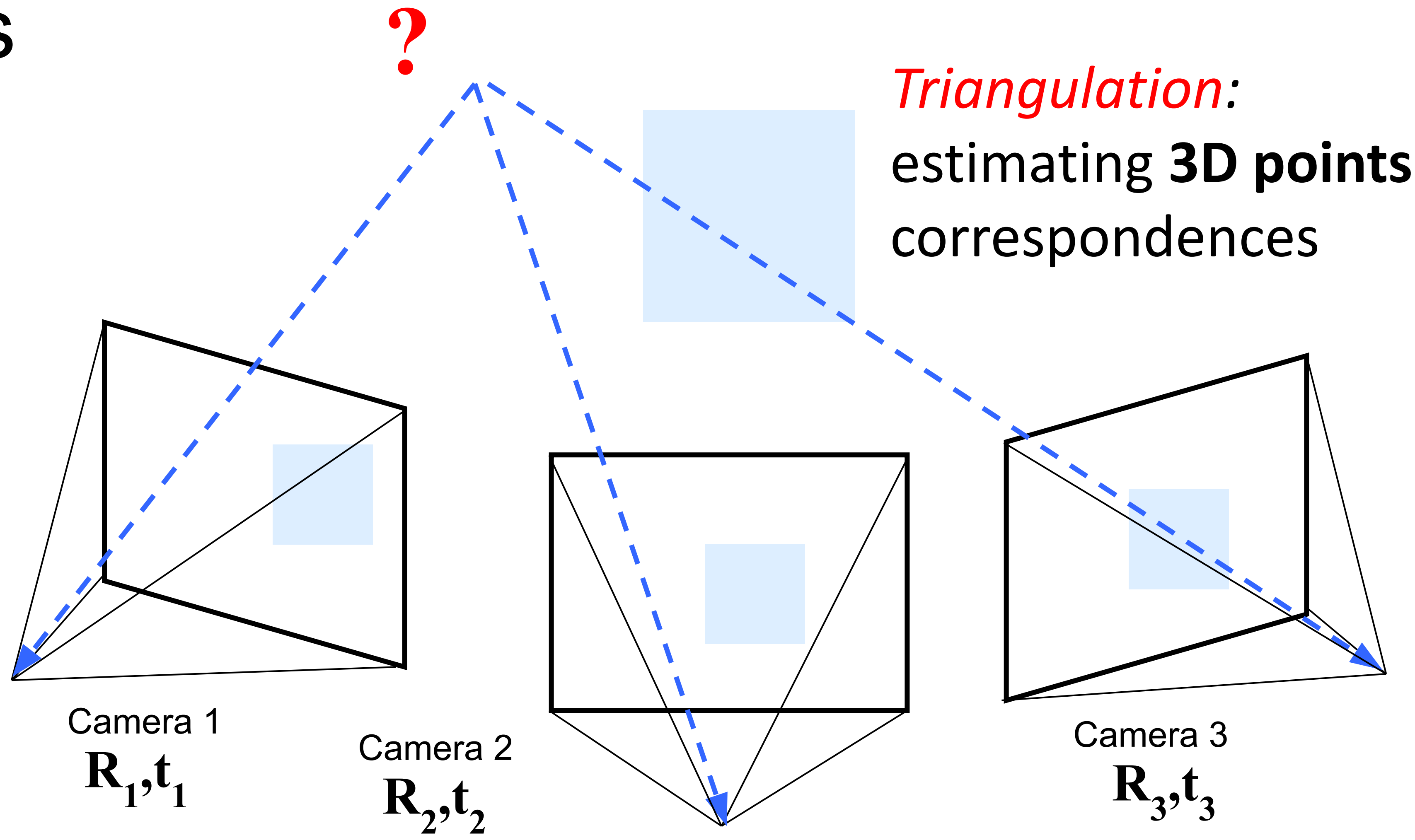
https://kornia.readthedocs.io/en/latest/applications/image_matching.html

What we want to do today: extract structure!

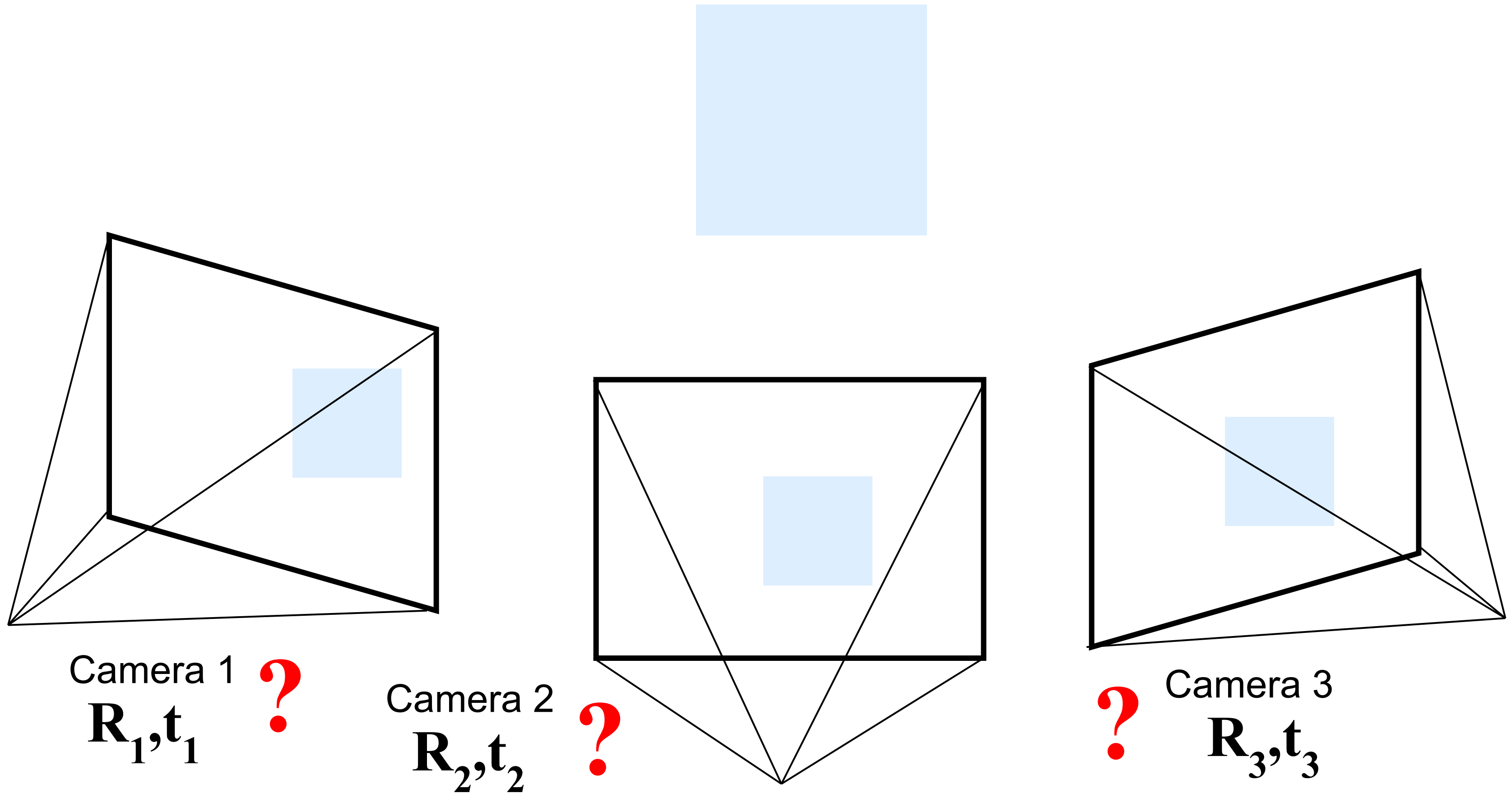


If we knew the camera parameters, we would be able to find the 3D world coordinates of things

Triangulation:
estimating **3D points** from **2D** correspondences



First we need to estimate motion (R, t) from 2D correspondences (we already did this)



Slide credit:
Noah Snavely

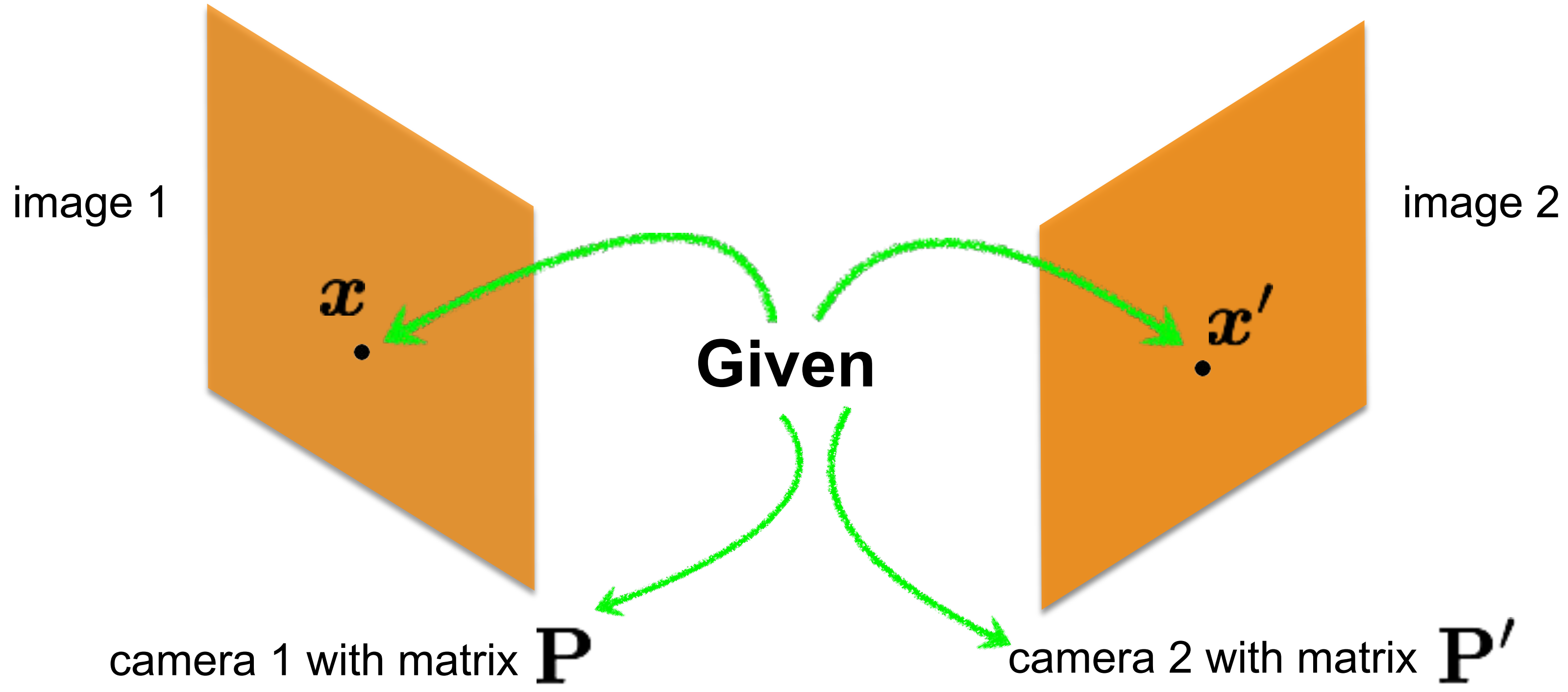
Today's agenda

- Triangulation
- Epipolar geometry
- Essential matrix
- Fundamental matrix
- Structure from motion

Today's agenda

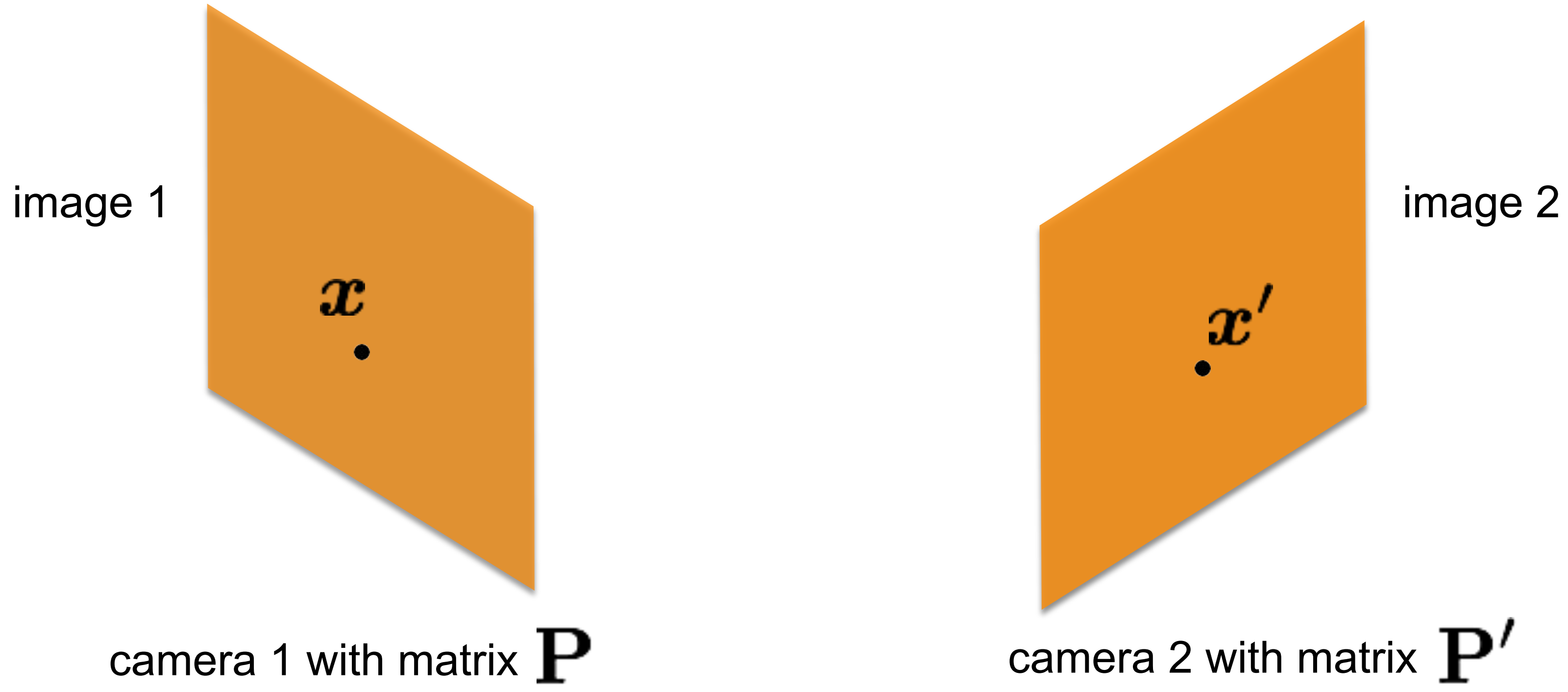
- Triangulation
- Epipolar geometry
- Essential matrix
- Fundamental matrix
- Structure from motion

Triangulation



Triangulation

Where is the 3D point that maps to the two x’s?



Triangulation formalization

Given a set of (noisy) matched points

$$\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$$

and camera matrices

$$\mathbf{P}, \mathbf{P}'$$

Estimate the 3D point

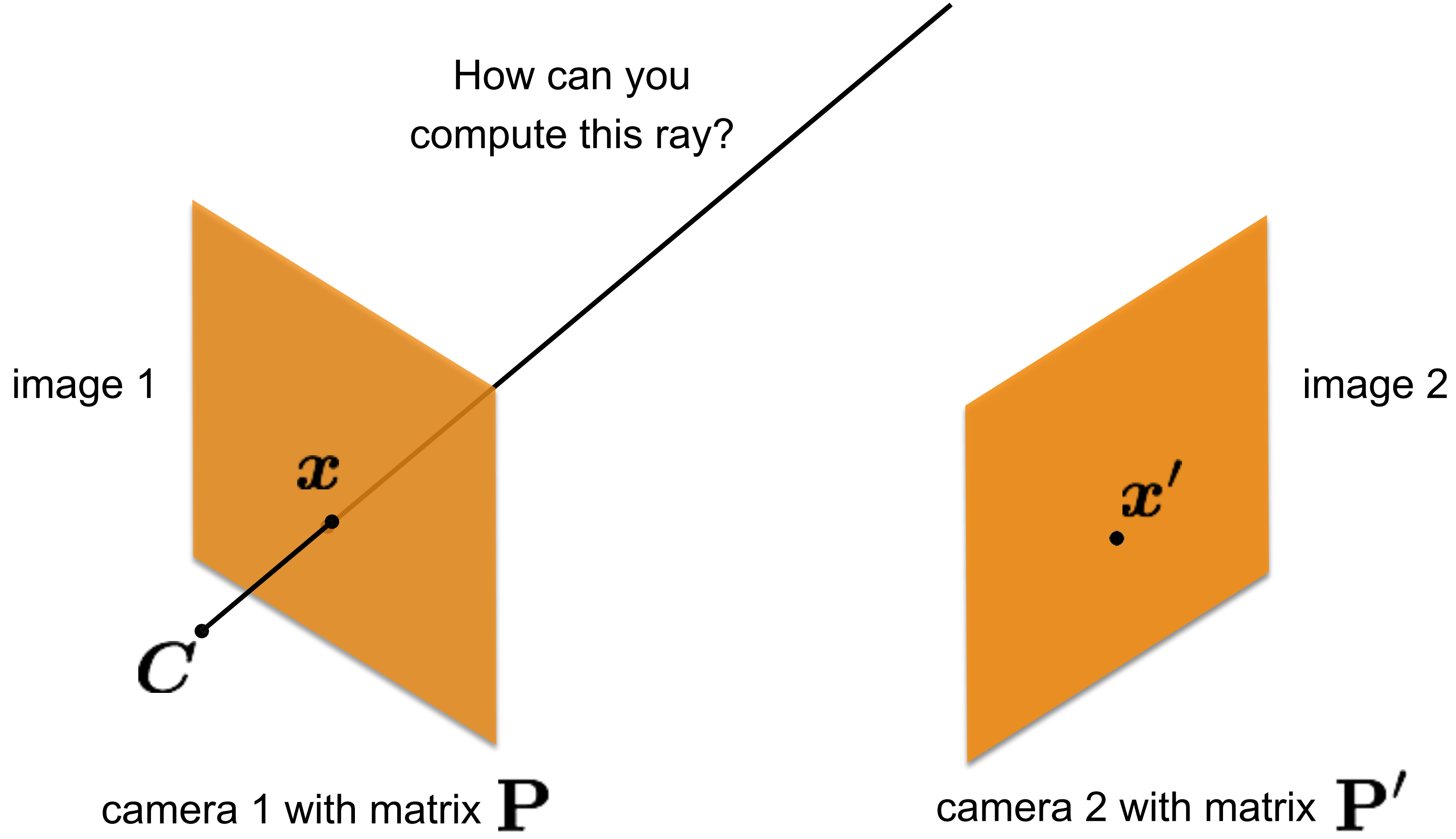
$$\mathbf{X}$$

Triangulation equation

$$\underbrace{\mathbf{x}}_{\text{known}} = \underbrace{\mathbf{P}}_{\text{known}} X$$

Q. Can we compute **X** from a single correspondence **x**?

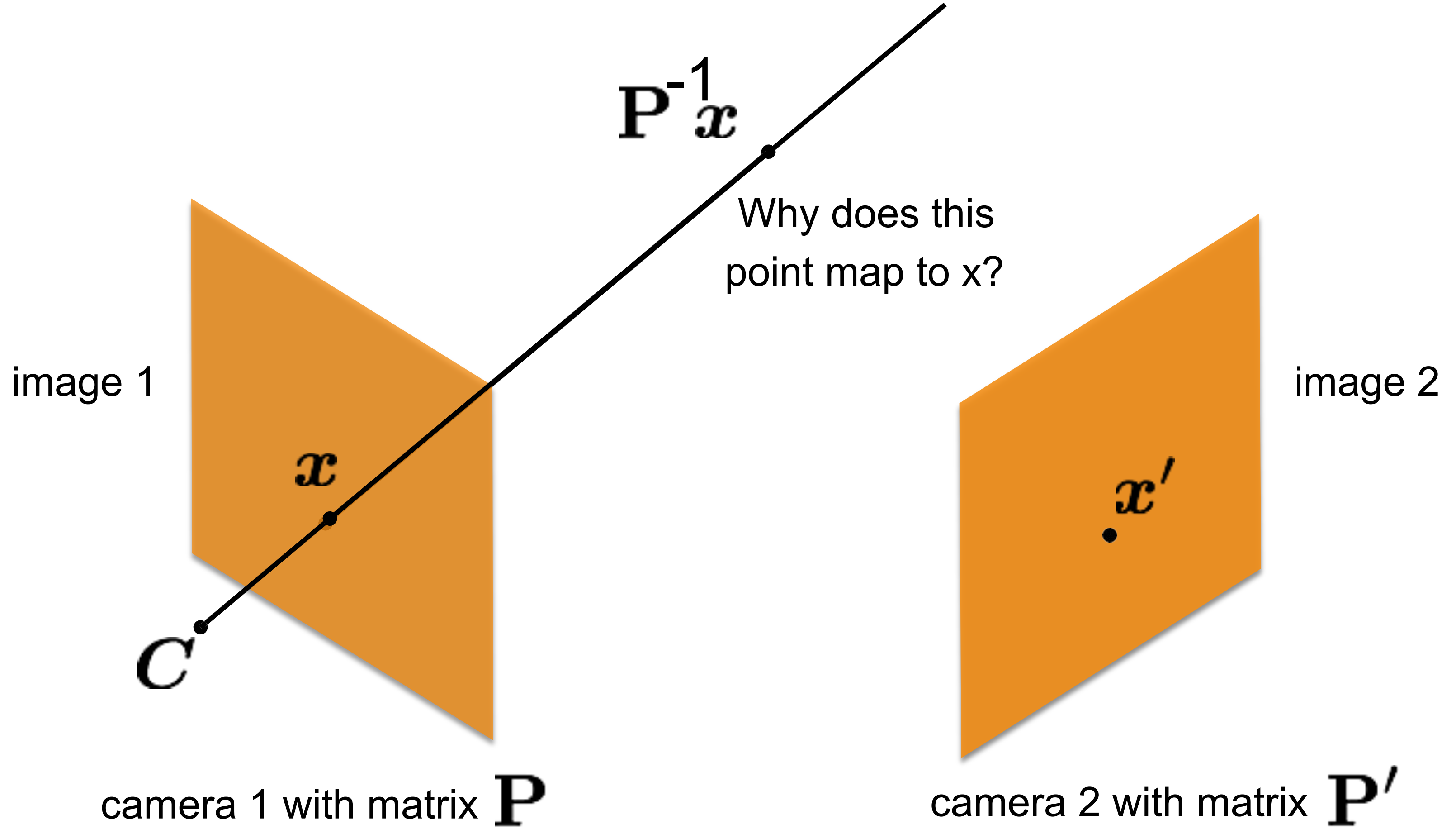
Triangulation



Triangulation

Apply the **pseudo-inverse** of **P** on **x**. Then connect the two points.

This procedure is called backprojection



$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

We lose information going from 3D to 2D.
Specifically, we lose depth information

$$\mathbf{x} = \alpha\mathbf{P}\boldsymbol{X}$$

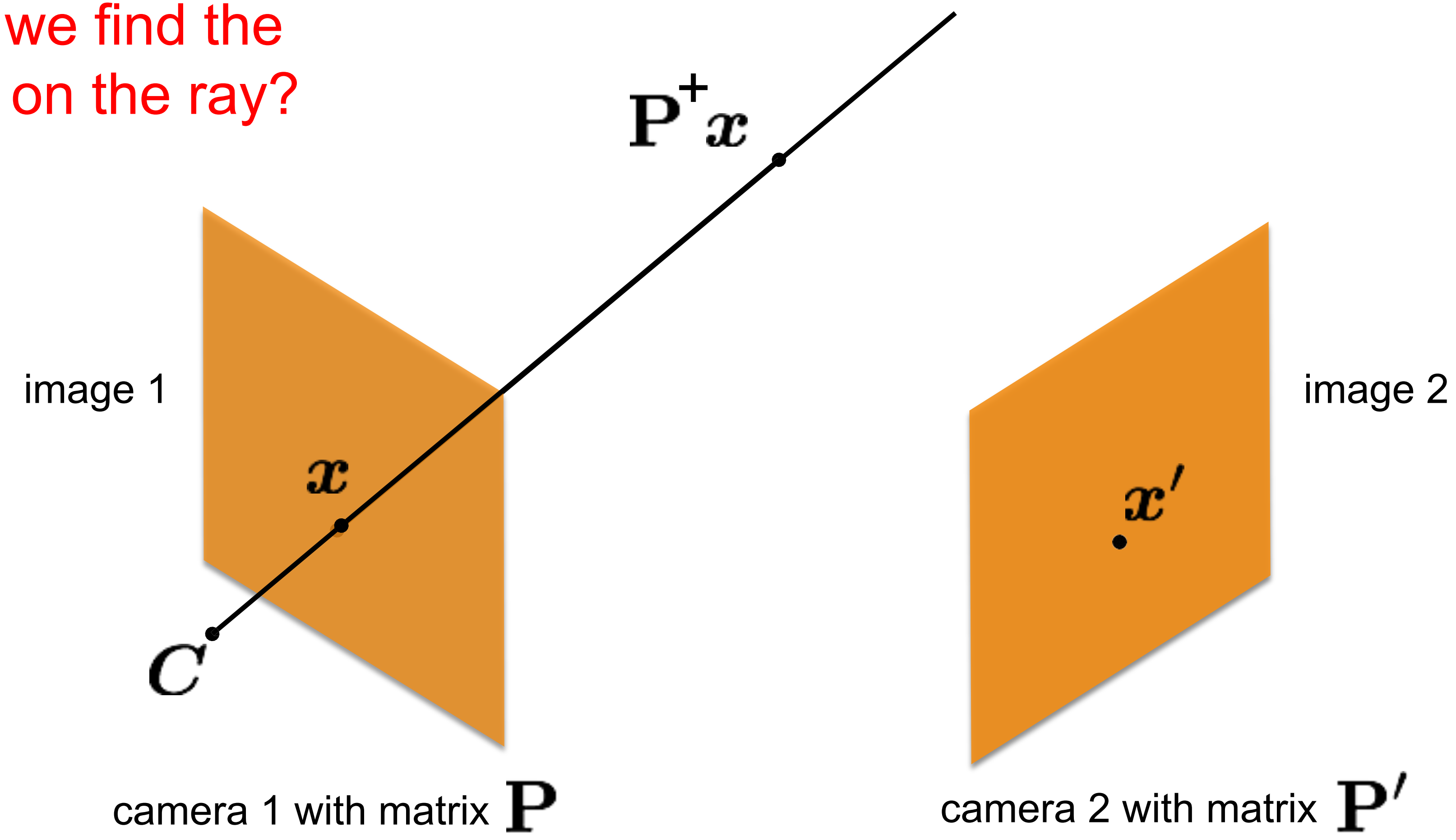
Scaling by α. is the same ray direction but differs by a scale factor corresponding to depth

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

How do we solve for unknowns in a similarity relation?

Triangulation

Q. How do we find the exact point on the ray?

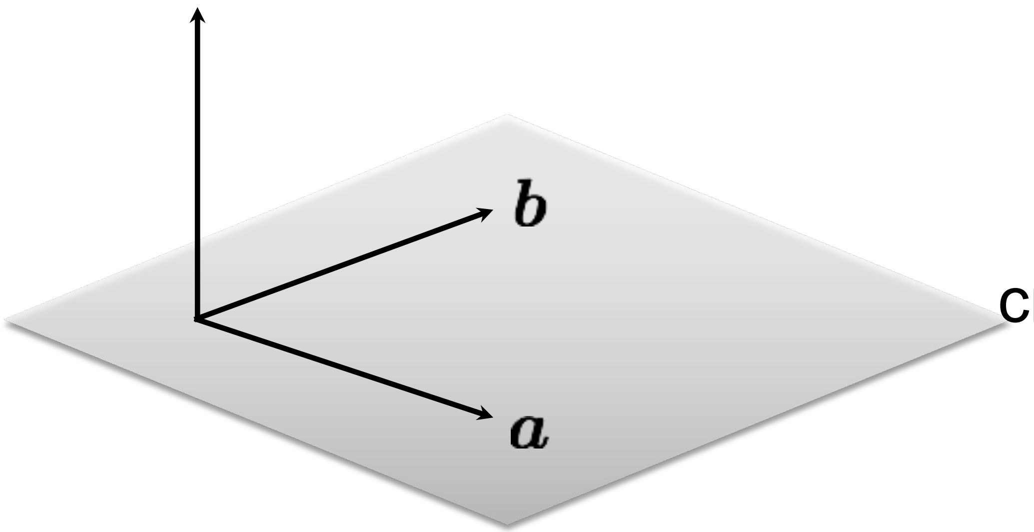


Reminder: cross products from linear algebra

Vector (cross) product

takes two vectors and returns a vector perpendicular to both

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



$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix}$$

cross product of two vectors in the same direction is zero vector

$$\boldsymbol{a} \times \boldsymbol{a} = 0$$

remember this!!!

$$\boldsymbol{c} \cdot \boldsymbol{a} = 0 \qquad \boldsymbol{c} \cdot \boldsymbol{b} = 0$$

Reminder: cross products from linear algebra

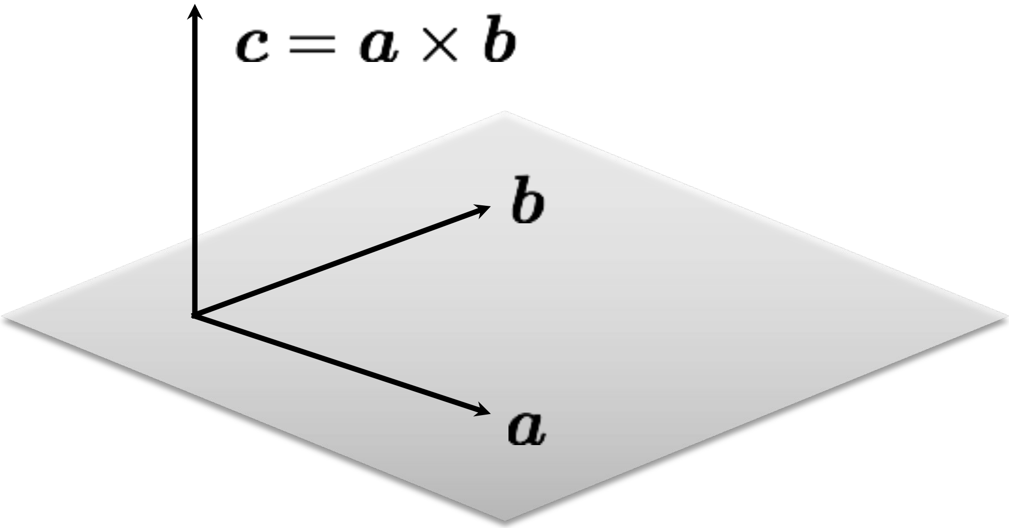
$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix}$$

Can also be written as a matrix multiplication

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_\times \boldsymbol{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Skew symmetric

Compare with: dot product



$$\boldsymbol{c} \cdot \boldsymbol{a} = 0 \qquad \boldsymbol{c} \cdot \boldsymbol{b} = 0$$

Dot product of two orthogonal vectors is zero!

Back to triangulation

$$\mathbf{x} = \alpha \mathbf{P} \boldsymbol{X}$$

Same direction but differs by a scale factor

How can we rewrite this using vector products?

$$\mathbf{x} = \alpha \mathbf{P} \boldsymbol{X}$$

Same direction but differs by a scale factor

$$\mathbf{x} \times \mathbf{P} \boldsymbol{X} = \mathbf{0}$$

Cross product of two vectors of same direction is zero
(this equality removes the scale factor)

$$\mathbf{x} = \alpha \mathbf{P} \boldsymbol{X}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} \text{---} & \boldsymbol{p}_1^\top & \text{---} \\ \text{---} & \boldsymbol{p}_2^\top & \text{---} \\ \text{---} & \boldsymbol{p}_3^\top & \text{---} \end{bmatrix} \begin{bmatrix} | \\ \boldsymbol{X} \\ | \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} \boldsymbol{p}_1^\top \boldsymbol{X} \\ \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_3^\top \boldsymbol{X} \end{bmatrix}$$

$$\mathbf{x} \times \mathbf{P}\boldsymbol{X} = \mathbf{0}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \times \begin{bmatrix} \boldsymbol{p}_1^\top \boldsymbol{X} \\ \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_3^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \\ x\boldsymbol{p}_2^\top \boldsymbol{X} - y\boldsymbol{p}_1^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \\ x\boldsymbol{p}_2^\top \boldsymbol{X} - y\boldsymbol{p}_1^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{x} \times \mathbf{P}\boldsymbol{X} = \mathbf{0}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \times \begin{bmatrix} \boldsymbol{p}_1^\top \boldsymbol{X} \\ \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_3^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \\ x\boldsymbol{p}_2^\top \boldsymbol{X} - y\boldsymbol{p}_1^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \\ x\boldsymbol{p}_2^\top \boldsymbol{X} - y\boldsymbol{p}_1^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Third line is a linear combination of the first and second lines. (x times the first line plus y times the second line)

So, we only get 2 equations to calculate 3 unknowns: X, Y, Z

Remove third row, and rearrange as system on unknowns

$$\begin{bmatrix} y\boldsymbol{p}_3^\top \boldsymbol{X} - \boldsymbol{p}_2^\top \boldsymbol{X} \\ \boldsymbol{p}_1^\top \boldsymbol{X} - x\boldsymbol{p}_3^\top \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} y\boldsymbol{p}_3^\top - \boldsymbol{p}_2^\top \\ \boldsymbol{p}_1^\top - x\boldsymbol{p}_3^\top \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

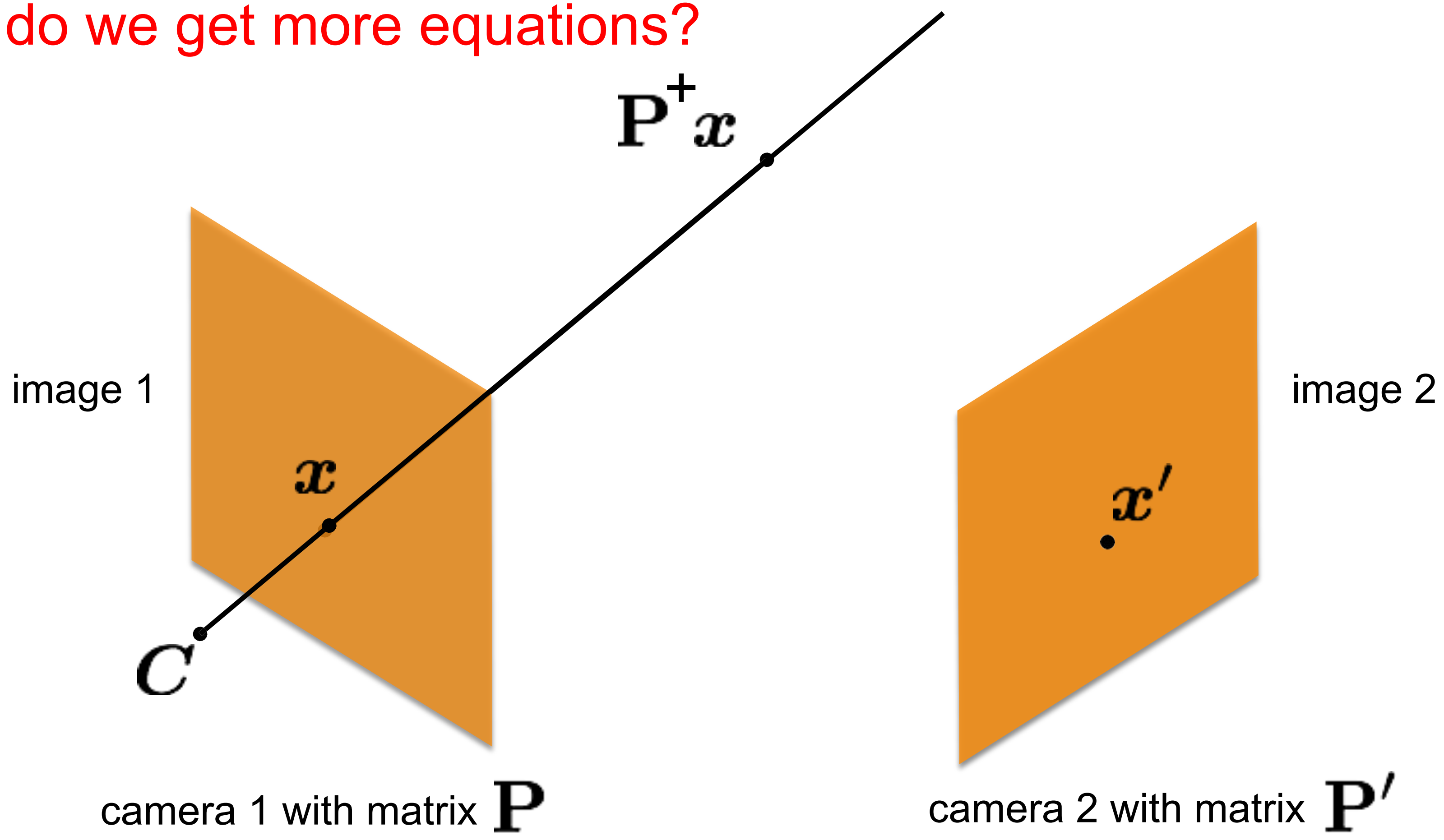
$$\mathbf{A}_i \boldsymbol{X} = \mathbf{0}$$

This is the proof that we can not solve for X… we only have two equations.

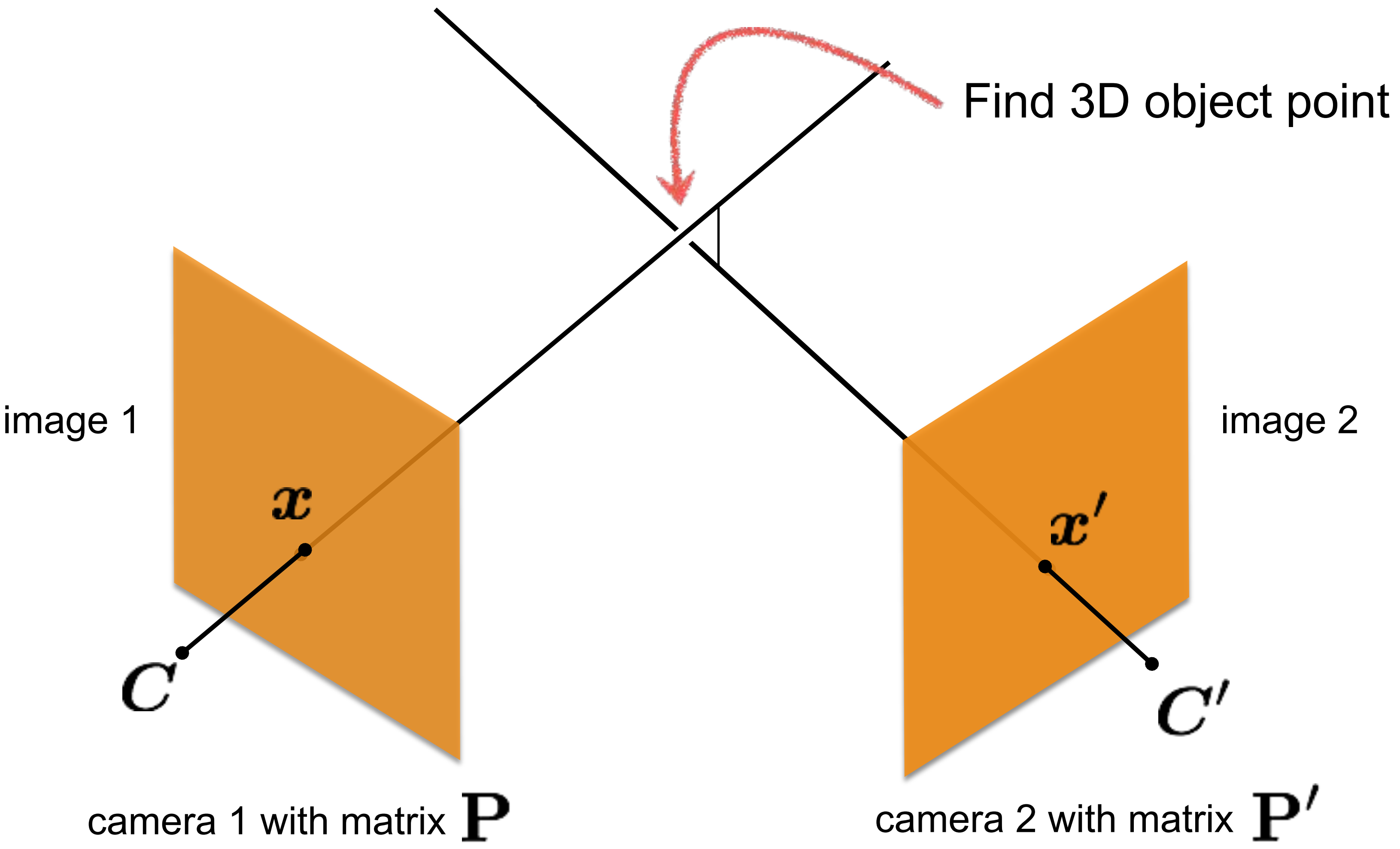
Q. So, how do we get more equations?

Triangulation

Q. So, how do we get more equations?



How do we find this intersection?



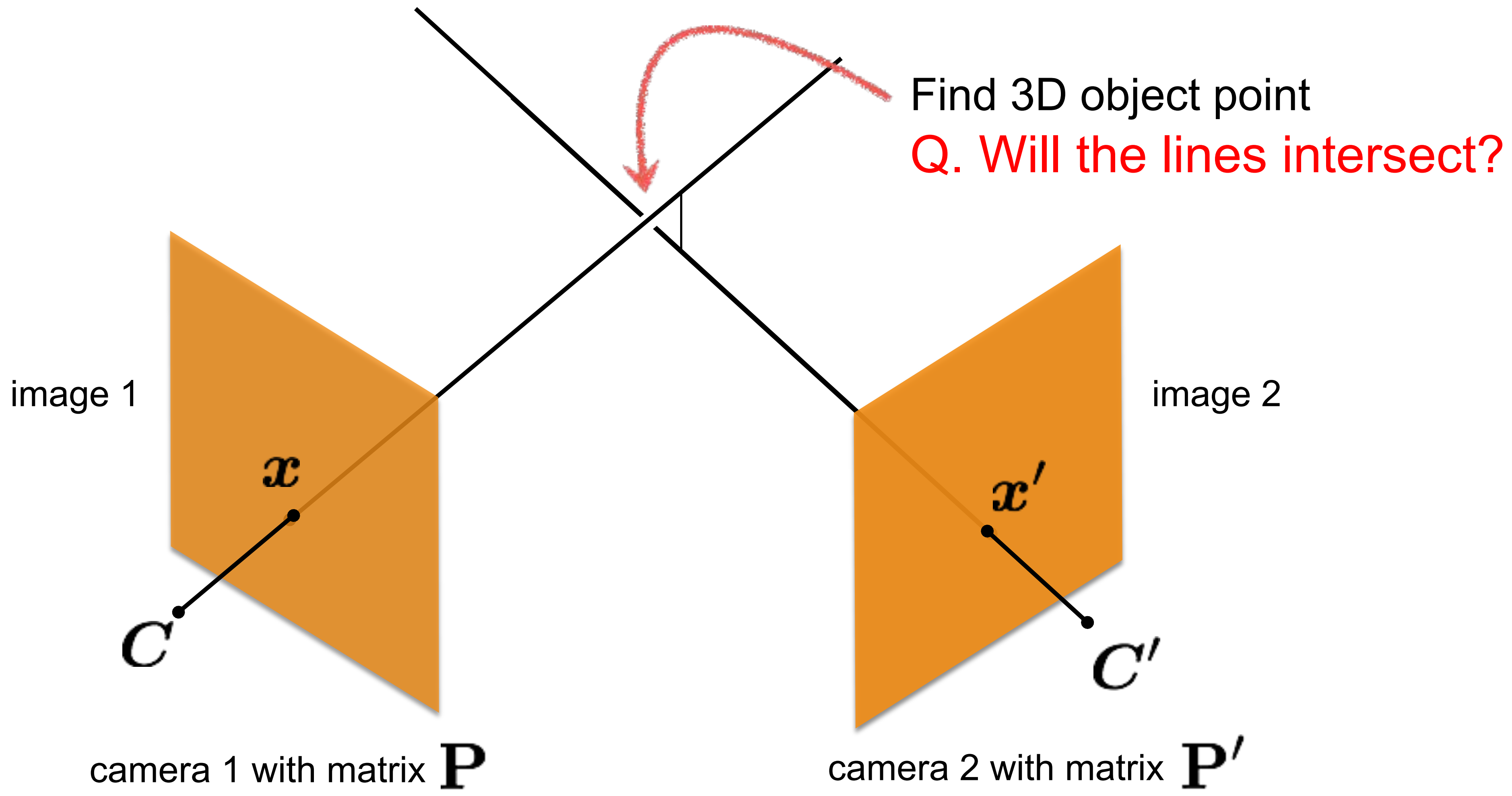
Collect more equations from other cameras

$$\begin{array}{r} \text{Two rows} \\ \text{from camera} \\ \text{one} \\ \\ \text{Two rows} \\ \text{from camera} \\ \text{two} \end{array} \quad \begin{bmatrix} y\boldsymbol{p}_3^\top - \boldsymbol{p}_2^\top \\ \boldsymbol{p}_1^\top - x\boldsymbol{p}_3^\top \\ y'\boldsymbol{p}_3'^\top - \boldsymbol{p}_2'^\top \\ \boldsymbol{p}_1'^\top - x'\boldsymbol{p}_3'^\top \end{bmatrix} \boldsymbol{X} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}\boldsymbol{X} = \mathbf{0}$$

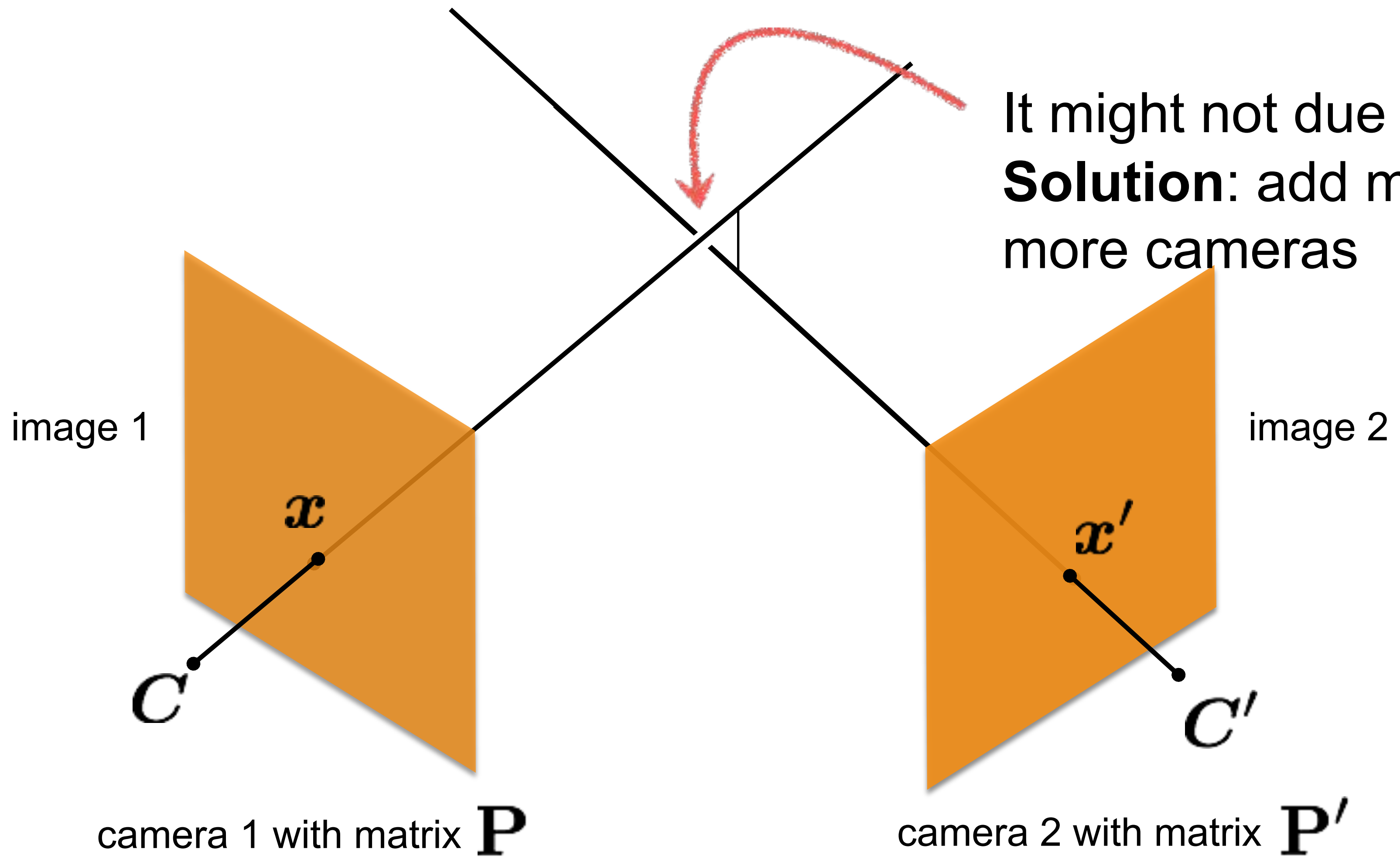
Now, we can solve for X using SVD

Triangulation



Triangulation

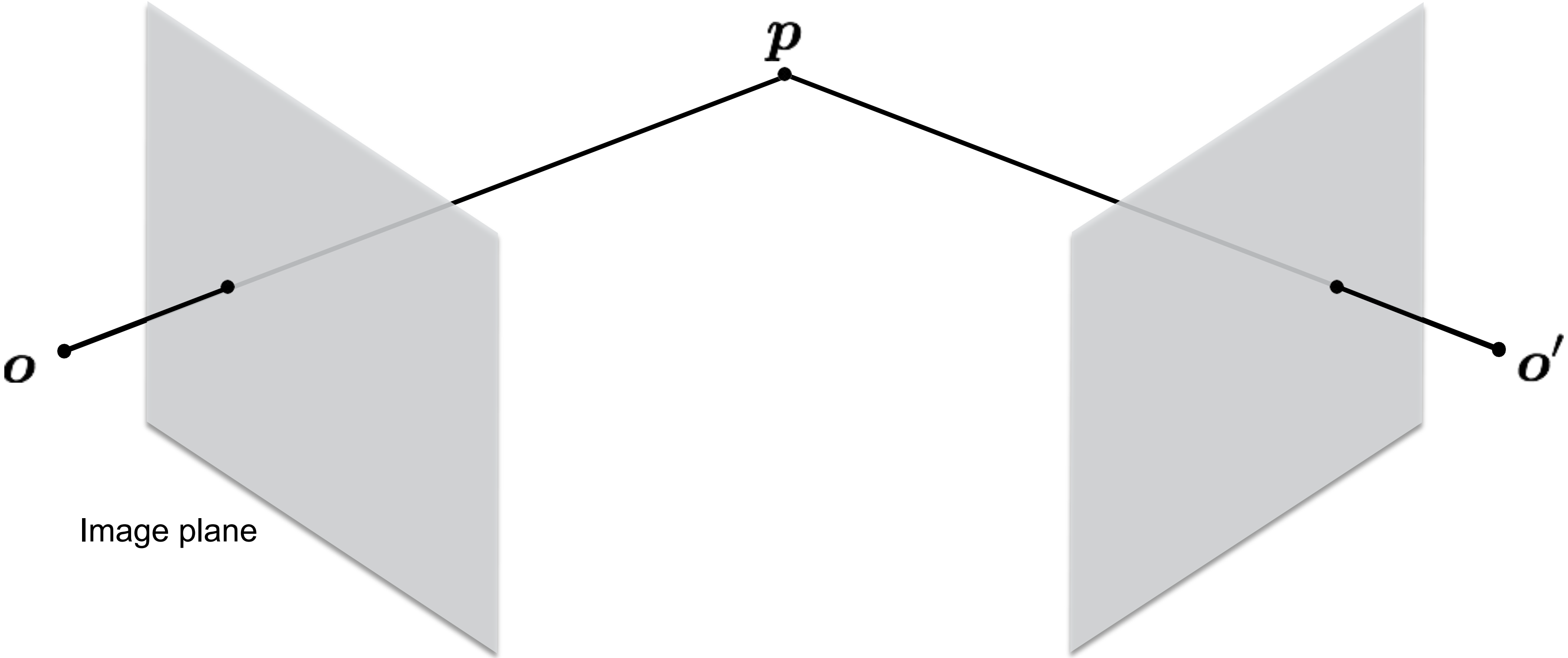
It might not due to noise
Solution: add more rows using more cameras



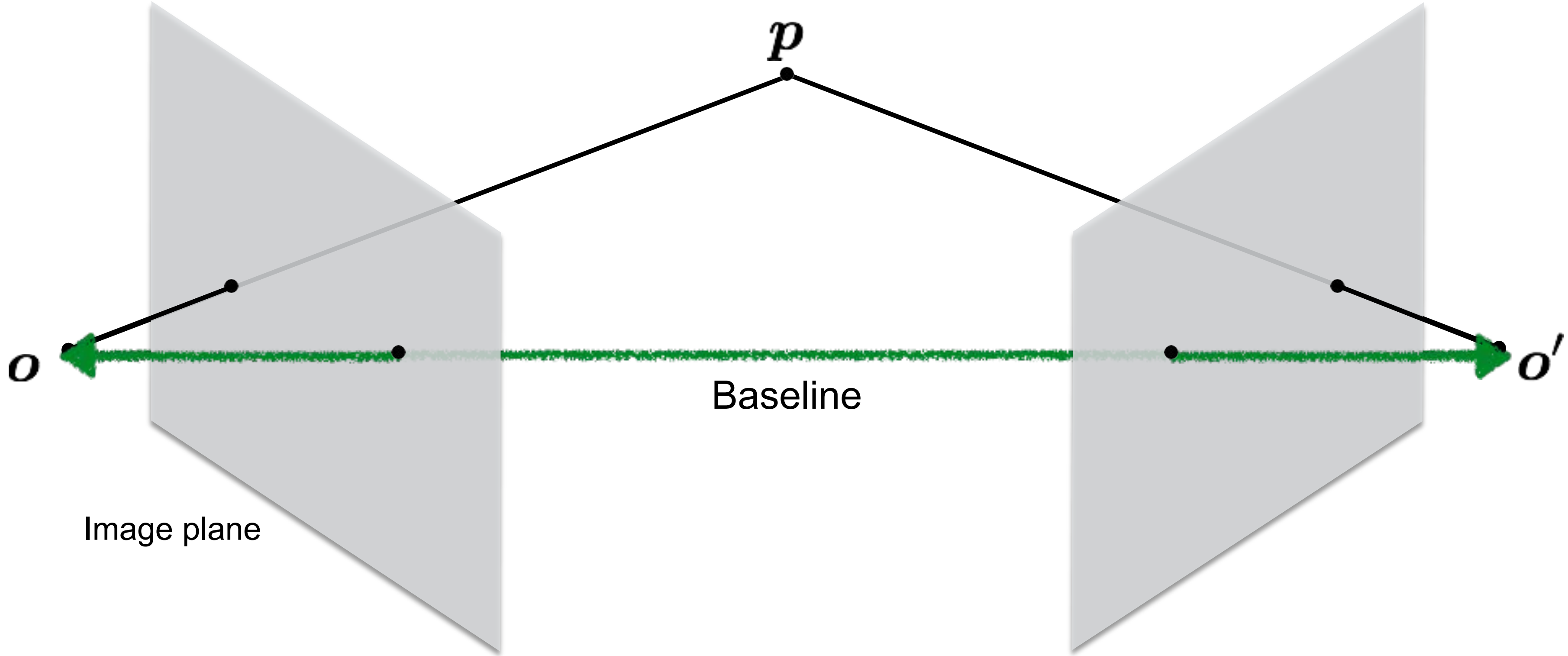
Today's agenda

- Triangulation
- Epipolar geometry
- Essential matrix
- Fundamental matrix
- Structure from motion

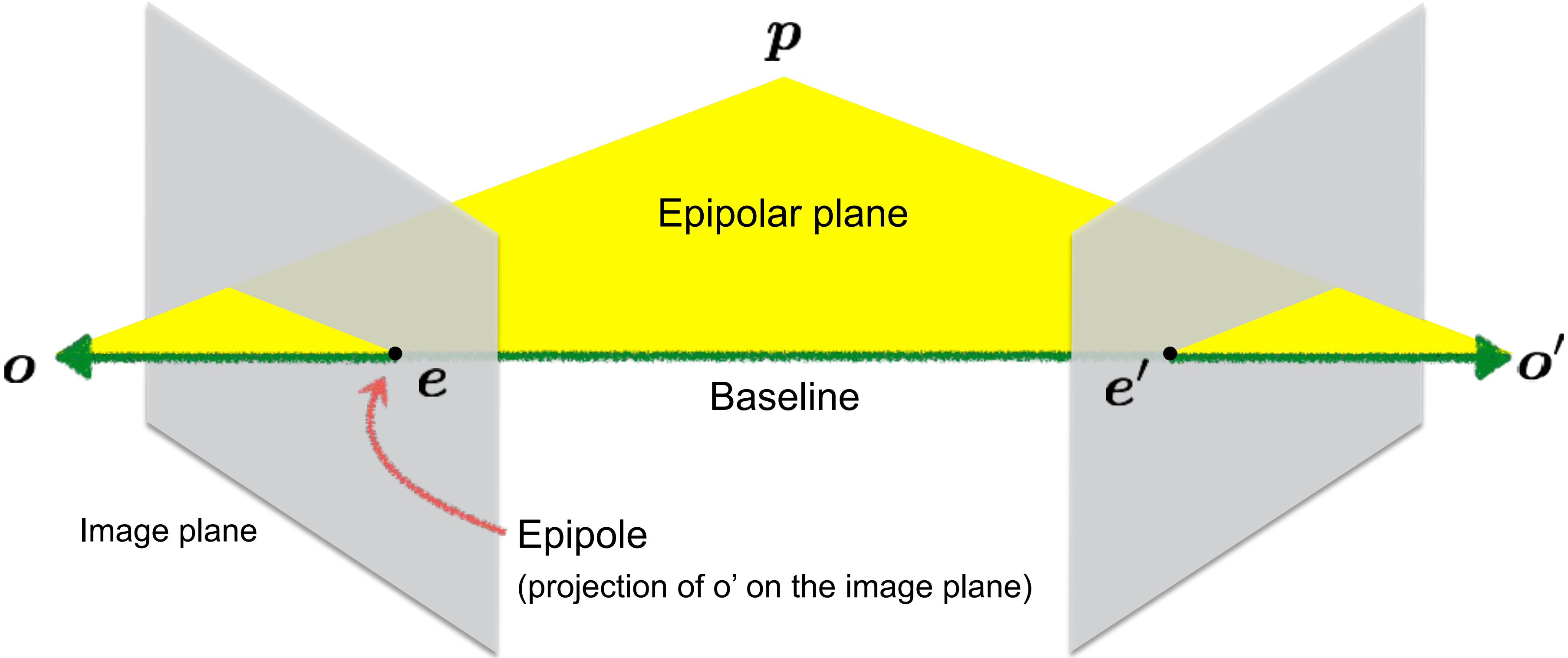
Epipolar geometry



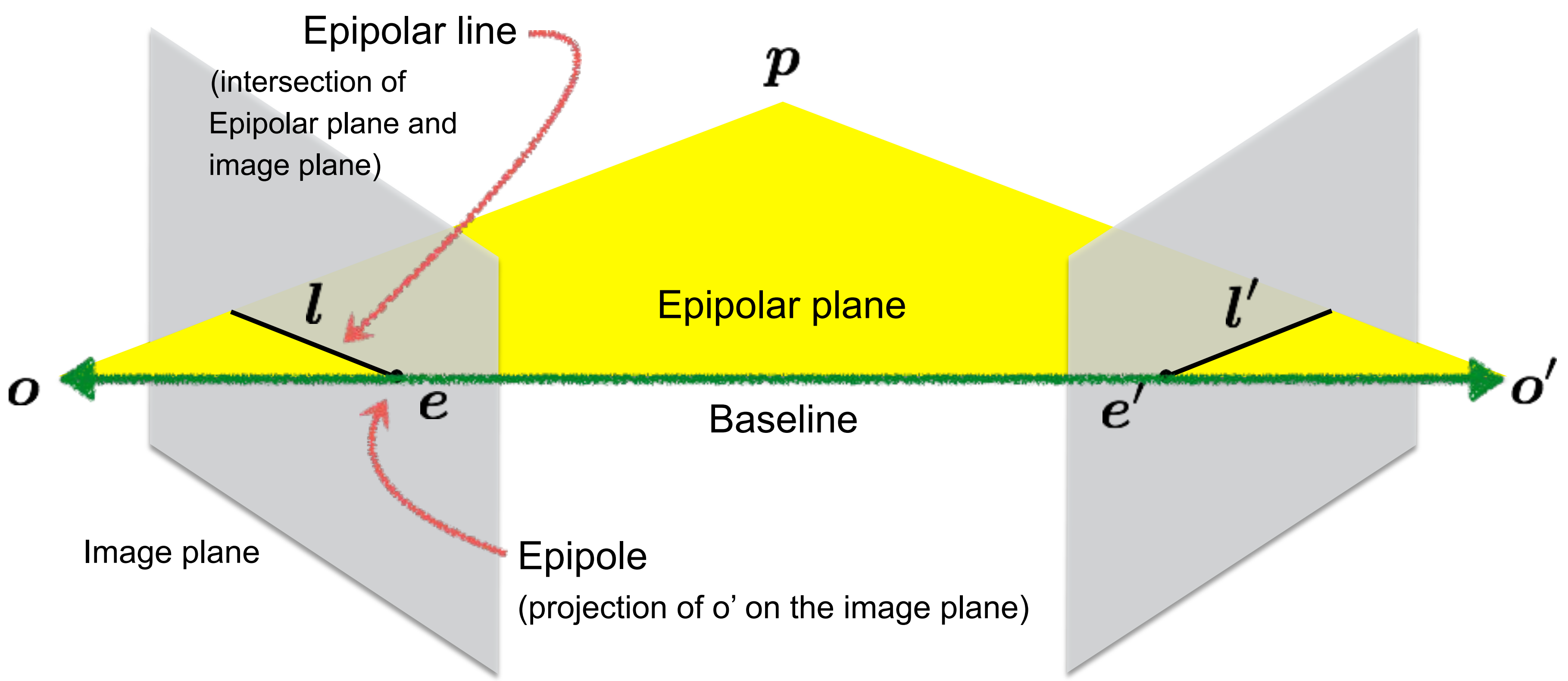
Epipolar geometry



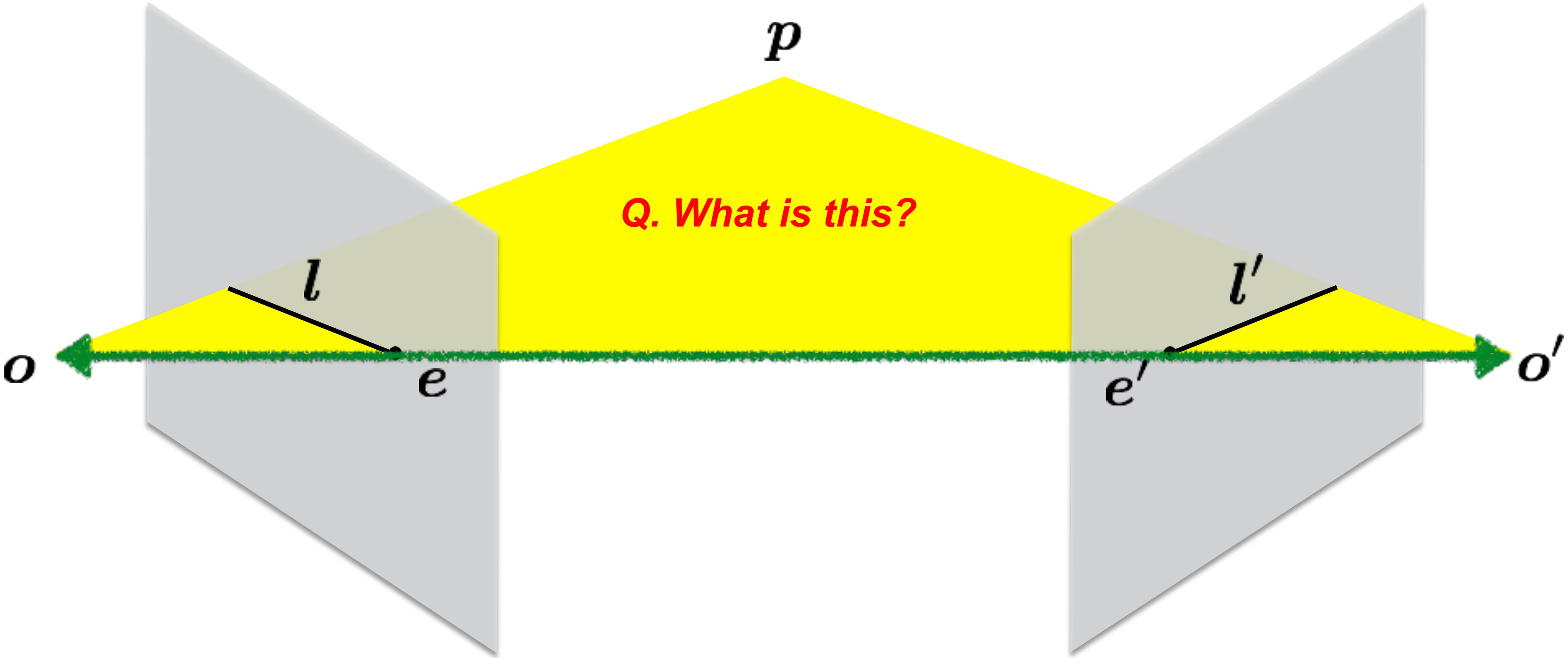
Epipolar geometry



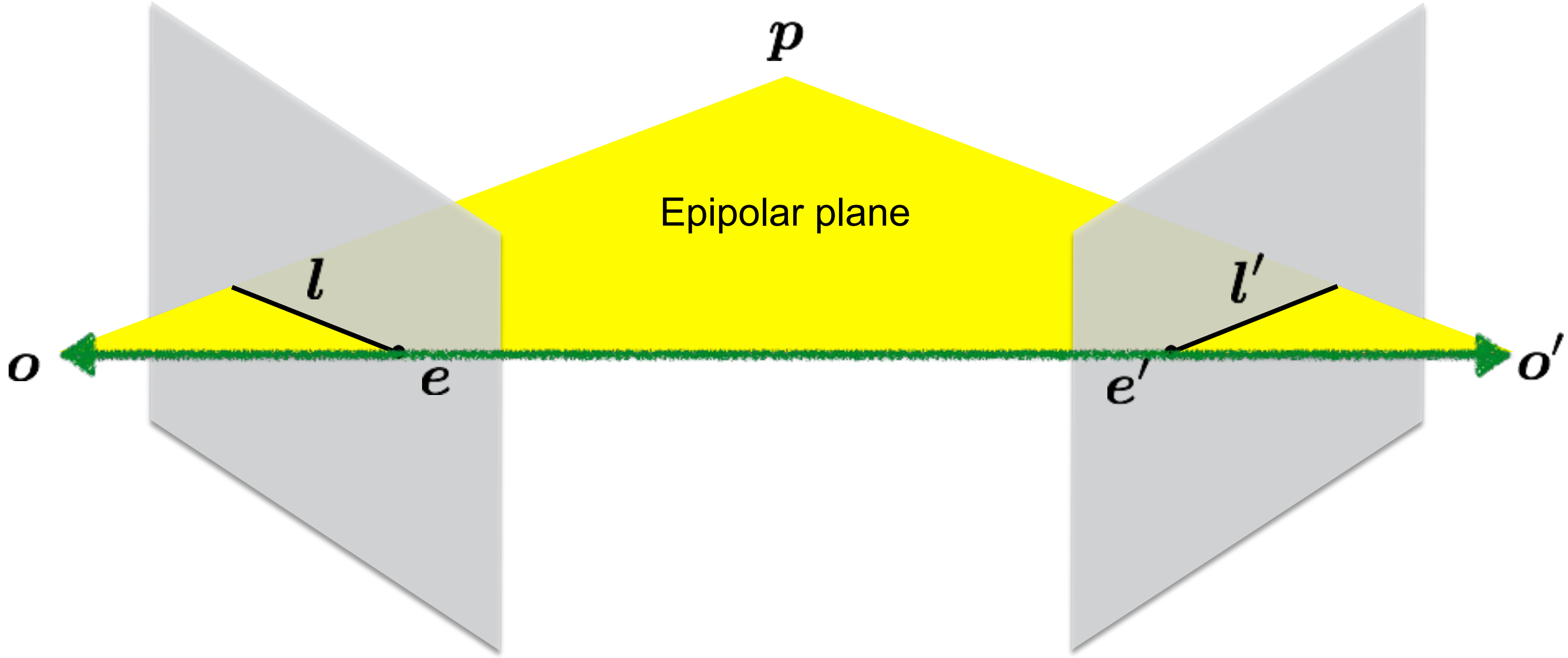
Epipolar geometry



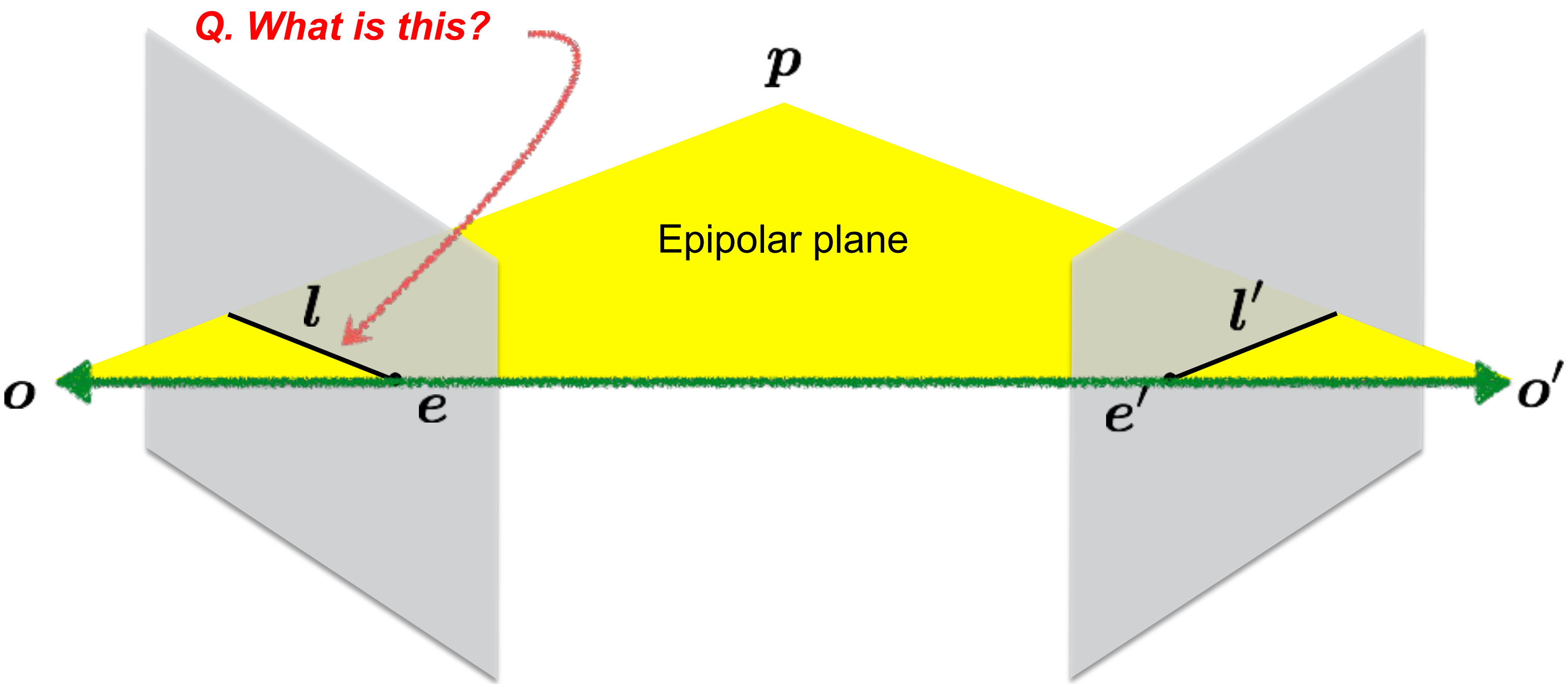
Let's see if you remember:



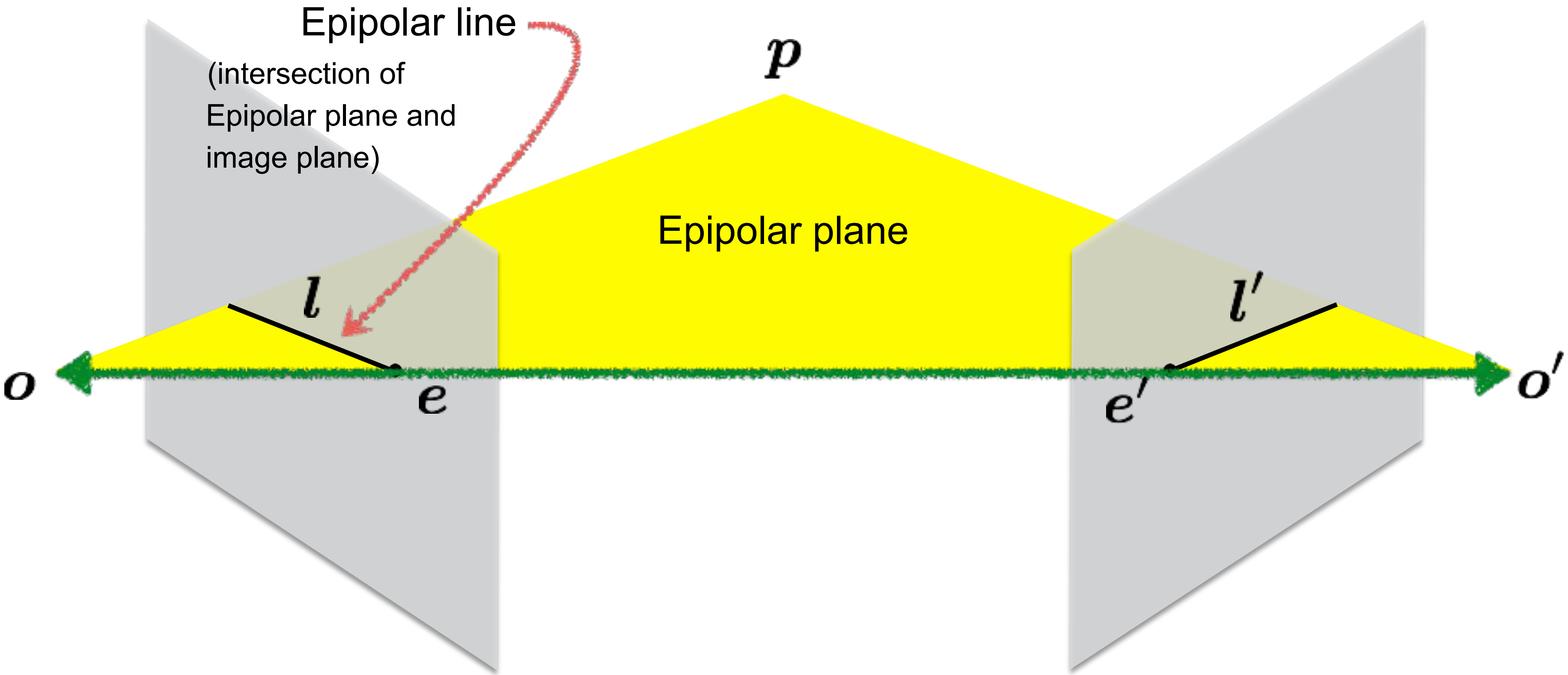
Let's see if you remember:



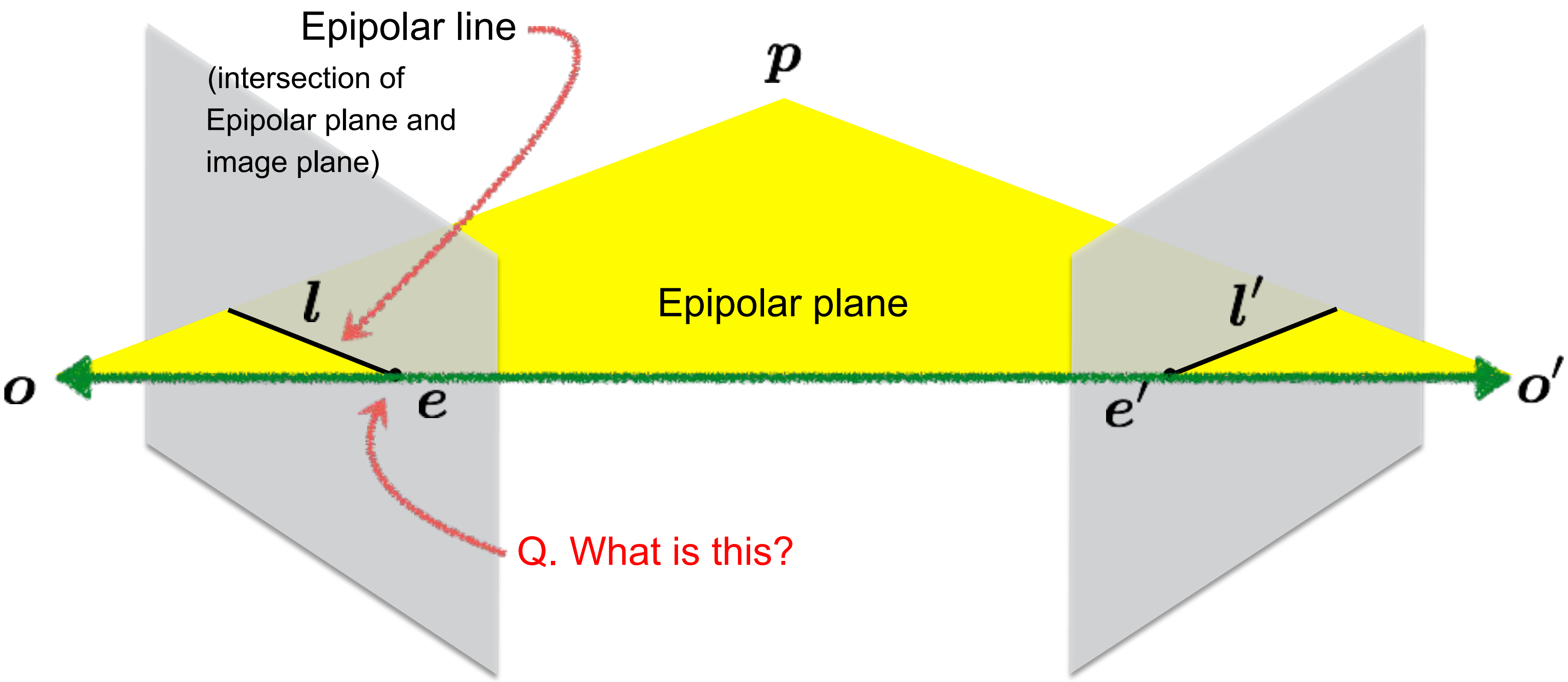
Let's see if you remember:



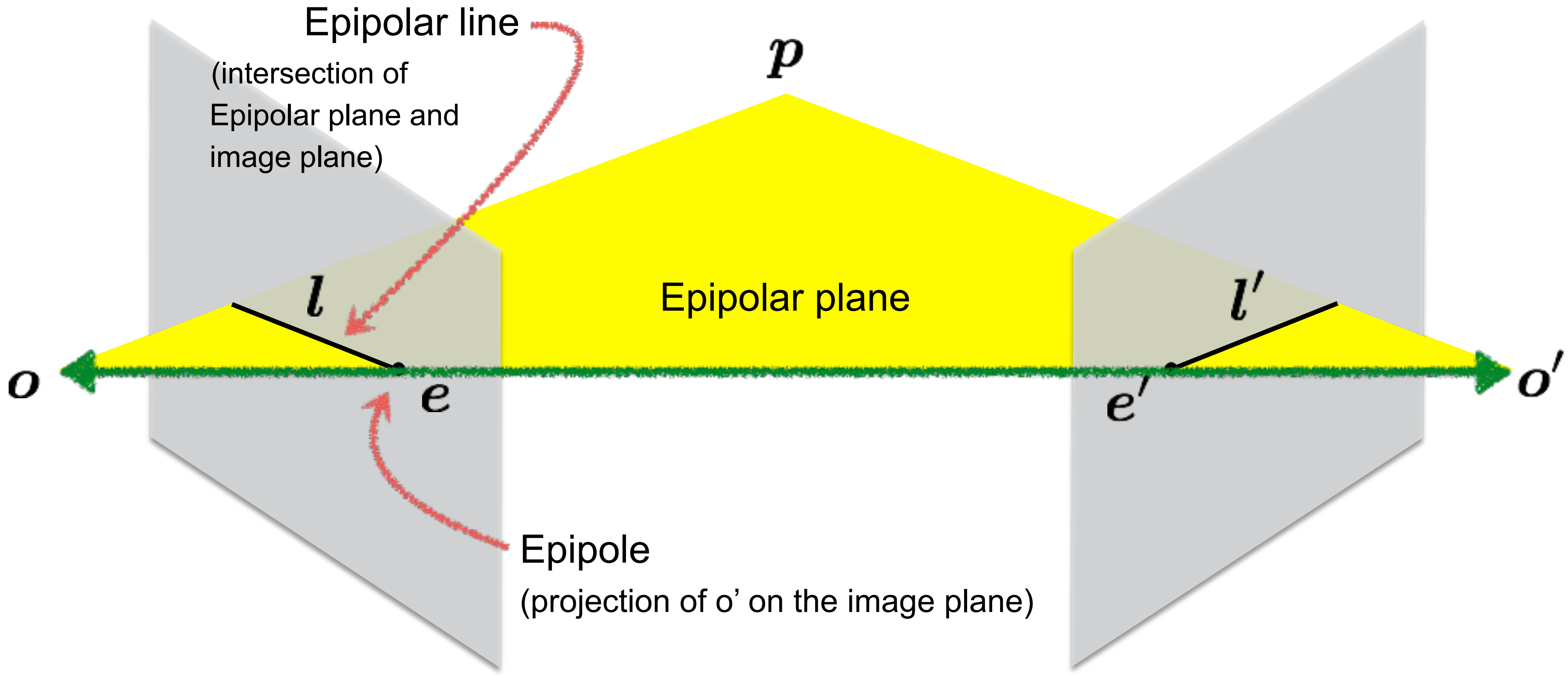
Let’s see if you remember:



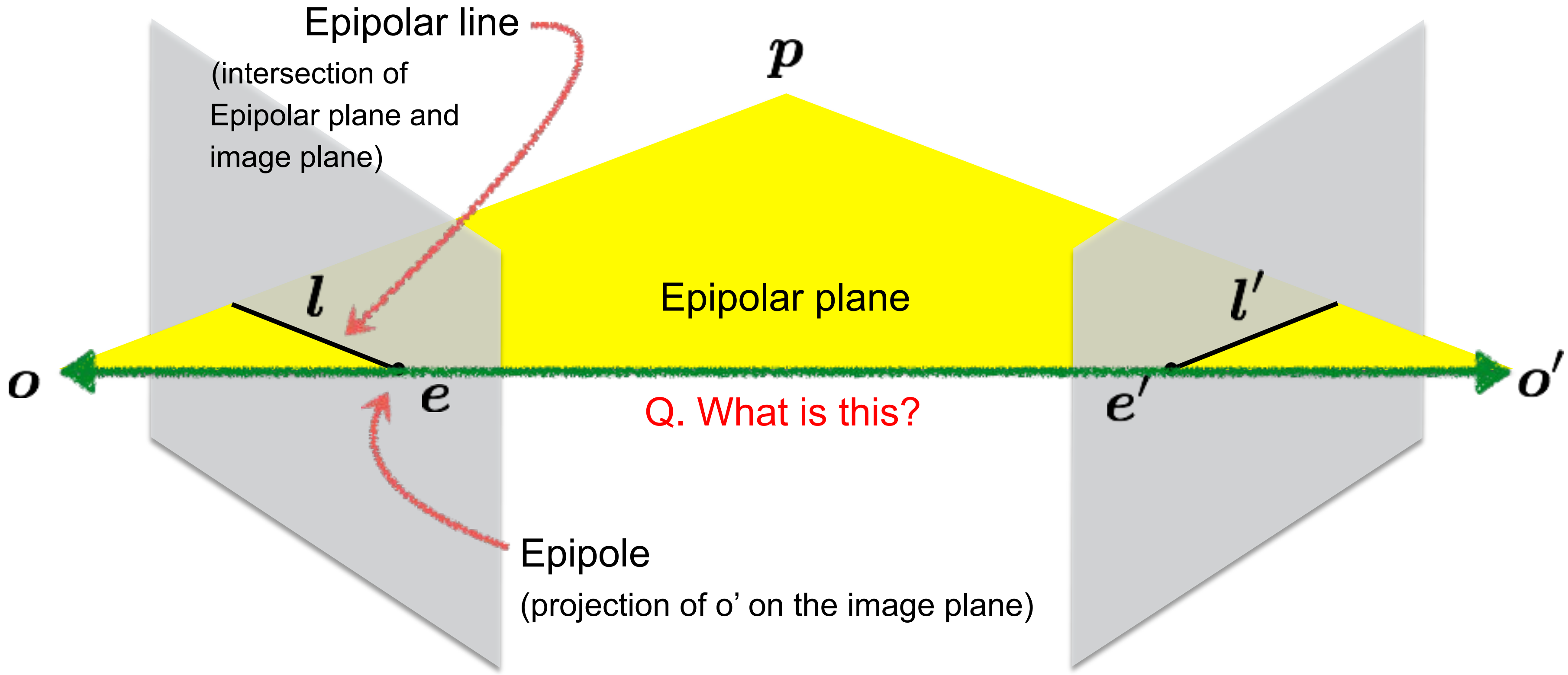
Let's see if you remember:



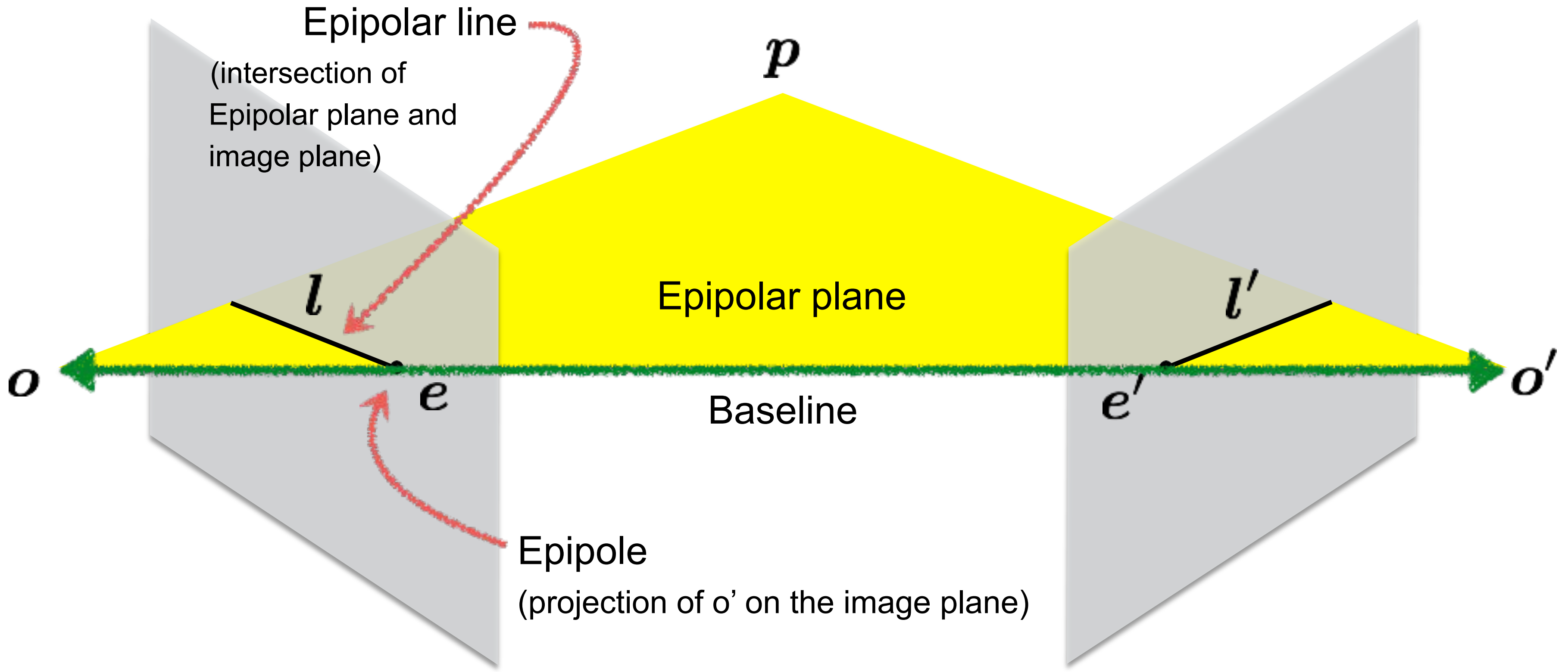
Let's see if you remember:



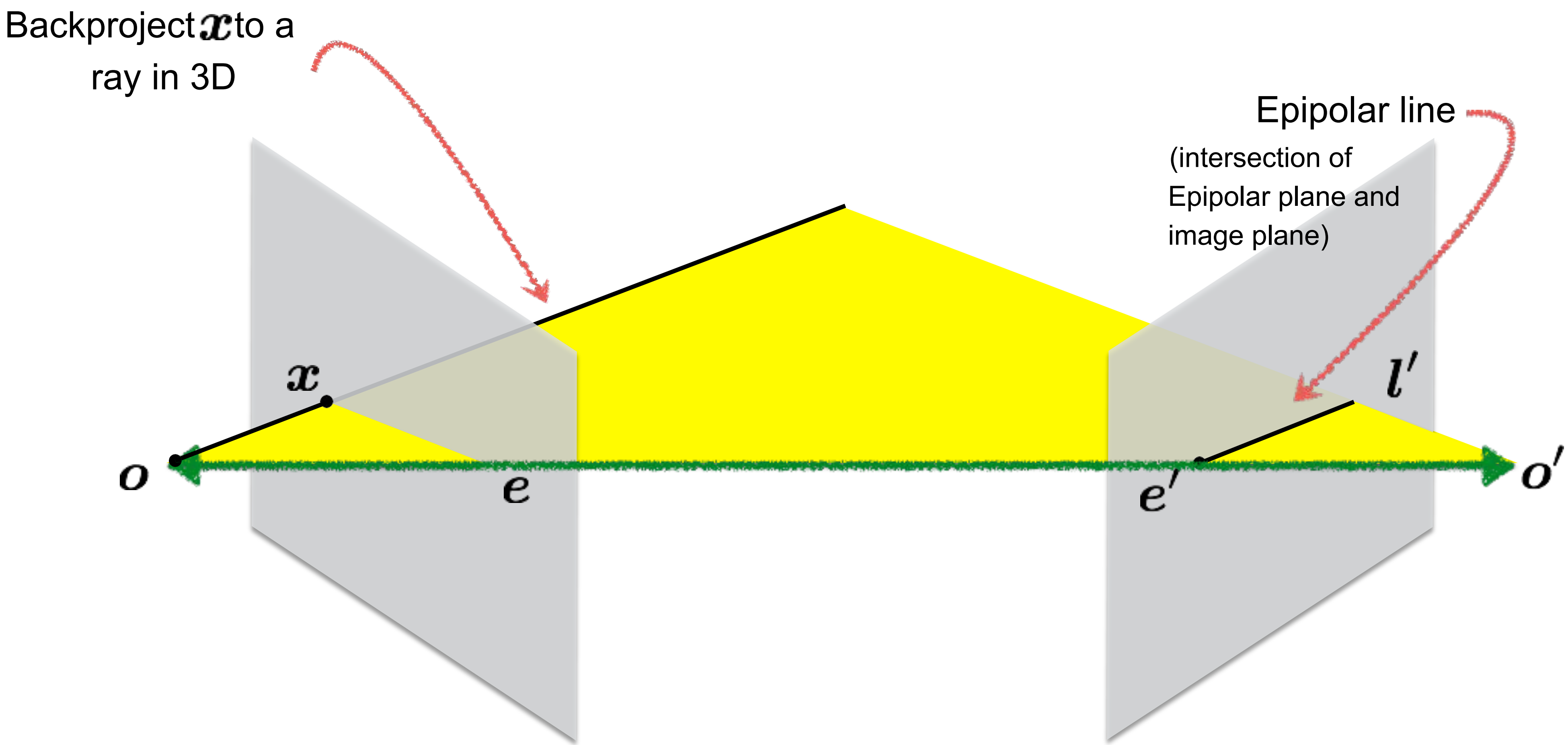
Let’s see if you remember:



Let's see if you remember:

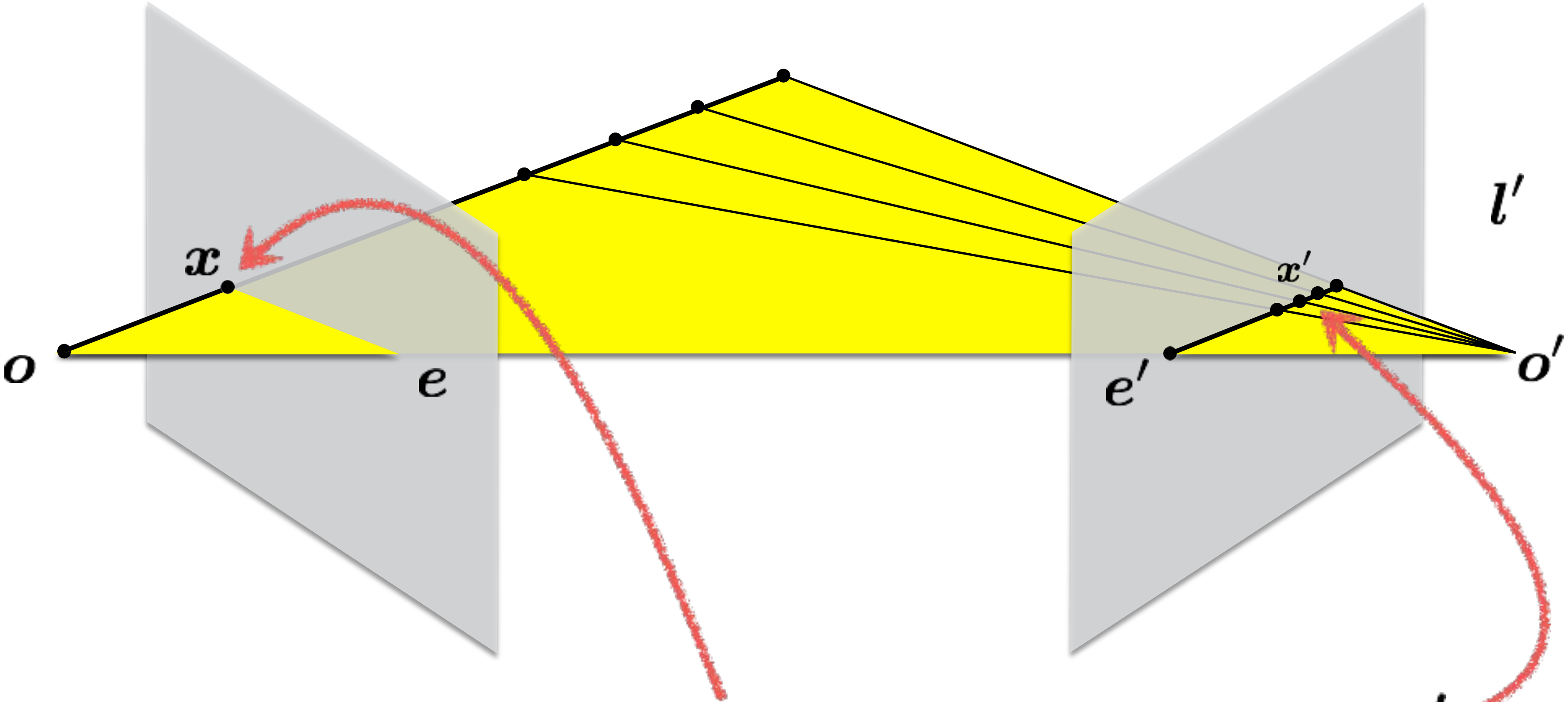


Epipolar constraint



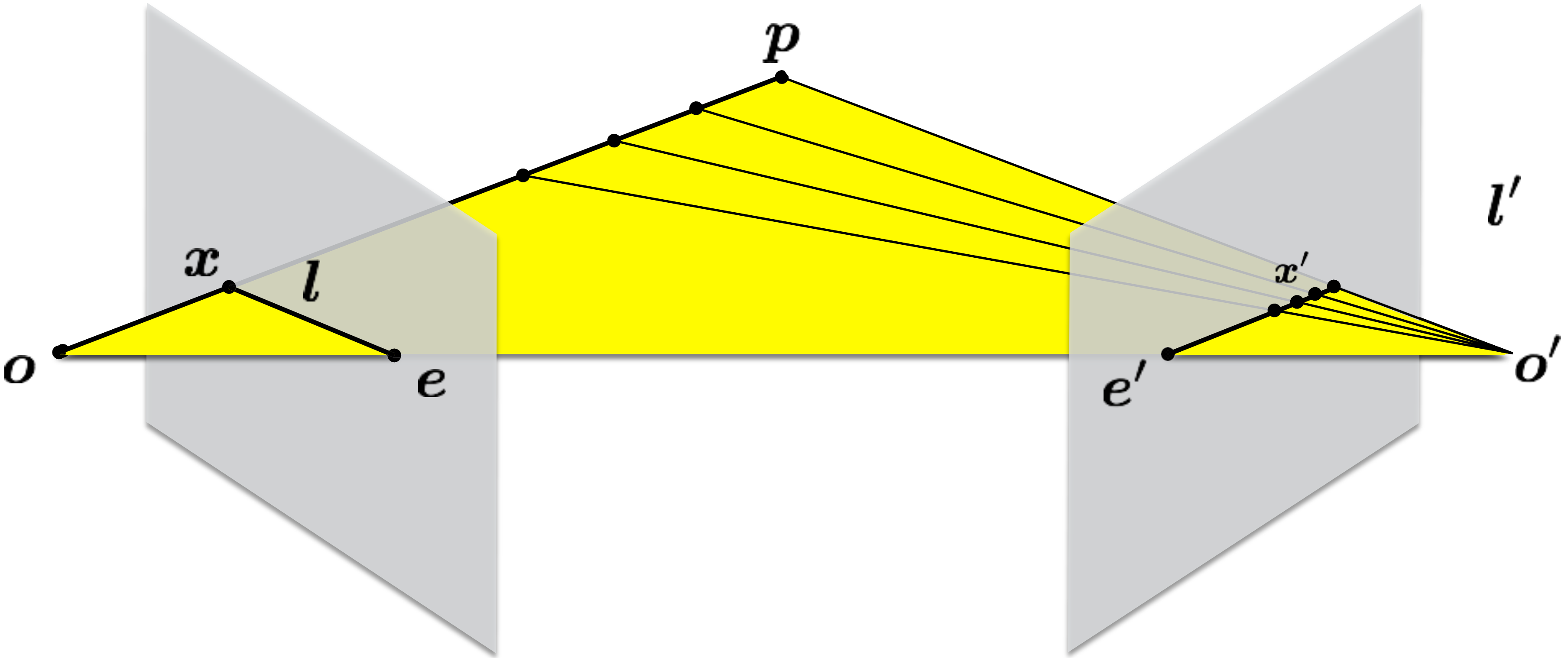
Another way to construct the epipolar plane, this time given x

Epipolar constraint



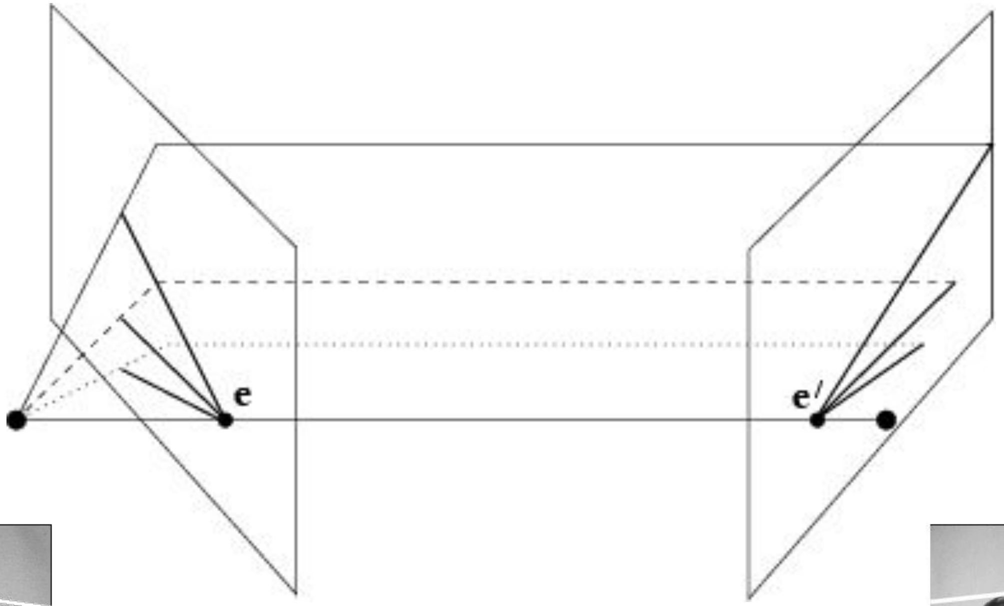
Potential matches for x lie on the epipolar line l'

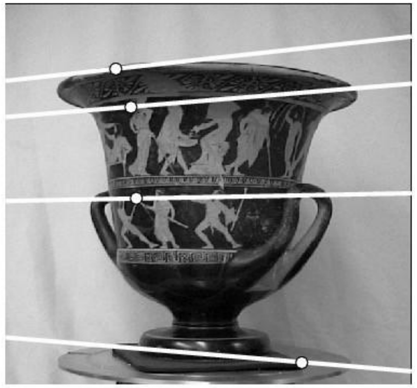
Questions



1. The point **x** (left image) maps to _______ in the right image
2. The baseline connects the ______ and _______
3. The epipolar line (left image) maps to a _______ in the right image
4. An epipole **e** is a projection of the _______ on the image plane
5. All epipolar lines in an image intersect at the _______ on the image plane

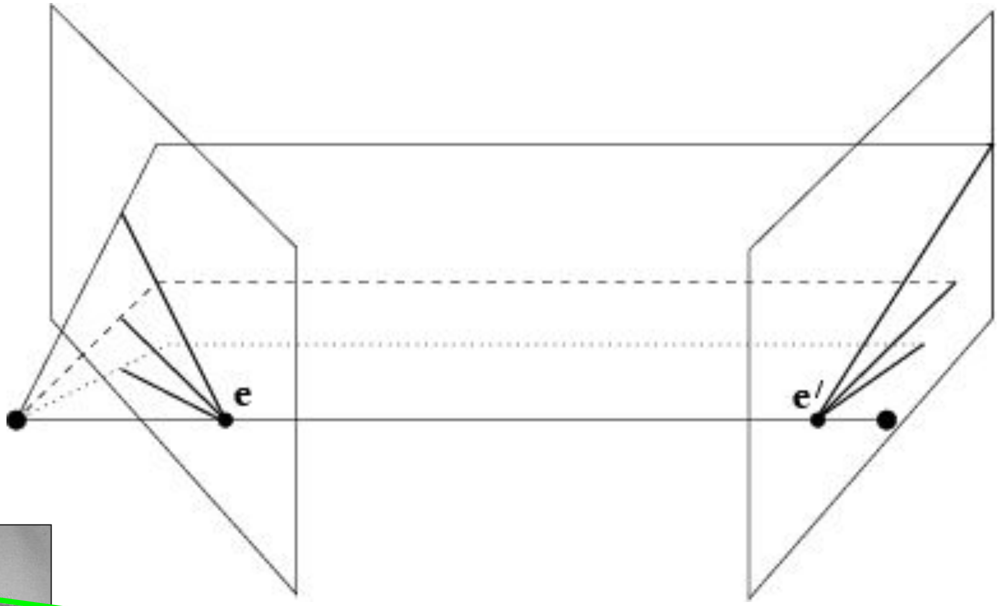
Converging cameras

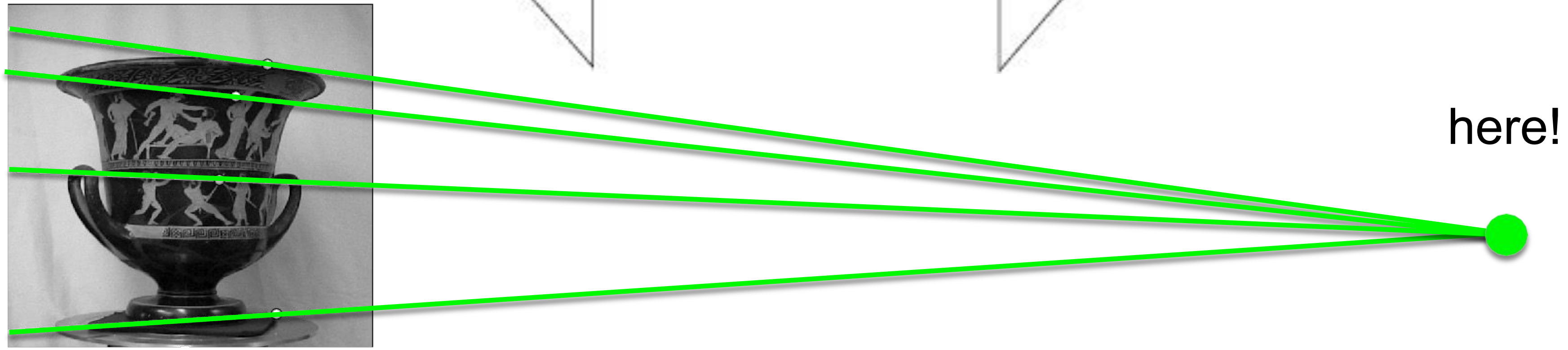




Where is the epipole in this image?

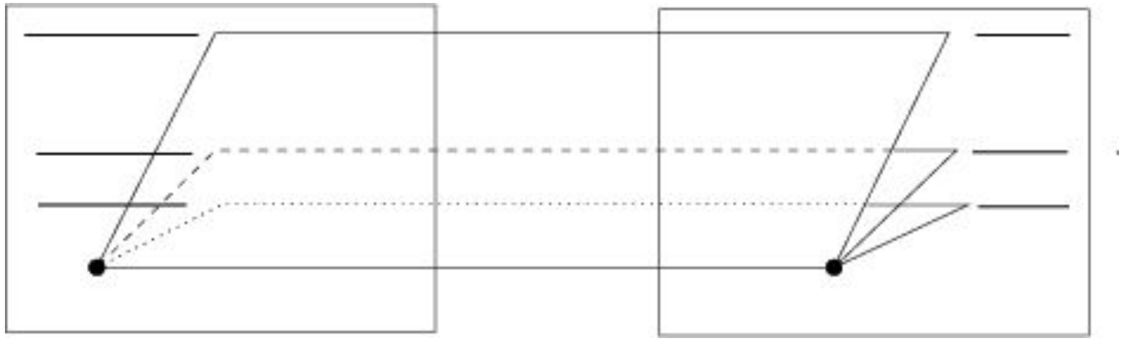
Converging cameras



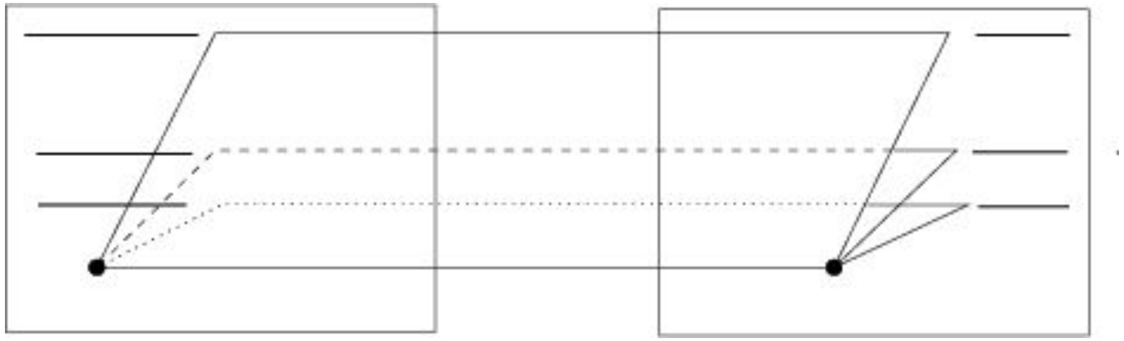


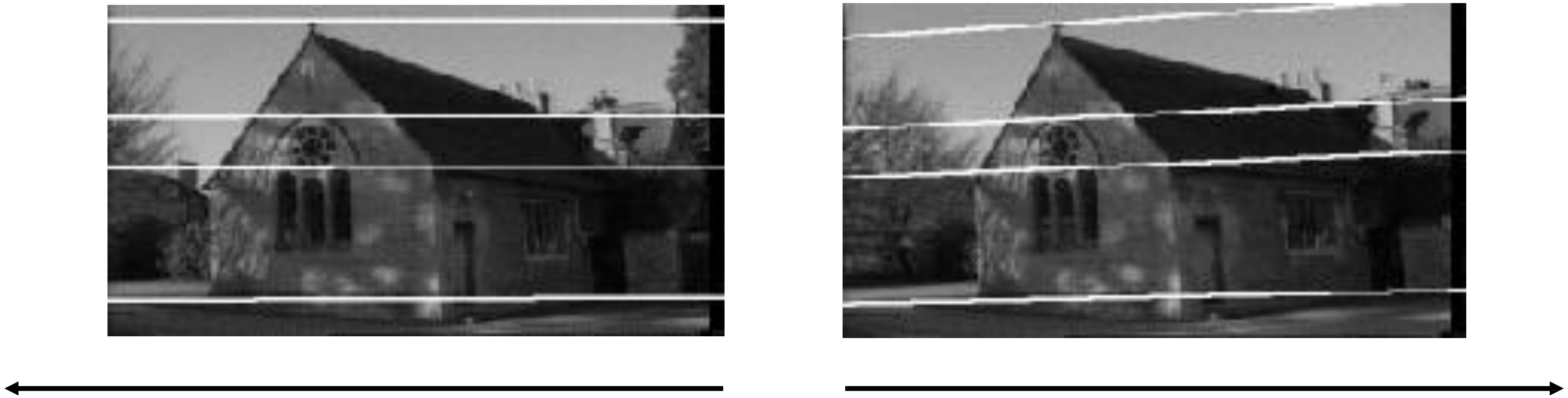
It’s not always in the image

Where is the epipole when the epipolar lines are parallel?



The epipoles can be at infinity

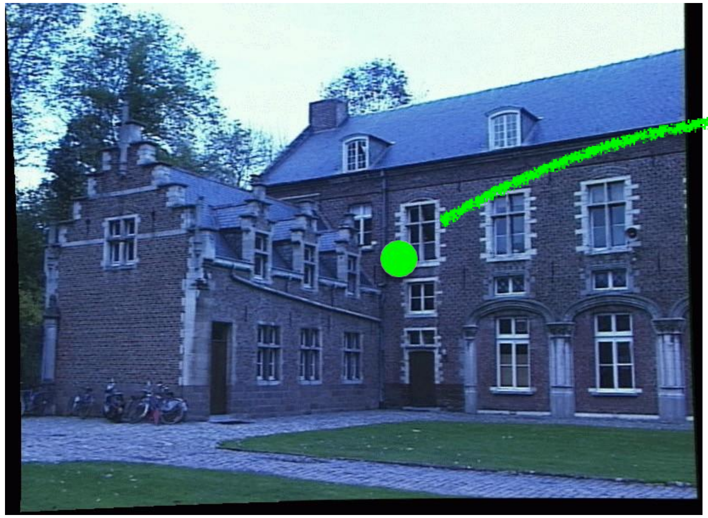




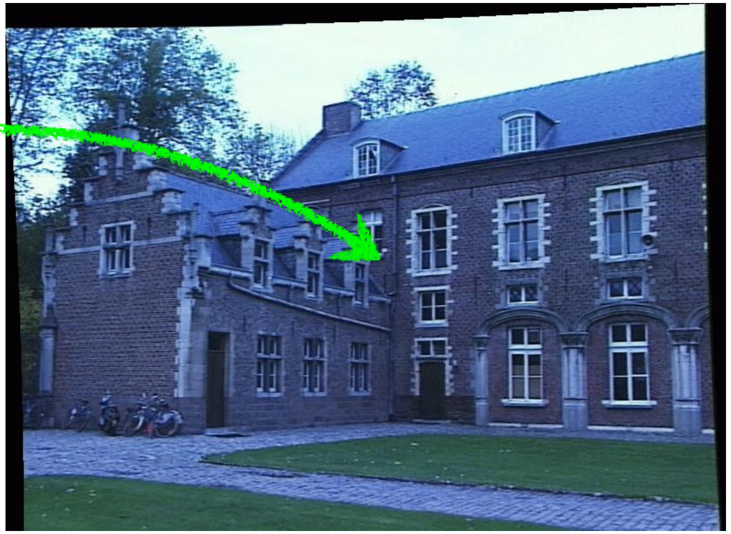
epipole at infinity

The epipolar constraint is an important concept for stereo vision

Task: Match point in left image to point in right image



Left image

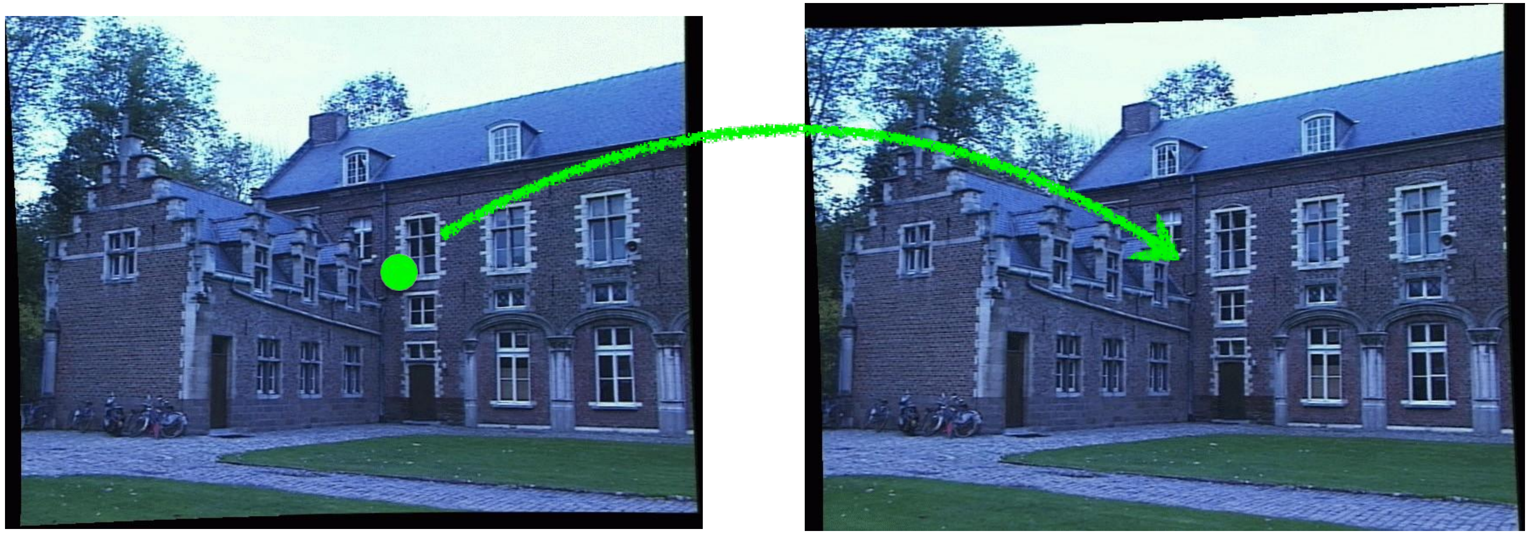


Right image

How would you do it?

The epipolar constraint is an important concept for stereo vision

Task: Match point in left image to point in right image

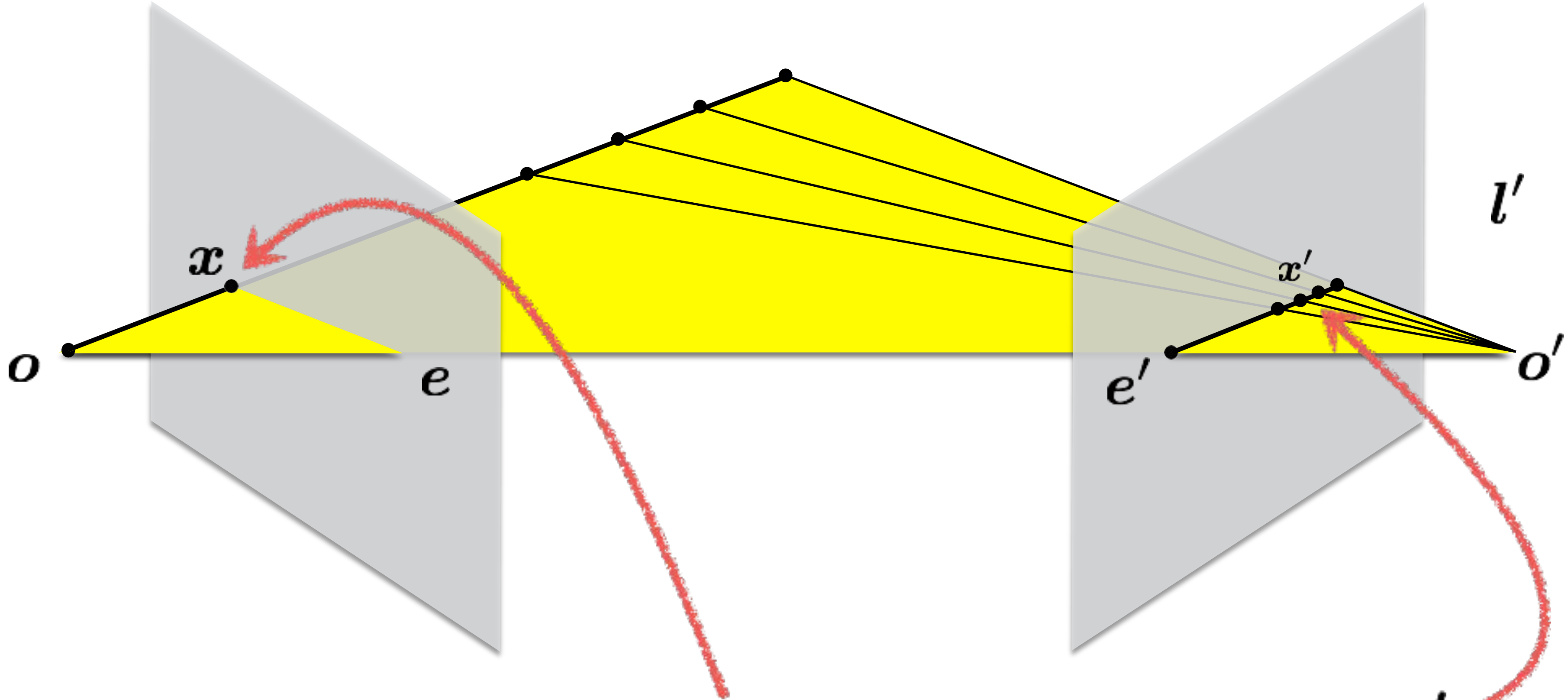


Left image

Right image

How would you do it using epipolar geometry?

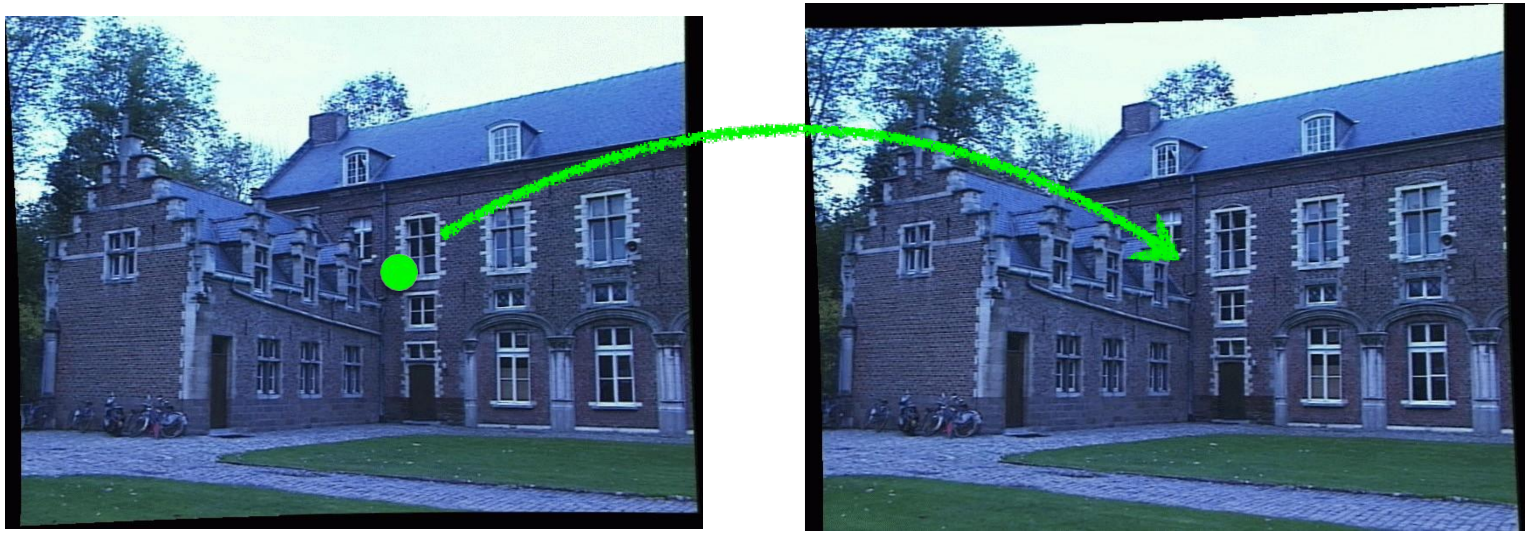
Let's use the epipolar constraint



Potential matches for x lie on the epipolar line l'

How do you compute the epipolar line?

Task: Match point in left image to point in right image



Left image Right image

Want to avoid search over entire image

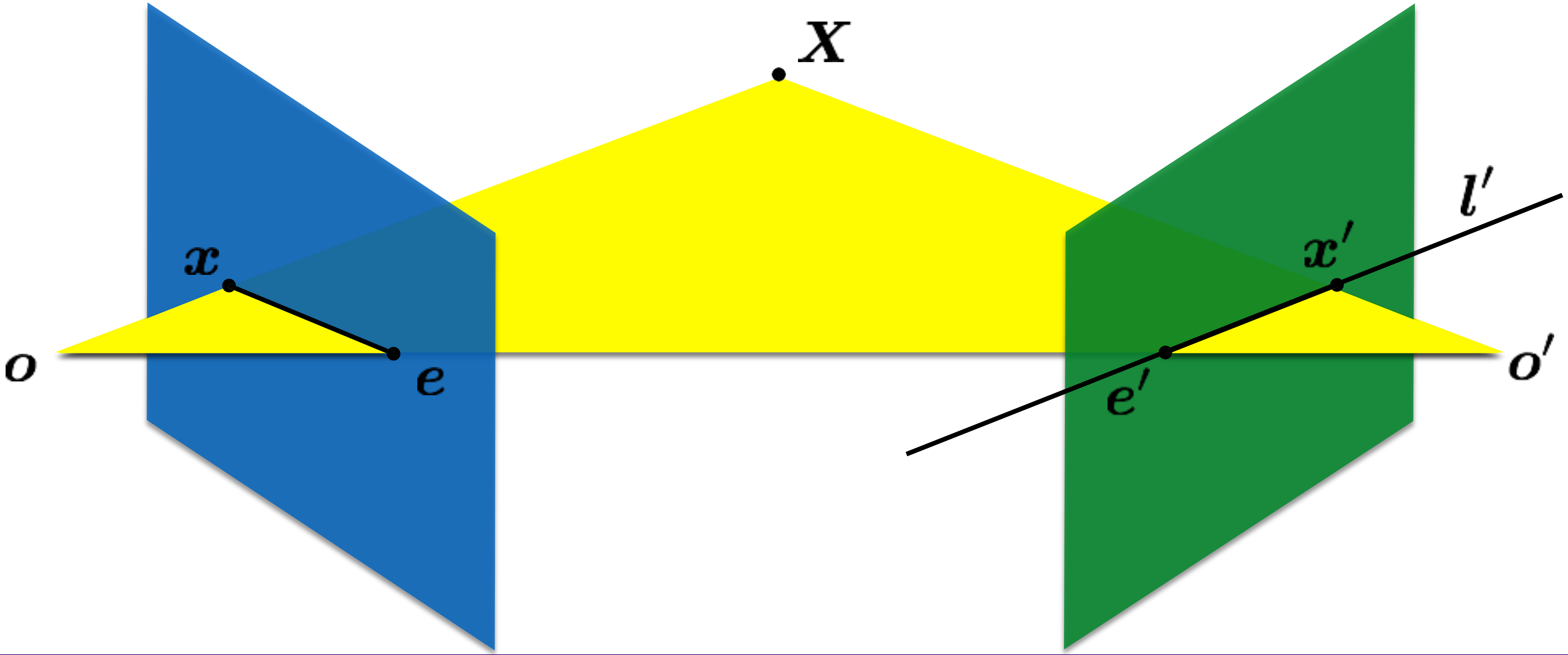
Epipolar constraint reduces search to a single line

Today's agenda

- Triangulation
- Epipolar geometry
- Essential matrix
- Fundamental matrix
- Structure from motion

Given a point in one image,
multiplying by the **essential matrix** will tell us
the **epipolar line** in the second view.

$$\boxed{\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'}$$



Motivation

The Essential Matrix is a 3 x 3 matrix that encodes
epipolar geometry

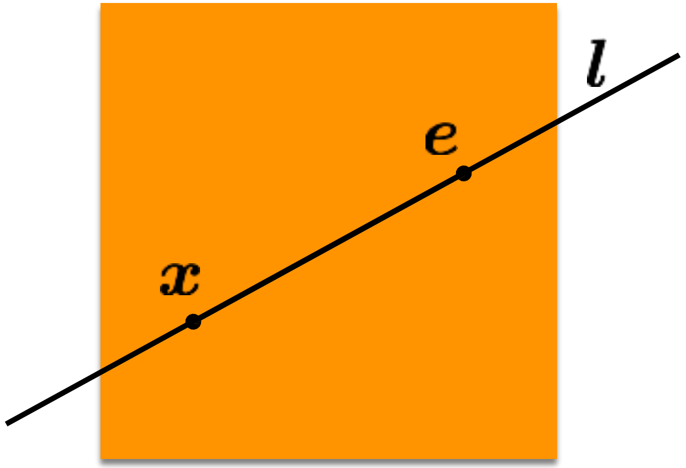
Given a point in one image,
multiplying by the **essential matrix** will tell us
the **epipolar line** in the second image.

Epipolar Line

$$ax + by + c = 0$$

in vector form

$$\boldsymbol{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$



If the point $\boldsymbol{x}$ is on the epipolar line $\boldsymbol{l}$ then

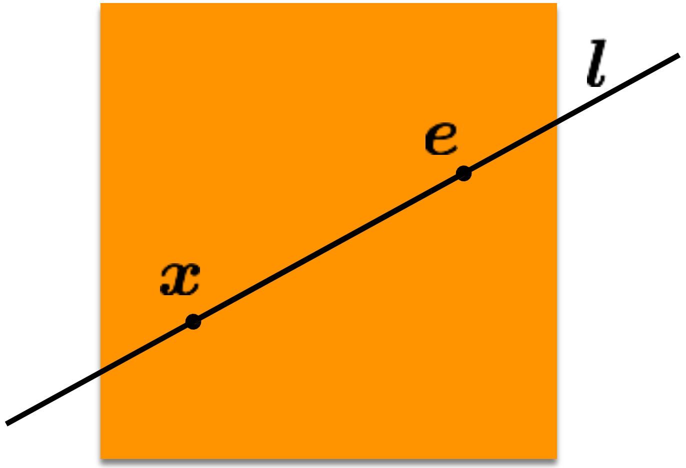
$$\boldsymbol{x}^\top \boldsymbol{l} = \ ?$$

Epipolar Line

$$ax + by + c = 0$$

in vector form

$$\boldsymbol{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

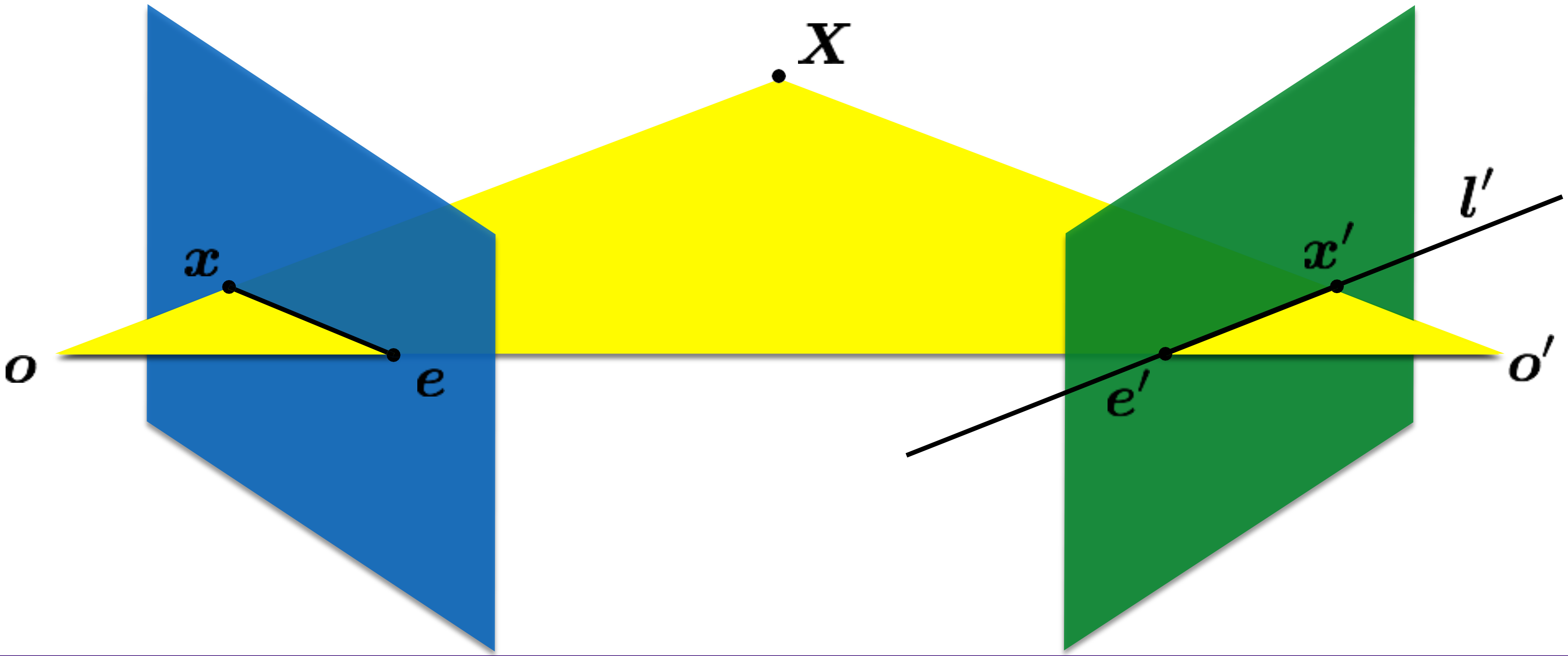


If the point $\boldsymbol{x}$ is on the epipolar line $\boldsymbol{l}$ then

$$\boldsymbol{x}^\top \boldsymbol{l} = 0$$

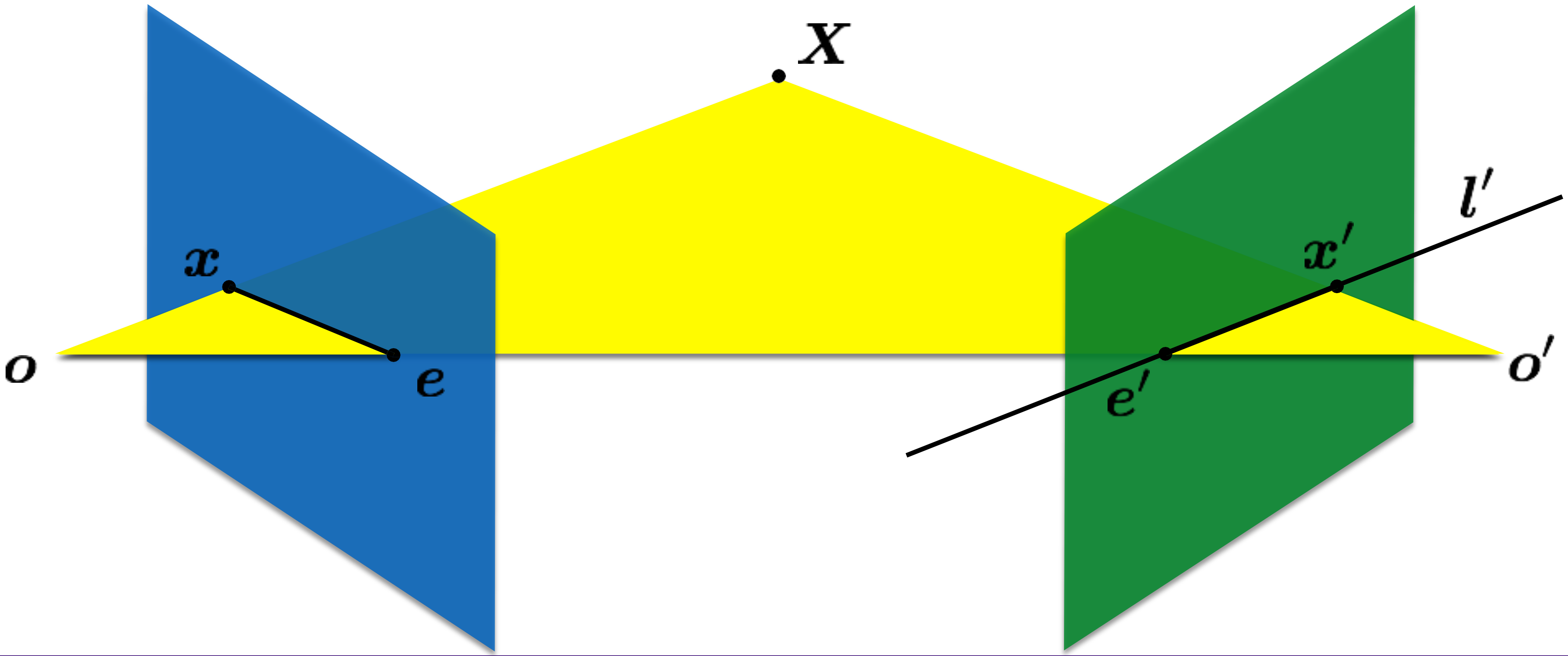
So if $\boldsymbol{x}'^{\top}\boldsymbol{l}' = 0$ and $\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'$ then

$$\boldsymbol{x}'^{\top}\mathbf{E}\boldsymbol{x} = \ ?$$



So if $\boldsymbol{x}'^{\top}\boldsymbol{l}' = 0$ and $\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'$ then

$$\boxed{\boldsymbol{x}'^{\top}\mathbf{E}\boldsymbol{x} = 0}$$



Essential Matrix vs Homography

What's the difference between the essential matrix and a homography?

They are both 3 x 3 matrices but …

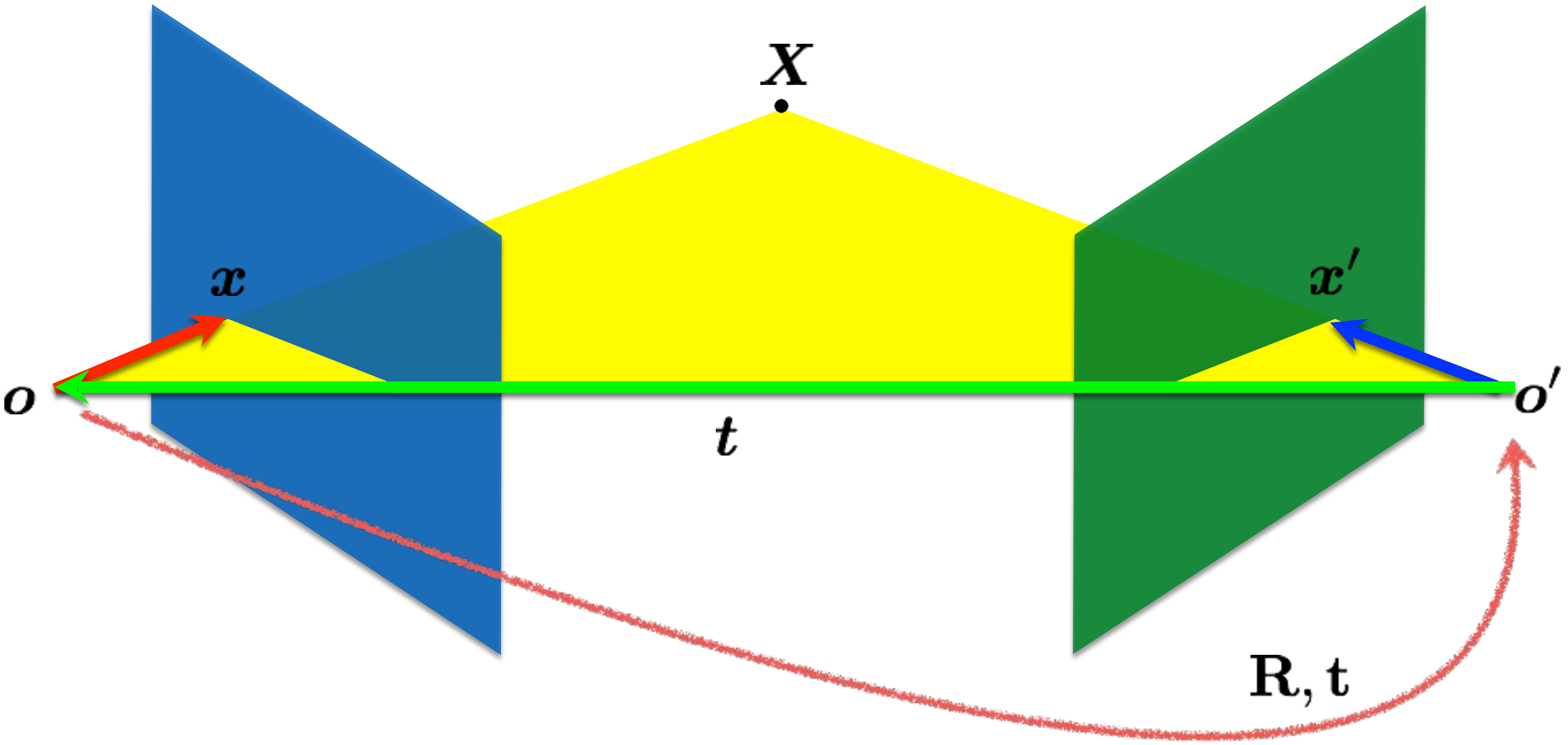
$$\boldsymbol{l}' = \mathbf{E}\boldsymbol{x}$$

Essential matrix maps a **point** to a **line**

$$\boldsymbol{x}' = \mathbf{H}\boldsymbol{x}$$

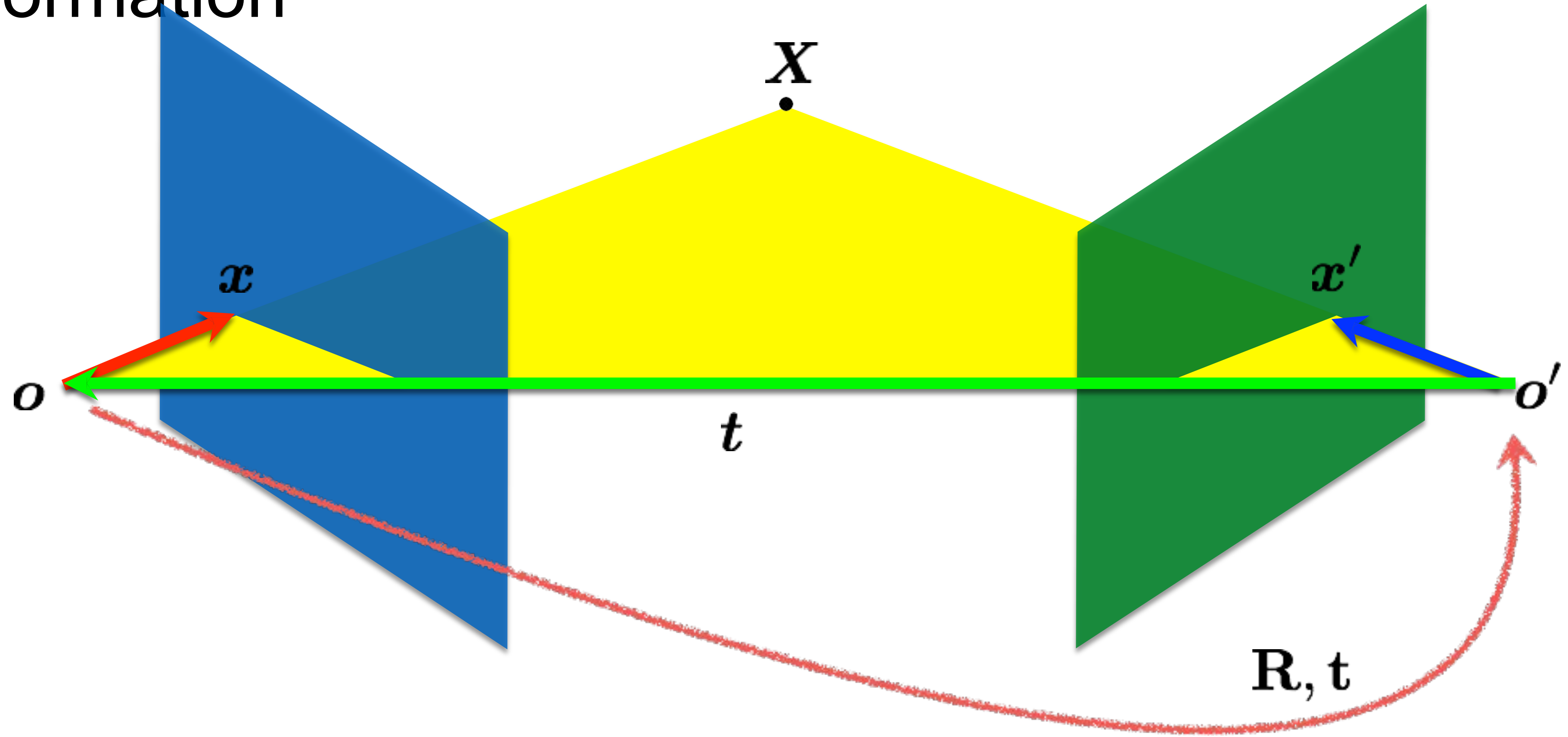
Homography maps a **point** to a **point**

Where does the essential matrix come from?

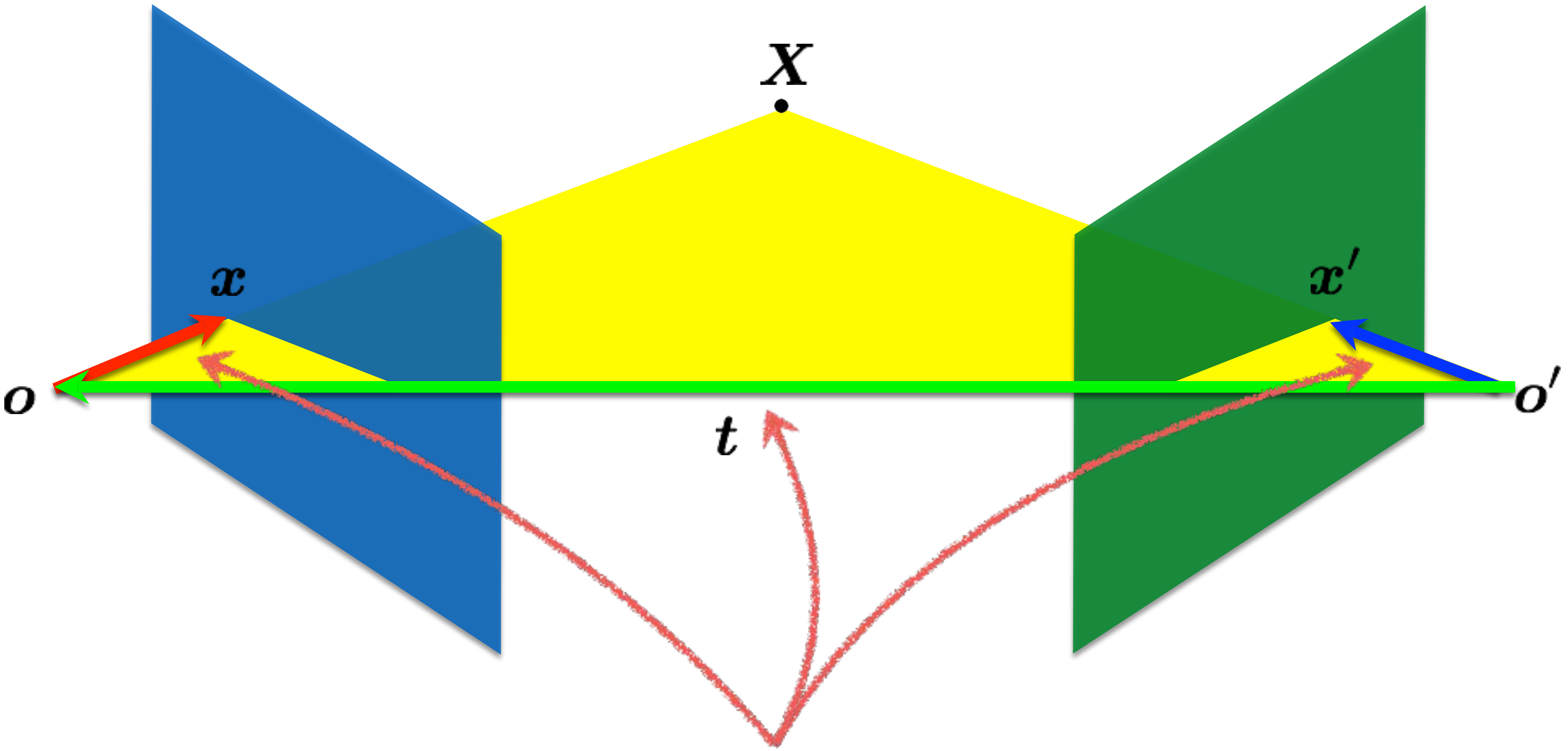


$$x' = \mathbf{R}(x - t)$$

Camera-camera transformation is just like **world-camera** transformation

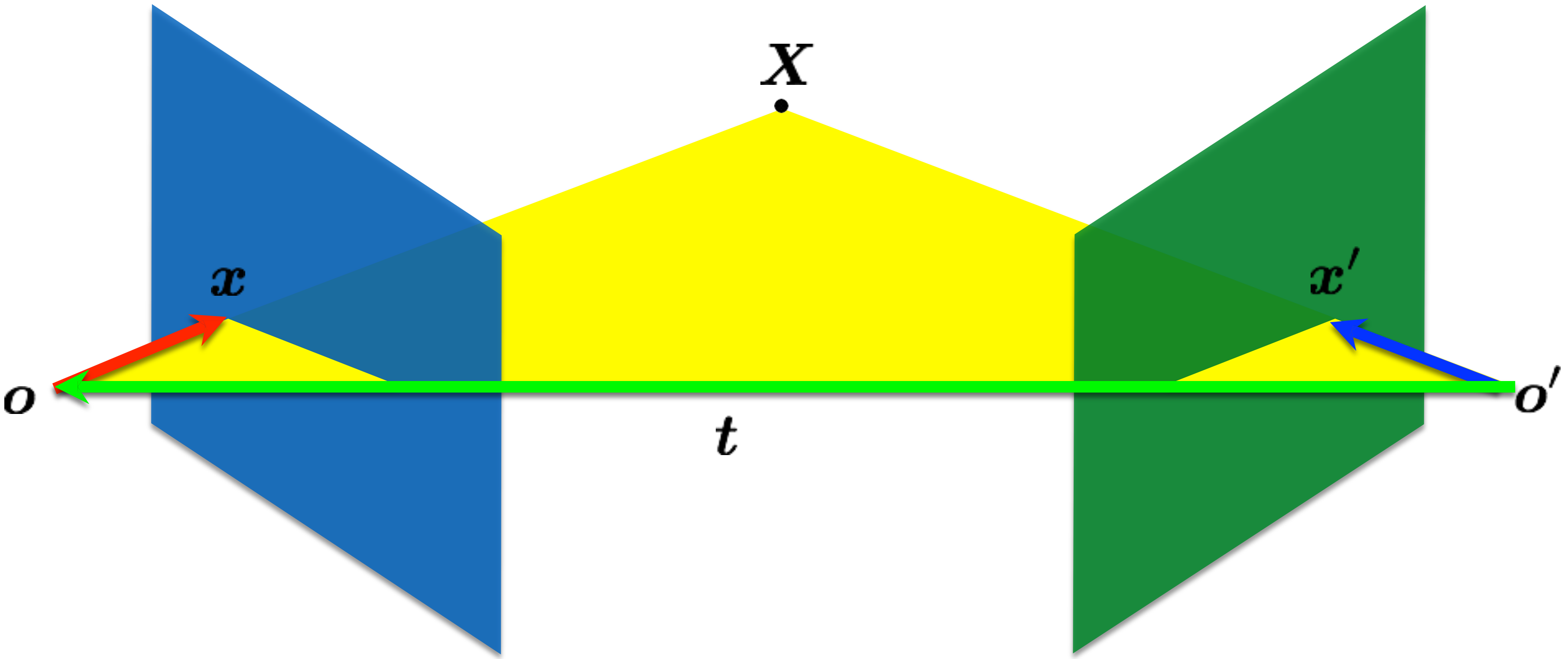


$$x' = \mathbf{R}(x - t)$$



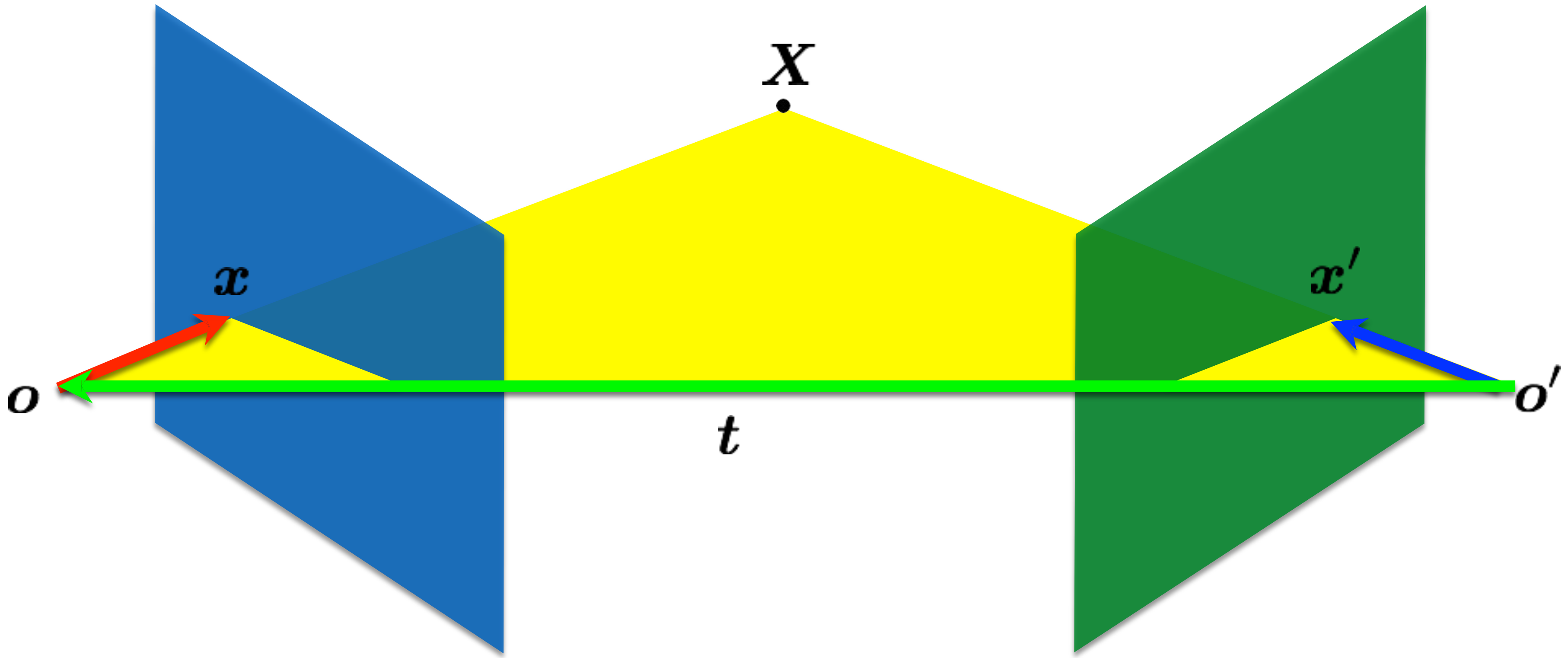
These three vectors are coplanar

$$\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$$



If these three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$ then

$$\boldsymbol{x}^\top(\boldsymbol{t} \times \boldsymbol{x}) = \ ?$$



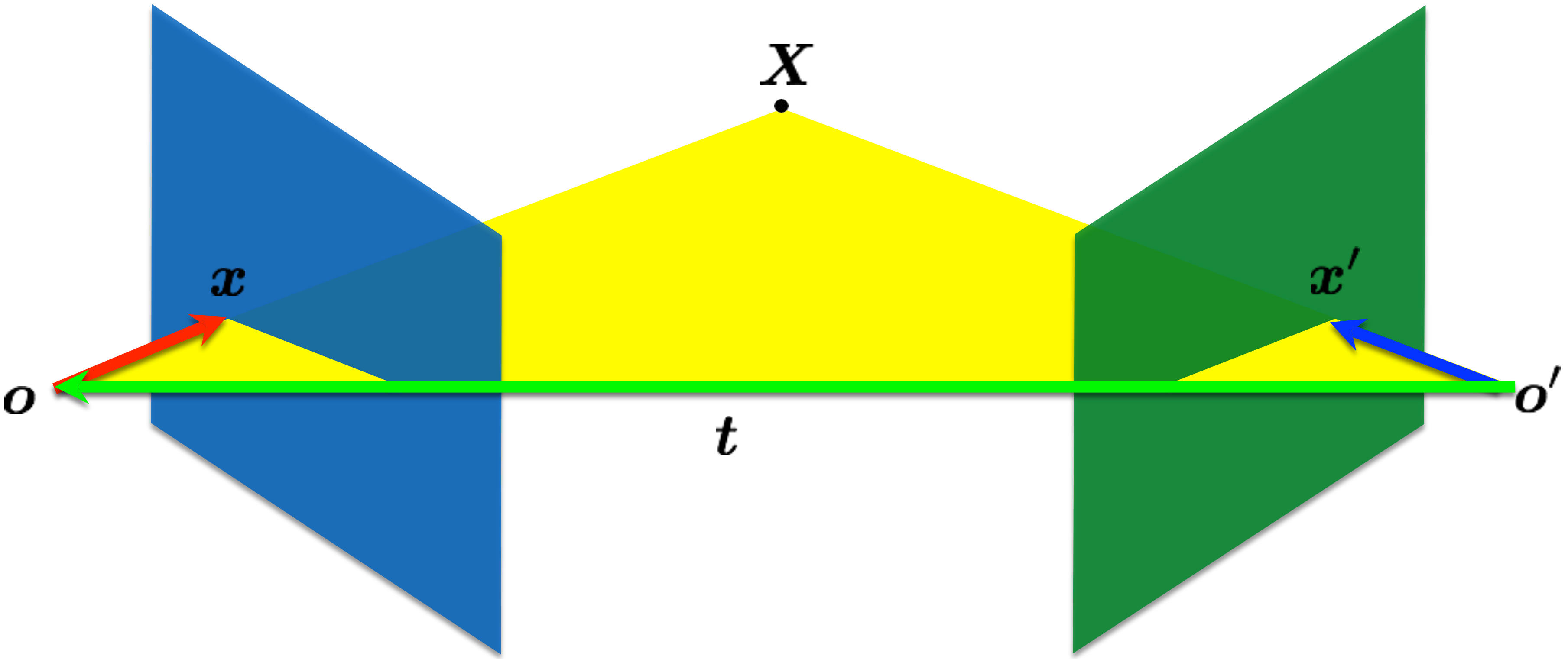
If these three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$ then

dot product of orthogonal vectors

$$\boldsymbol{x}^\top(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

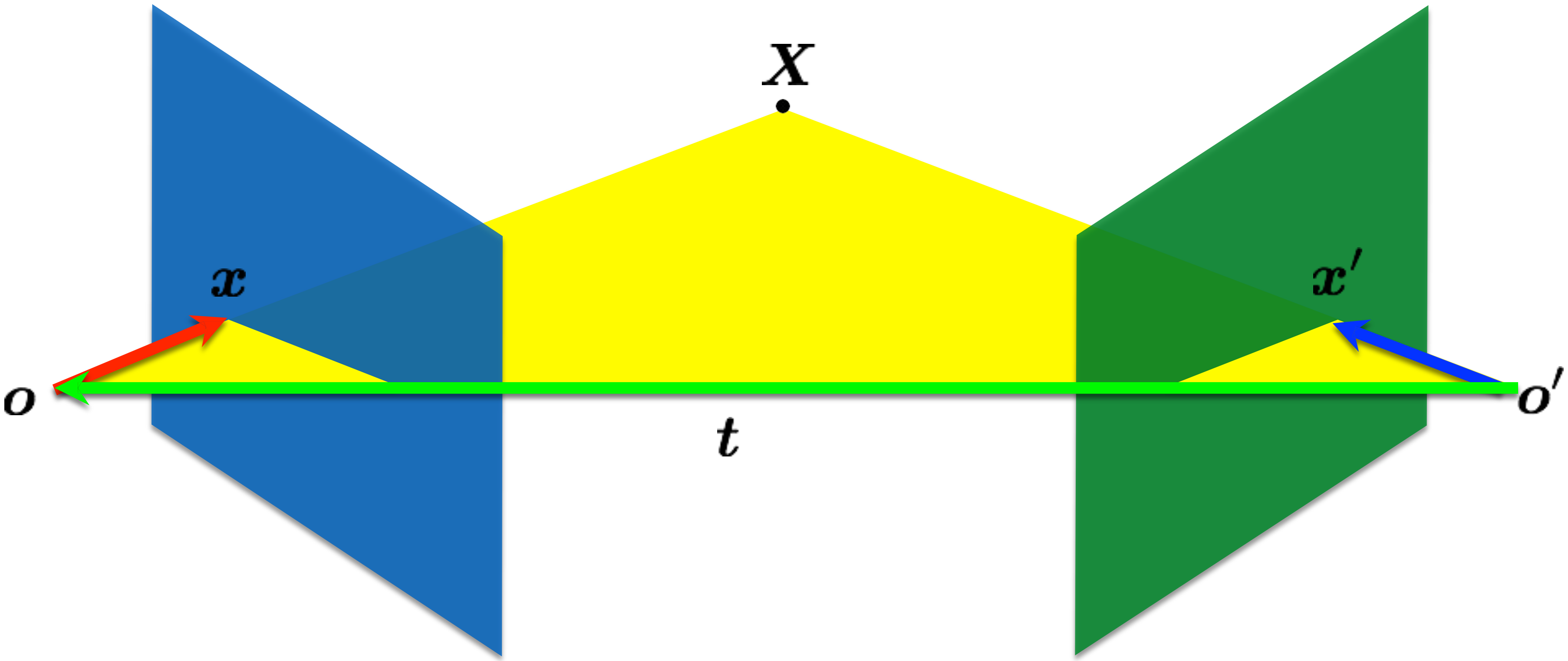
cross-product: vector orthogonal to plane

What is (x - t)? It is another line on the epipolar plane



If these three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$ then

$$(\boldsymbol{x} - \boldsymbol{t})^\top (\boldsymbol{t} \times \boldsymbol{x}) = \ ?$$



If these three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$ then

$$(\boldsymbol{x} - \boldsymbol{t})^\top (\boldsymbol{t} \times \boldsymbol{x}) = 0$$

putting it together

$$\boldsymbol{x}' = \mathbf{R}(\boldsymbol{x} - \boldsymbol{t}) \qquad\qquad (\boldsymbol{x} - \boldsymbol{t})^\top(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

Substituting (x-t):

$$\boldsymbol{x}' = \mathbf{R}(\boldsymbol{x} - \boldsymbol{t}) \qquad\qquad (\boldsymbol{x} - \boldsymbol{t})^\top(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

$$(\boldsymbol{x}'^\top \mathbf{R})(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

Cross product reminder:

$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$$

Can also be written as a matrix multiplication

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_\times \boldsymbol{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Skew symmetric

Use the skew-symmetric
matrix to represent the cross product

$$\boldsymbol{x}' = \mathbf{R}(\boldsymbol{x} - \boldsymbol{t}) \qquad (\boldsymbol{x} - \boldsymbol{t})^\top(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

$$(\boldsymbol{x}'^\top \mathbf{R})(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

$$(\boldsymbol{x}'^\top \mathbf{R})([\mathrm{t}_\times]\boldsymbol{x}) = 0$$

Use the skew-symmetric
matrix to represent the cross product

$$\boldsymbol{x}' = \mathbf{R}(\boldsymbol{x} - \boldsymbol{t}) \qquad\qquad (\boldsymbol{x} - \boldsymbol{t})^\top (\boldsymbol{t} \times \boldsymbol{x}) = 0$$

$$(\boldsymbol{x}'^\top \mathbf{R})(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

$$(\boldsymbol{x}'^\top \mathbf{R})([\mathbf{t}_\times]\boldsymbol{x}) = 0$$

$$\boldsymbol{x}'^\top (\mathbf{R}[\mathbf{t}_\times])\boldsymbol{x} = 0$$

This is the essential matrix

$$\boldsymbol{x}' = \mathbf{R}(\boldsymbol{x} - \boldsymbol{t}) \qquad (\boldsymbol{x} - \boldsymbol{t})^\top (\boldsymbol{t} \times \boldsymbol{x}) = 0$$

$$(\boldsymbol{x}'^\top \mathbf{R})(\boldsymbol{t} \times \boldsymbol{x}) = 0$$

$$(\boldsymbol{x}'^\top \mathbf{R})([\mathbf{t}_\times]\boldsymbol{x}) = 0$$

$$\boldsymbol{x}'^\top (\mathbf{R}[\mathbf{t}_\times])\boldsymbol{x} = 0$$

$$\boldsymbol{x}'^\top \mathbf{E} \boldsymbol{x} = 0$$

Essential Matrix

[Longuet-Higgins 1981]

Summary

Longuet-Higgins equation $\boldsymbol{x}'^\top \mathbf{E}\boldsymbol{x} = 0$

Epipolar lines

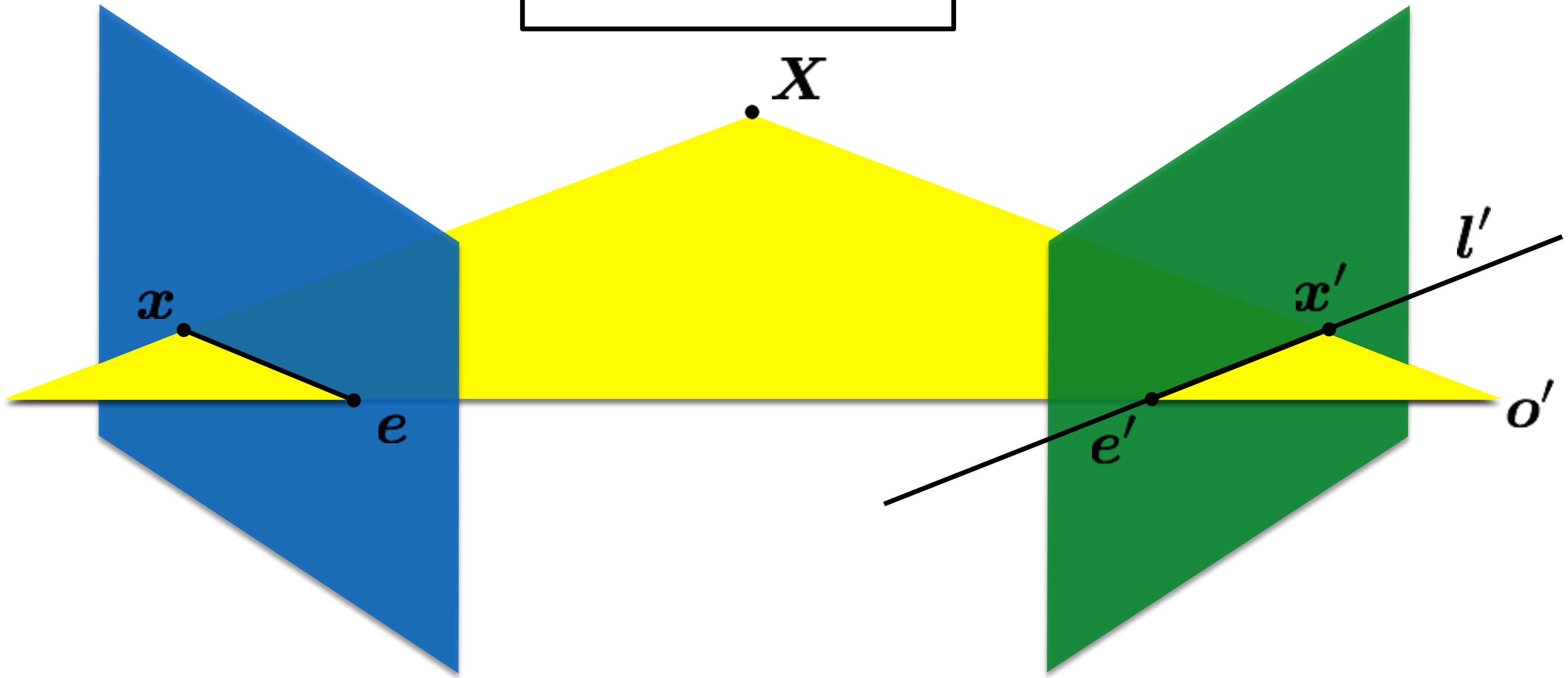
$\boldsymbol{x}^\top \boldsymbol{l} = 0$ $\boldsymbol{x}'^\top \boldsymbol{l}' = 0$

$\boldsymbol{l}' = \mathbf{E}\boldsymbol{x}$ $\boldsymbol{l} = \mathbf{E}^T \boldsymbol{x}'$

$\boldsymbol{e}'^\top \mathbf{E} = \mathbf{0}$ $\mathbf{E}\boldsymbol{e} = \mathbf{0}$

Everything we have done so far assumes:
we have camera coordinates of pixels
but **x** can only be calculated in image coordinates

$$\boxed{\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'}$$



Today's agenda

- Triangulation
- Epipolar geometry
- Essential matrix
- Fundamental matrix
- Structure from motion

$$\hat{\boldsymbol{x}}'^{\top}\mathbf{E}\hat{\boldsymbol{x}} = 0$$

The essential matrix operates on **2D points coordinates** in the camera coordinate system

$$\hat{\boldsymbol{x}}' = \mathbf{K}'^{-1}\boldsymbol{x}'$$

$$\underbrace{\hat{\boldsymbol{x}}}_{\text{camera point}} = \mathbf{K}^{-1}\underbrace{\boldsymbol{x}}_{\text{image point}}$$

$$\hat{\boldsymbol{x}}'^{\top}\mathbf{E}\hat{\boldsymbol{x}} = 0$$

The essential matrix operates on **2D points coordinates** in the camera coordinate system

$$\hat{\boldsymbol{x}}' = \mathbf{K}'^{-1}\boldsymbol{x}'$$

$$\hat{\boldsymbol{x}} = \mathbf{K}^{-1}\boldsymbol{x}$$

camera point

image point

Writing out the epipolar constraint in terms of image coordinates

$$\boldsymbol{x}'^{\top}(\mathbf{K}'^{-\top}\mathbf{E}\mathbf{K}^{-1})\boldsymbol{x} = 0$$

$$\boldsymbol{x}'^{\top}\mathbf{F}\boldsymbol{x} = 0$$

Same equation works in image coordinates!

$$\boldsymbol{x}'^{\top}\mathbf{F}\boldsymbol{x} = 0$$

it maps pixels to epipolar lines

F

$\cancel{E}$

Longuet-Higgins equation	$\boldsymbol{x}'^{\top}\mathbf{E}\boldsymbol{x} = 0$	
Epipolar lines	$\boldsymbol{x}^{\top}\boldsymbol{l} = 0$	$\boldsymbol{x}'^{\top}\boldsymbol{l}' = 0$
	$\boldsymbol{l}' = \mathbf{E}\boldsymbol{x}$	$\boldsymbol{l} = \mathbf{E}^{T}\boldsymbol{x}'$
Epipoles	$\boldsymbol{e}'^{\top}\mathbf{E} = \mathbf{0}$	$\mathbf{E}\boldsymbol{e} = \mathbf{0}$

(points in **image** coordinates)

Breaking down the fundamental matrix

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

$$\mathbf{F} = \mathbf{K}'^{-\top}[\mathbf{t}_{\times}]\mathbf{R}\mathbf{K}^{-1}$$

Depends on both intrinsic and extrinsic parameters

The 8-point algorithm solves for F given a list of corresponding points (x, x')

Assume you have *M* matched *image* points

$$\{\boldsymbol{x}_m, \boldsymbol{x}'_m\} \qquad m = 1, \ldots, M$$

Each correspondence should satisfy

$$\boldsymbol{x}'^{\top}_m \mathbf{F} \boldsymbol{x}_m = 0$$

Each corresponding set of points (x, x')
will give us one equation

$$\boldsymbol{x}_m'^\top \mathbf{F} \boldsymbol{x}_m = 0$$

$$\begin{bmatrix} x_m' & y_m' & 1 \end{bmatrix} \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \begin{bmatrix} x_m \\ y_m \\ 1 \end{bmatrix} = 0$$

Each corresponding set of points (x, x')
will give us one equation

$$\boldsymbol{x}_m'^{\top}\mathbf{F}\boldsymbol{x}_m = 0$$

$$\begin{bmatrix} x_m' & y_m' & 1 \end{bmatrix} \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \begin{bmatrix} x_m \\ y_m \\ 1 \end{bmatrix} = 0$$

$$\begin{aligned} x_m x_m' f_1 + x_m y_m' f_2 + x_m f_3 + \\ y_m x_m' f_4 + y_m y_m' f_5 + y_m f_6 + \\ x_m' f_7 + y_m' f_8 + f_9 = 0 \end{aligned}$$

Like always, we can re-write it as a linear equation with M corresponding pairs of points:

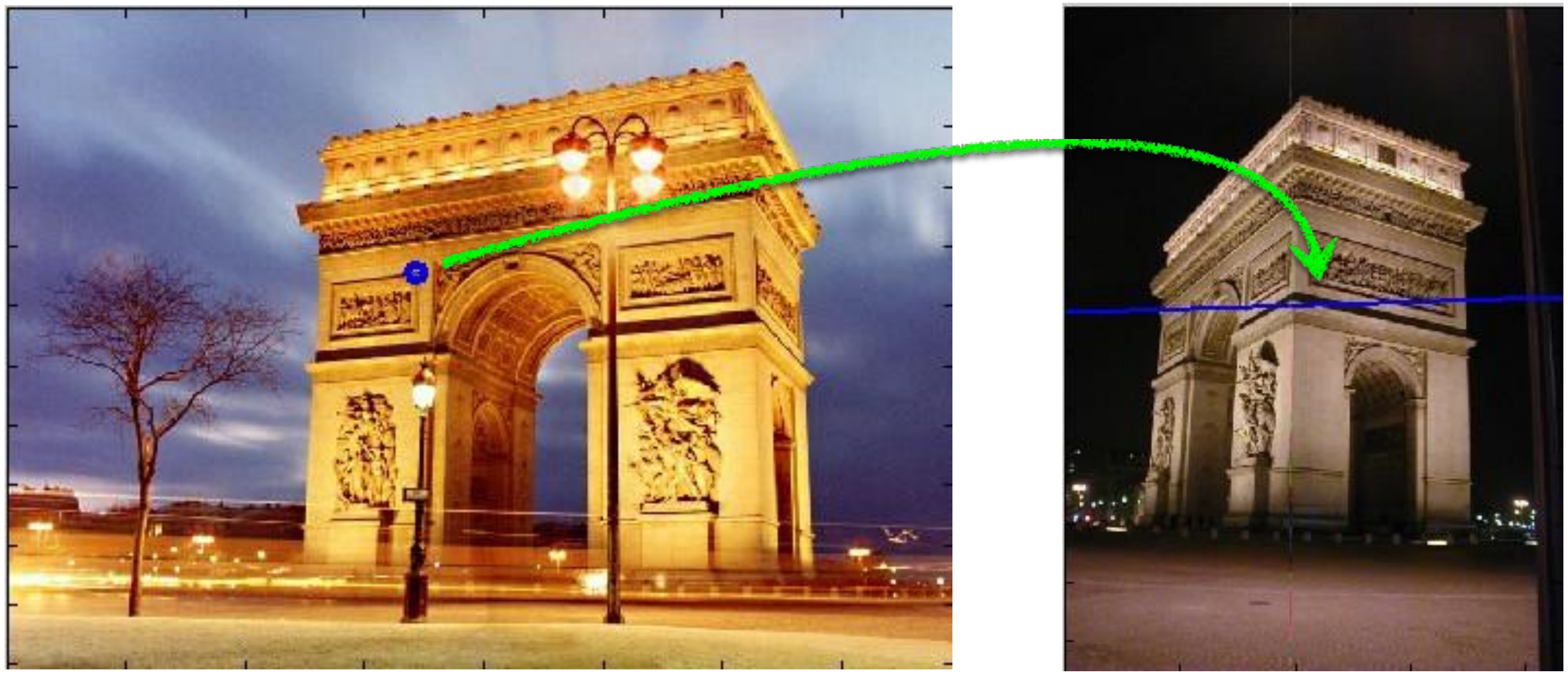
$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_Mx'_M & x_My'_M & x_M & y_Mx'_M & y_My'_M & y_M & x'_M & y'_M & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{bmatrix} = \mathbf{0}$$

Solve using SVD!

You can find correspondences using Harris + RANSAC

You can use the corresponding points to calculate F

Once you have F, you can map points to epipolar lines:



Here are a bunch of epipolar lines across these two images

Today's agenda

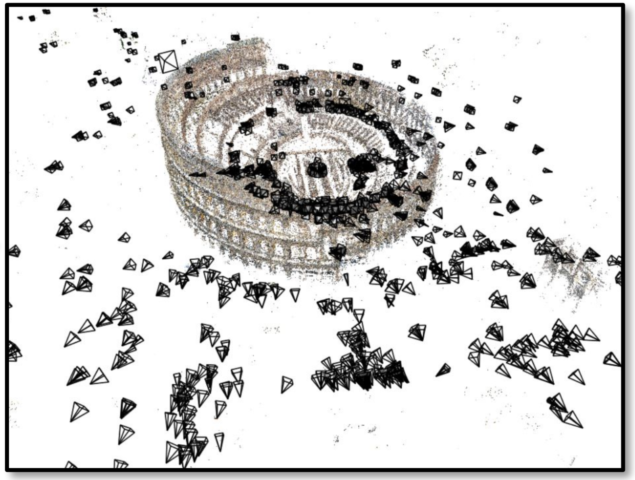
- Triangulation
- Epipolar geometry
- Essential matrix
- Fundamental matrix
- Structure from motion

Structure-from-Motion

Given many images, how can we

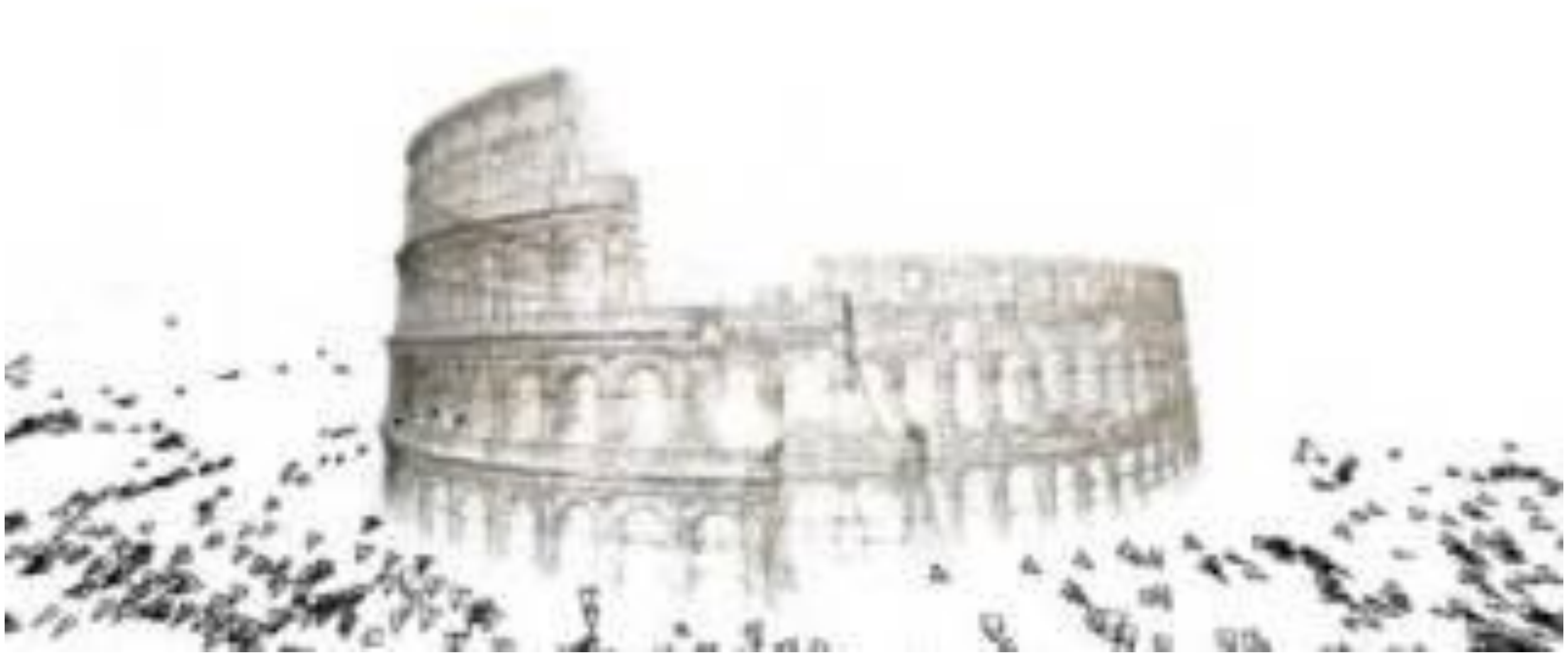
a) figure out where they were all taken from?

b) build a 3D model of the scene?



N. Snavely, S. Seitz, and R. Szeliski, Photo tourism: Exploring photo collections in 3D, SIGGRAPH 2006.
http://phototour.cs.washington.edu/

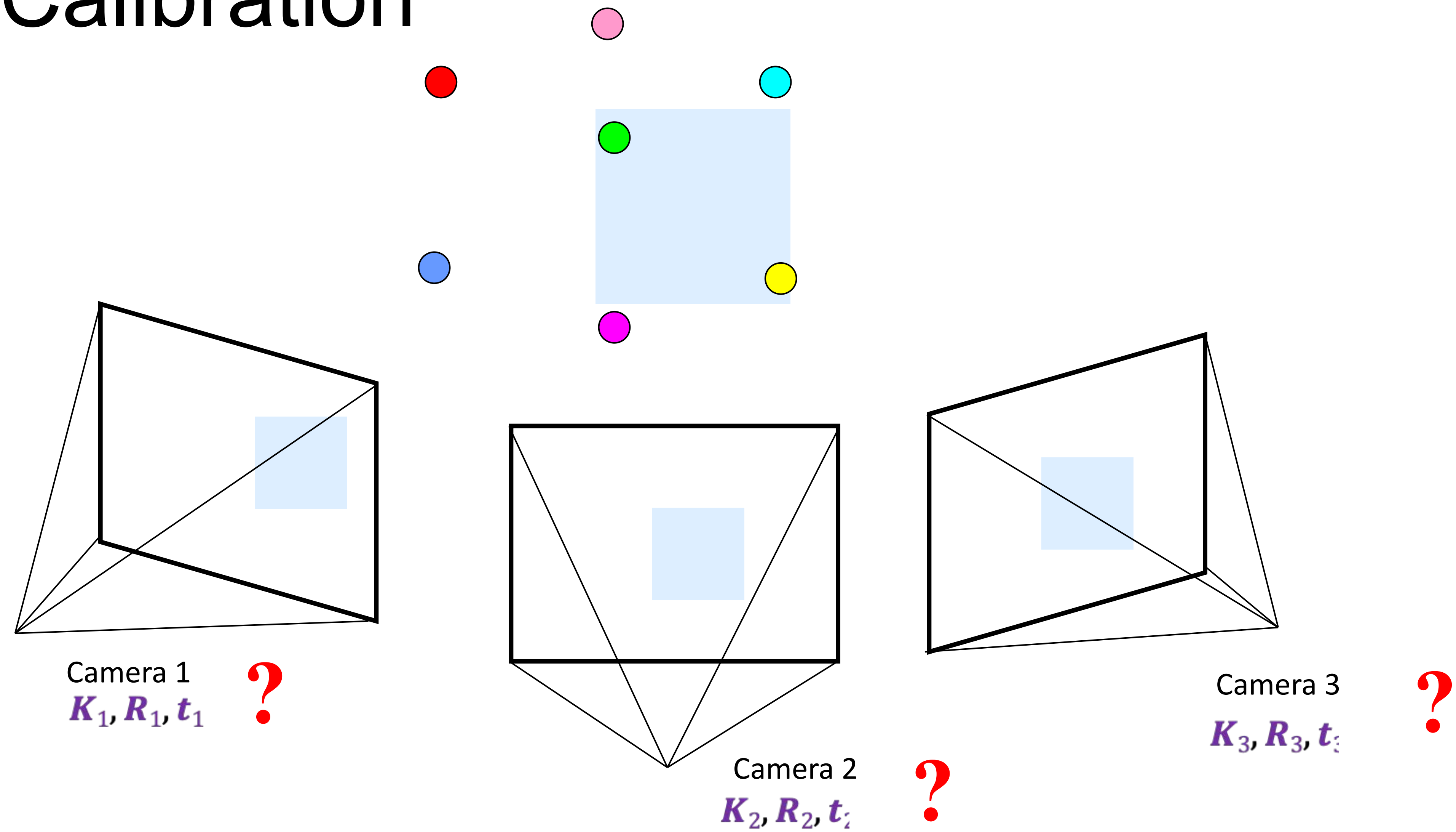
Large-scale structure-from-motion



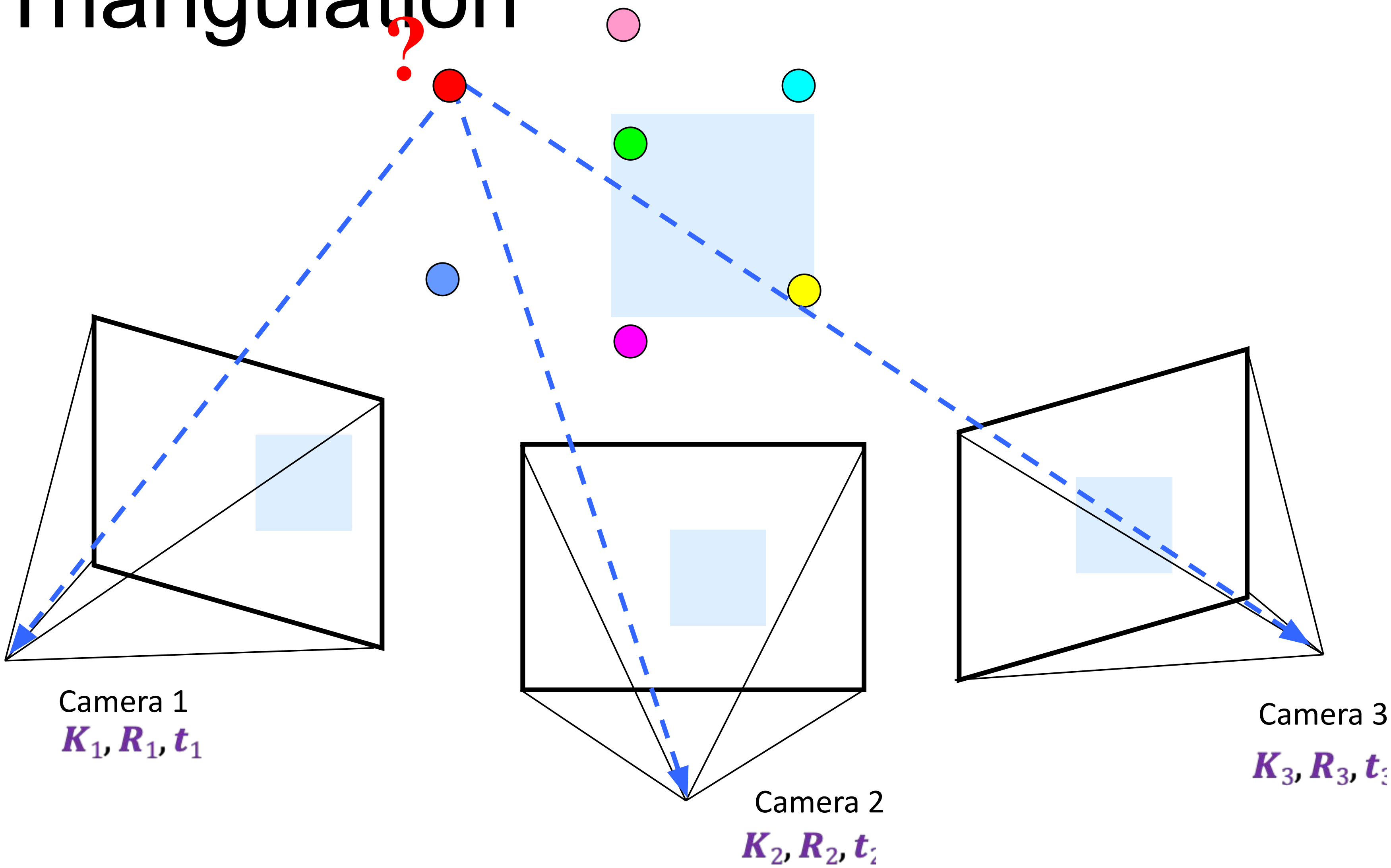
Dubrovnik, Croatia. 4,619 images (out of an initial 57,845 downloaded from Flickr). 3.5M points!
Total reconstruction time: 17.5 hours on 352 cores

Building Rome in a Day, *Agarwal et al, ICCV'09*
http://grail.cs.washington.edu/rome/

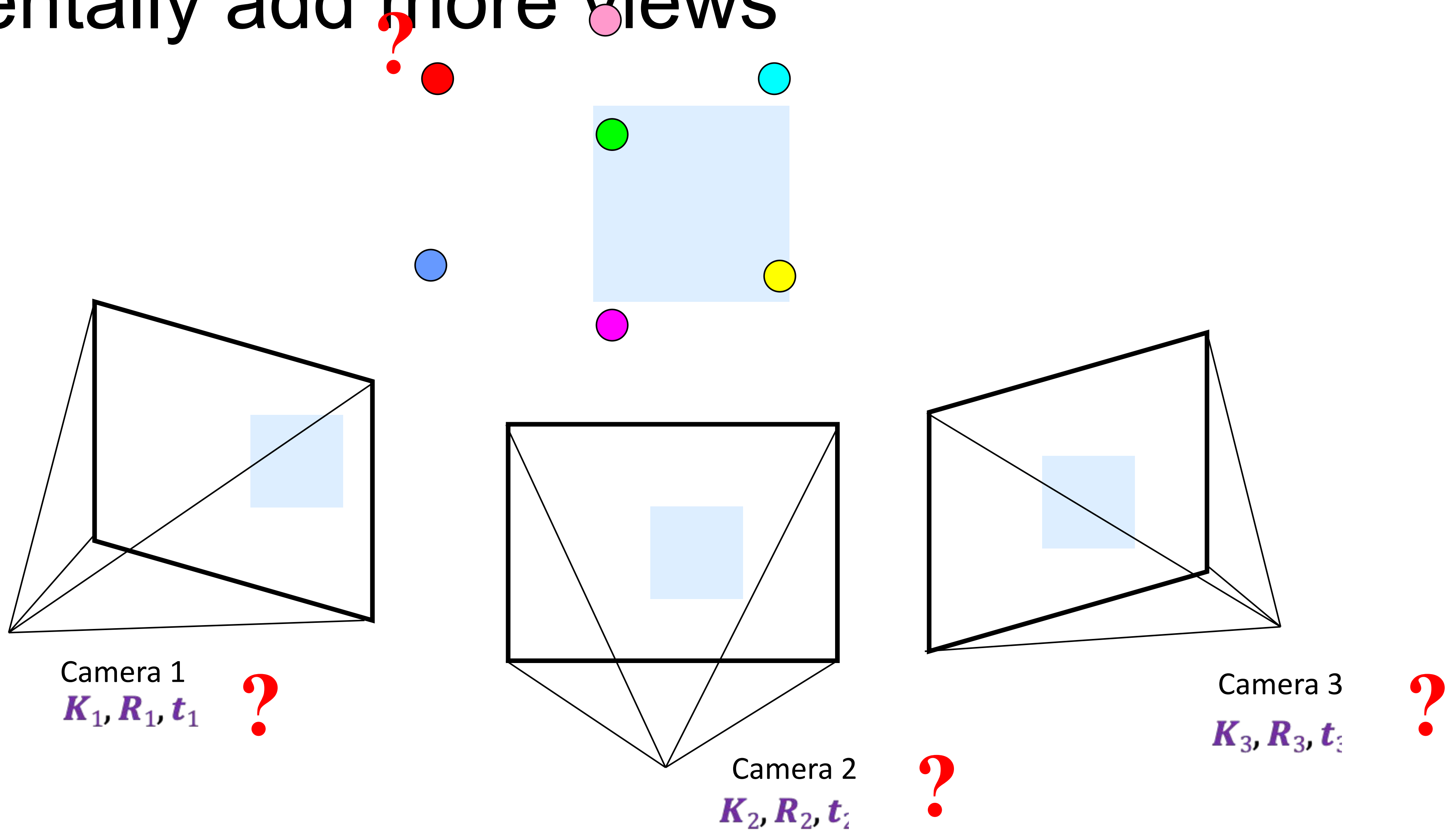
First: Calibration



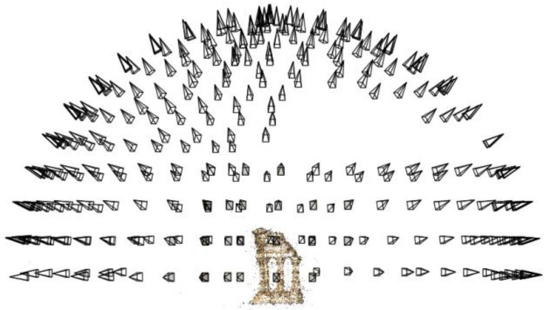
Next: Triangulation



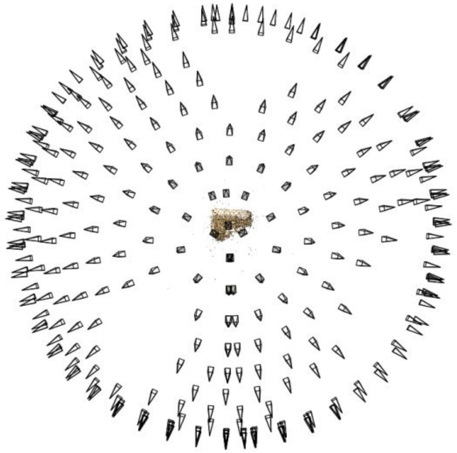
That's it: you get structure-from-motion as you incrementally add more views



Structure-from-Motion



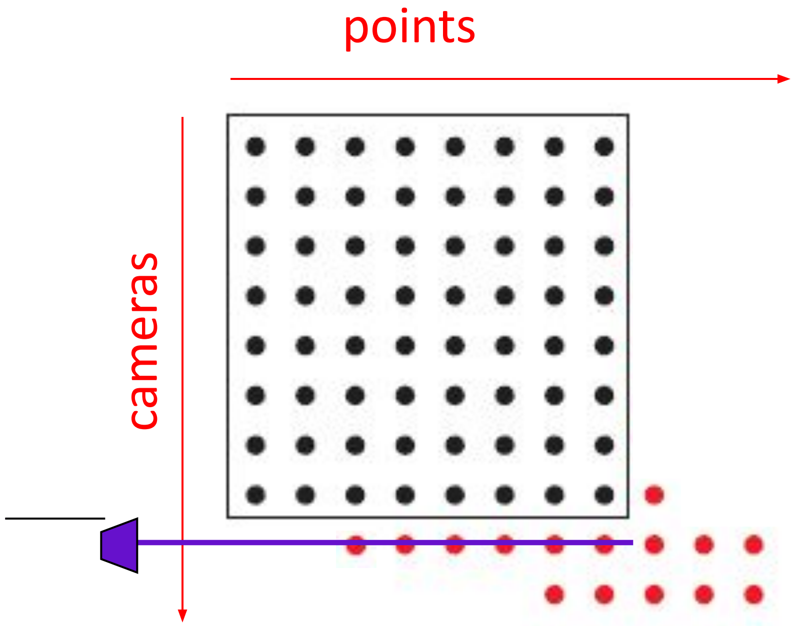
Reconstruction
(side)



(top)

Incremental Structure-from-Motion

- Estimate motion between two images by calculating the fundamental matrix
- Estimate 3D structure by triangulation
- For each additional view:
  - Determine motion of new camera using all the known 3D points that have correspondence in the new image
  - Add new structure by estimating the new points in the new image



Incremental structure from motion

Time-lapse reconstruction of Dubrovnik, Croatia, viewed from above

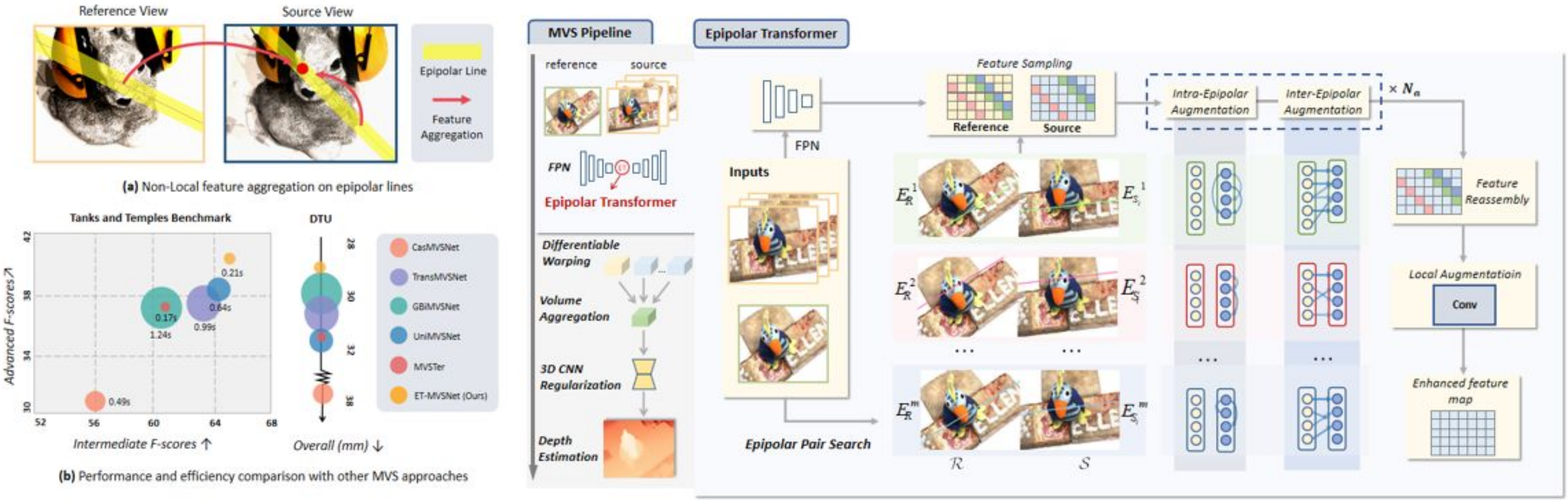
COLMAP

Sparse model of central Rome using 21K photos produced by COLMAP's SfM pipeline.

Dense models of several landmarks produced by COLMAP's MVS pipeline.

https://colmap.github.io/

SfM in the age of Deep Learning



ET-MVSNet: When Epipolar Constraint Meets Non-local Operators in Multi-View Stereo (ICCV'23)

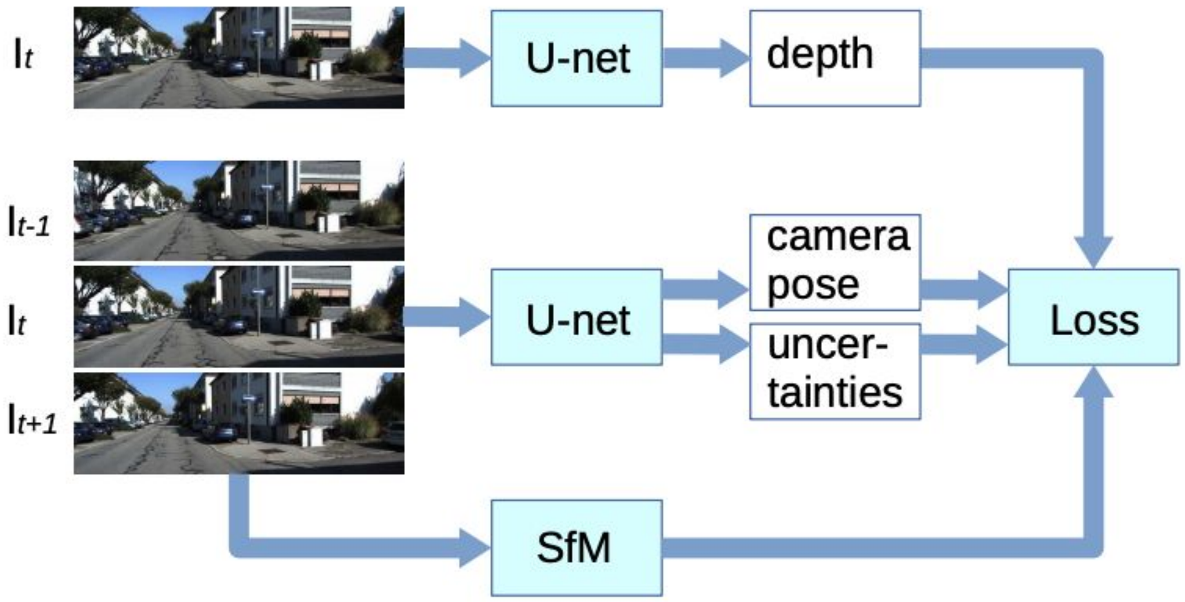
See also MVSFormer: Multi-View Stereo by Learning Robust Image Features and Temperature-based Depth (TMLR'23)

Supervising the new with the old: learning SFM from SFM

Maria Klodt[0000−0003−3015−9584] and Andrea Vedaldi[0000−0003−1374−2858]

Visual Geometry Group, University of Oxford
{klodt,vedaldi}@robots.ox.ac.uk

Abstract. Recent work has demonstrated that it is possible to learn deep neural networks for monocular depth and ego-motion estimation from unlabelled video sequences, an interesting theoretical development with numerous advantages in applications. In this paper, we propose a number of improvements to these approaches. First, since such self-supervised approaches are based on the brightness constancy assumption, which is valid only for a subset of pixels, we propose a probabilistic learning formulation where the network predicts distributions over variables rather than specific values. As these distributions are conditioned on the observed image, the network can learn which scene and object types are likely to violate the model assumptions, resulting in more robust learning. We also propose to build on dozens of years of experience in developing handcrafted structure-from-motion (SFM) algorithms. We do so by using an off-the-shelf SFM system to generate a supervisory signal for the deep neural network. While this signal is also noisy, we show that our probabilistic formulation can learn and account for the defects of SFM, helping to integrate different sources of information and boosting the overall performance of the network.



(b) proposed network architecture: the depth and pose-uncertainty networks are supervised by traditional SfM.

https://openaccess.thecvf.com/content_ECCV_2018/papers/Maria_Klodt_Supervising_the_new_ECCV_2018_paper.pdf

Today's agenda

- Triangulation
- Epipolar geometry
- Essential matrix
- Fundamental matrix
- Structure from motion

Next lecture

The frontiers of Computer Vision