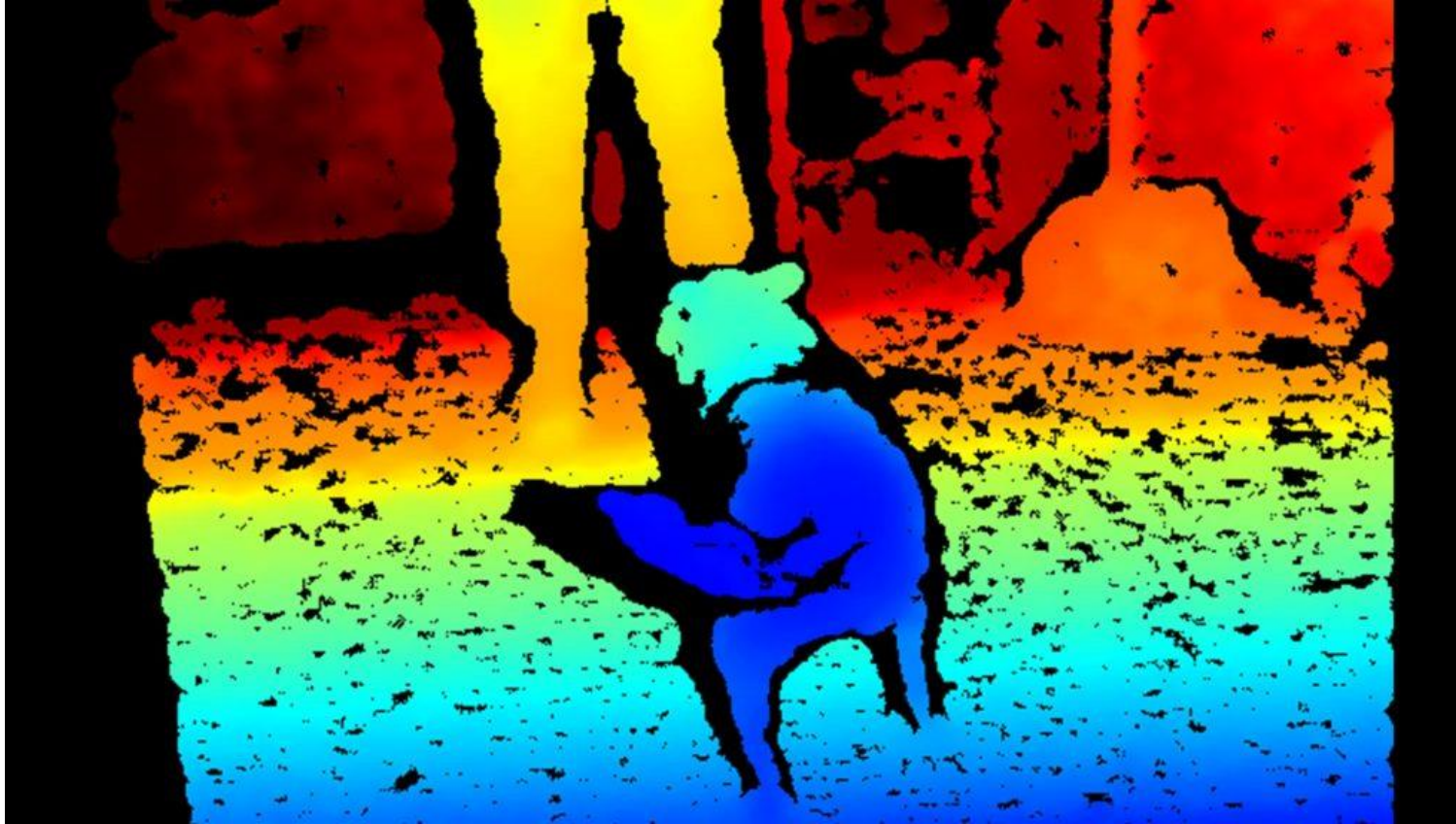
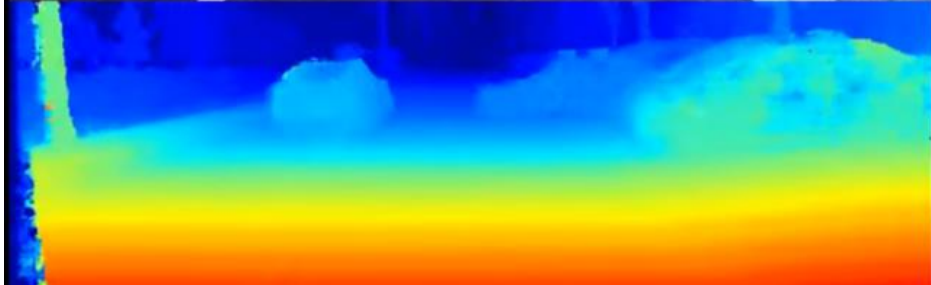


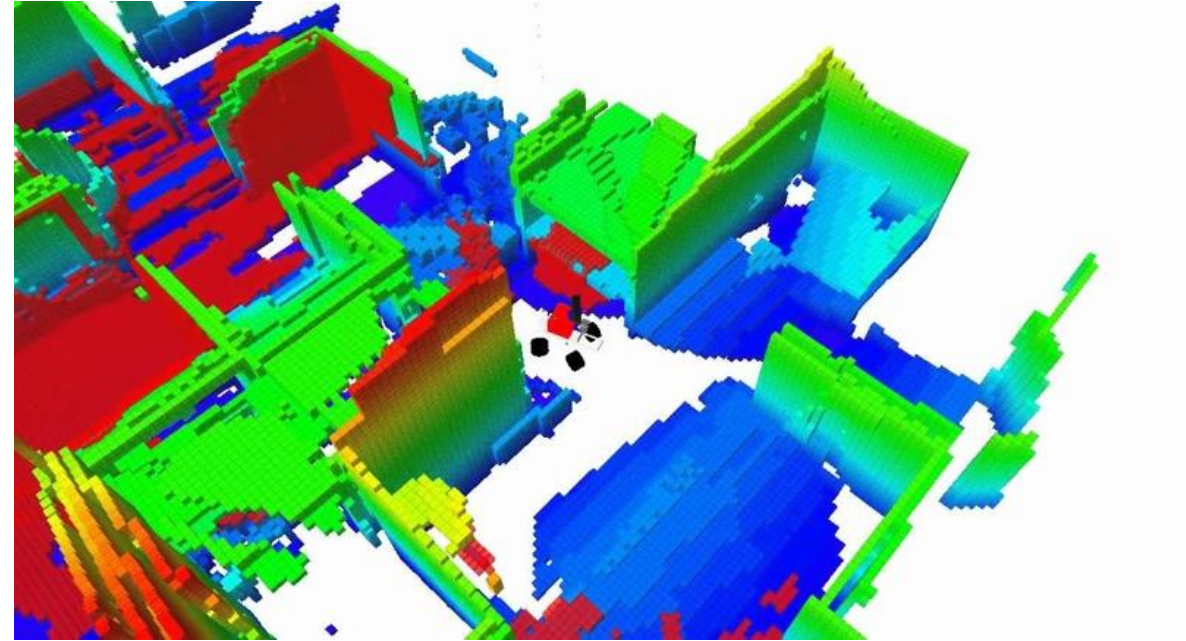
# Stereo



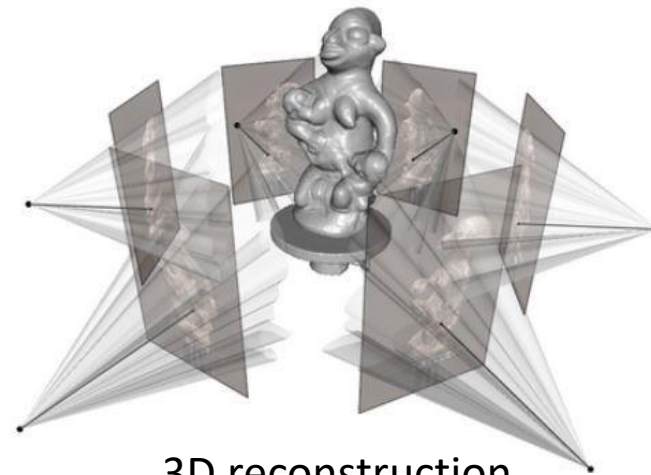
# What can be done with stereo vision?



Autonomous driving



SLAM- robot navigation



3D reconstruction

# References

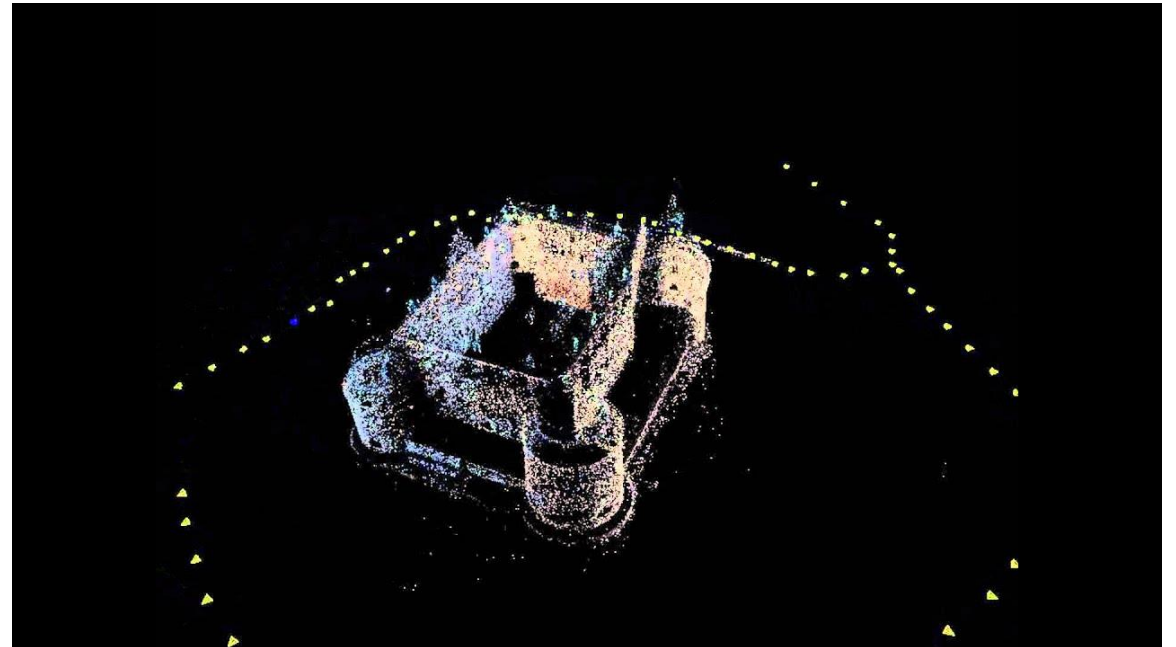
- <http://szeliski.org/Book/>
- <http://www.cs.cornell.edu/courses/cs5670/2019sp/lectures/lectures.html>
- <http://www.cs.cmu.edu/~16385/>

# Contents

- **structure from motion**
- Epipolar geometry
  - Essential matrix
  - Fundamental matrix
  - Estimating the fundamental matrix
- Camera rectification
- Triangulation
- Stereo matching
- Other 3D sensors

# Structure from motion

- **Structure from motion (SfM)** is the process of estimating the 3-D structure of a scene from a set of 2-D images. SfM is used in many applications, such as 3-D scanning and augmented reality.
  - [Mathworks]
- SfM is also known as **3D reconstruction**.
- **Stereo vision** is a subcategory of SfM in which we are dealing only with 2 images.



# Structure and motion

	Structure (3D model of world)	Motion (6 DOFs of cameras)	Measurements
Pose Estimation (camera pose estimation)	known	<b>estimate</b>	3D to 2D correspondences
Triangulation	<b>estimate</b>	known	2D to 2D correspondences
3D reconstruction/ SfM/ stereo vision	<b>estimate</b>	<b>estimate</b>	2D to 2D correspondences + triangulation

# Structure and motion

- So essentially one can say that “structure from motion” is the wrong name...
  - Structure and motion is more precise, but nobody will understand what are you talking about.
- In this class we are dealing with 3D reconstruction and Triangulation.



# Contents

- structure from motion
- **Epipolar geometry**
  - Essential matrix
  - Fundamental matrix
  - Estimating the fundamental matrix
- Camera rectification
- Triangulation
- Stereo matching
- Other 3D sensors



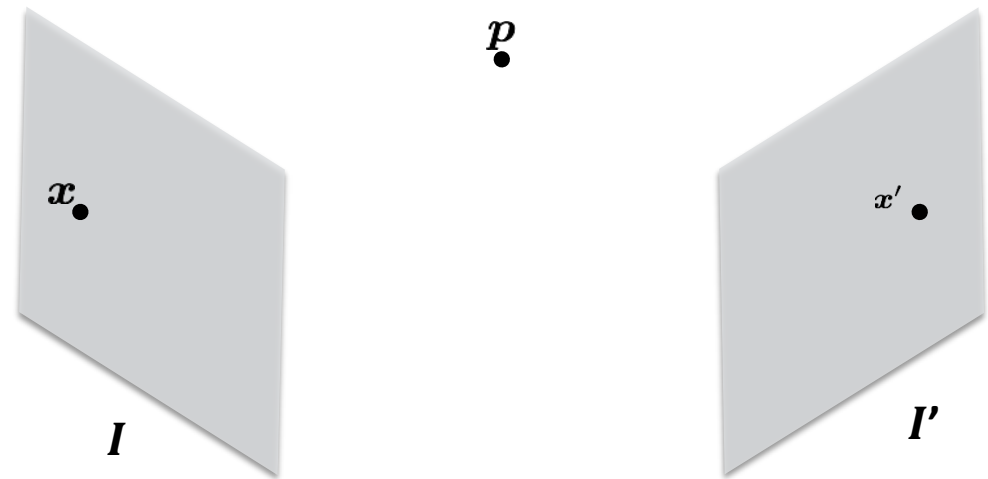
# Epipolar geometry

- **Epipolar geometry** is the geometry of stereo vision. When two cameras view a 3D scene from two distinct positions, there are a number of geometric relations between the 3D points and their projections onto the 2D images that lead to constraints between the image points.
  - [Wikipedia]

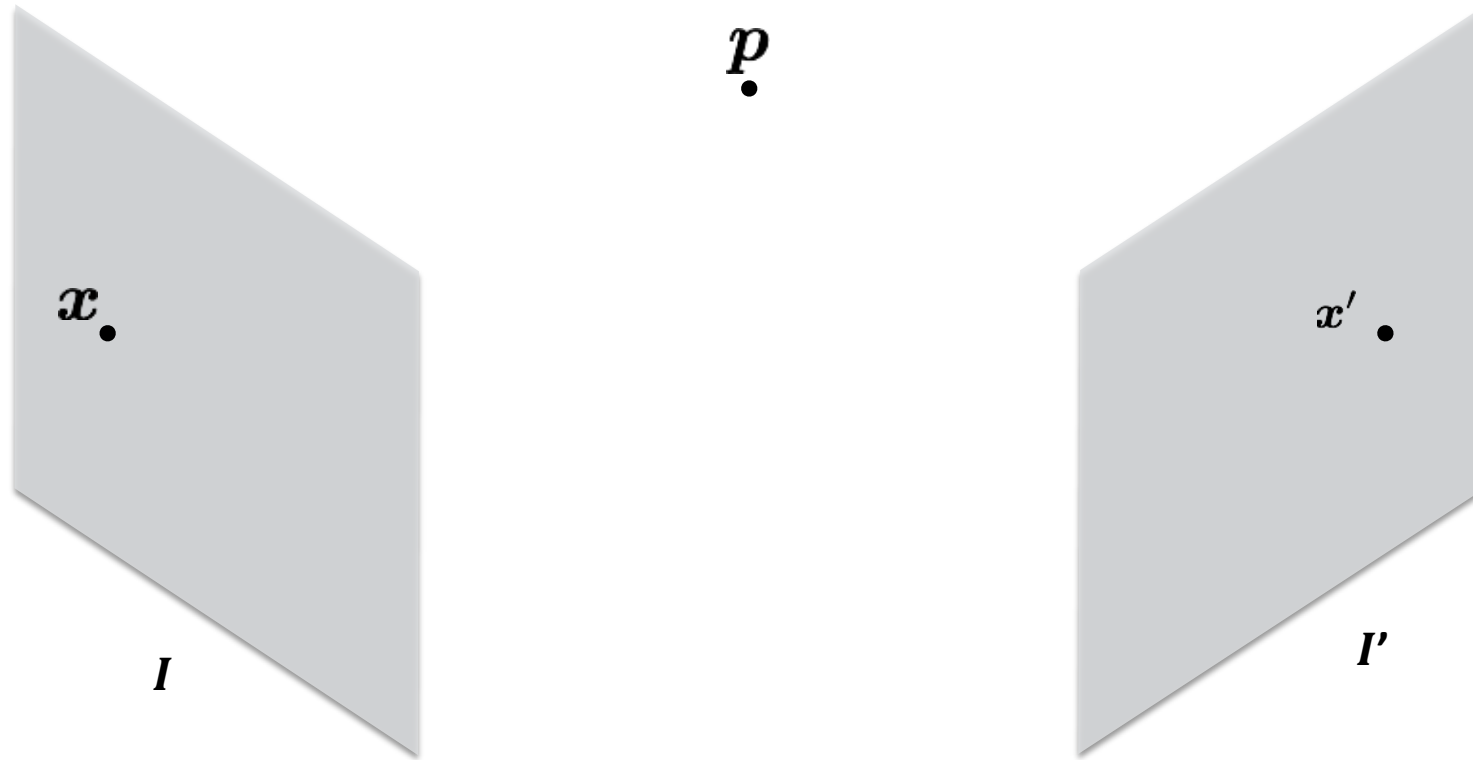
# Epipolar geometry - The triangulation problem

- Given:
  - two 2D points in the **normalized image coordinate system**  $(x, x')$  in two different images  $(I, I')$  that describes the same point  $p$  in 3D space.
  - Rotation and translation between the two cameras.
- Find  $p$ .

- Normalized image coordinate system:  $x = K^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$

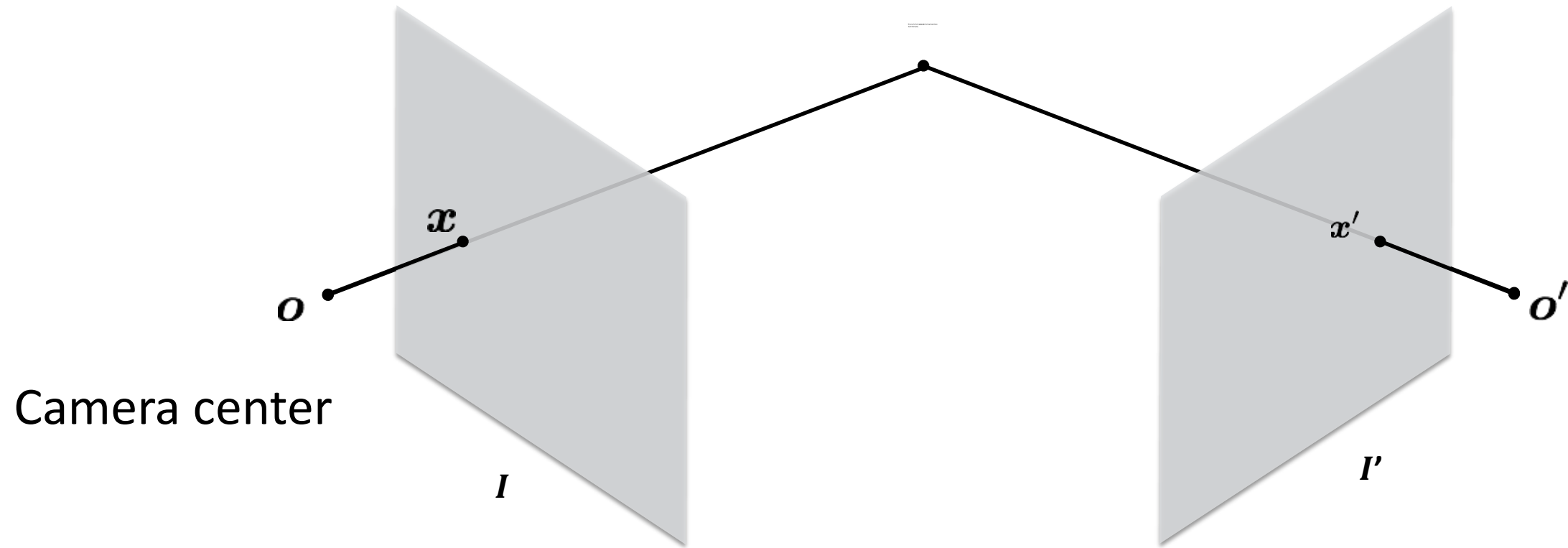


# Epipolar geometry



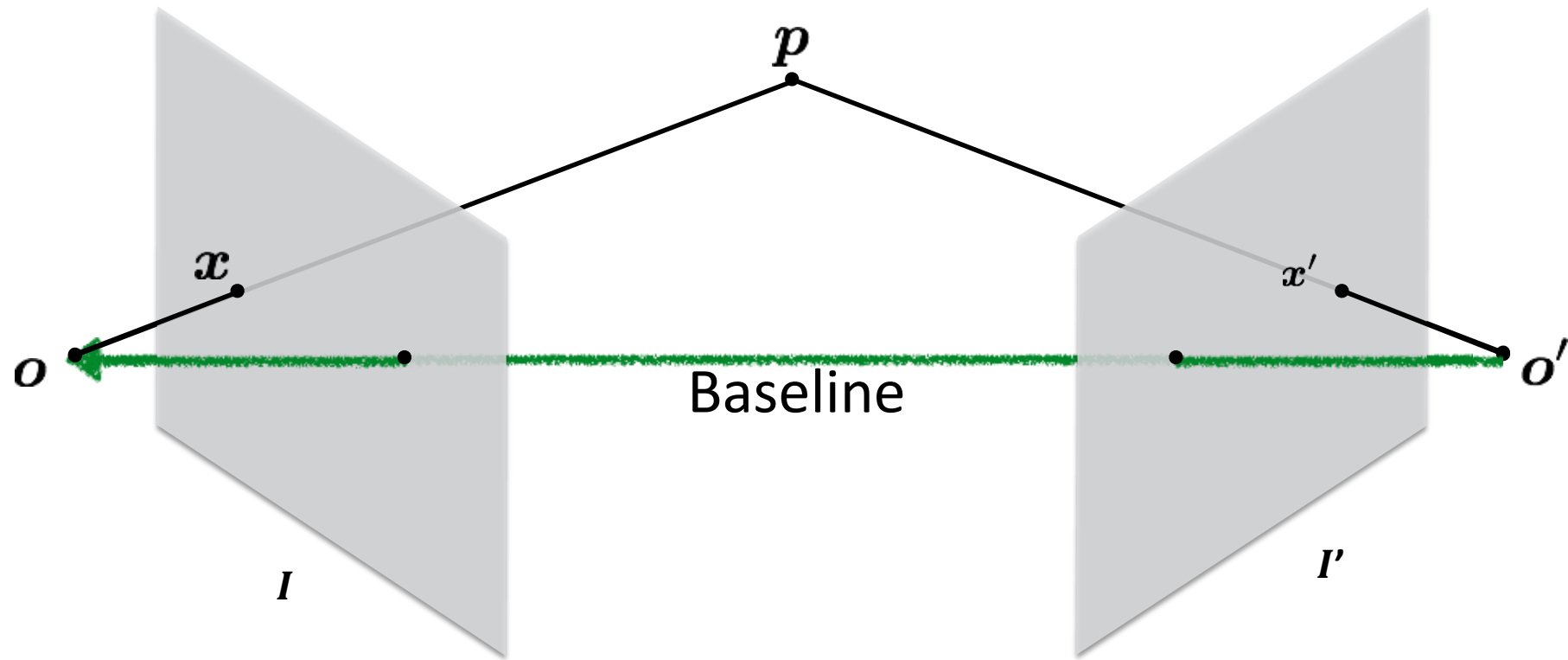
# Epipolar geometry

- We can trace lines from the **camera center** of each image, through the given 2D point to the 3D point  $p$ .



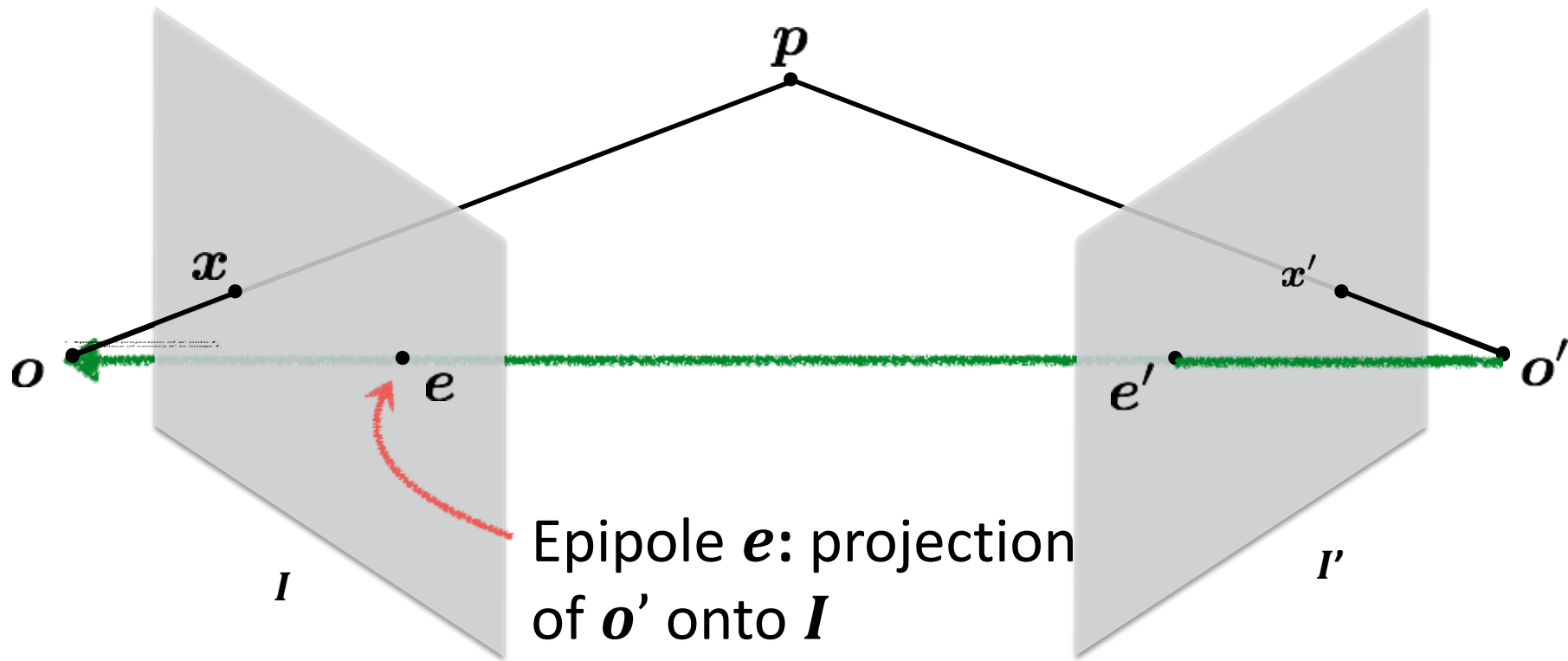
# Epipolar geometry

- **Baseline** is a vector that represent the translation between two cameras



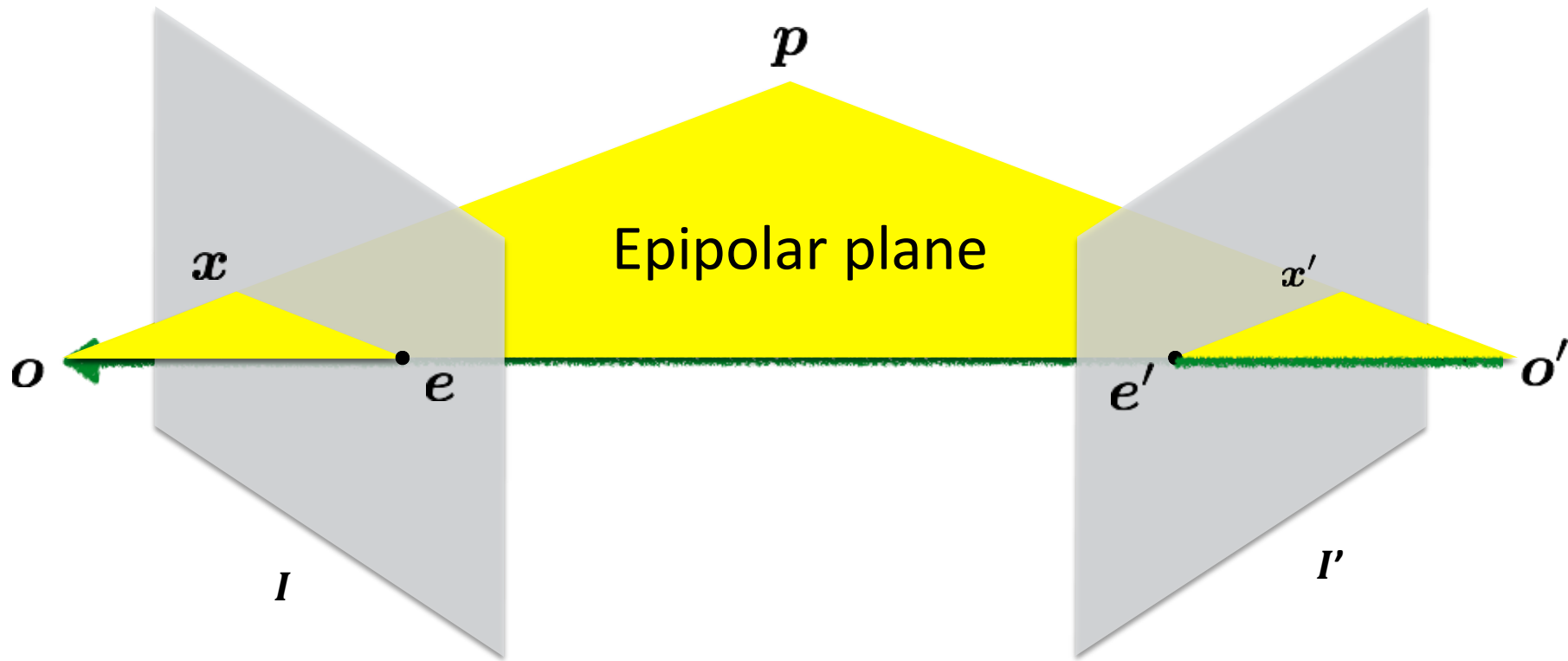
# Epipolar geometry

- **Epipole  $e$ :** projection of  $o'$  onto  $I$ .
  - The place of camera  $o'$  in image  $I$ .



# Epipolar geometry

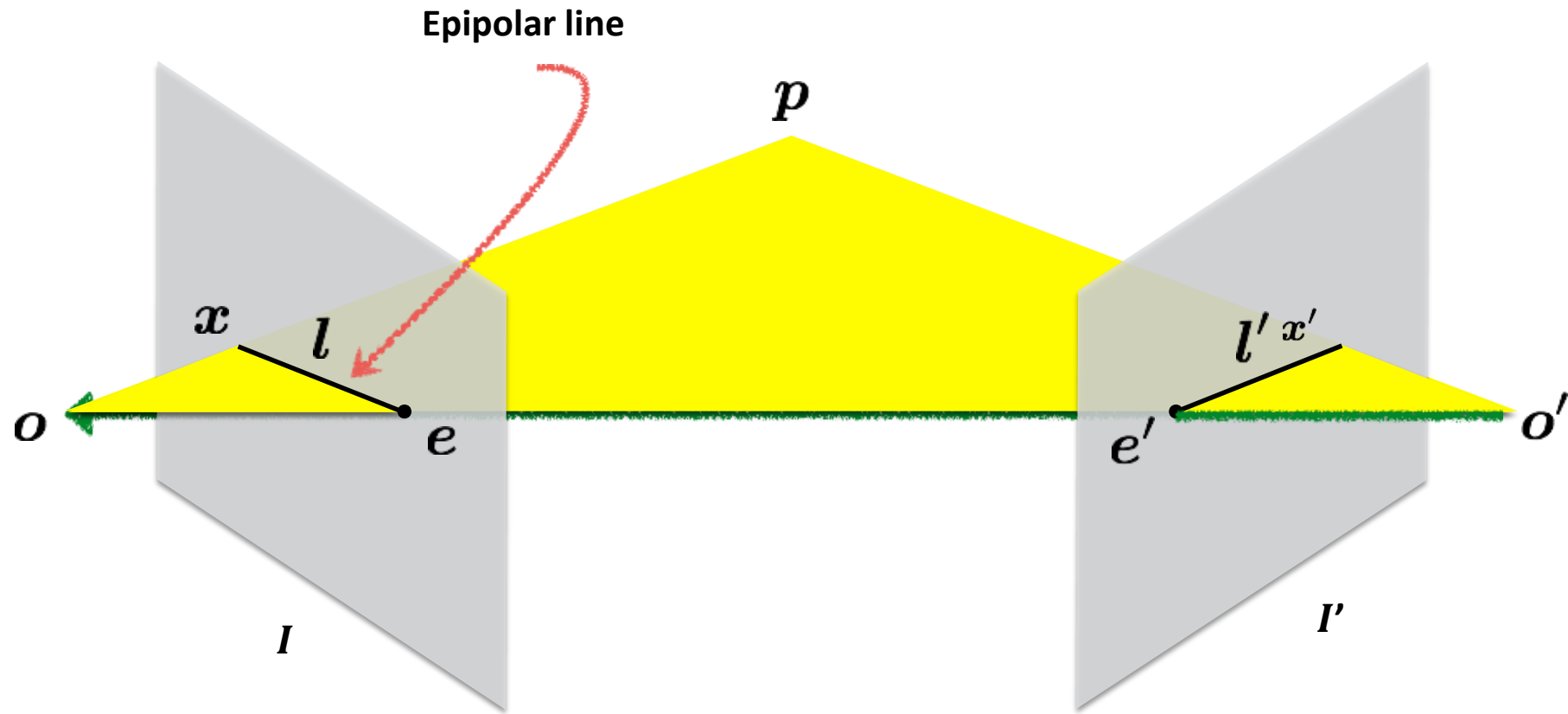
- **Epipolar plane:** the plane that is constructed from the 3 points  $(p, o, o')$ .





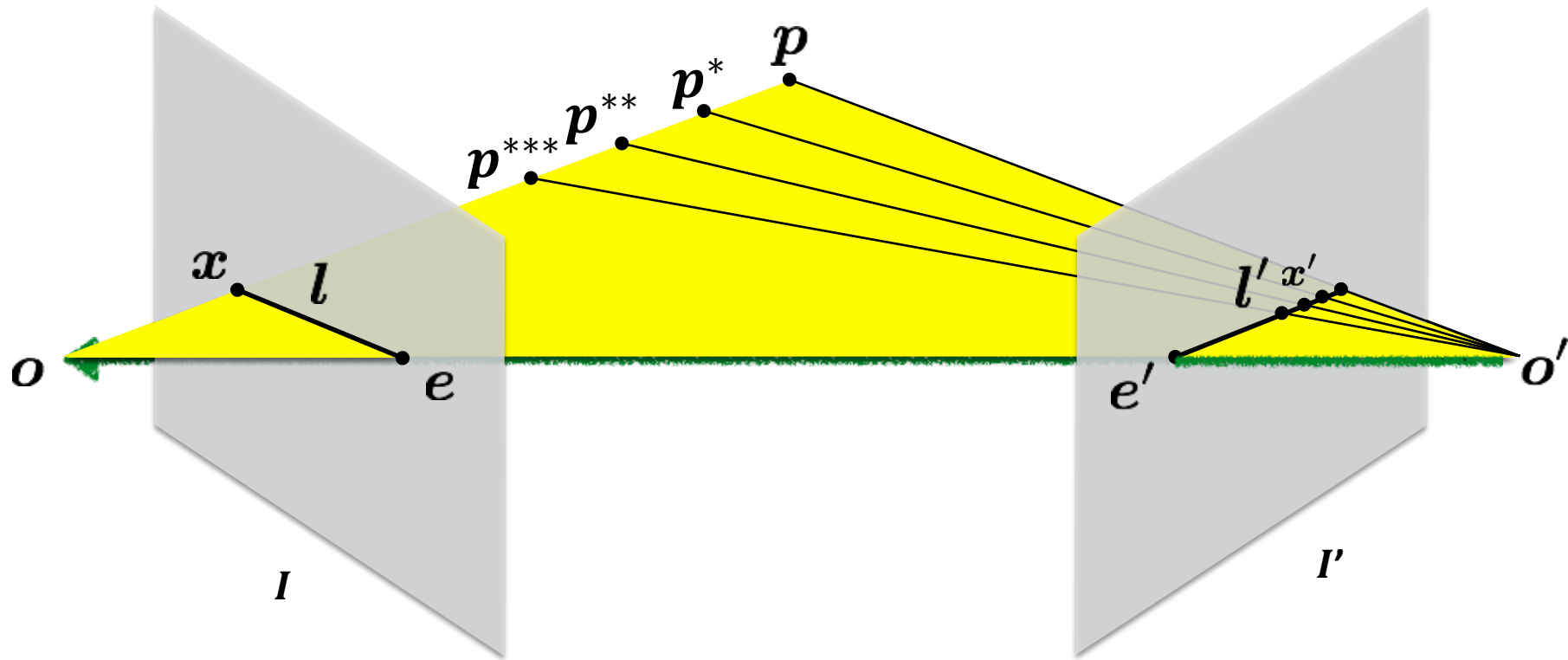
# Epipolar geometry

- **Epipolar line:** intersection of Epipolar plane and image plane.



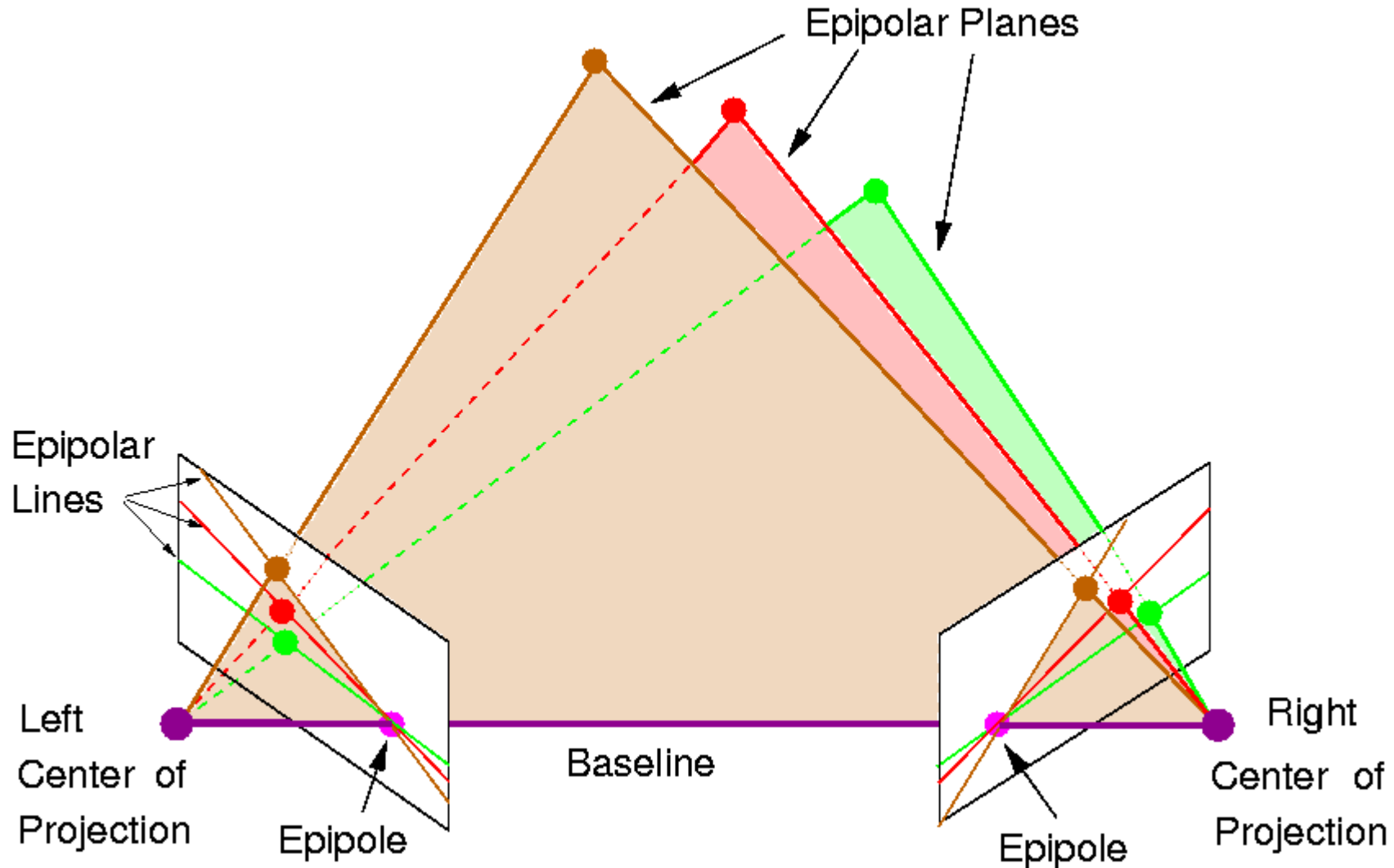
# Epipolar constraint

- **The epipolar constraint:** a point  $x$  in image  $I$  is mapped onto an epipolar line  $l'$  in image  $I'$ .
  - This happens since we don't know  $p$  in advance.



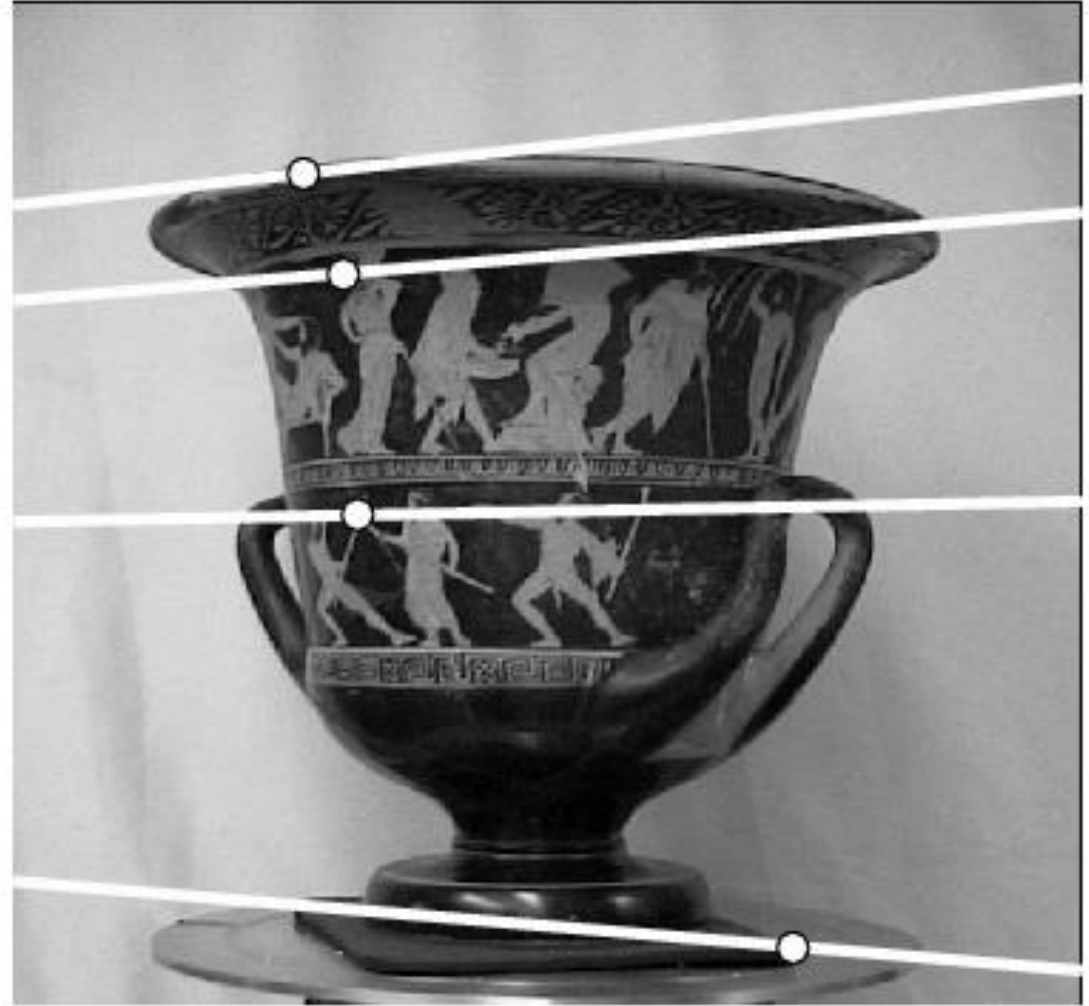
# Epipolar geometry

- Note: all epipolar lines pass through the epipole.



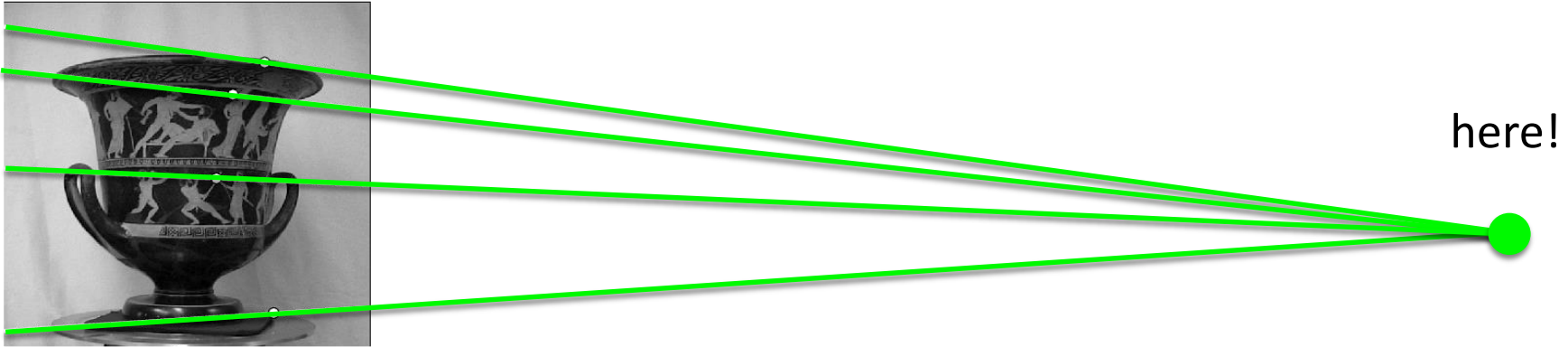
# Epipolar geometry

- Where is the epipole in this images?



# Epipolar geometry

- Where is the epipole in this images? The epipole doesn't have to be inside the image!



# Epipolar geometry

- Where is the epipole in this image?



# Epipolar geometry

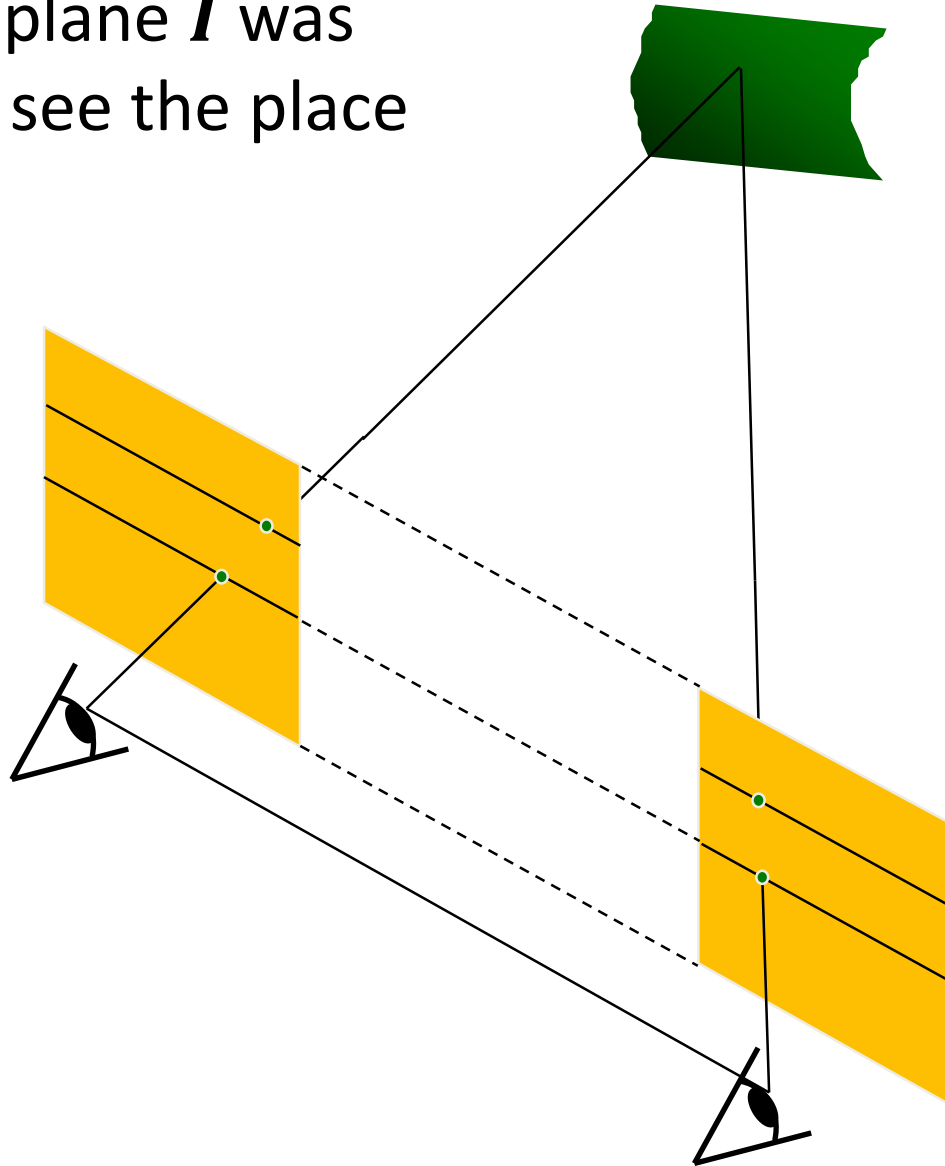
- Where is the epipole in this image? The epipolar lines doesn't converge since the baseline translation is parallel to the image plane!





# Epipolar geometry

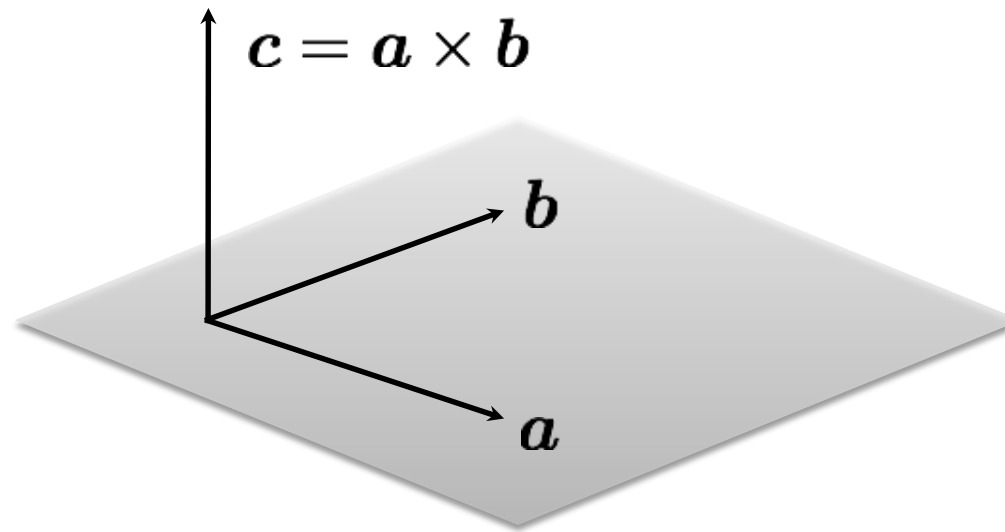
- Even if the image plane  $I$  was infinite, you can't see the place of  $o'$ .



# Contents

- structure from motion
- Epipolar geometry
  - **Essential matrix**
  - Fundamental matrix
  - Estimating the fundamental matrix
- Camera rectification
- Triangulation
- Stereo matching
- Other 3D sensors

# Recall: Dot Product



$$c \cdot a = 0$$

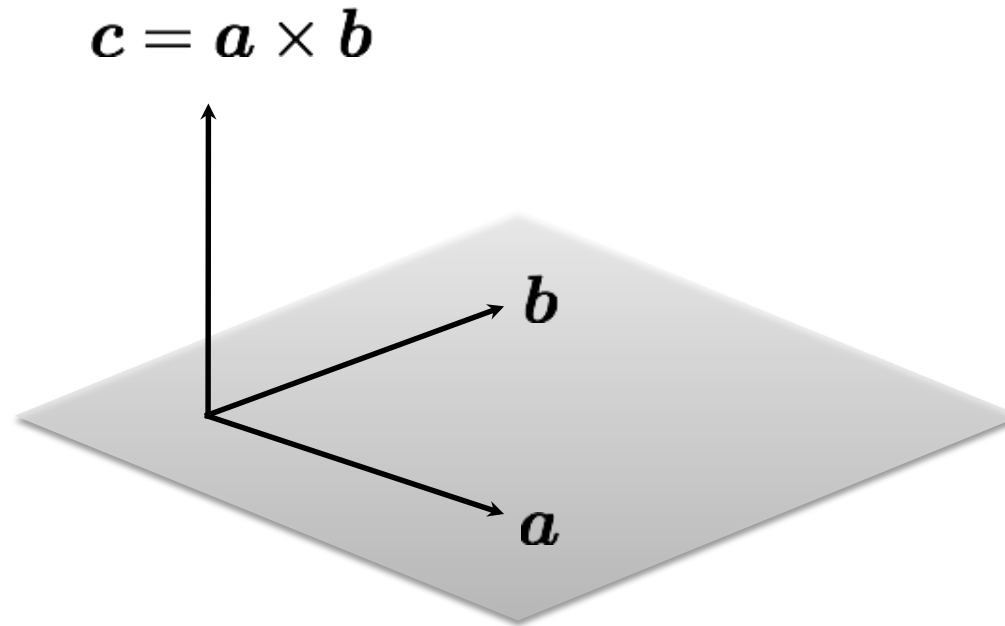
$$c \cdot b = 0$$

dot product of two orthogonal vectors is zero

# Recall: Cross Product

## Vector (cross) product

takes two vectors and returns a vector perpendicular to both



$$c \cdot a = 0$$

$$c \cdot b = 0$$

# Recall: Cross Product

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

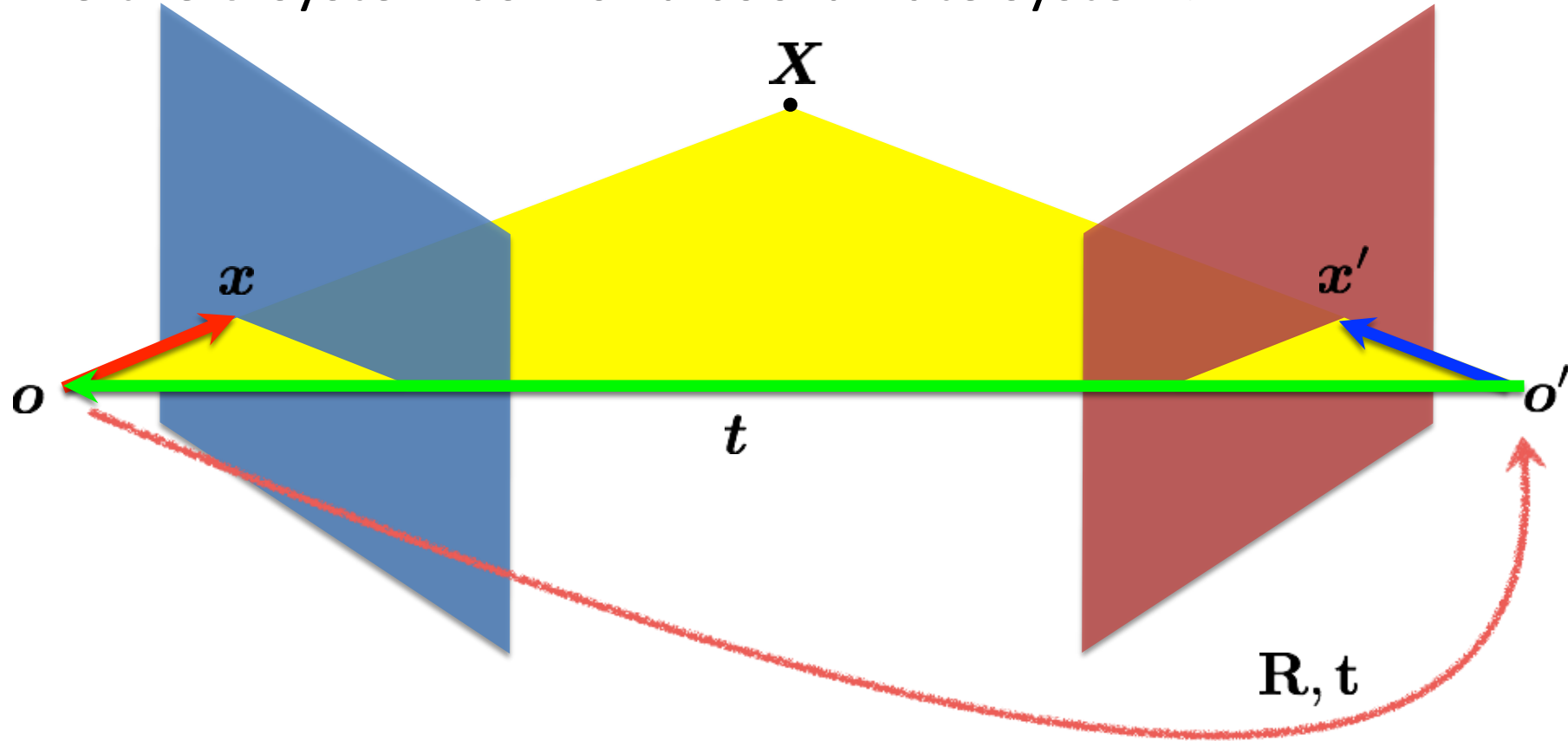
Can also be written as a matrix multiplication

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

**Skew symmetric**

# Building the essential matrix

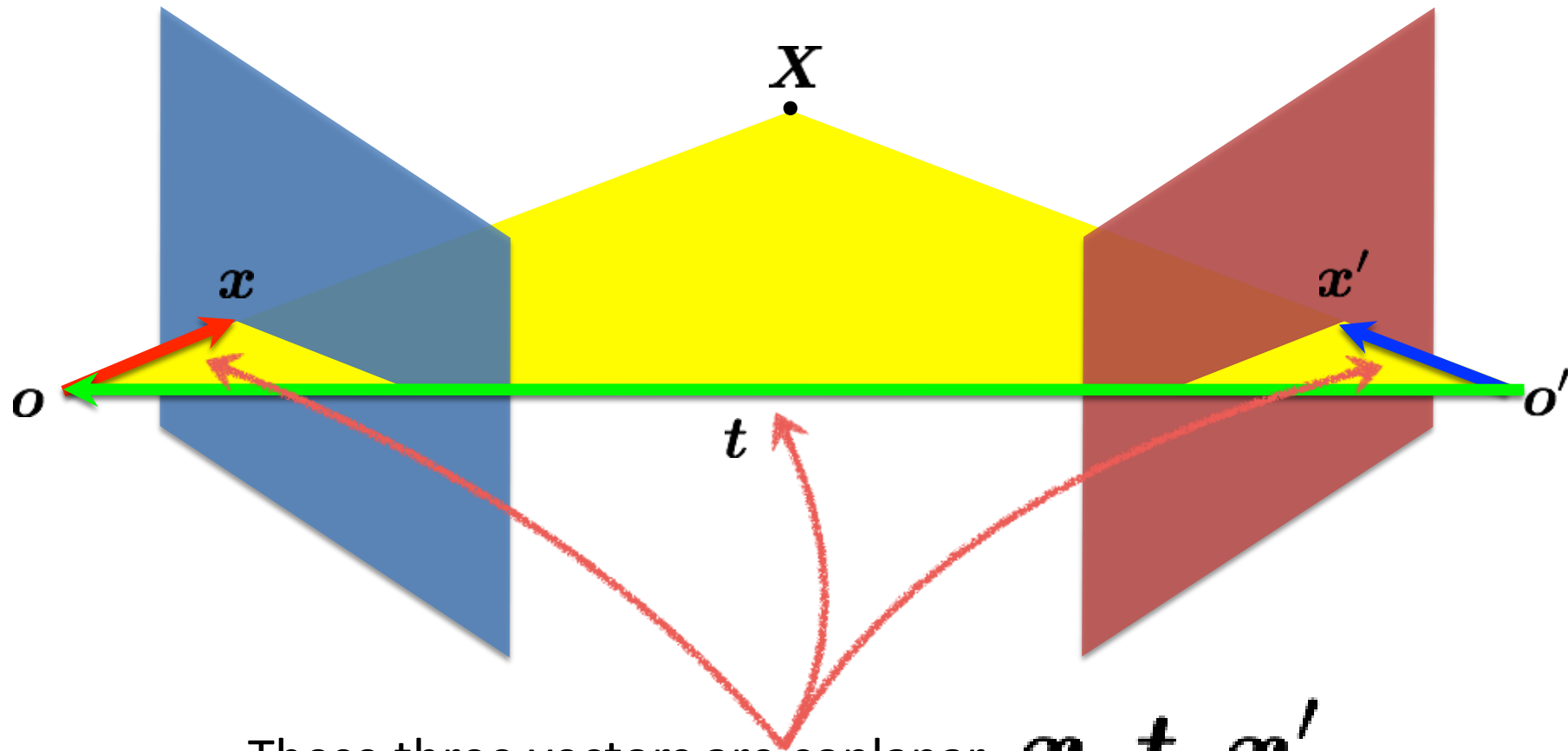
- Let's define the  $\mathbf{o}$  system as world coordinate system.



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

\* $t$  here is  $c$  from camera calibration class

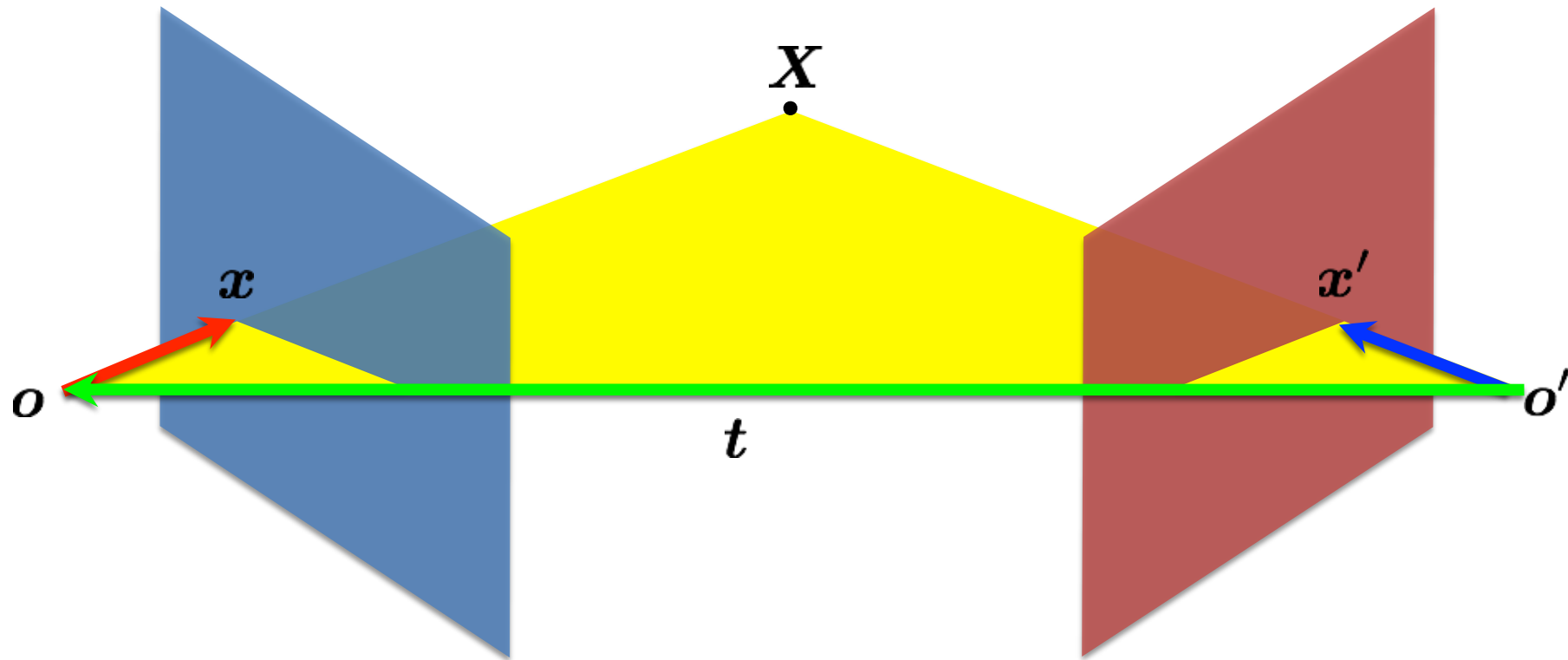
# Building the essential matrix



These three vectors are coplanar  $x, t, x'$



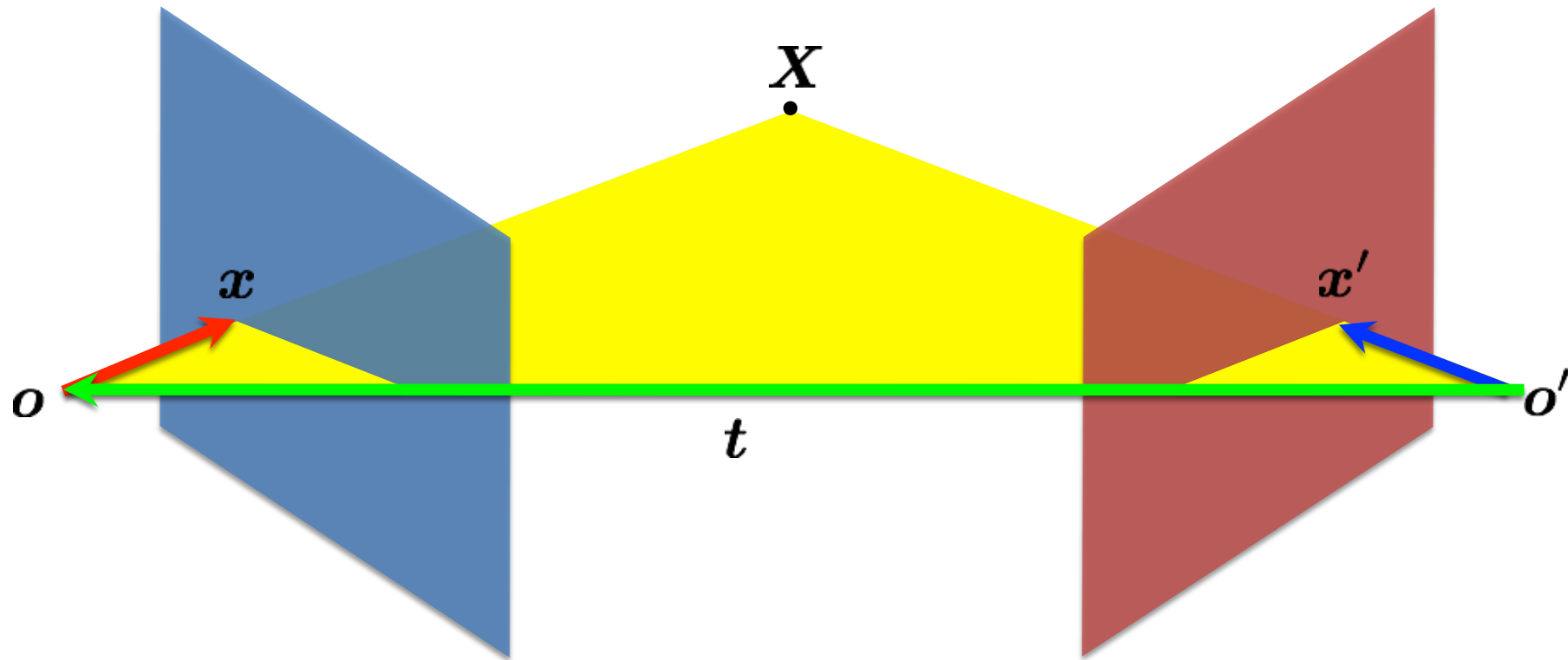
# Building the essential matrix



These three vectors are coplanar  $x, t, x'$

$$x^\top (t \times x) = ?$$

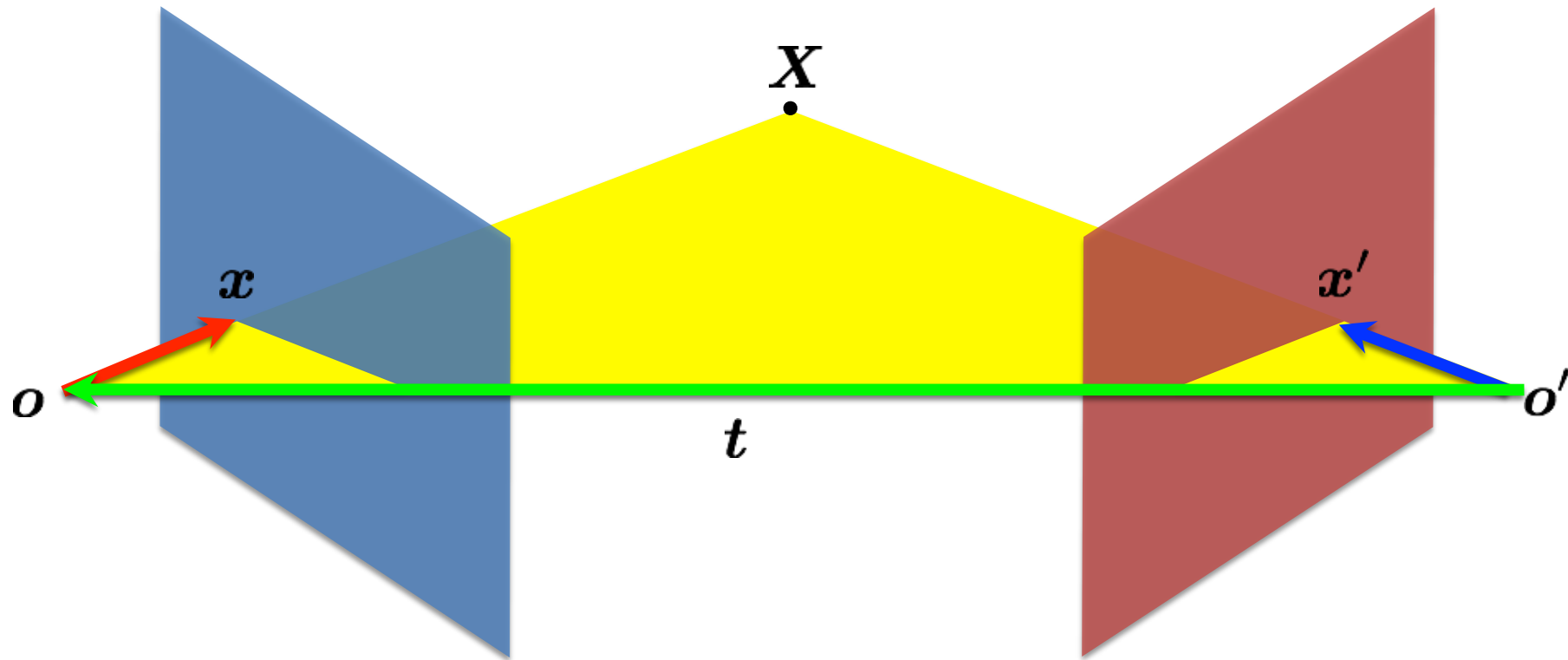
# Building the essential matrix



These three vectors are coplanar  $x, t, x'$

$$x^\top (t \times x) = 0$$

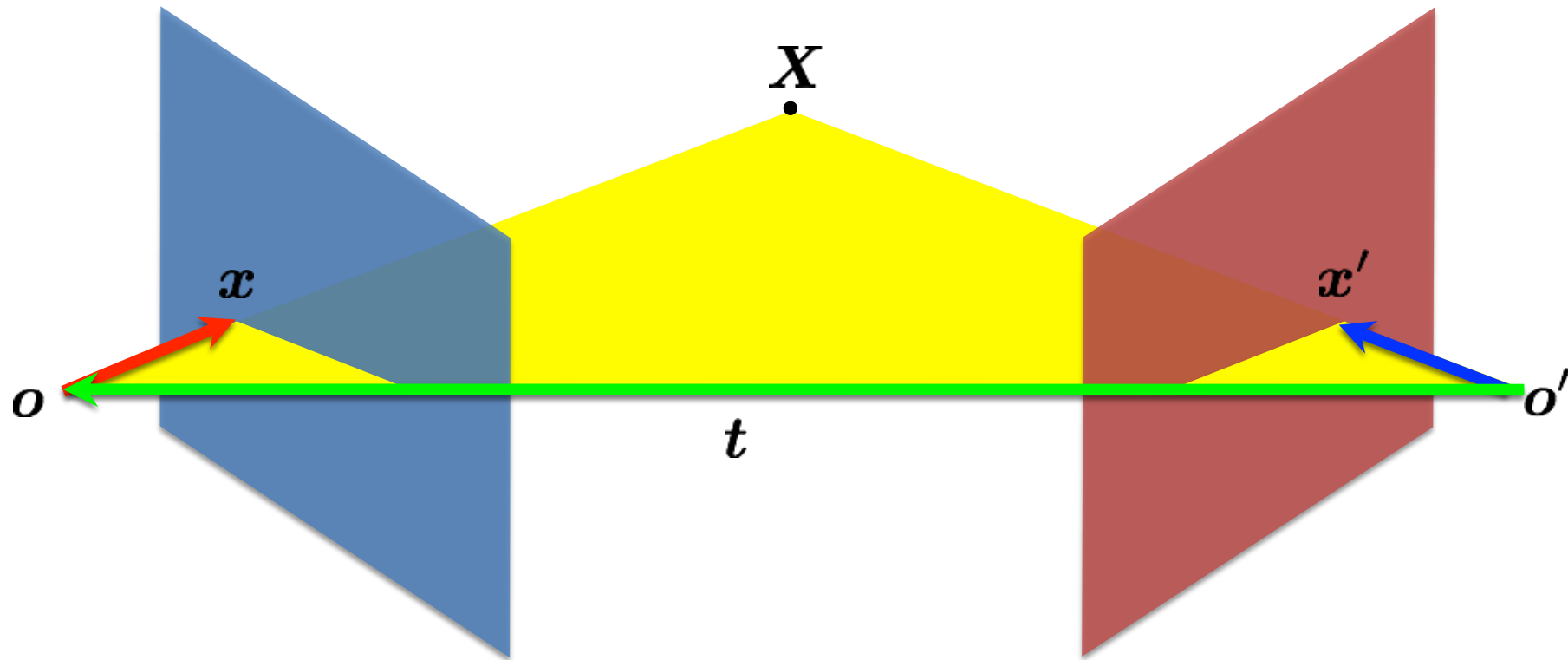
# Building the essential matrix



These three vectors are coplanar  $x, t, x'$

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

# Building the essential matrix



These three vectors are coplanar  $x, t, x'$

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

# Building the essential matrix

rigid motion

coplanarity

$$\begin{aligned} R^T &= R^{-1} \begin{cases} \mathbf{x}' = \mathbf{R}(\mathbf{x} - \mathbf{t}) \\ R^T \mathbf{x}' = \mathbf{x} - \mathbf{t} \\ \mathbf{x}'^T \mathbf{R} = (\mathbf{x} - \mathbf{t})^T \end{cases} \end{aligned}$$

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

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

# Building the essential matrix

rigid motion

coplanarity

$$\begin{aligned} R^T &= R^{-1} \begin{cases} \mathbf{x}' = \mathbf{R}(\mathbf{x} - \mathbf{t}) \\ \mathbf{R}^T \mathbf{x}' = \mathbf{x} - \mathbf{t} \\ \mathbf{x}'^T \mathbf{R} = (\mathbf{x} - \mathbf{t})^T \end{cases} \end{aligned}$$

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

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

$$(\mathbf{x}'^T \mathbf{R})([\mathbf{t}]_x \mathbf{x}) = 0$$

# Building the essential matrix

rigid motion

coplanarity

$$\begin{aligned} R^T &= R^{-1} \begin{cases} \mathbf{x}' = \mathbf{R}(\mathbf{x} - \mathbf{t}) \\ R^T \mathbf{x}' = \mathbf{x} - \mathbf{t} \\ \mathbf{x}'^T R = (\mathbf{x} - \mathbf{t})^T \end{cases} \end{aligned}$$

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

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

$$(\mathbf{x}'^T R)([\mathbf{t}]_x \mathbf{x}) = 0$$

$$\mathbf{x}'^T (R[\mathbf{t}]_x) \mathbf{x} = 0$$



# Building the essential matrix

rigid motion

coplanarity

$$R^T = R^{-1} \begin{cases} \mathbf{x}' = \mathbf{R}(\mathbf{x} - \mathbf{t}) \\ \mathbf{R}^T \mathbf{x}' = \mathbf{x} - \mathbf{t} \\ \mathbf{x}'^T \mathbf{R} = (\mathbf{x} - \mathbf{t})^T \end{cases}$$

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

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

$$(\mathbf{x}'^T \mathbf{R})([\mathbf{t}]_x \mathbf{x}) = 0$$

$$\mathbf{x}'^T (\mathbf{R}[\mathbf{t}]_x) \mathbf{x} = 0$$

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

# Building the essential matrix

rigid motion

coplanarity

$$\begin{aligned} R^T &= R^{-1} \begin{cases} \mathbf{x}' = \mathbf{R}(\mathbf{x} - \mathbf{t}) \\ R^T \mathbf{x}' = \mathbf{x} - \mathbf{t} \\ \mathbf{x}'^T R = (\mathbf{x} - \mathbf{t})^T \end{cases} \end{aligned}$$

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

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

$$(\mathbf{x}'^T R)([\mathbf{t}]_x \mathbf{x}) = 0$$

$$\mathbf{x}'^T (R[\mathbf{t}]_x) \mathbf{x} = 0$$

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

**Essential Matrix**

$$E = R[\mathbf{t}]_x$$

# properties of the E matrix

(points in normalized image coordinates)

Essential matrix definition  $x'^T E x = 0$

# Epipolar lines equation

- Recall this version of line equation:

$$ax + by + c = 0$$

For homogenous vector  $x = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$  and  $l = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ :

$$x^T l = l^T x = 0$$

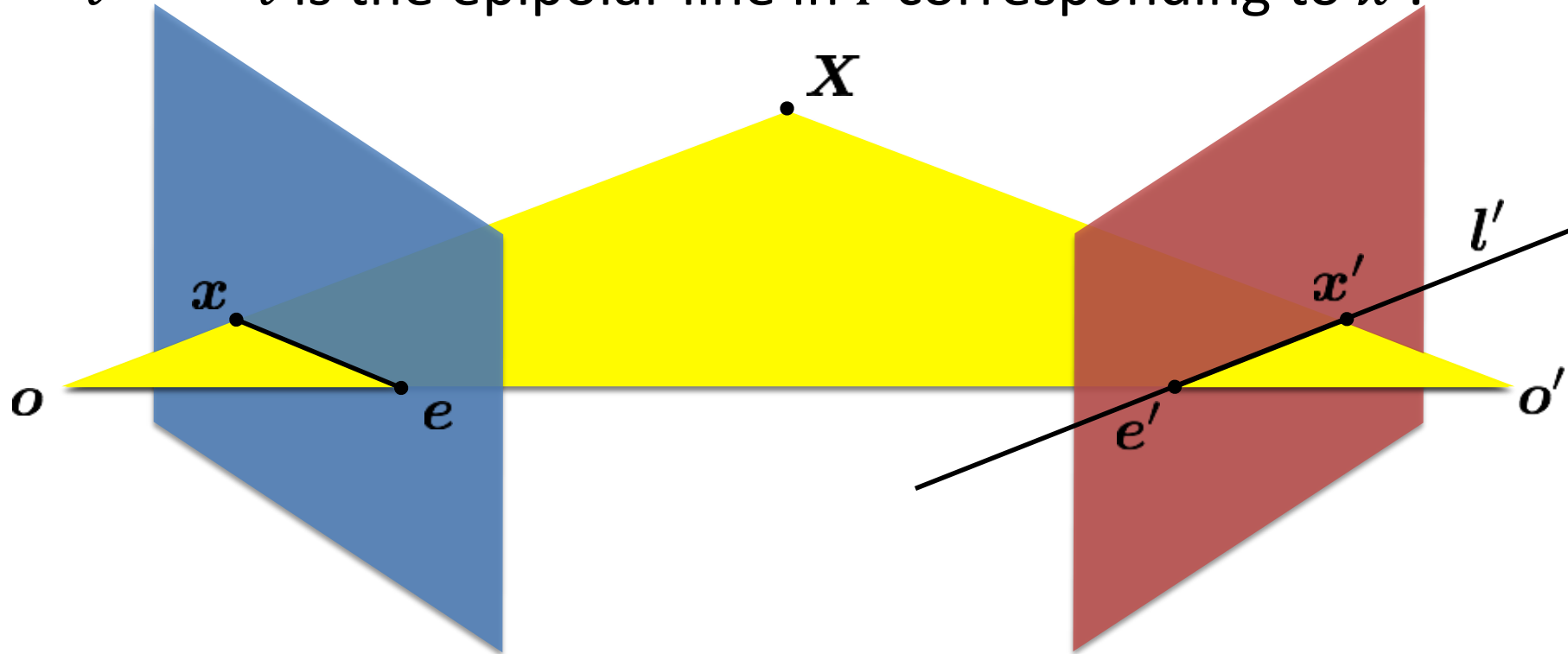
# Epipolar lines equation

- $x'^T E x = x'^T l' = 0$

$E x = l'$  is the epipolar line in  $I'$  corresponding to  $x$ .

- $x'^T E x = l^T x = 0$

$(x'^T E)^T = l^{TT} = l$  is the epipolar line in  $I$  corresponding to  $x'$ .



# properties of the E matrix

(points in normalized image coordinates)

Essential matrix definition

$$x'^T E x = 0$$

Epipolar lines

- $E x = l'$  is the epipolar line in  $I'$  corresponding to  $x$
- $(x'^T E)^T = l'^T = l$  is the epipolar line in  $I$  corresponding to  $x'$ .

- Full proof for the epipolar lines can be found here:

[http://www.cs.cornell.edu/courses/cs4670/2015sp/lectures/lec21\\_stereo\\_web.pdf](http://www.cs.cornell.edu/courses/cs4670/2015sp/lectures/lec21_stereo_web.pdf)

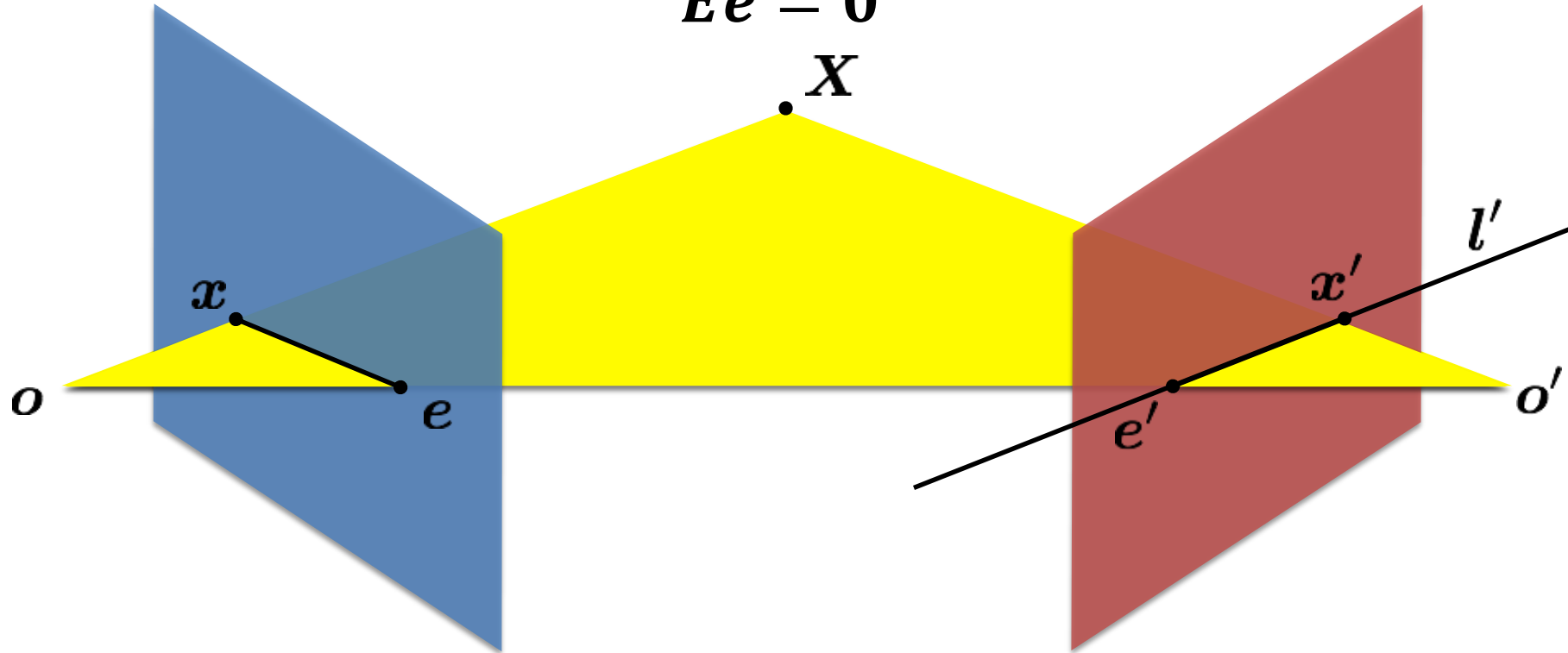
# Epipolar lines equation

- $e'^T l' = 0 \quad \forall x \text{ s.t. } Ex = l'$ , meaning:

$$e'^T E = 0$$

- $l^T e = 0 \quad \forall x' \text{ s.t. } x'^T E = l^T$  meaning:

$$Ee = 0$$



# properties of the E matrix

(points in normalized image coordinates)

Essential matrix definition  $x'^T E x = 0$

Epipolar lines

- $E x = l'$  is the epipolar line in  $I'$  corresponding to  $x$
- $(x'^T E)^T = l'^T = l$  is the epipolar line in  $I$  corresponding to  $x'$ .

Epipoles  $e'^T E = 0$   $E e = 0$



# Contents

- structure from motion
- Epipolar geometry
  - Essential matrix
  - **Fundamental matrix**
  - Estimating the fundamental matrix
- Camera rectification
- Triangulation
- Stereo matching
- Other 3D sensors

# Fundamental matrix

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

The Essential matrix operates on image points expressed in

**normalized coordinates**

(points have been aligned (normalized) to camera coordinates)

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

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

camera point

image point

# Fundamental matrix

$$\hat{x}'^T \mathbf{E} \hat{x} = 0$$

The Essential matrix operates on image points expressed in **normalized coordinates**  
(points have been aligned (normalized) to camera coordinates)

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

camera point                      image point

Writing out the epipolar constraint in terms of image coordinates

$$x'^T \mathbf{K}'^{-T} \mathbf{E} \mathbf{K}^{-1} x = 0$$

$$x'^T (\mathbf{K}'^{-T} \mathbf{E} \mathbf{K}^{-1}) x = 0$$

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

**Essential  
Matrix**

$$\mathbf{F} = \mathbf{K}'^{-T} \mathbf{E} \mathbf{K}^{-1}$$

# properties of the **F** matrix

- All of the below functions works with **F** and un-normalized points the same!

(points in un-normalized image coordinates)

Fundamental matrix definition  $x'^T \mathbf{F} x = 0$

Epipolar lines

- $\mathbf{F}x = l'$  is the epipolar line in  $I'$  corresponding to  $x$
- $(x'^T \mathbf{F})^T = l'^T = l$  is the epipolar line in  $I$  corresponding to  $x'$ .

Epipoles  $e'^T \mathbf{F} = 0$   $\mathbf{F}e = 0$

# Contents

- structure from motion
- Epipolar geometry
  - Essential matrix
  - Fundamental matrix
  - **Estimating the fundamental matrix**
- Camera rectification
- Triangulation
- Stereo matching
- Other 3D sensors

# Estimating $F$

- If we don't know  $K_1$ ,  $K_2$ ,  $R$ , or  $t$ , can we estimate  $F$  for two images?
- Yes, given enough correspondences



# Estimating F – 8-point algorithm

- The fundamental matrix F is defined by

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

for any pair of matches  $\mathbf{x}$  and  $\mathbf{x}'$  in two images.

- Let  $\mathbf{x}=(u,v,1)^T$  and  $\mathbf{x}'=(u',v',1)^T$ ,  
each match gives a linear equation

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

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

# 8-point algorithm

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

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



## 8-point algorithm – Problem?

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

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

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

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

# 8-point algorithm

- Pros: it is linear, easy to implement and fast
- Cons: susceptible to noise.
  - Solutions: (all out of scope)
    - normalized 8 points algorithm.
    - 7 points algorithm.
    - Finding  $K, K'$  with single camera intrinsics calibration and then search for  $E$  (only 5 DOFs instead of 8/7).

# Contents

- structure from motion
- Epipolar geometry
  - Essential matrix
  - Fundamental matrix
  - Estimating the fundamental matrix
- **Camera rectification**
- Triangulation
- Stereo matching
- Other 3D sensors

# Structure and motion

	Structure (3D model of world)	Motion (6 DOFs of cameras)	Measurements
Pose Estimation (camera pose estimation)	known	<b>estimate</b>	3D to 2D correspondences
Triangulation	<b>estimate</b>	known	2D to 2D correspondences
3D reconstruction	<b>estimate</b>	<b>Estimated!</b>	2D to 2D correspondences + triangulation

We've found the motion, let's find the structure – back to triangulation problem!

# How would you reconstruct 3D points?

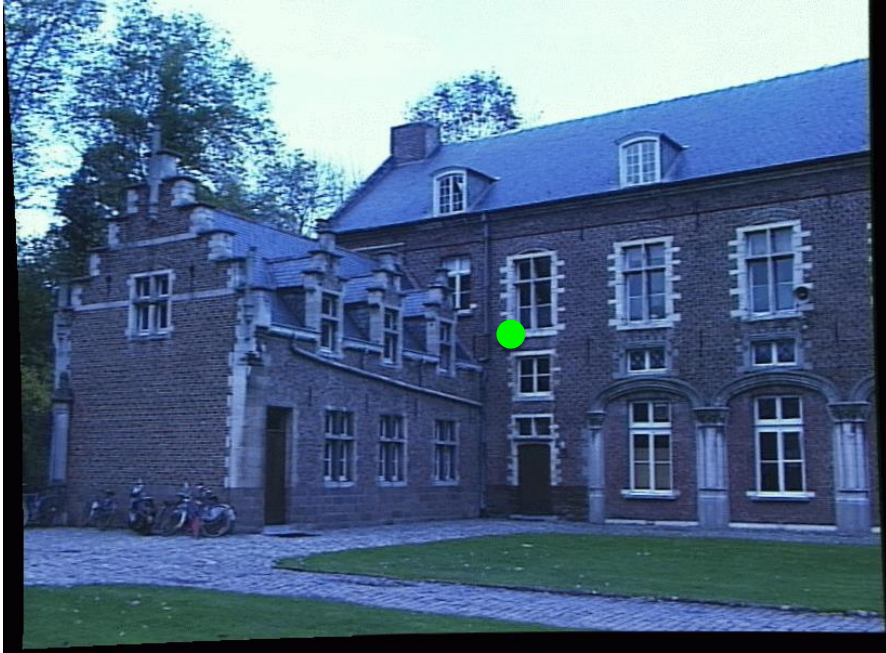


Left image



Right image

# How would you reconstruct 3D points?



Left image

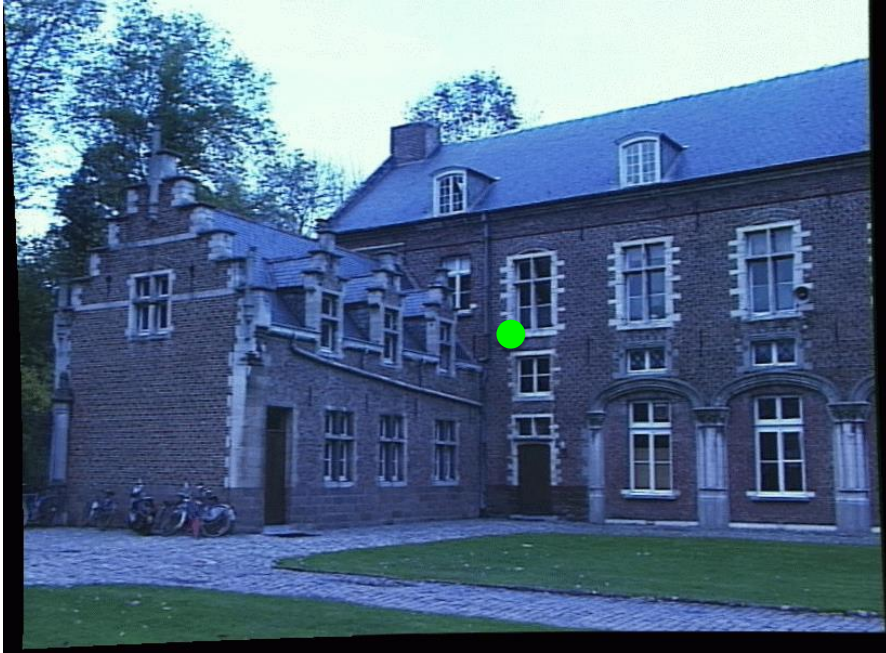


Right image

1. Select point in one image



# How would you reconstruct 3D points?



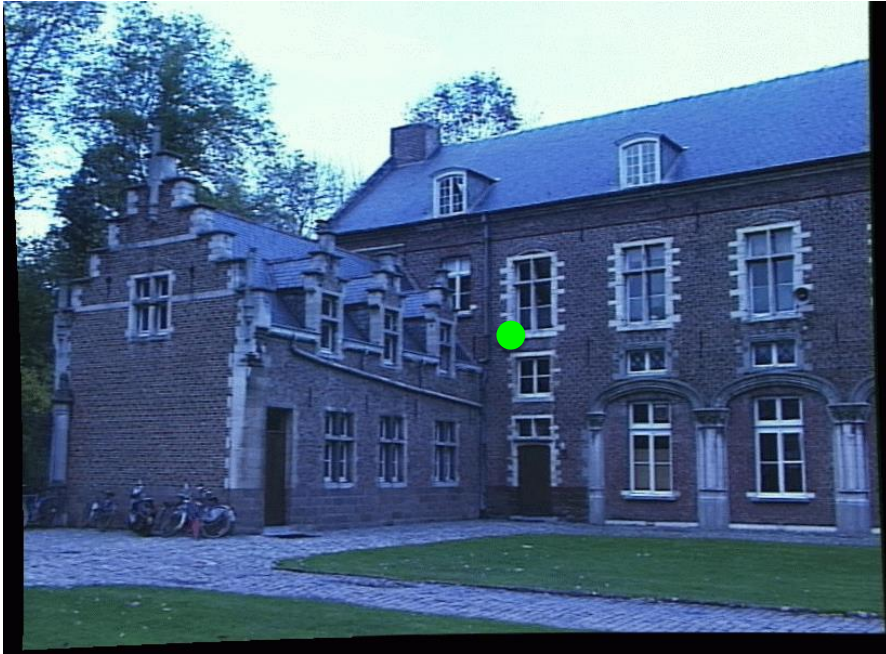
Left image



Right image

1. Select point in one image
2. Form epipolar line for that point in second image

# How would you reconstruct 3D points?



Left image

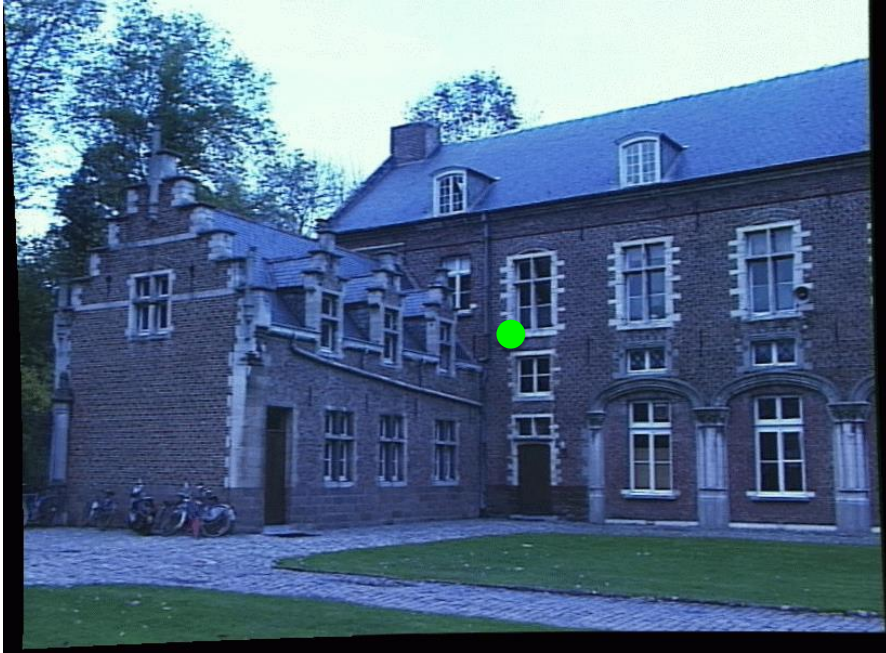


Right image

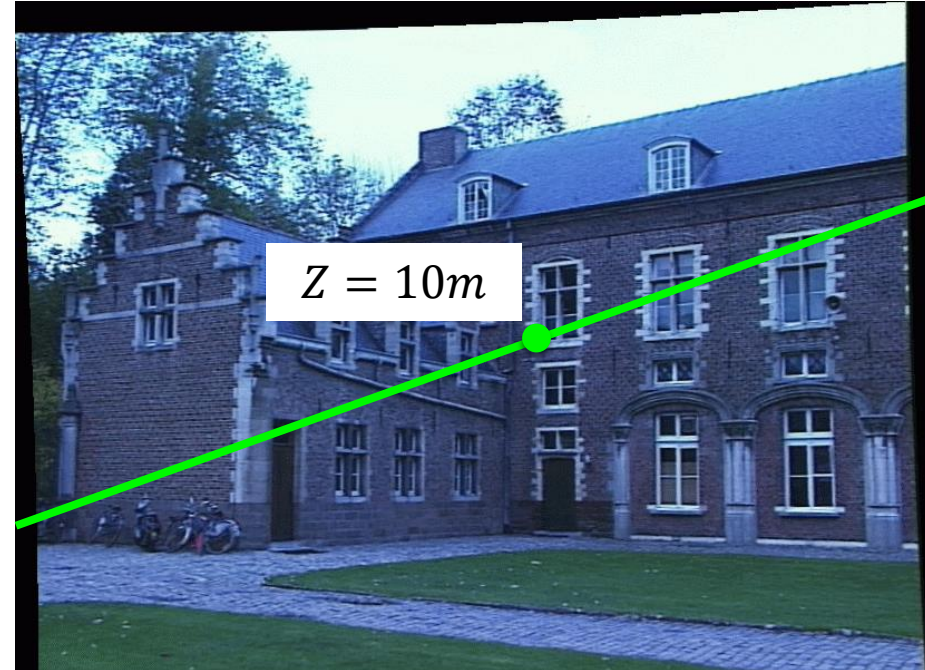
1. Select point in one image
2. Form epipolar line for that point in second image
3. Find matching point along line (how?)



# How would you reconstruct 3D points?



Left image

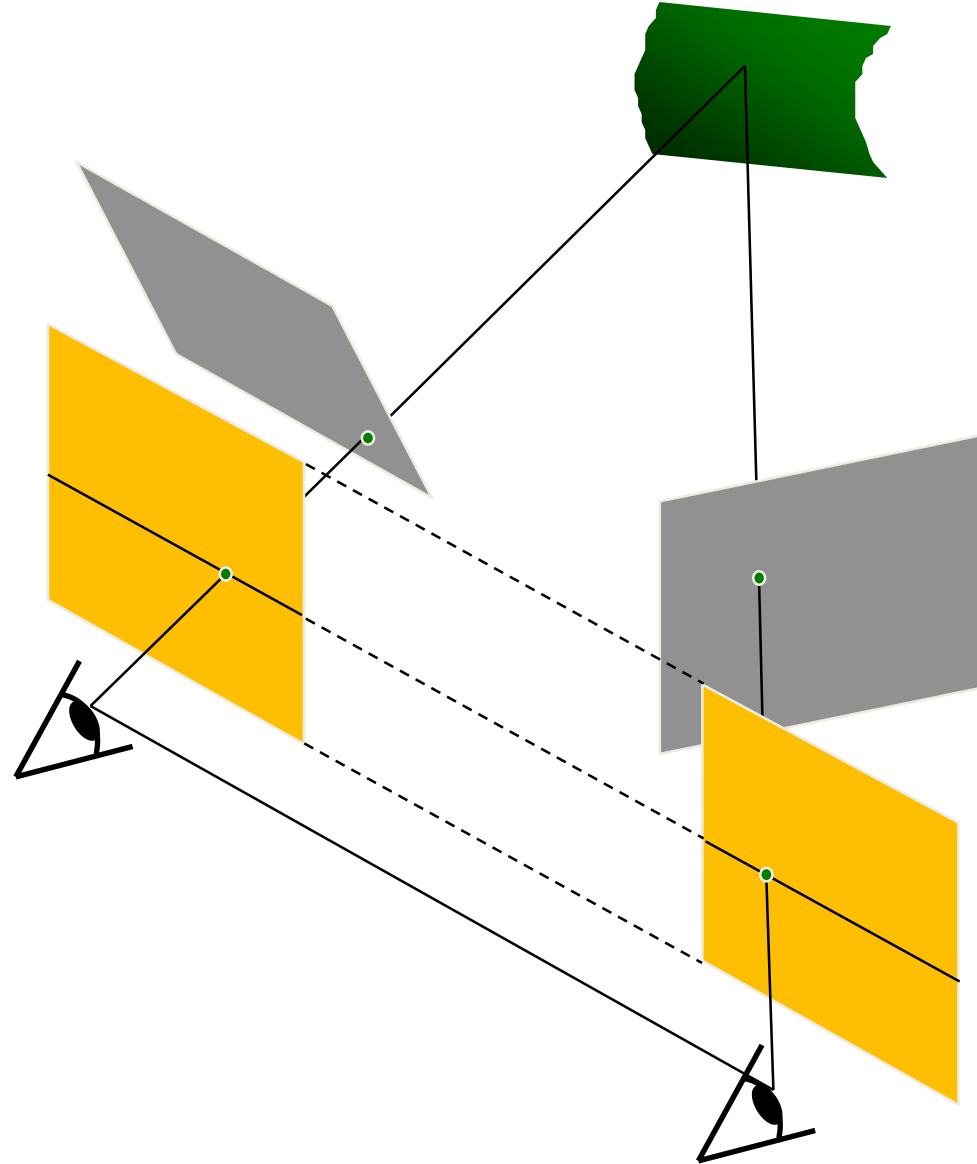


Right image

1. Select point in one image
2. Form epipolar line for that point in second image
3. Find matching point along line (how?)
4. Perform triangulation (how?)

# Stereo image rectification

- Out of scope...
- Let's say the images comes rectified (as in the yellow samples).





Original stereo pair



After rectification

# Contents

- structure from motion
- Epipolar geometry
  - Essential matrix
  - Fundamental matrix
  - Estimating the fundamental matrix
- Camera rectification
- **Triangulation**
- Stereo matching
- Other 3D sensors



# triangulation







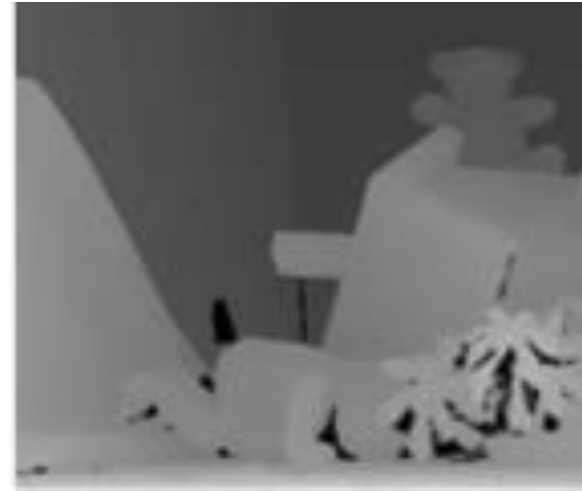
# triangulation



Objects that are close move more or less?

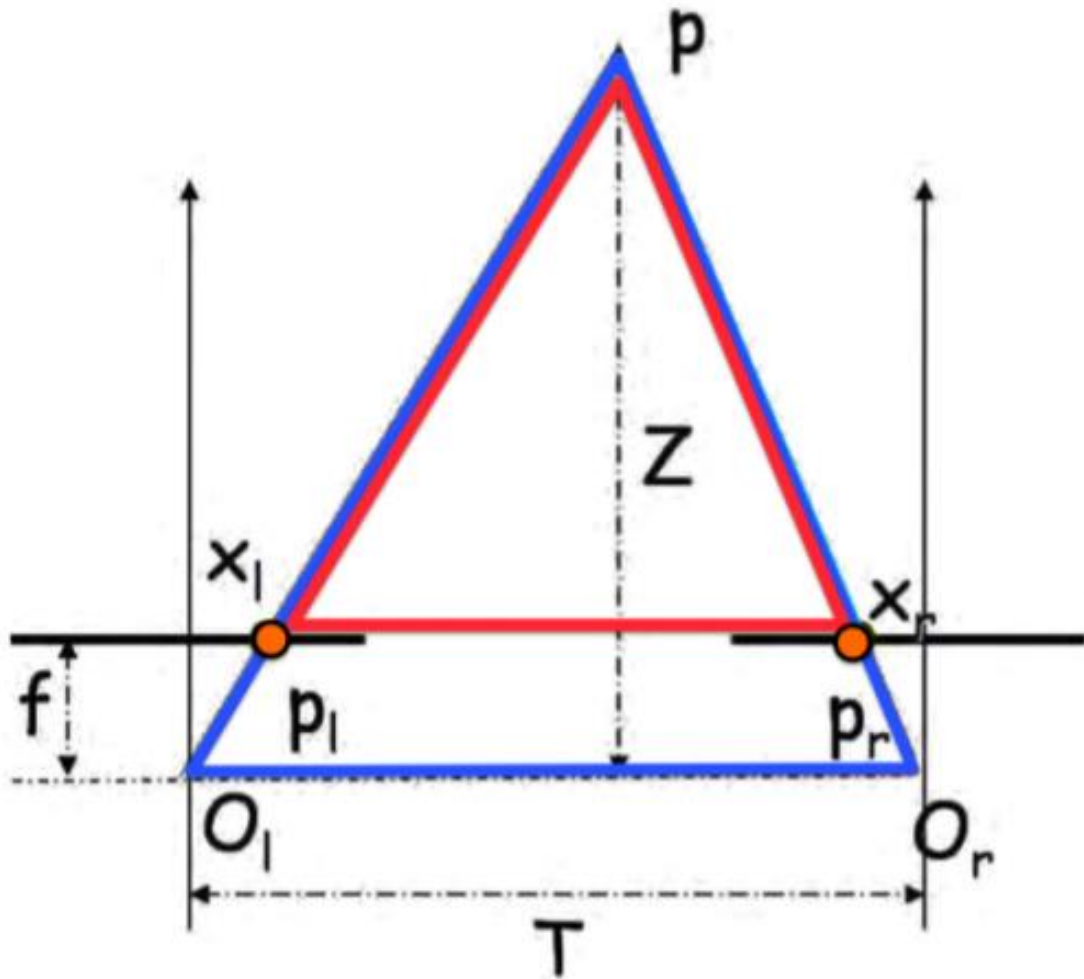


# triangulation



- The amount of horizontal movement is inversely proportional to the distance from the camera.
- The amount of horizontal movement == disparity.
- Distance from the camera == depth (or  $Z$ ).

# triangulation



Similar triangles:

$$\frac{T}{Z} = \frac{T + x_r - x_l}{Z - f}$$

The diagram shows the derivation of the triangulation formula. An orange arrow points from the similar triangles equation to a boxed equation. The boxed equation is  $Z = \frac{f \cdot T}{x_l - x_r}$ . The term  $f \cdot T$  is labeled "baseline" with an arrow. The term  $x_l - x_r$  is labeled "disparity" with an arrow. The entire equation is enclosed in an orange rounded rectangle.

$$Z = \frac{f \cdot T}{x_l - x_r}$$

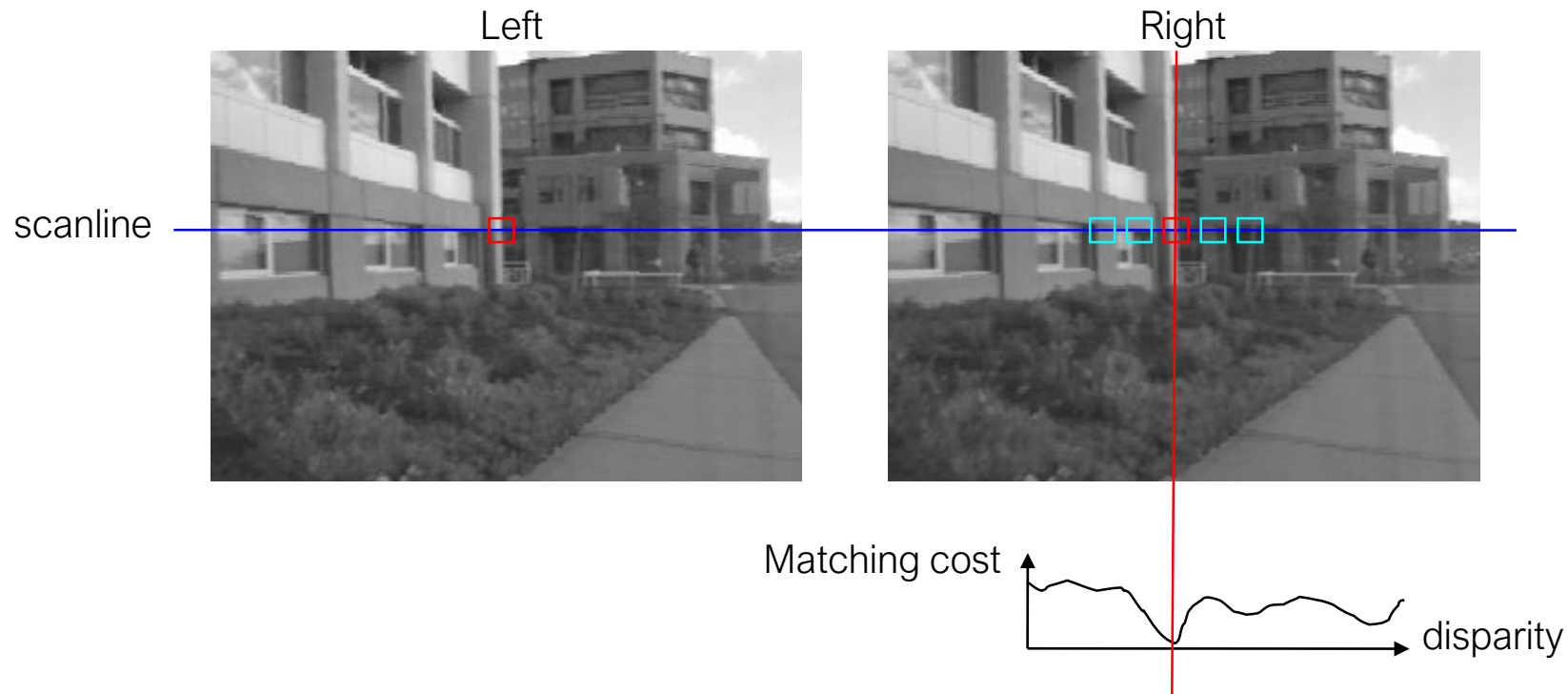
baseline

focal length      disparity

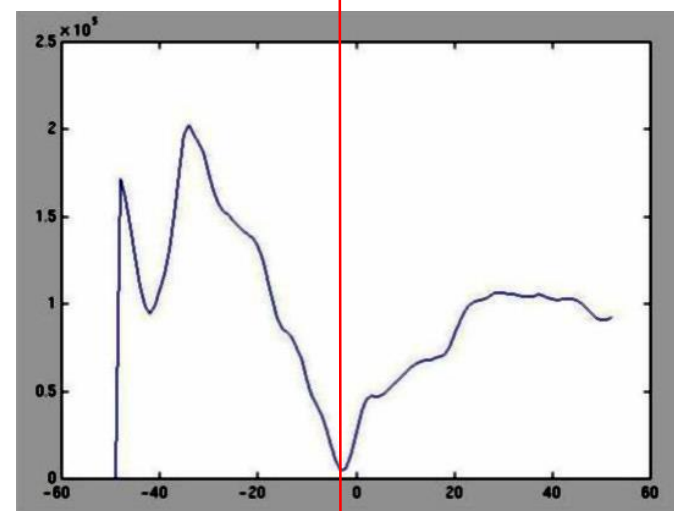
# Contents

- structure from motion
- Epipolar geometry
  - Essential matrix
  - Fundamental matrix
  - Estimating the fundamental matrix
- Camera rectification
- Triangulation
- **Stereo matching**
- Other 3D sensors

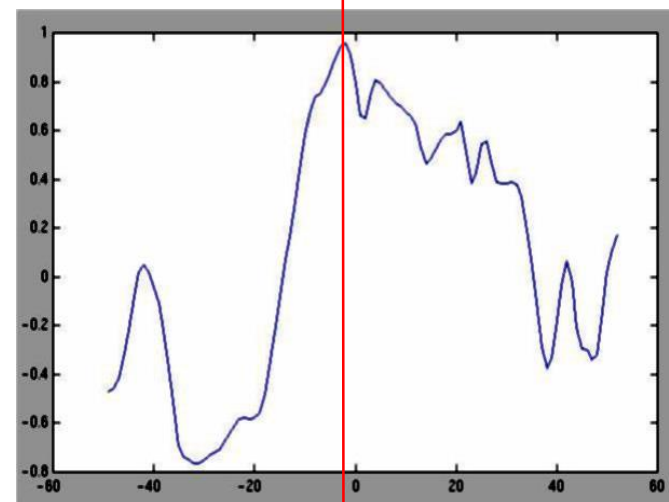
# Stereo Block Matching



- Slide a window along the epipolar line and compare contents of that window with the reference window in the left image
- Matching cost: SSD or normalized correlation



SSD

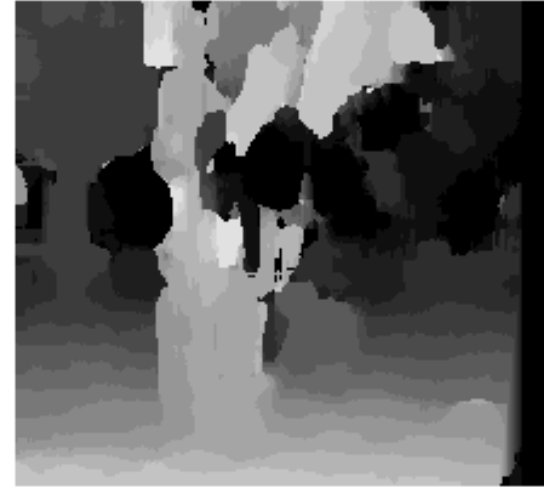


Normalized cross-correlation

# Effect of window size



$W = 3$



$W = 20$

# Effect of window size



$W = 3$

## **Smaller window**

- + More detail
- More noise



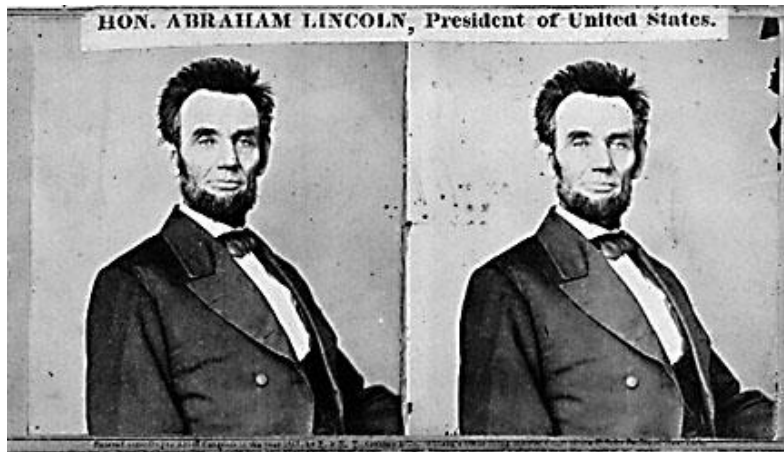
$W = 20$

## **Larger window**

- + Smoother disparity maps
- Less detail
- Fails near boundaries



## *When will stereo block matching fail?*

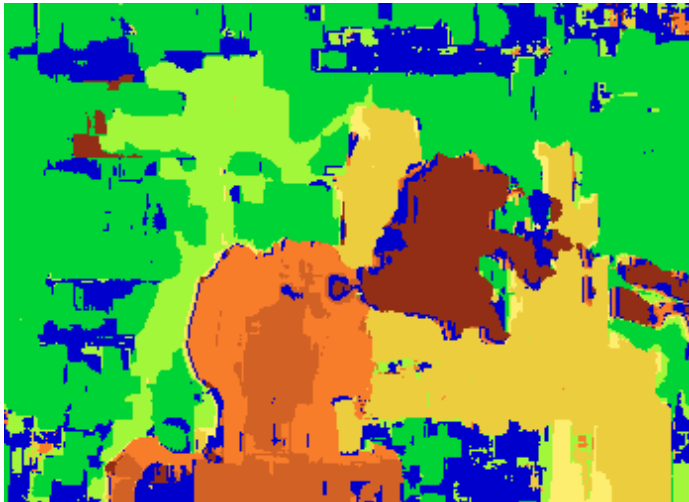


## *When will stereo block matching fail?*





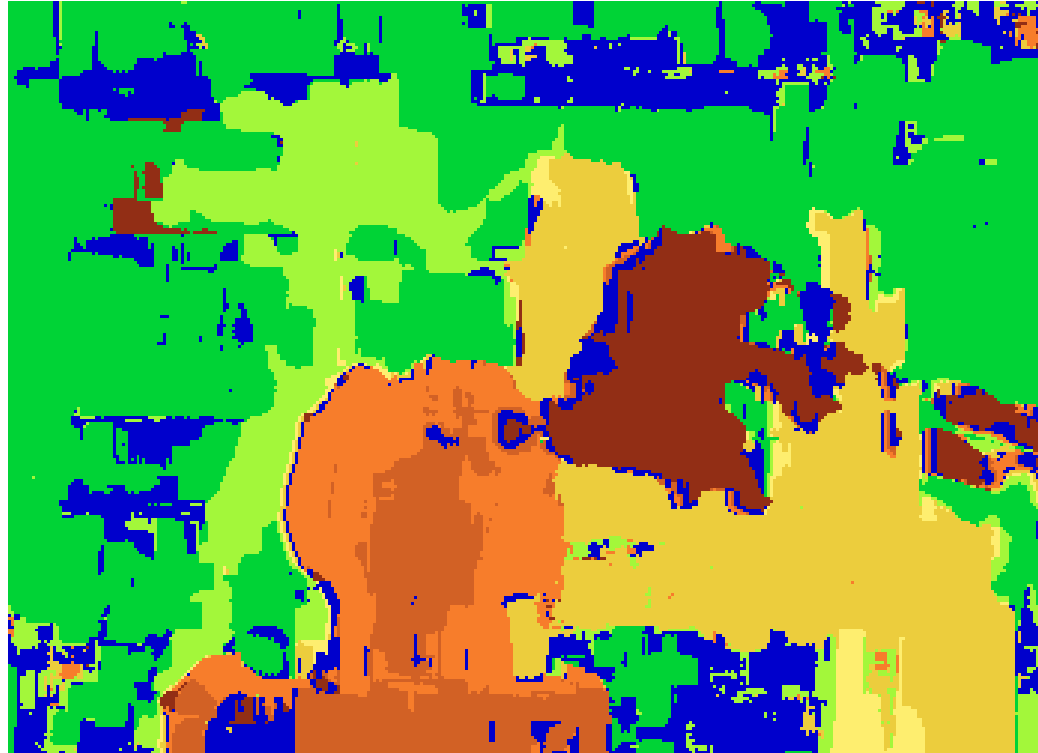
Block matching



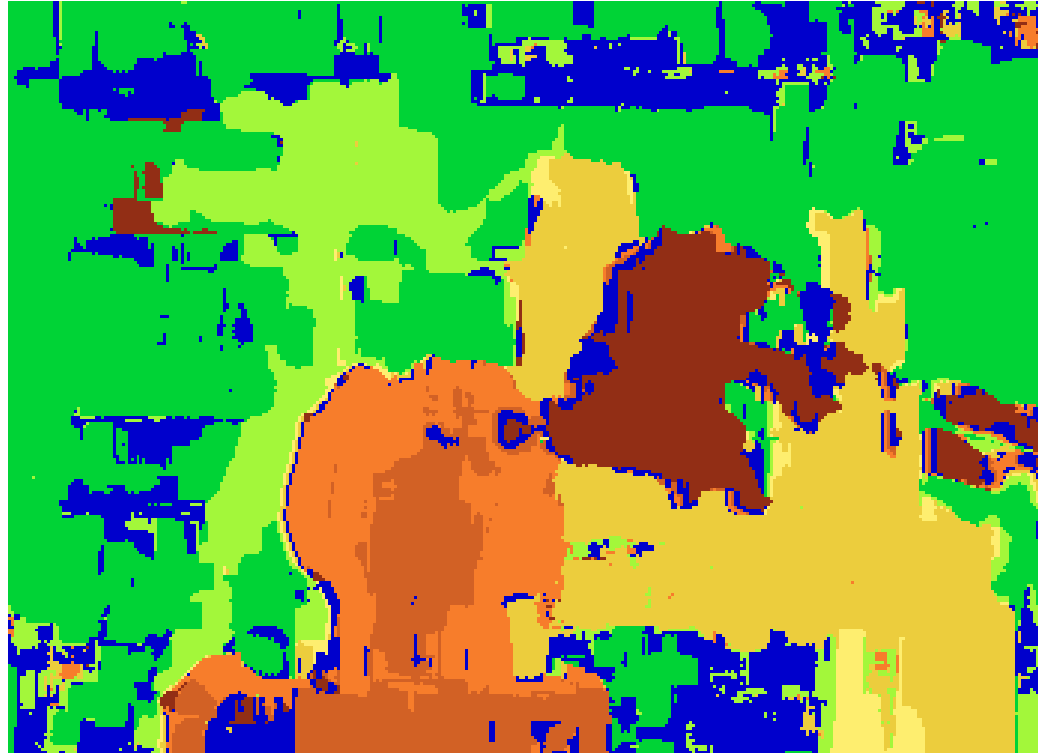
Ground truth



*What are some problems with the result?*



*How can we improve depth estimation?*

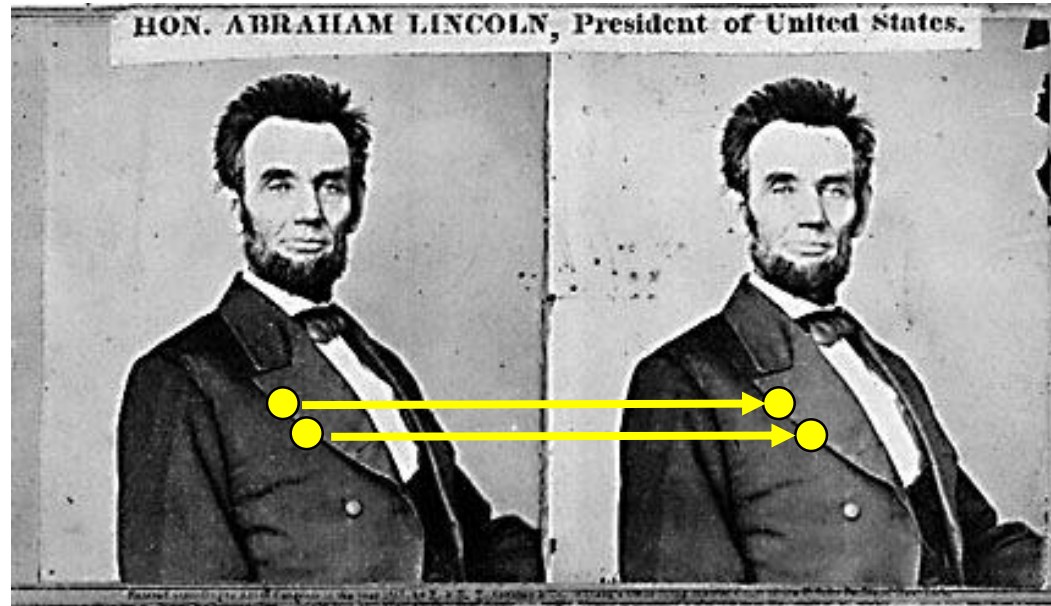


*How can we improve depth estimation?*

Too many discontinuities.  
We expect disparity values to change slowly.

Let's make an assumption:  
**depth should change smoothly**

# Energy Minimization



What defines a good stereo correspondence?

- 1. Match quality**

- Want each pixel to find a good match in the other image

- 2. Smoothness**

- If two pixels are adjacent, they should (usually) move about the same amount

energy function  
(for one pixel)

$$E(d) = \underbrace{E_d(d)}_{\text{data term}} + \lambda \underbrace{E_s(d)}_{\text{smoothness term}}$$

Want each pixel to find a good match  
in the other image  
(block matching result)

Adjacent pixels should (usually)  
move about the same amount  
(smoothness function)



$$E(d) = E_d(d) + \lambda E_s(d)$$

$$E_d(d) = \sum_{(x,y) \in I} C(x, y, d(x, y))$$

data term

SSD distance between windows  
centered at  $I(x, y)$  and  $J(x + d(x, y), y)$



$$E(d) = E_d(d) + \lambda E_s(d)$$

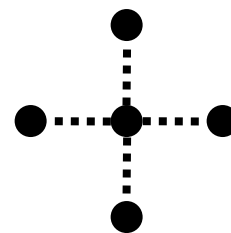
$$E_d(d) = \sum_{(x,y) \in I} C(x, y, d(x, y))$$

SSD distance between windows  
centered at  $I(x, y)$  and  $J(x + d(x, y), y)$

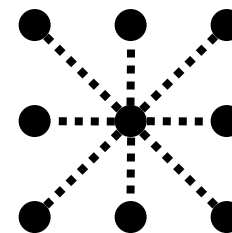
$$E_s(d) = \sum_{(p,q) \in \mathcal{E}} V(d_p, d_q)$$

smoothness term

$\mathcal{E}$  : set of neighboring pixels



4-connected  
neighborhood



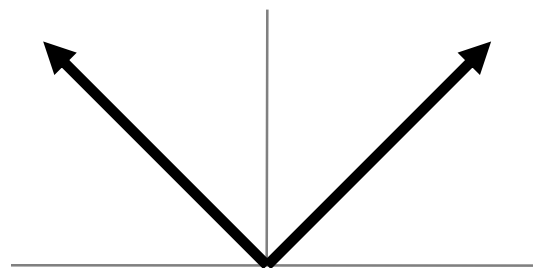
8-connected  
neighborhood

$$E_s(d) = \sum_{(p,q) \in \mathcal{E}} V(d_p, d_q)$$

smoothness term

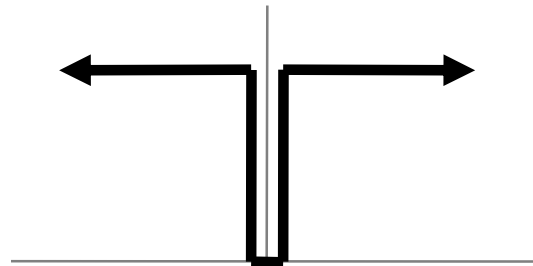
$$V(d_p, d_q) = |d_p - d_q|$$

$L_1$  distance



$$V(d_p, d_q) = \begin{cases} 0 & \text{if } d_p = d_q \\ 1 & \text{if } d_p \neq d_q \end{cases}$$

“Potts model”



# Dynamic Programming

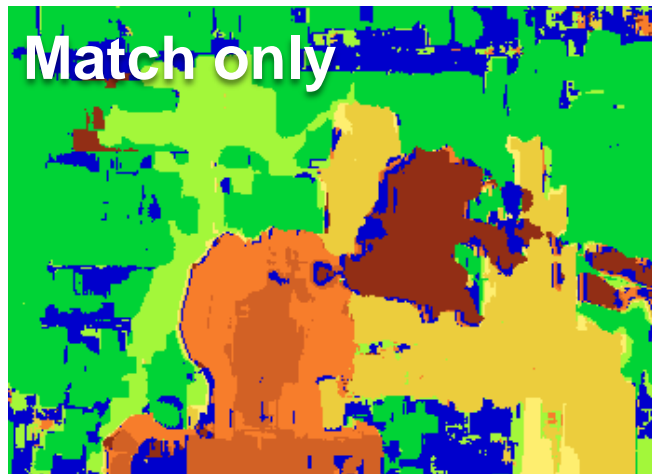
One possible solution...

$$E(d) = E_d(d) + \lambda E_s(d)$$

Can minimize this independently per scanline  
using dynamic programming (DP) ●.....●.....●

$D(x, y, d)$  : minimum cost of solution such that  $d(x,y) = d$

$$D(x, y, d) = C(x, y, d) + \min_{d'} \{D(x - 1, y, d') + \lambda |d - d'|\}$$



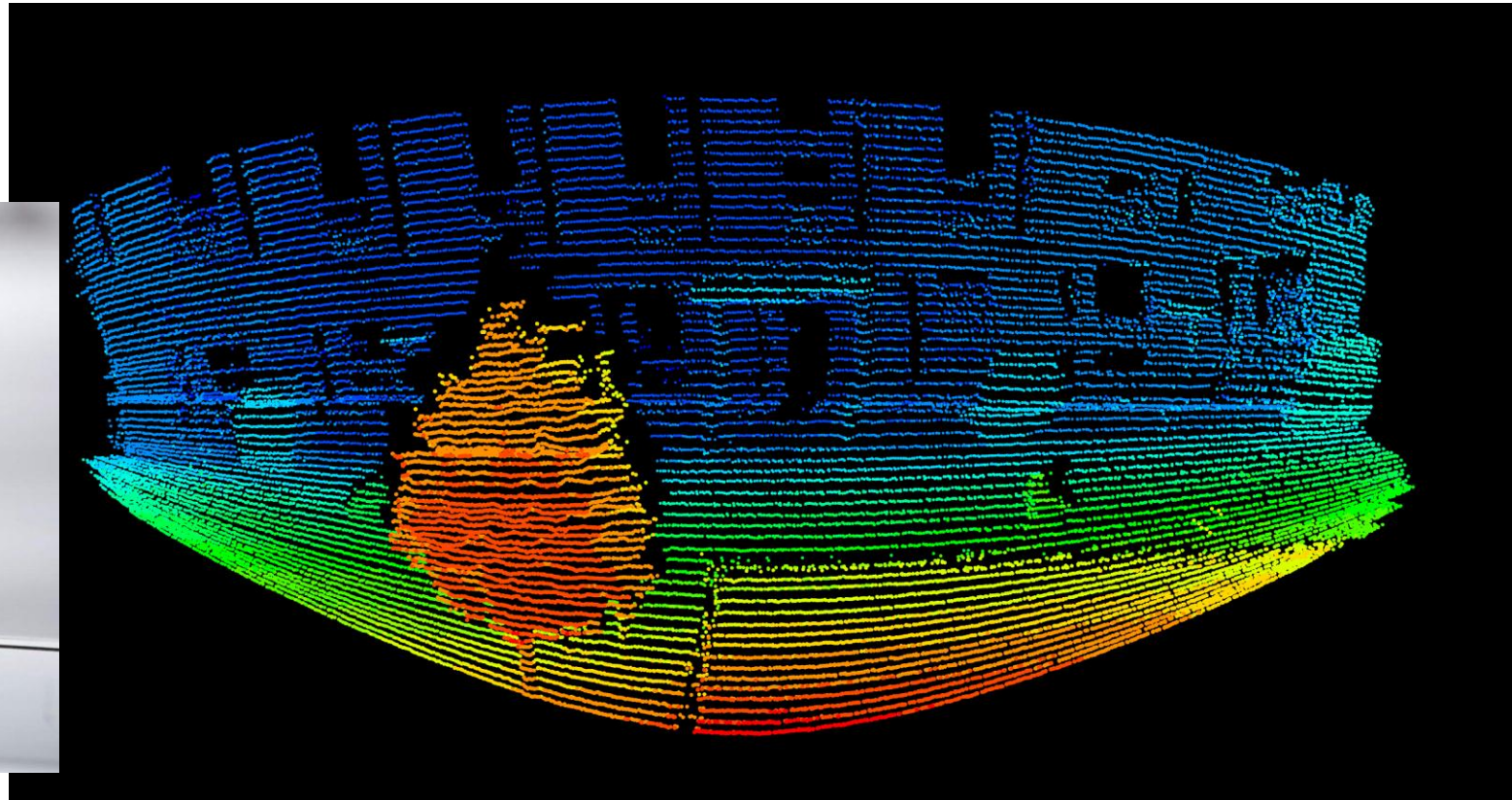
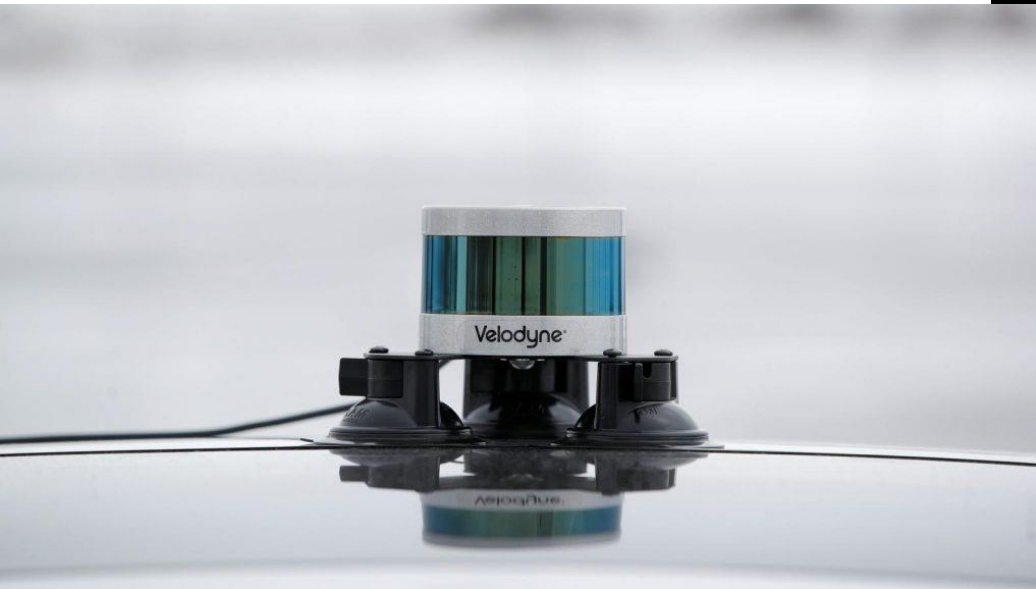
Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](#), PAMI 2001

# Contents

- structure from motion
- Epipolar geometry
  - Essential matrix
  - Fundamental matrix
  - Estimating the fundamental matrix
- Camera rectification
- Triangulation
- Stereo matching
- **Other 3D sensors**

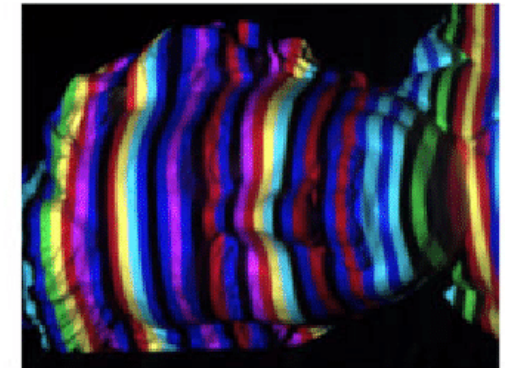
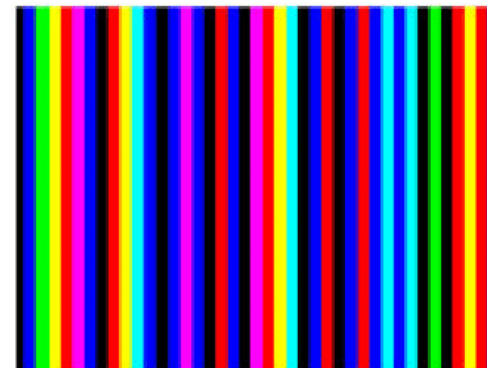
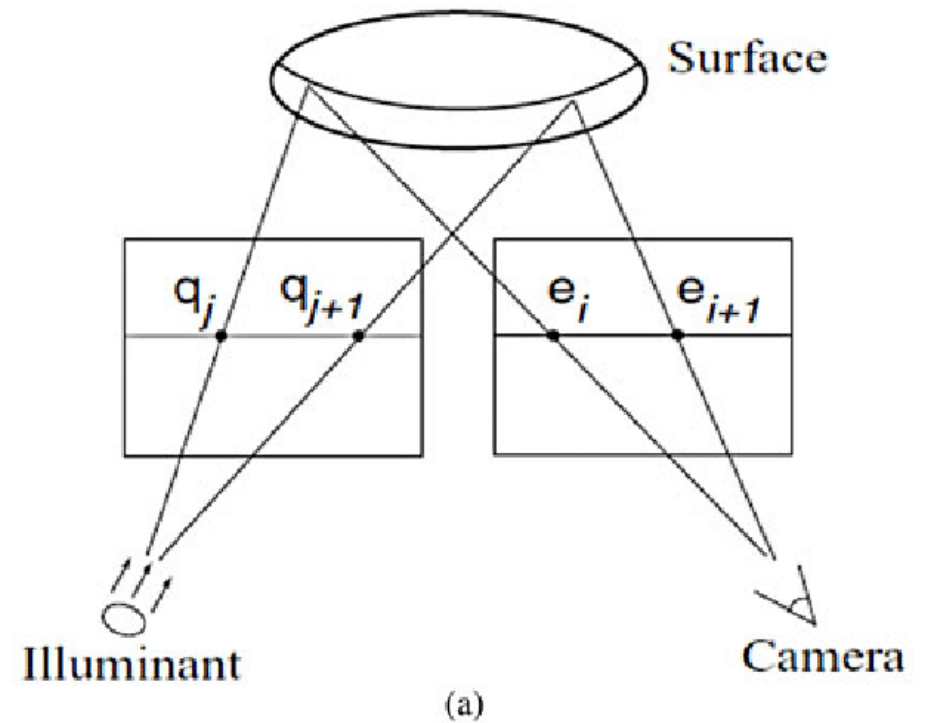
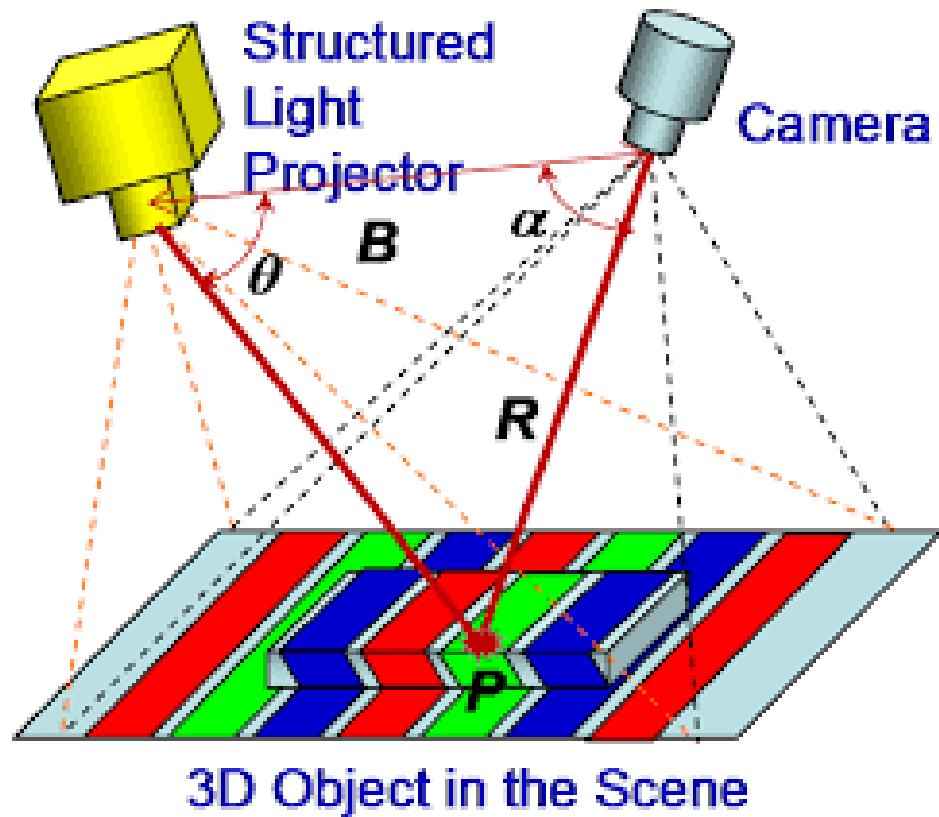
# Other types of 3D sensors

- **LIDAR**, which stands for **L**ight **D**etection and **R**anging (or light radar), is a remote sensing method that uses light in the form of a pulsed laser to measure ranges.
- Most known: velodyne projector.



# Other types of 3D sensors

- Structured light



# Other types of 3D sensors

- Coded light
- Realsense SR305
- <https://www.youtube.com/watch?v=PluL7WTIKrM>





# Other types of 3D sensors

- *Light Coding*
- *Used in Kinect v1- Kinect for xbox 360.*
- *Iphone x front camera*

