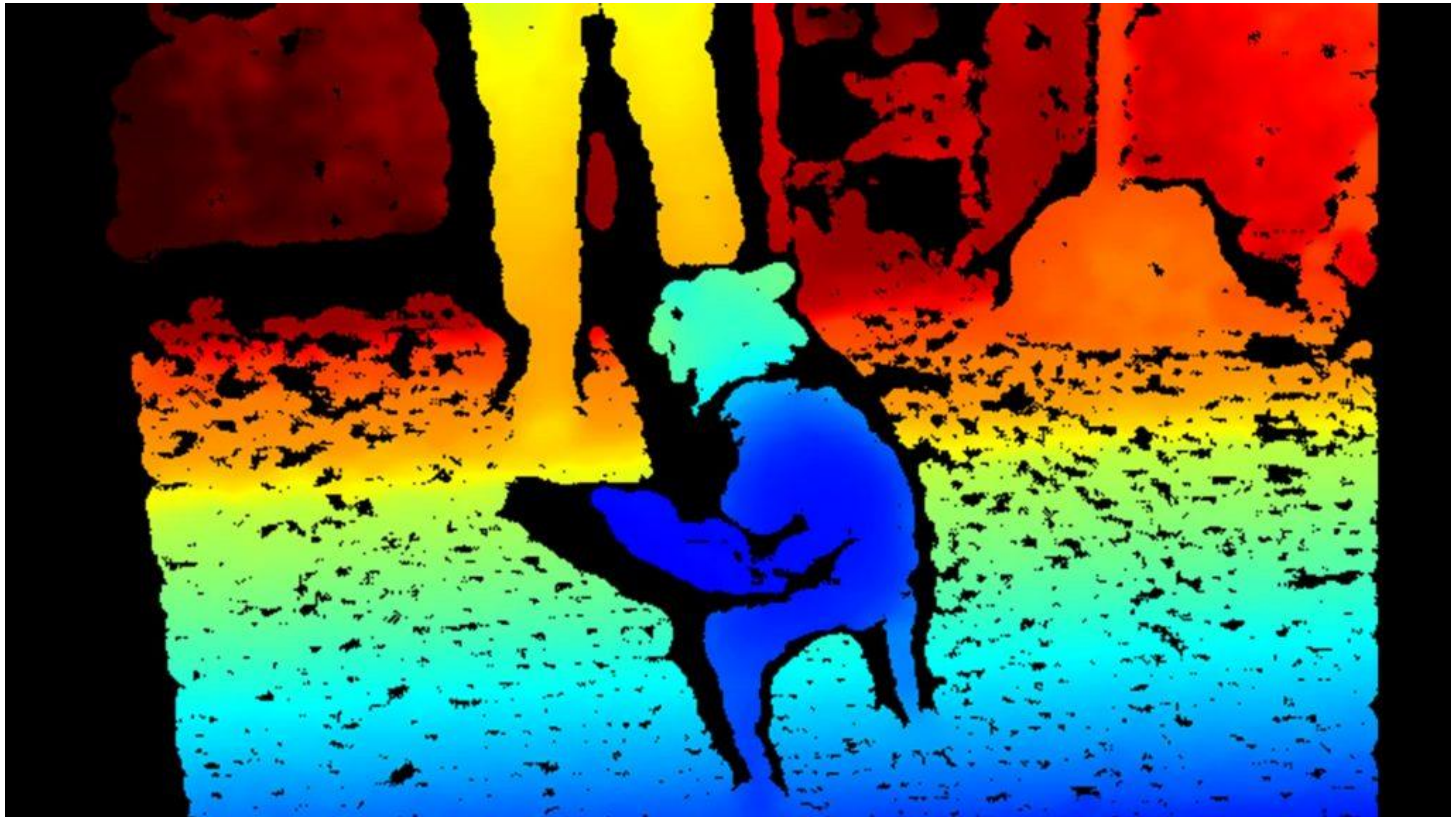
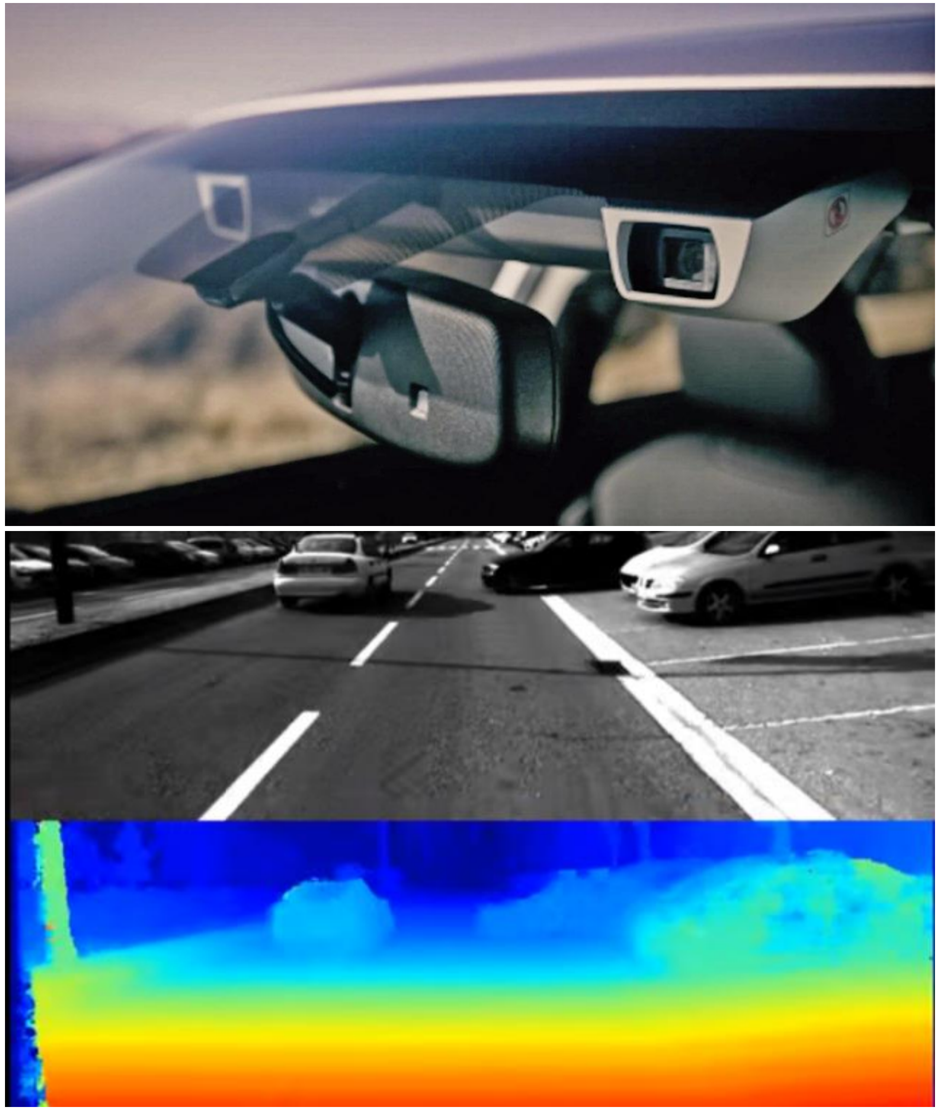
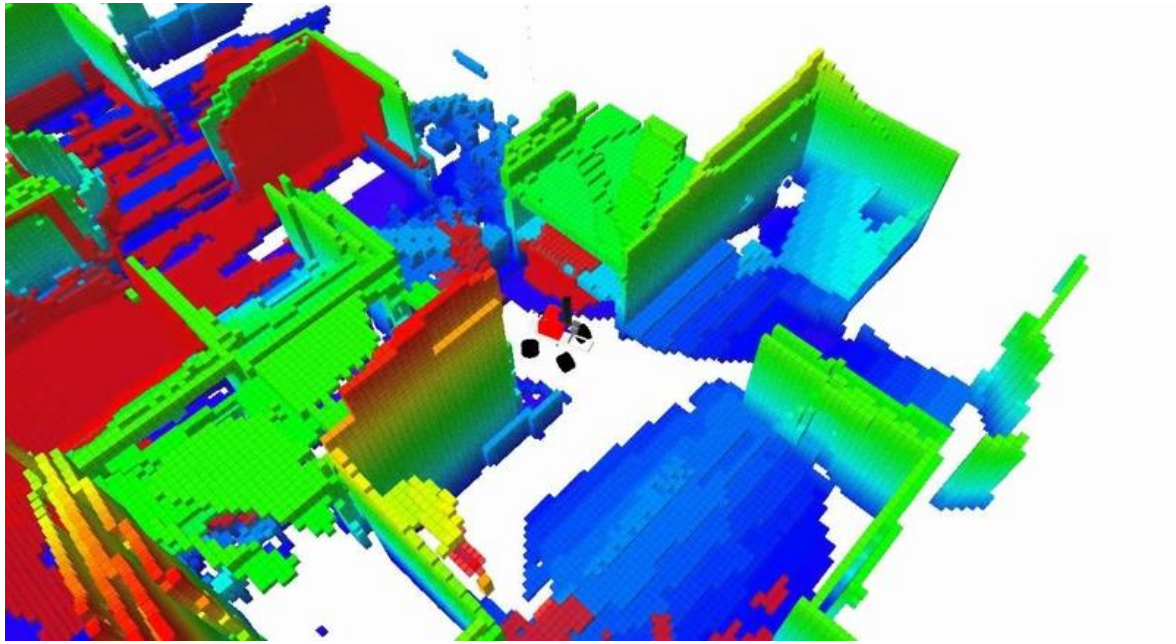
Stereo



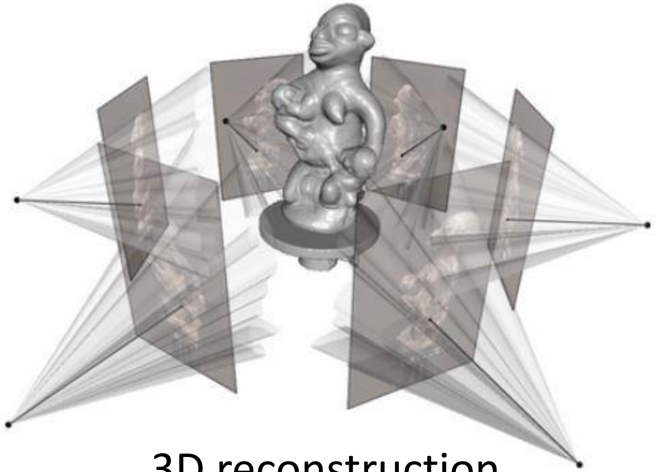
What can be done with stereo vision?



Autonomous driving



SLAM- robot navigation



3D reconstruction

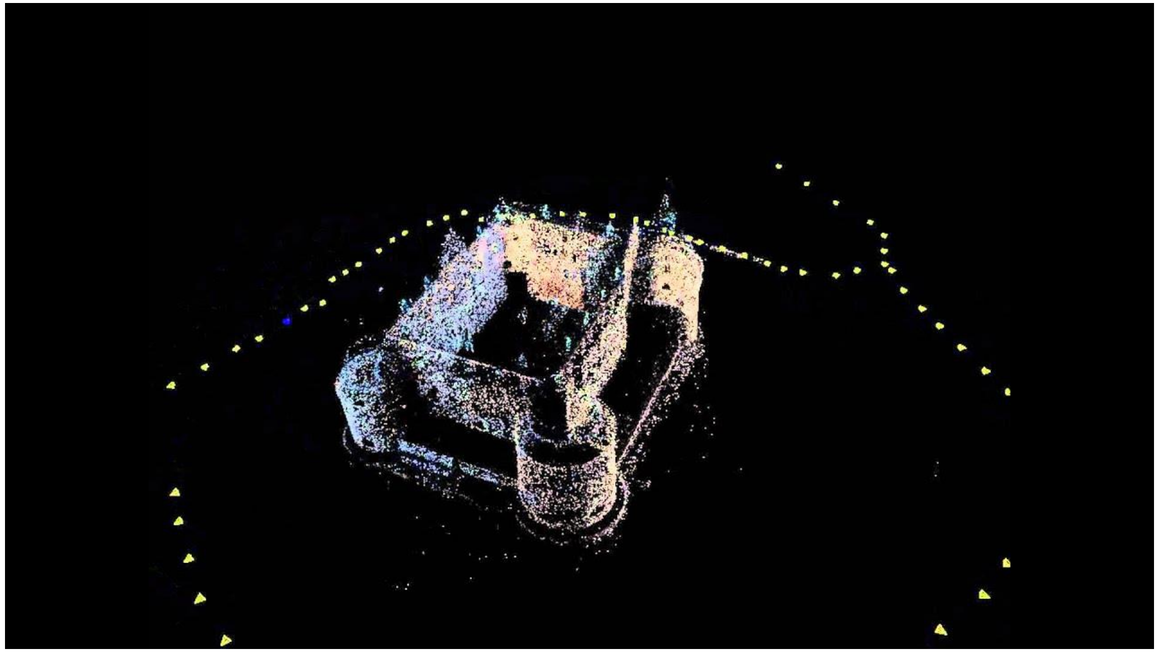
References

- 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

- **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)
Pose Estimation (camera pose estimation)	Known	**Estimate**
Triangulation	**Estimate**	Known
3D reconstruction/ SfM/ stereo vision	**Estimate**	**Estimate**

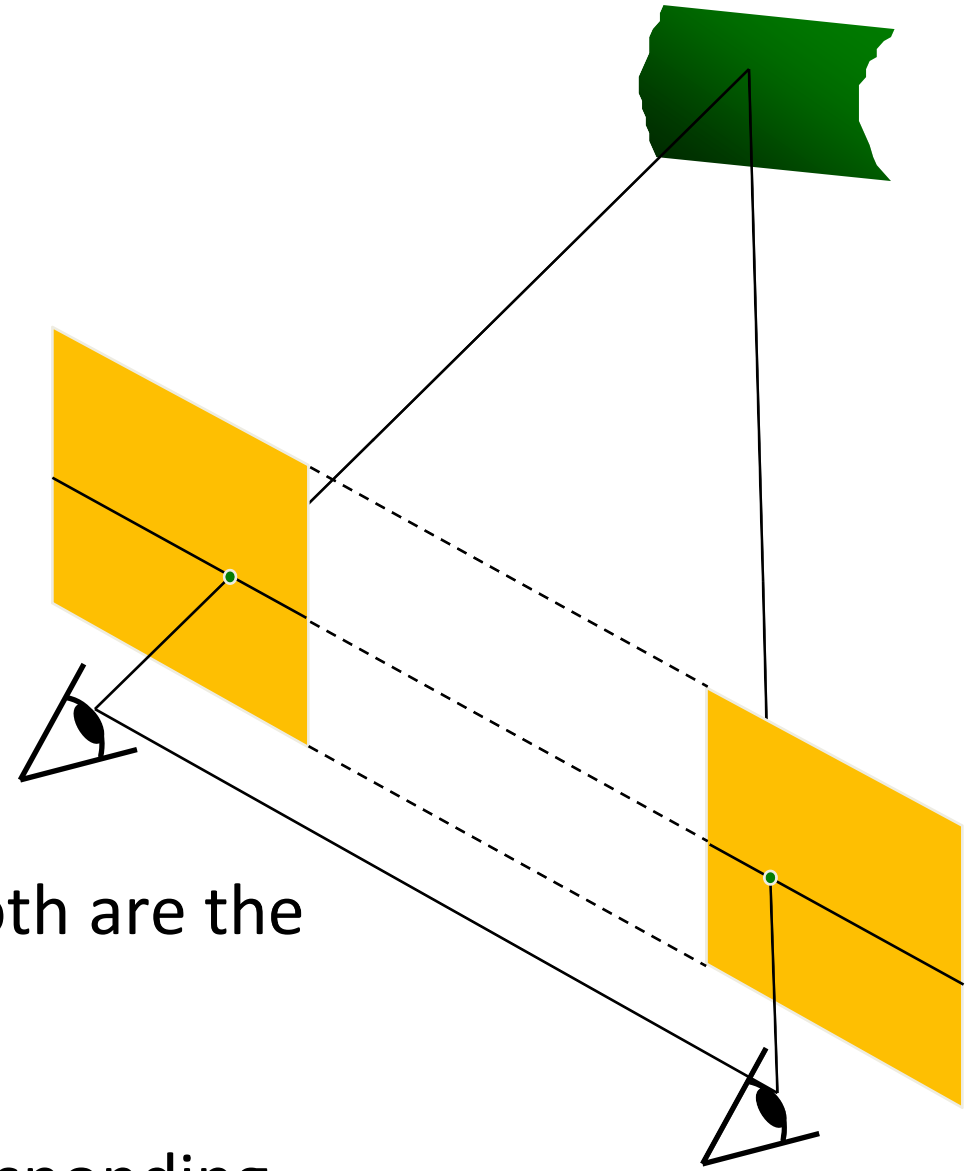
Structure and motion

- 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 will learn about 3D reconstruction from two cameras (and triangulation as a subtopic).

Contents

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

Triangulation

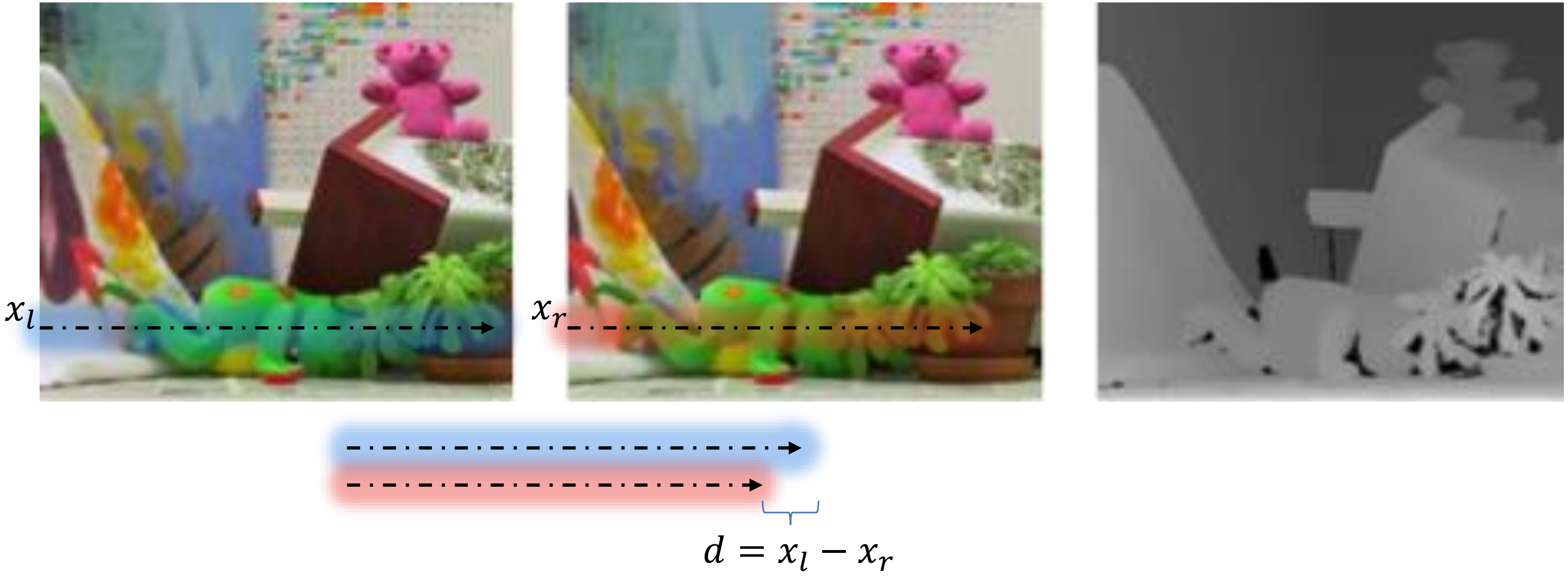


- Assume both cameras are rectified- 6 DOF of both are the same except the horizontal translation.
- Assume same focal length f in both cameras
- Assume we know for each pixel in left the corresponding pixel in right.
- From this we want to get a depth image using triangulation.

Left

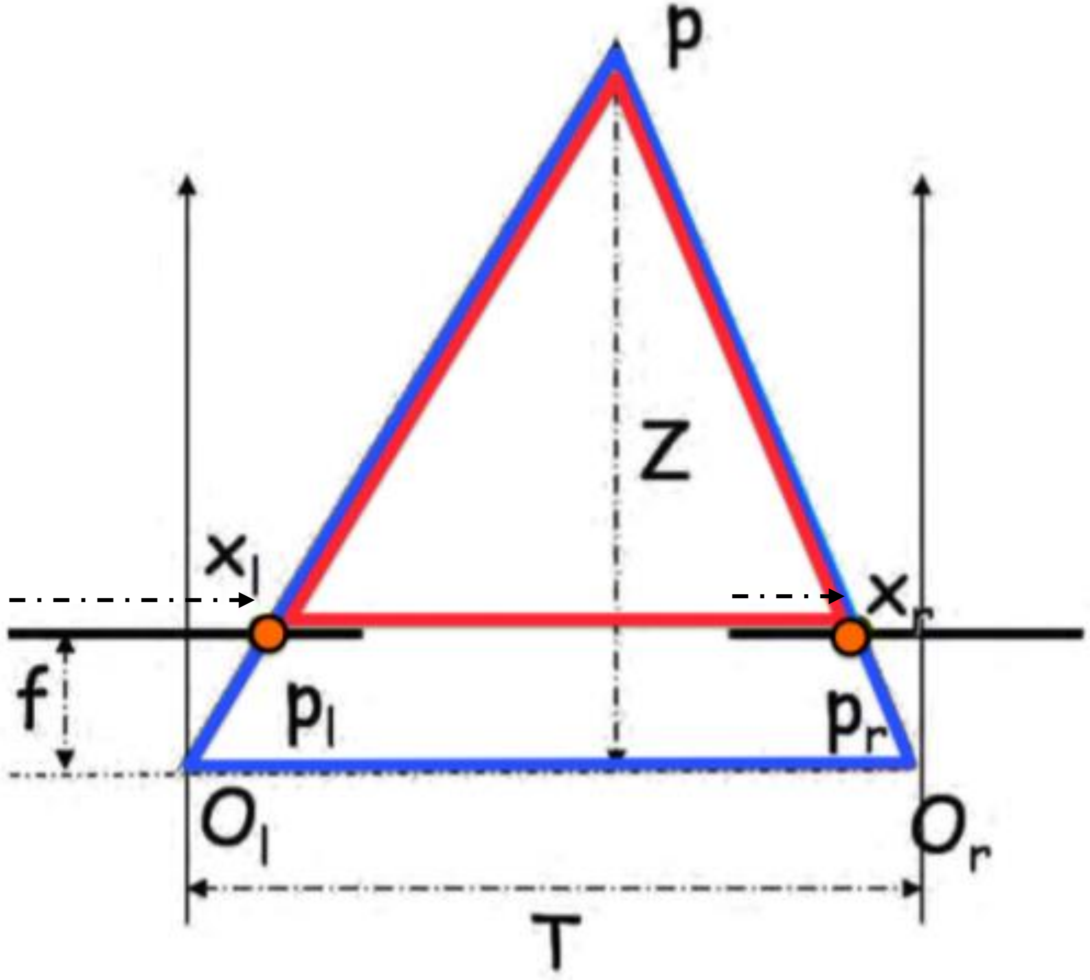
Right

Triangulation



- The amount of horizontal movement is inversely proportional to the distance from the camera.
- The amount of horizontal movement == disparity ($d = x_l - x_r$).
- Distance from the camera == depth (or Z).
- Note: x_l& x_r are in normalized image coordinate system: $x = K^{-1}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$

Triangulation



Similar triangles:

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

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

baseline

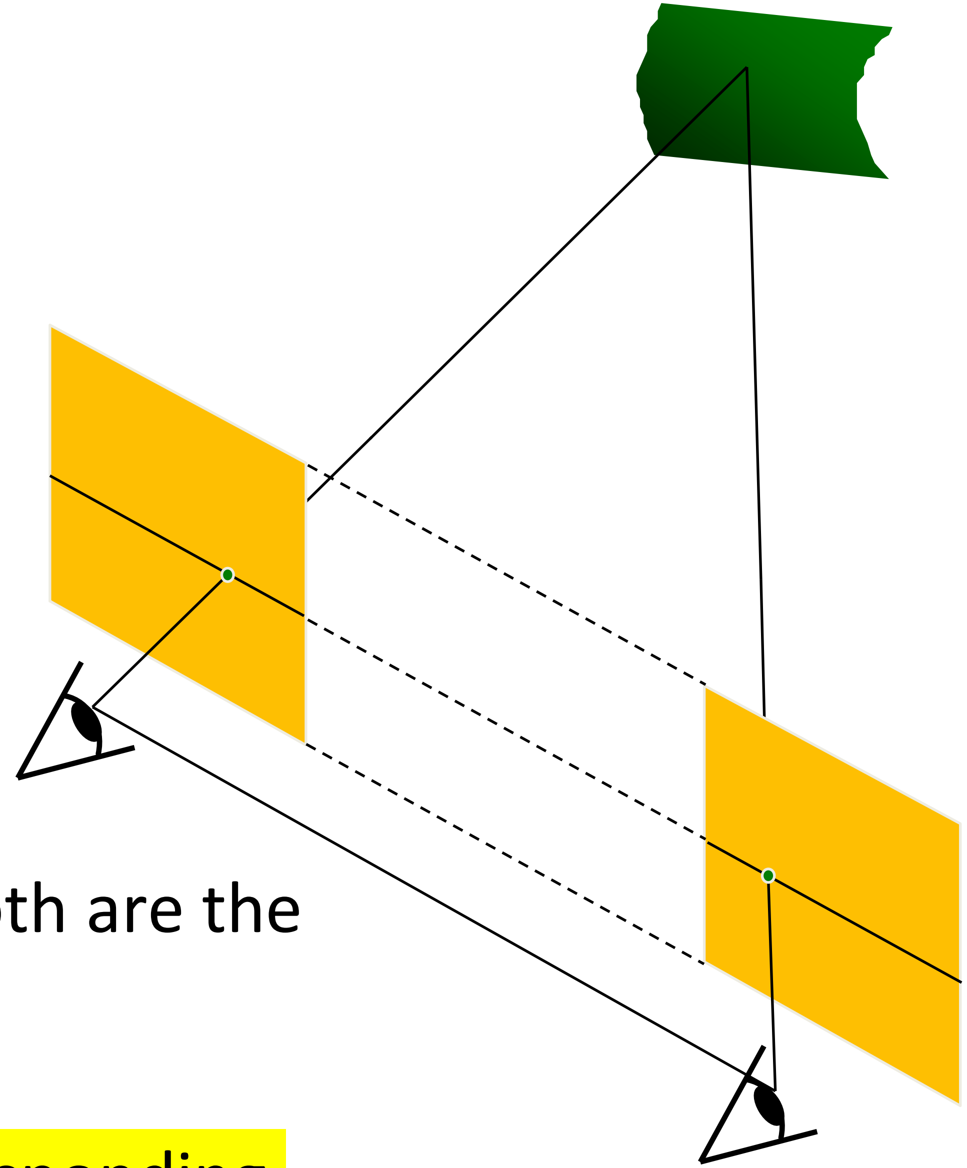
focal length

disparity

Contents

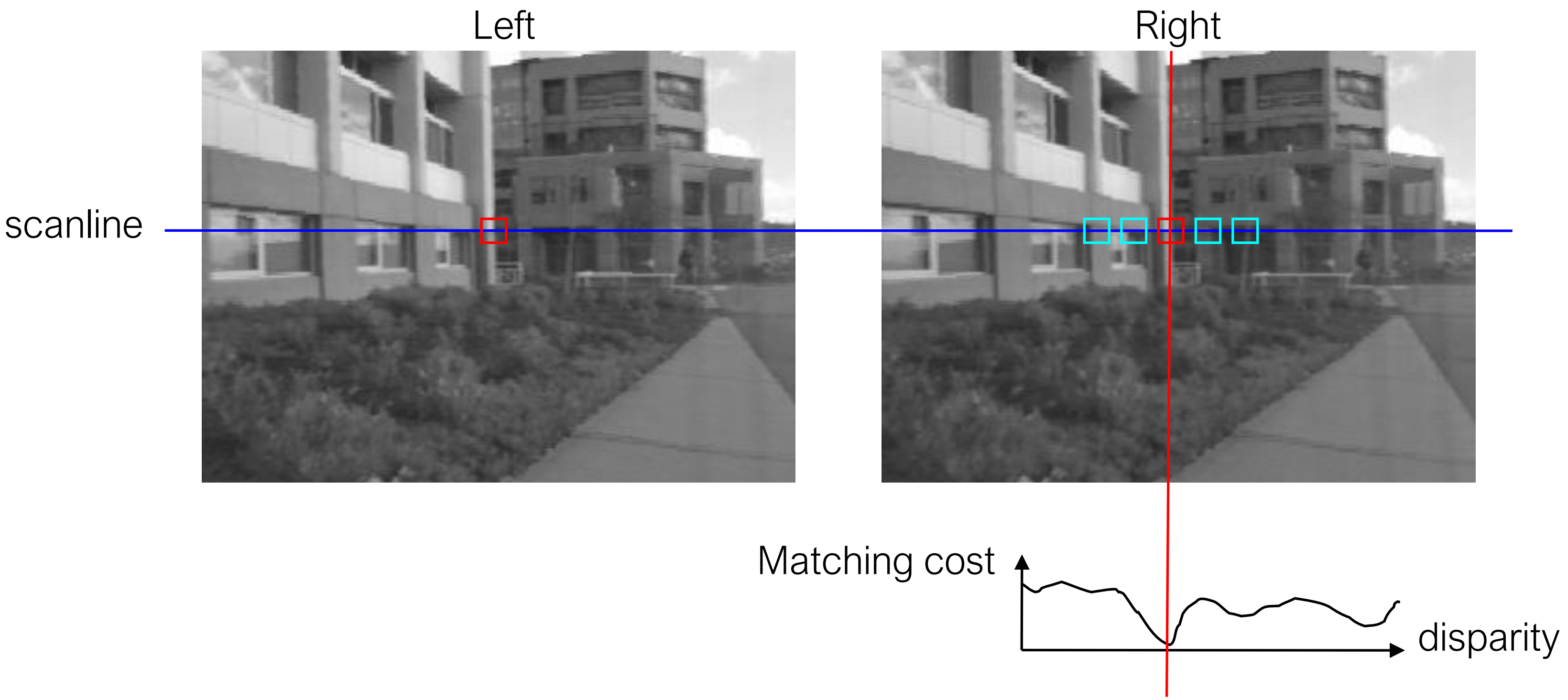
- Structure from motion
- Triangulation
- **Stereo matching**
- Camera rectification
- Epipolar geometry
  – Essential matrix
  – Fundamental matrix
  – Estimating the fundamental matrix
- Other 3D sensors

Triangulation

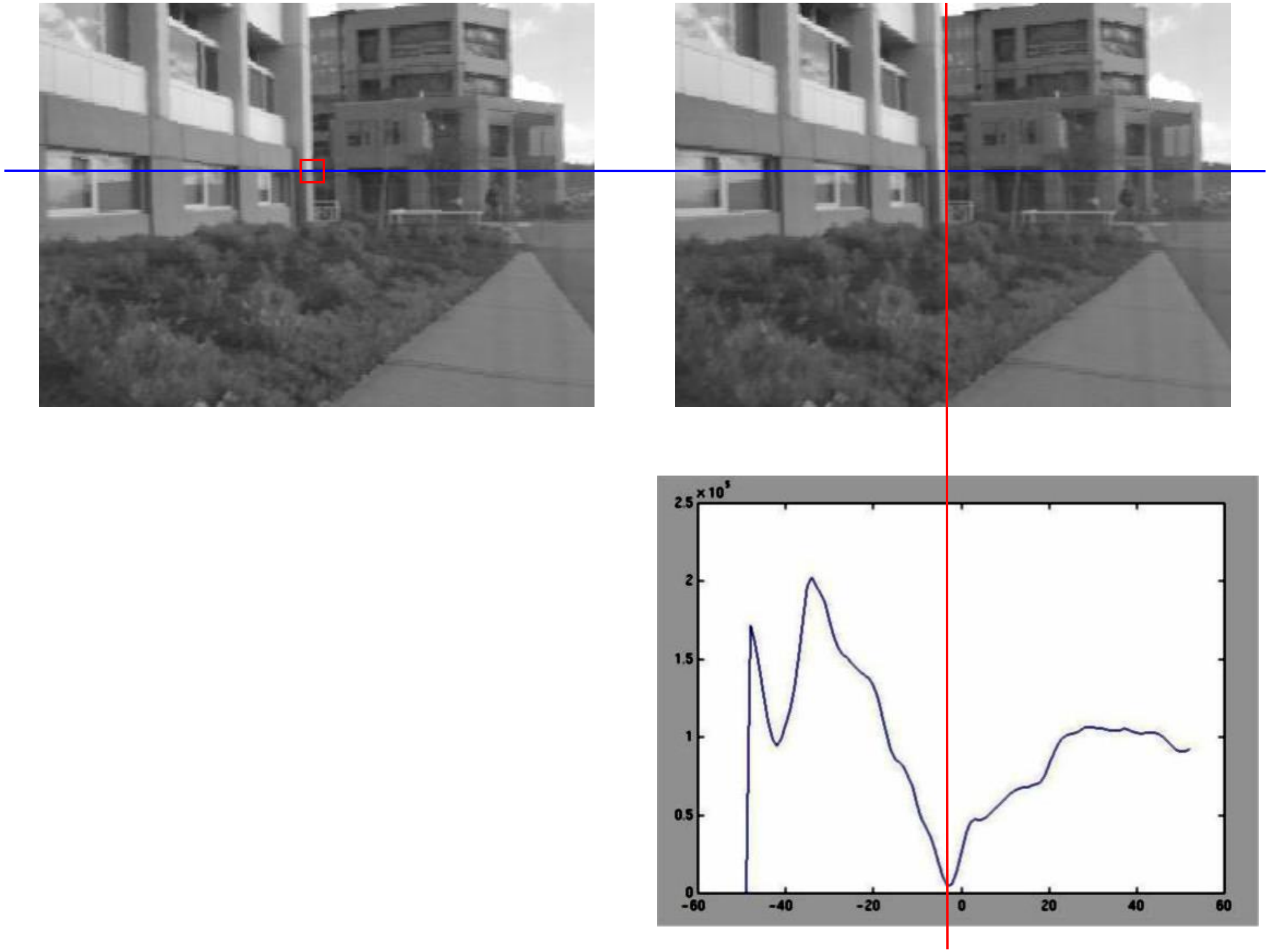


- Assume both cameras are rectified- 6 DOF of both are the same except the horizontal translation.
- Assume same focal length f in both cameras
- Assume we know for each pixel in left the corresponding pixel in right.
- From this we want to get a depth image using triangulation.

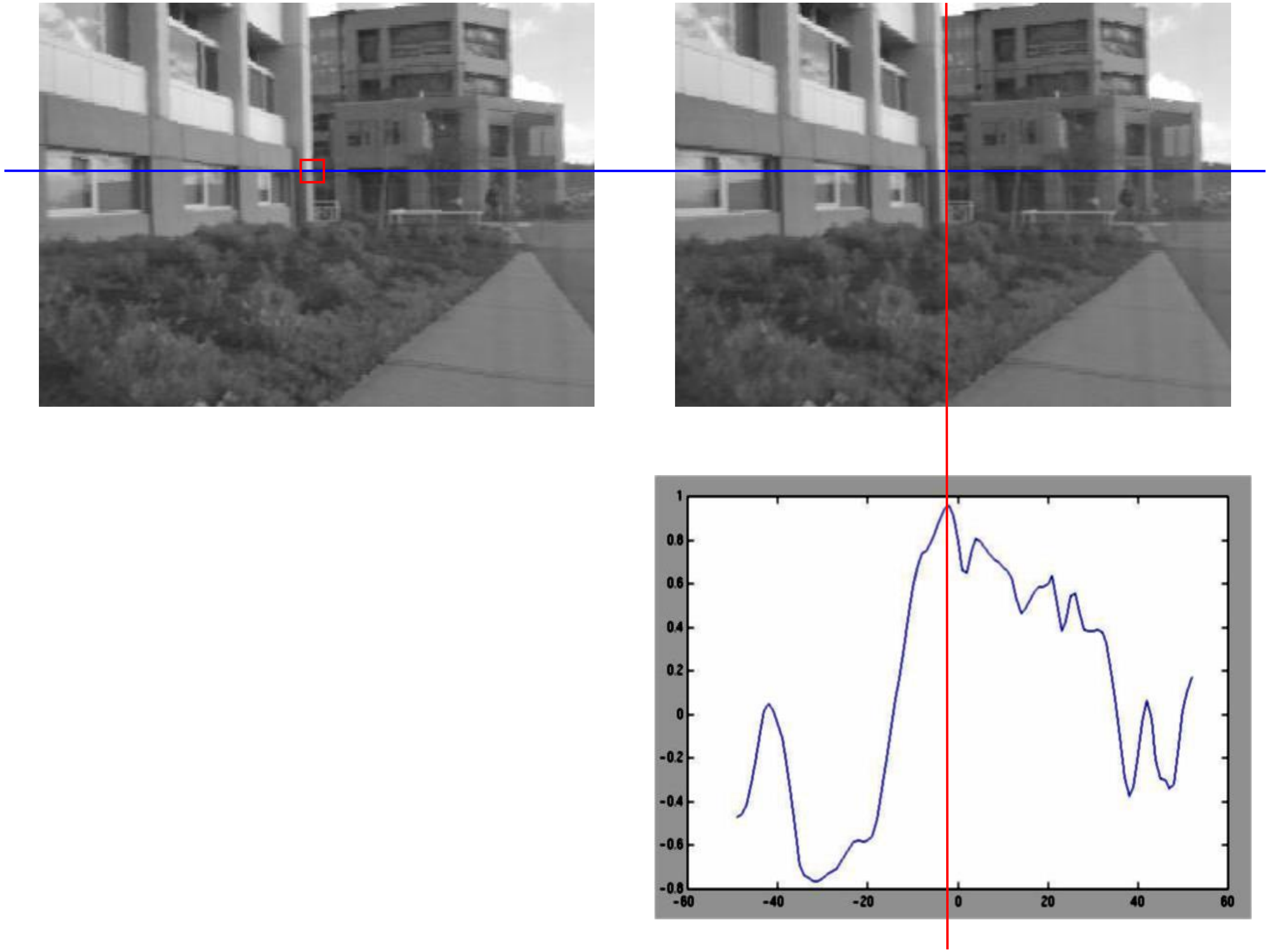
Stereo Block Matching



- 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



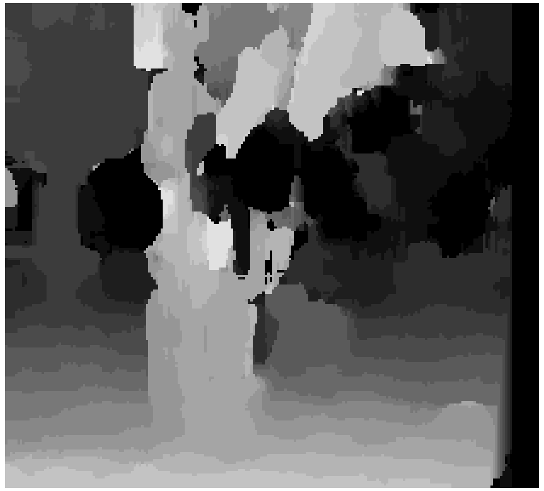
SSD



Normalized cross-correlation

Effect of window size

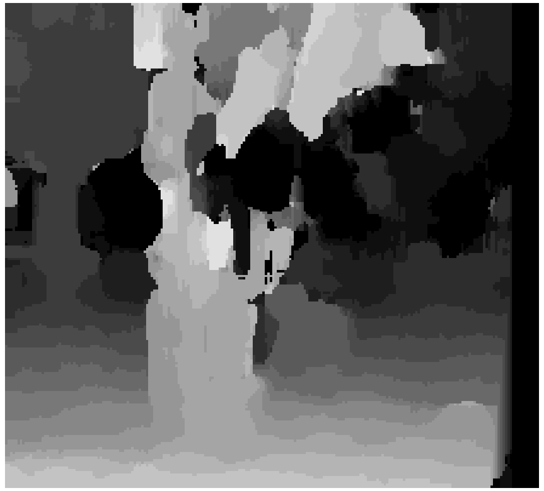
W = 3



W = 20

Effect of window size

W = 3

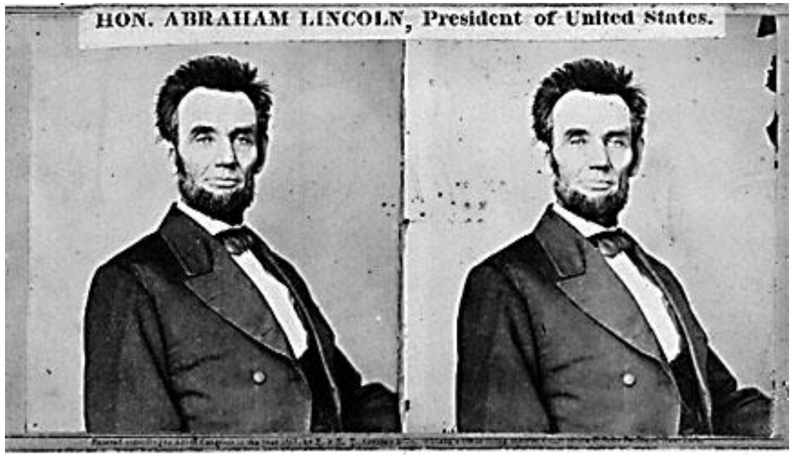


W = 20

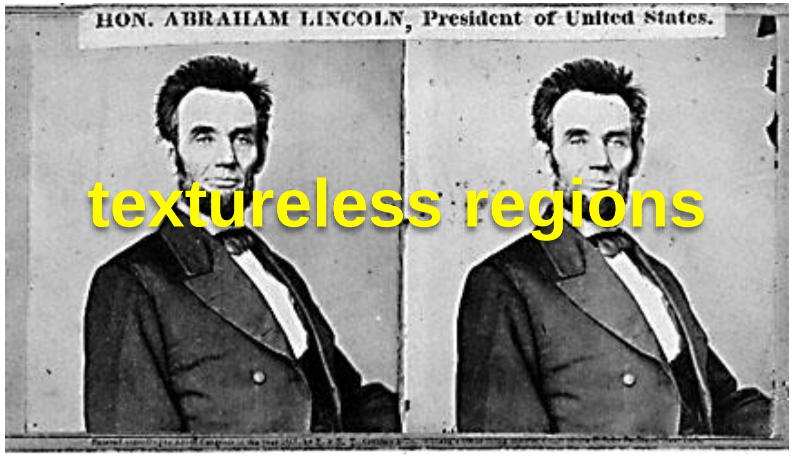
Smaller window
- \+ More detail
- \- More noise

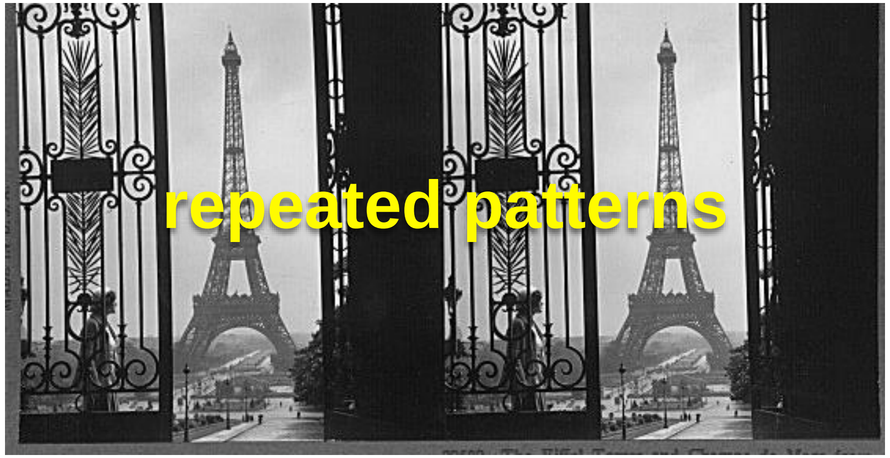
Larger window
- \+ Smoother disparity maps
- \- Less detail
- \- Fails near boundaries

When will stereo block matching fail?

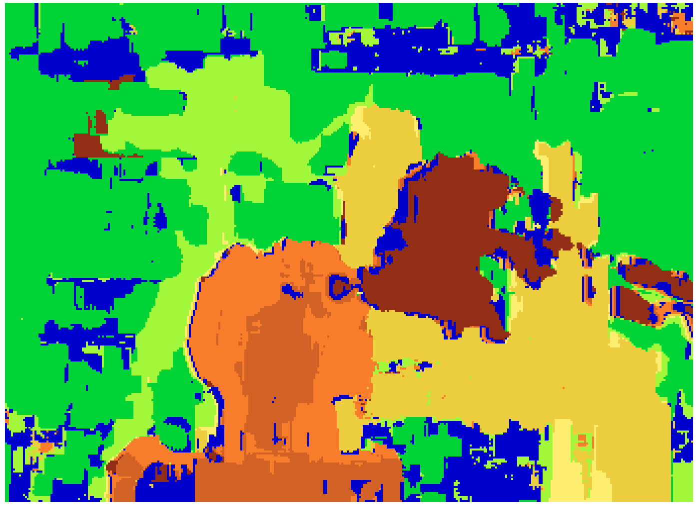


When will stereo block matching fail?



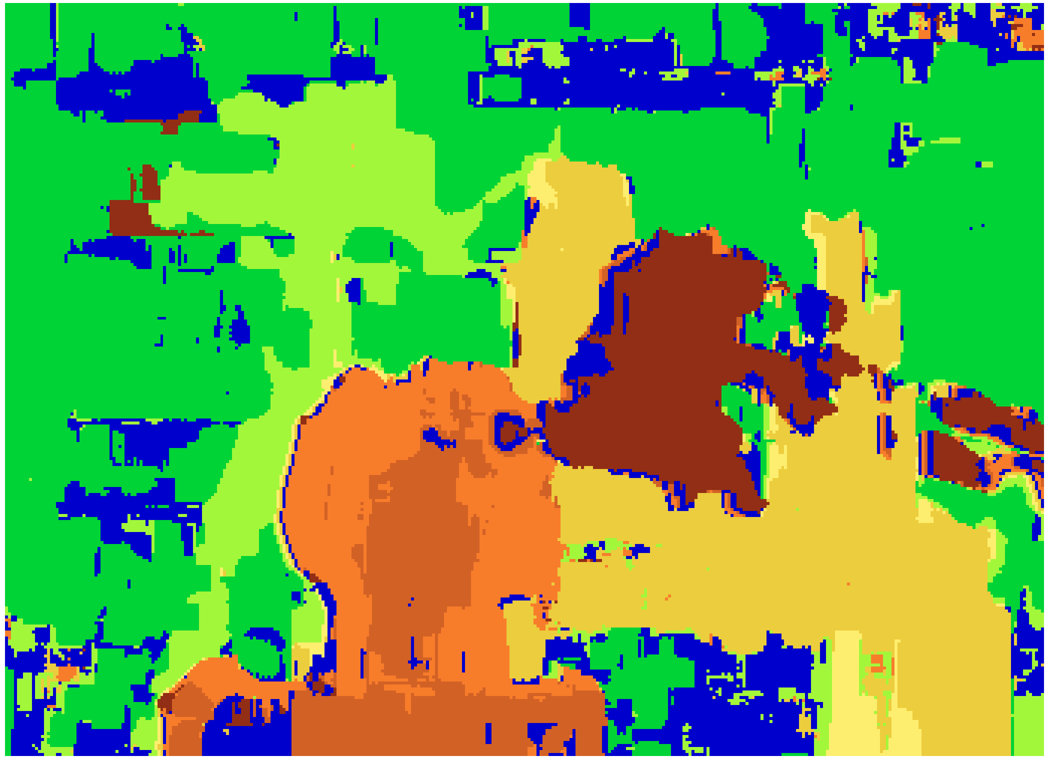


Block matching

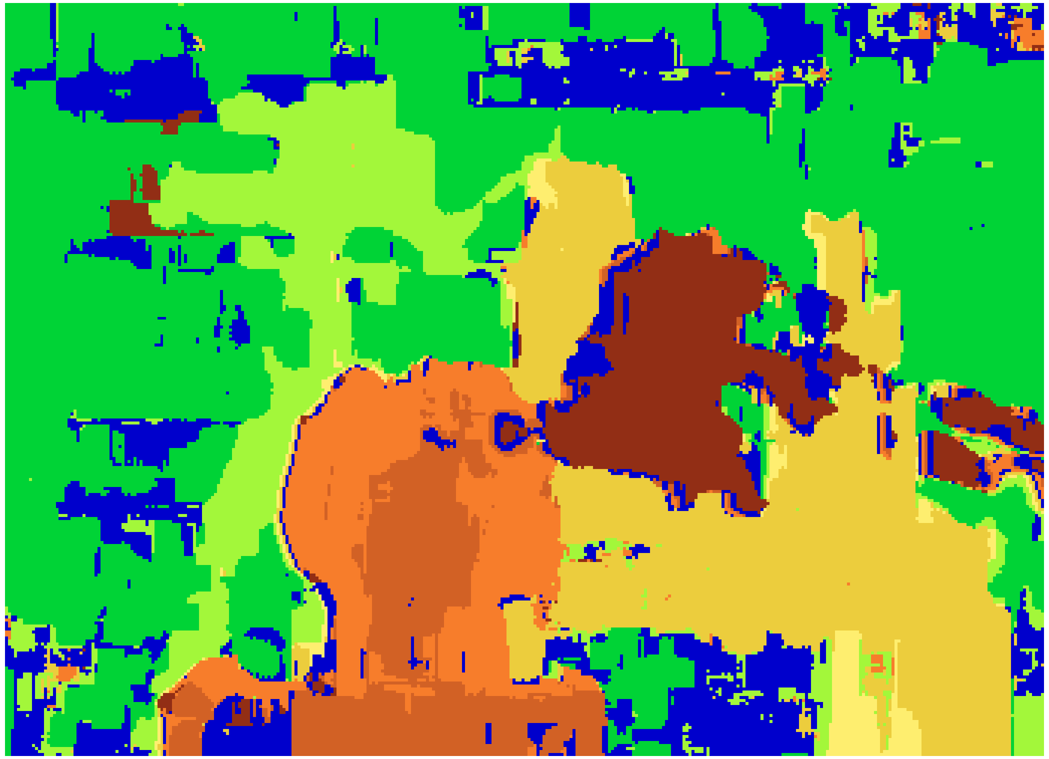


Ground truth

What are some problems with the result?



How can we improve depth estimation?

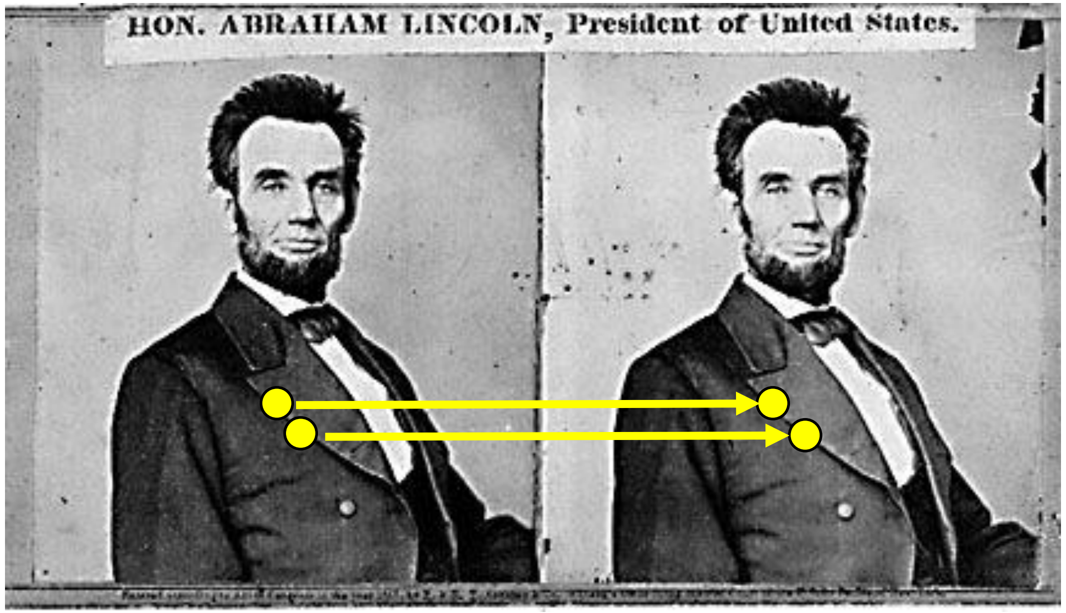


How can we improve depth estimation?

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)

$$E(d) = \boxed{E_d(d)} + \lambda E_s(d)$$

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

data term

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

$$E(d) = E_d(d) + \lambda \boxed{E_s(d)}$$

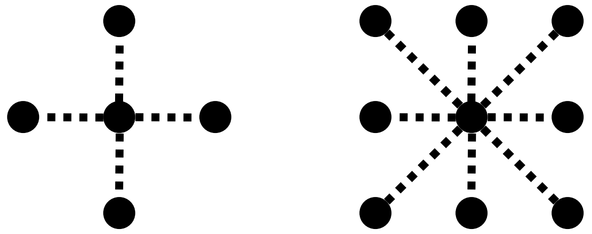
$$E_d(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

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

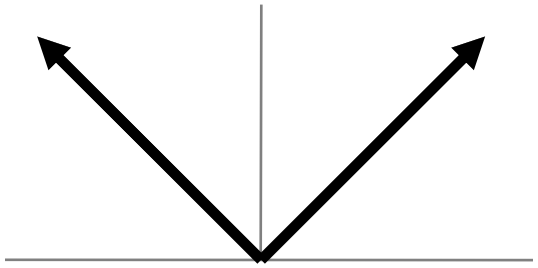
8-connected neighborhood

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

smoothness term

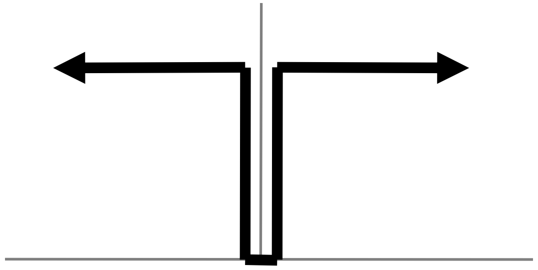
$$V(d_p, d_q) = |d_p - d_q|$$

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

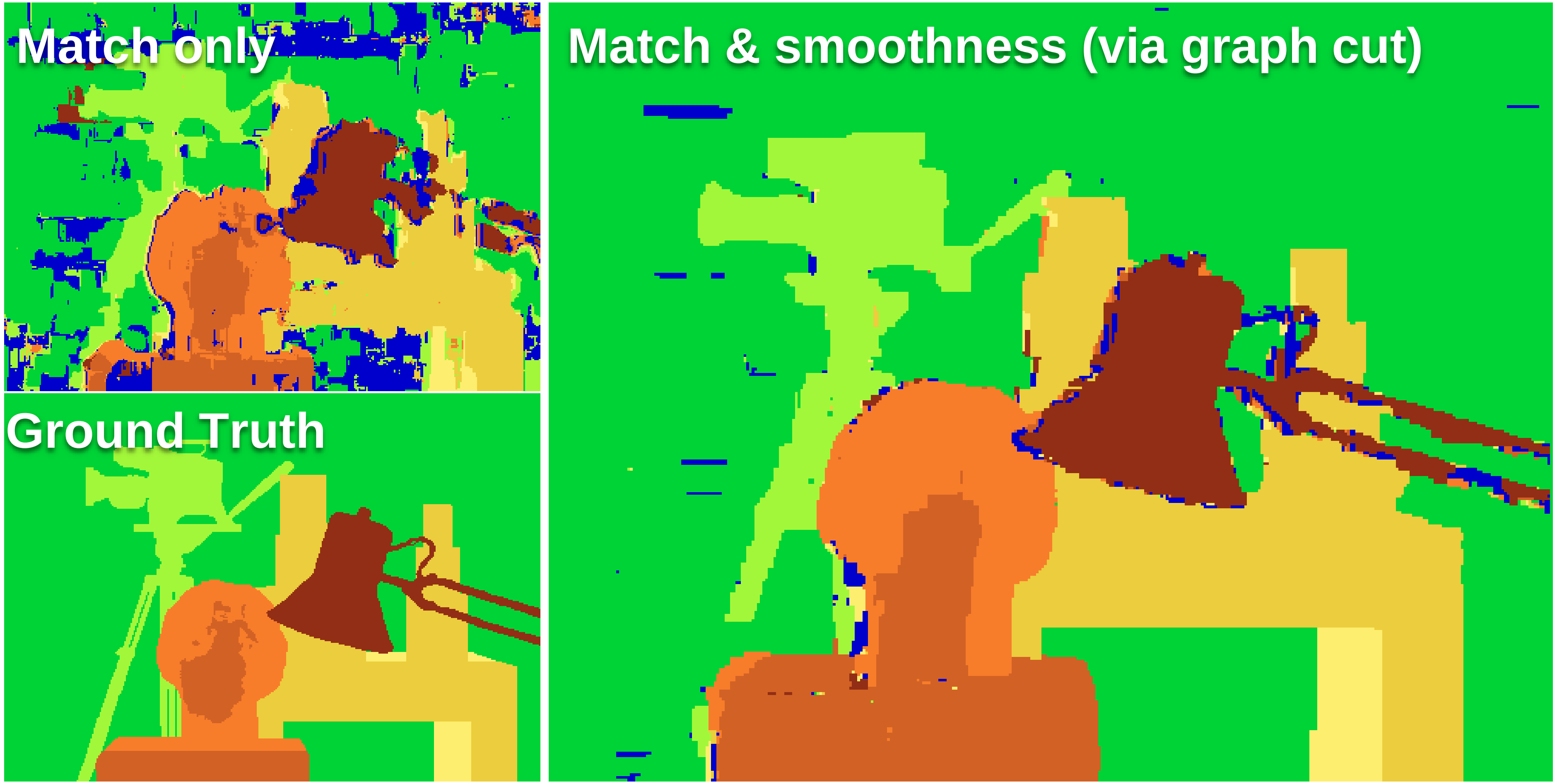
One possible solution…

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

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

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

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



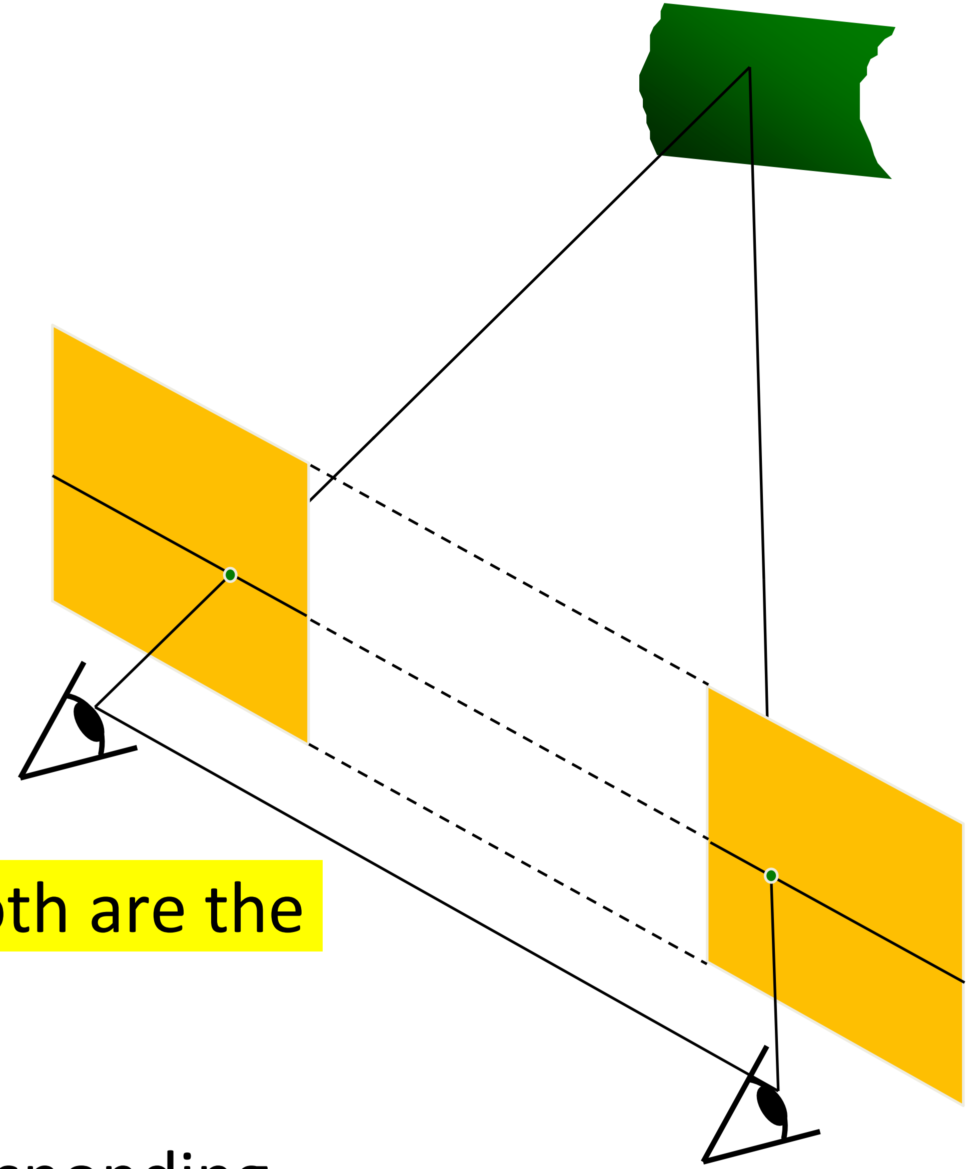
Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](), PAMI 2001

Contents

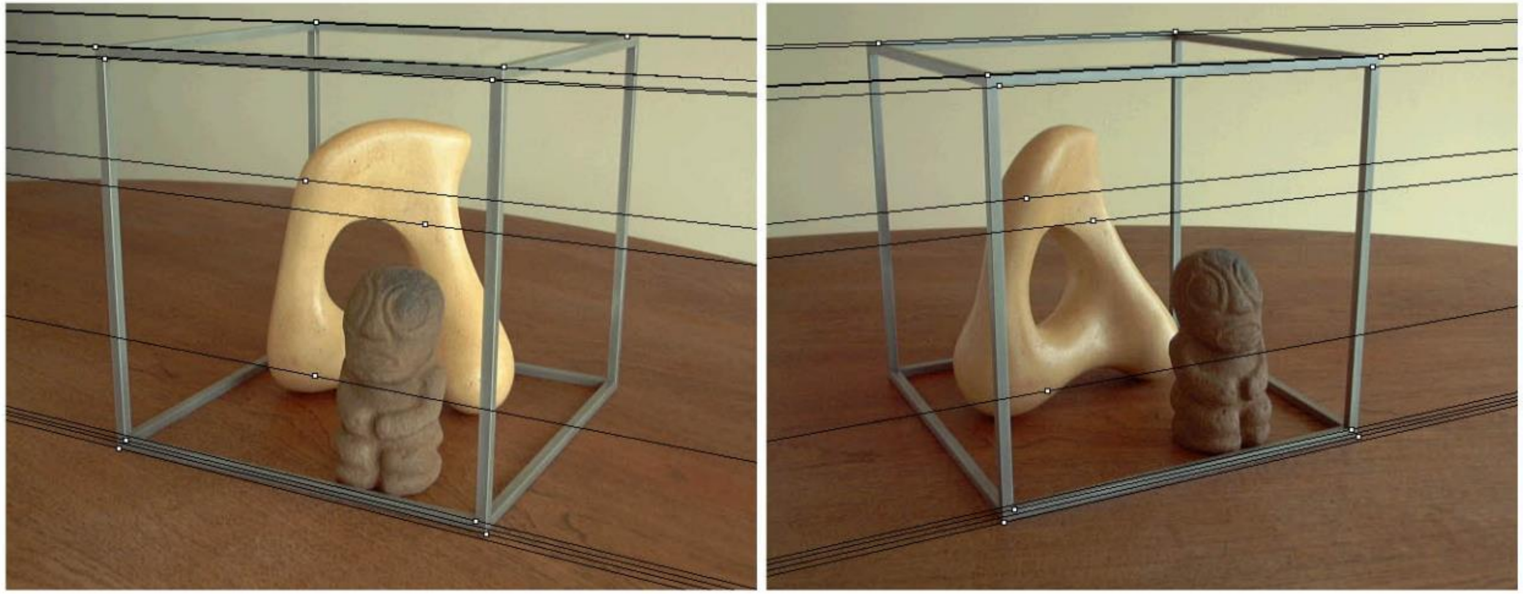
- Structure from motion
- Triangulation
- Stereo matching
- **Camera rectification**
- Epipolar geometry
  - Essential matrix
  - Fundamental matrix
  - Estimating the fundamental matrix
- Other 3D sensors

Triangulation



- Assume both cameras are rectified- 6 DOF of both are the same except the horizontal translation.
- Assume same focal length f in both cameras.
- Assume we know for each pixel in left the corresponding pixel in right.
- From this we want to get a depth image using triangulation.

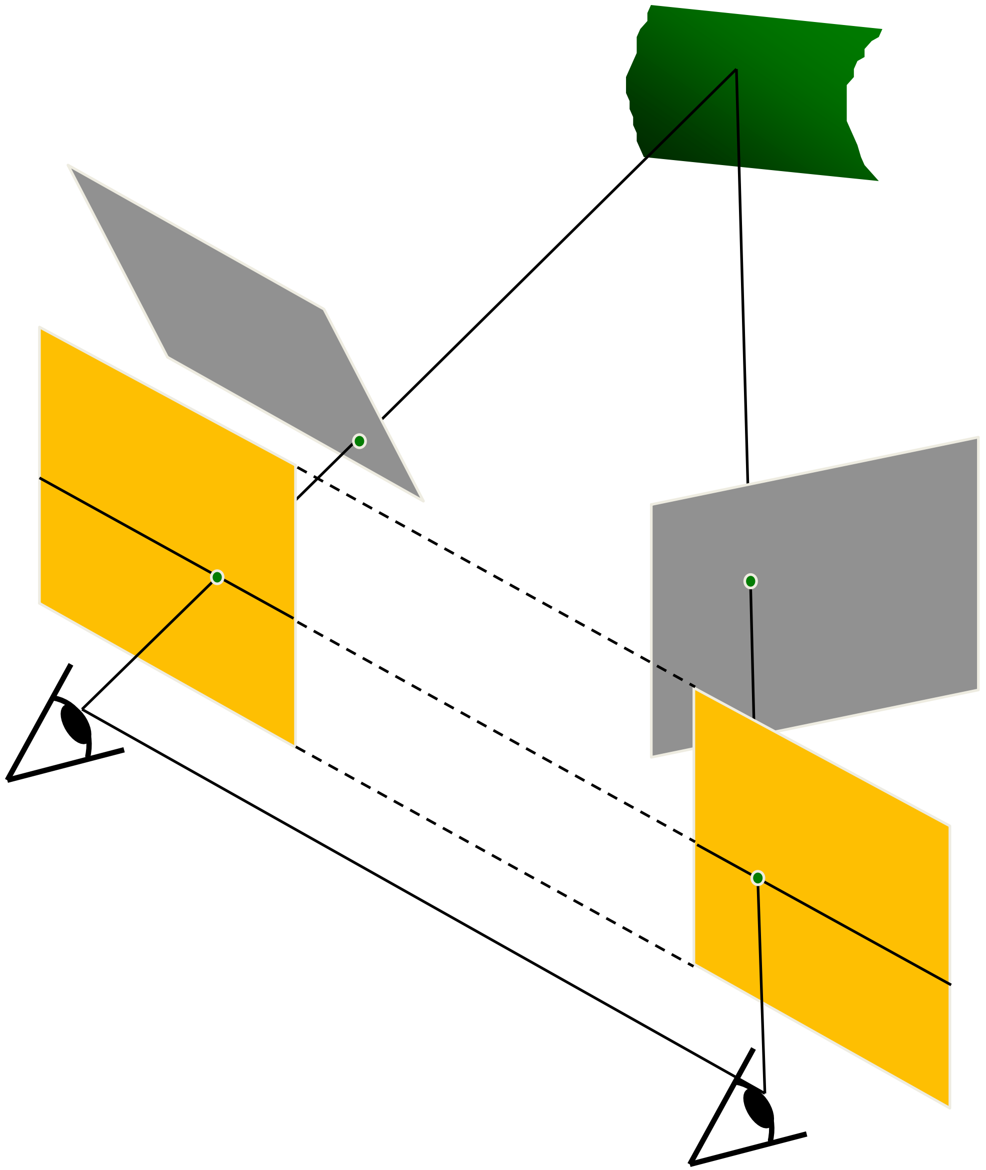


Original stereo pair

After rectification

Stereo image rectification

- Out of scope…
- Let’s say the images comes rectified (as in the yellow samples).
  - Rectification proof here: https://www.cs.cmu.edu/~16385/s17/Slides/13.1_Stereo_Rectification.pdf
- We want to find the relative R, t of the images.



Contents

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

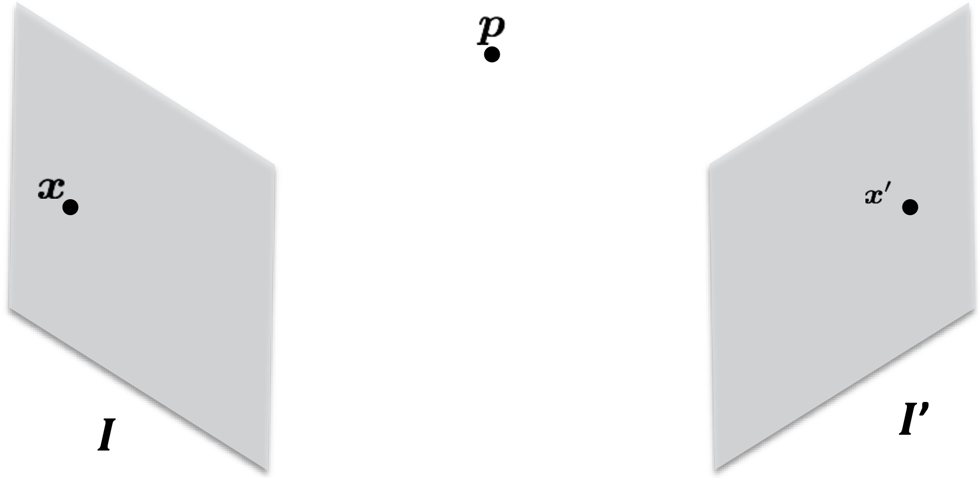
Epipolar geometry

- **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]

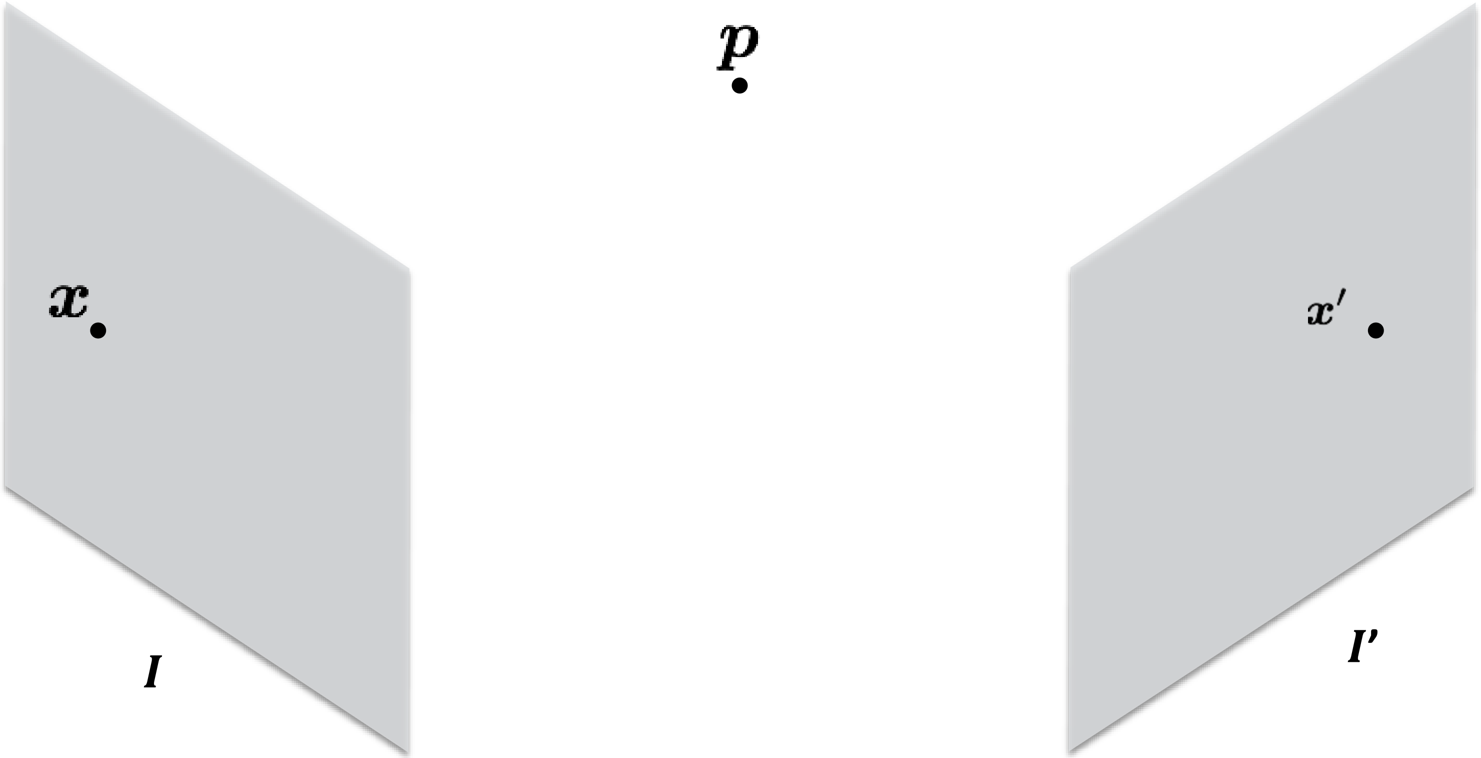
Epipolar geometry - The triangulation problem

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

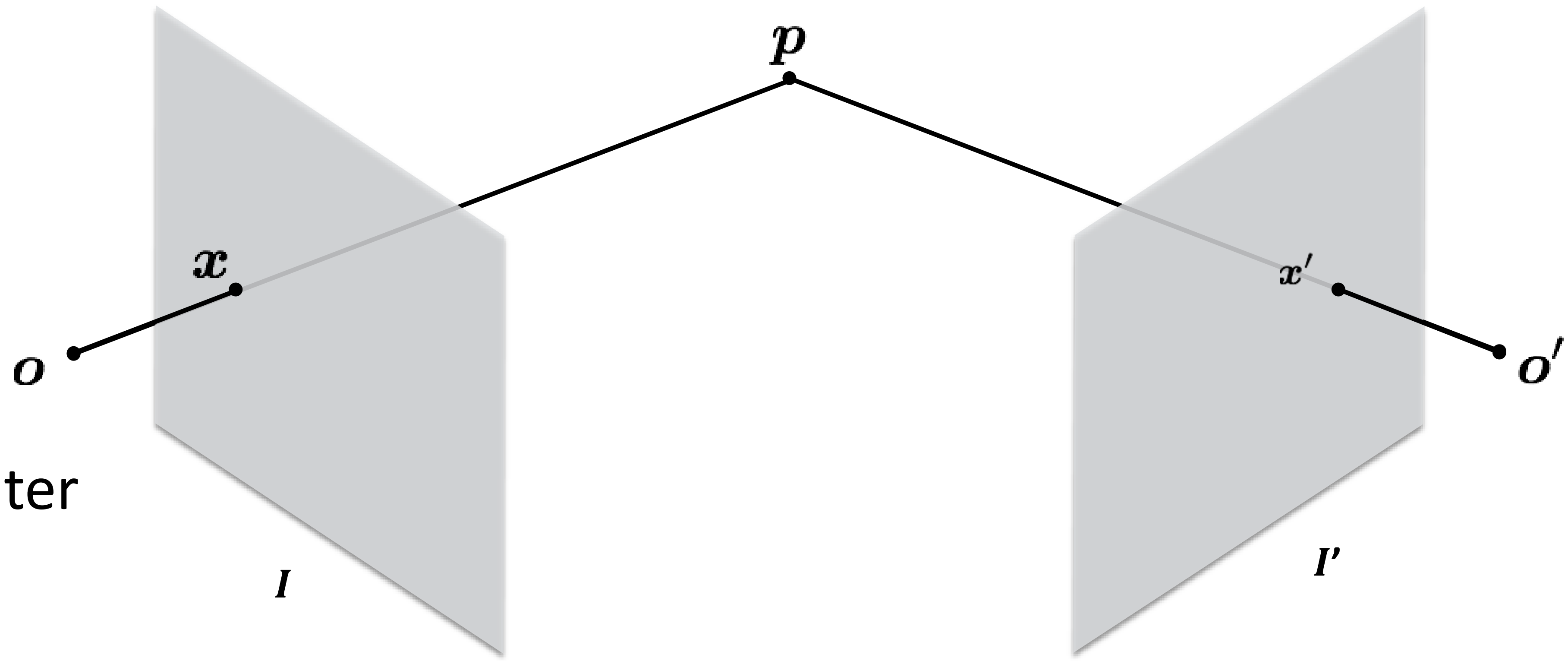


Epipolar geometry



Epipolar geometry

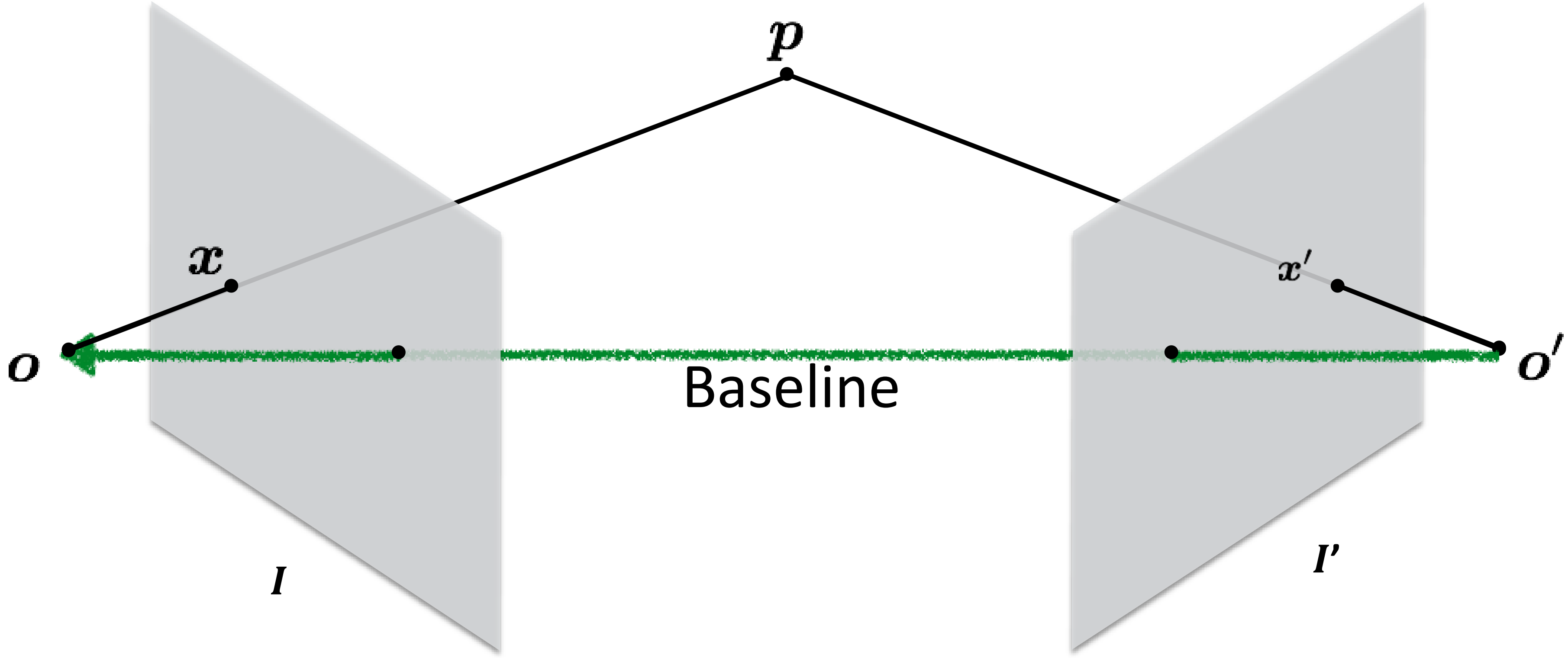
- We can trace lines from the **camera center** of each image, through the given 2D point to the 3D point p.



Camera center

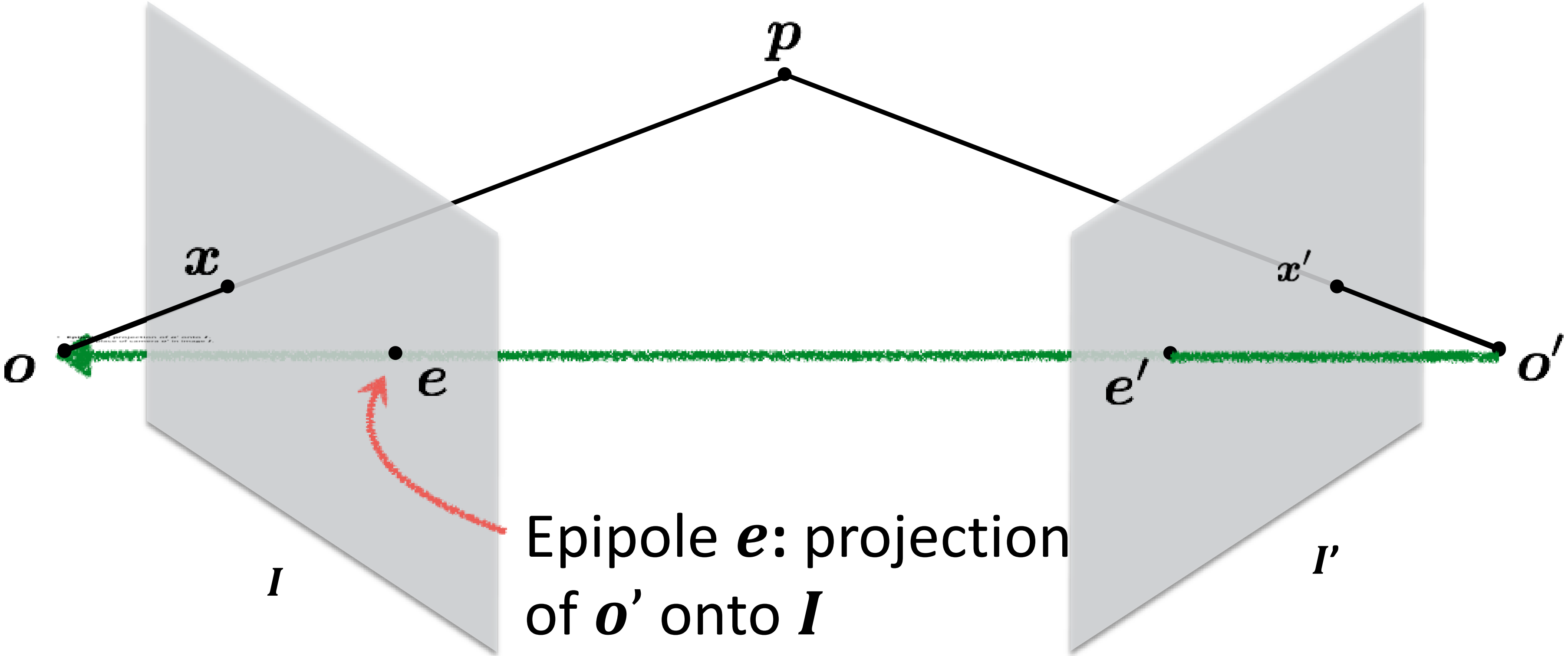
Epipolar geometry

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



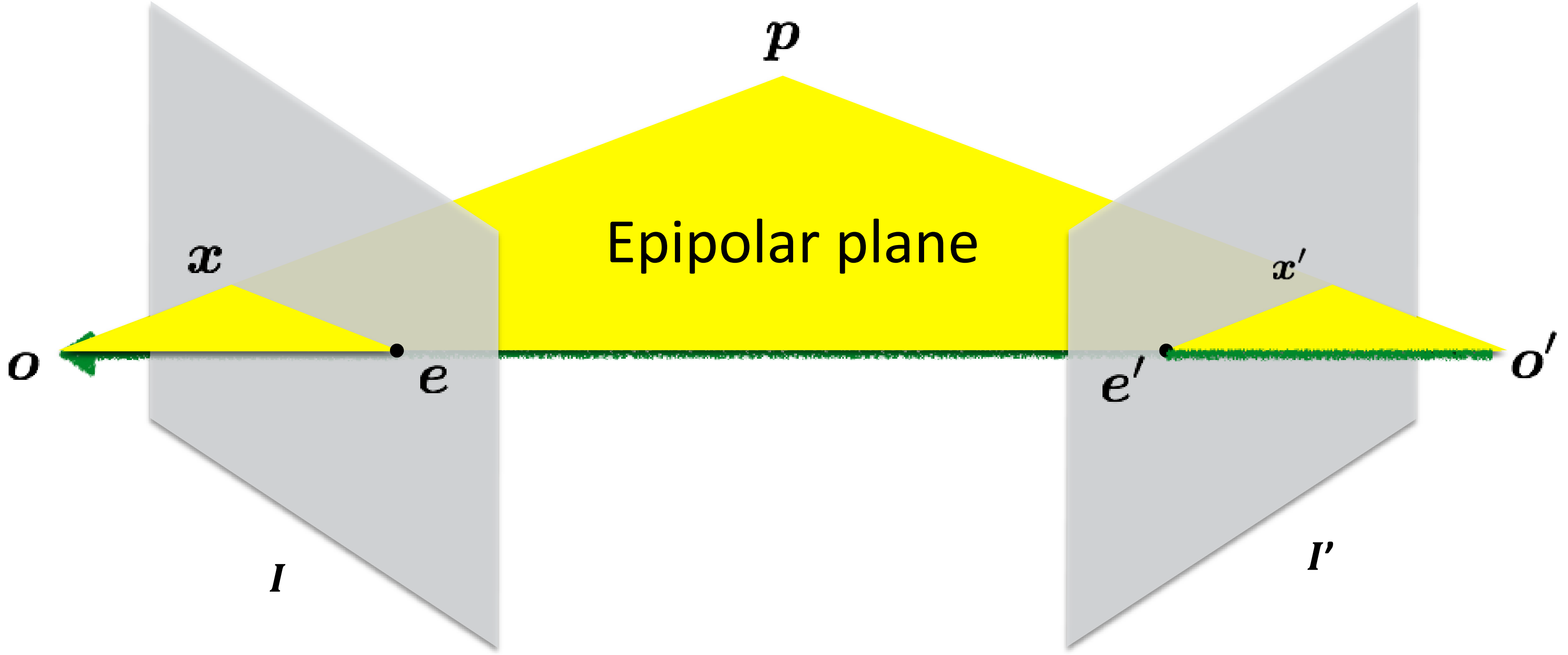
Epipolar geometry

- **Epipole $\boldsymbol{e}$:** projection of $\boldsymbol{o}$’ onto $\boldsymbol{I}$.
  - The place of camera $\boldsymbol{o}'$ in image $\boldsymbol{I}$.



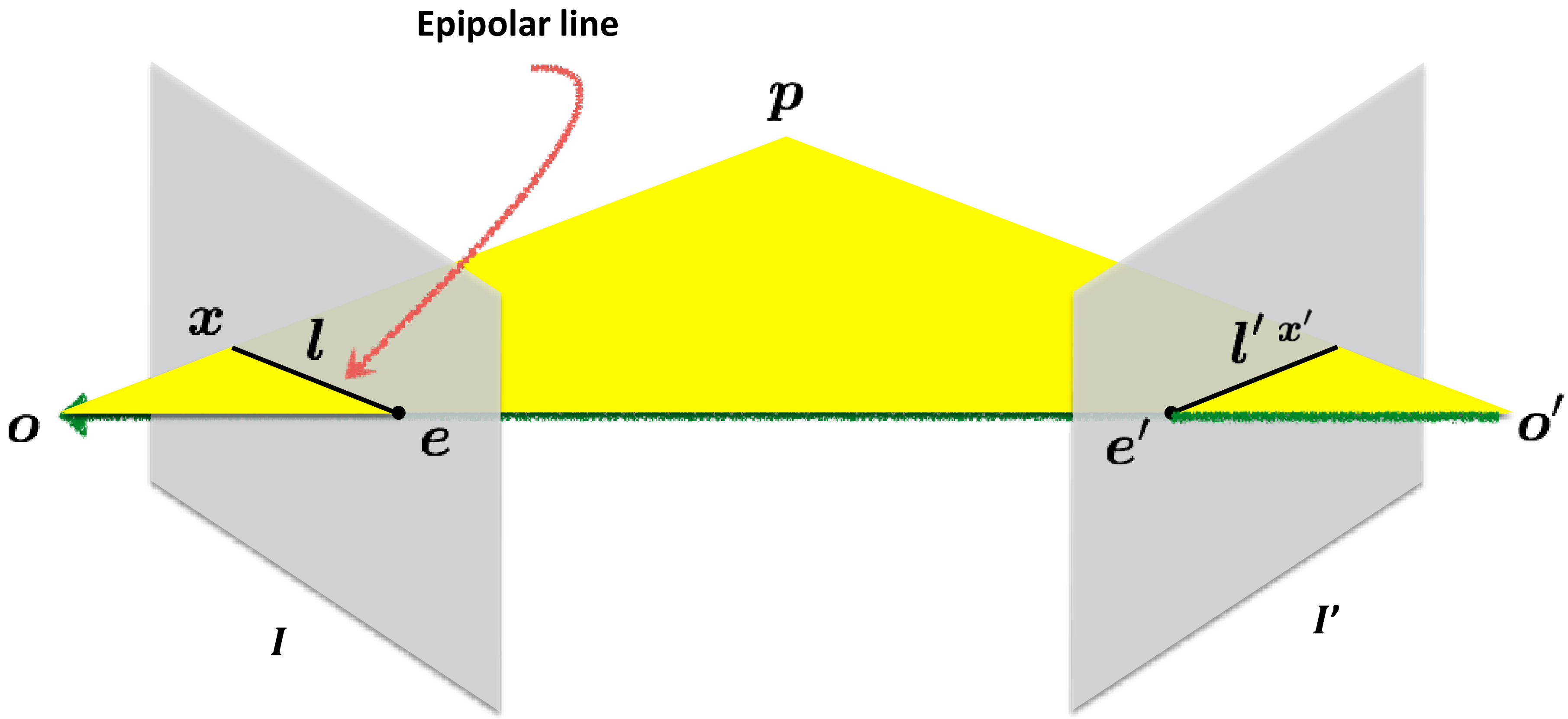
Epipolar geometry

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



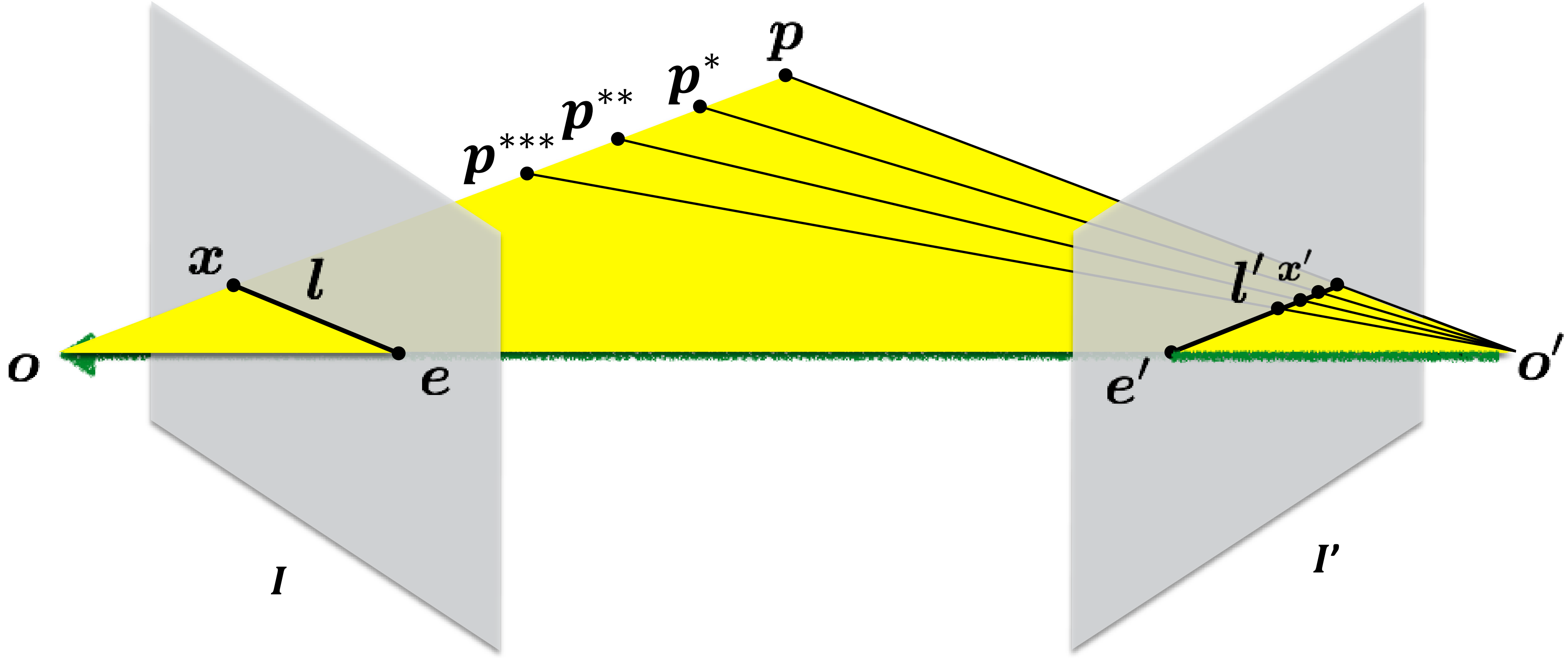
Epipolar geometry

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



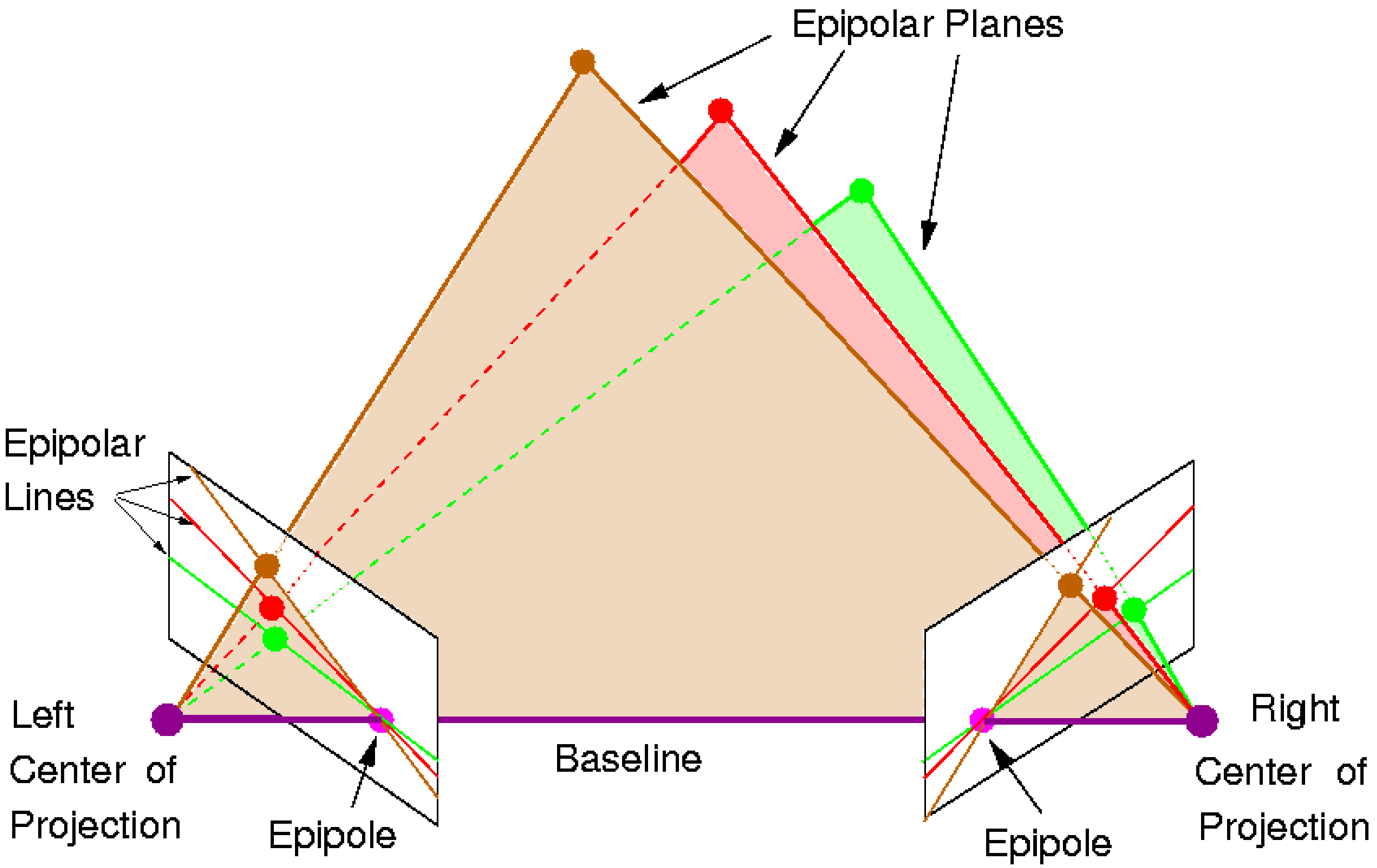
Epipolar constraint

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



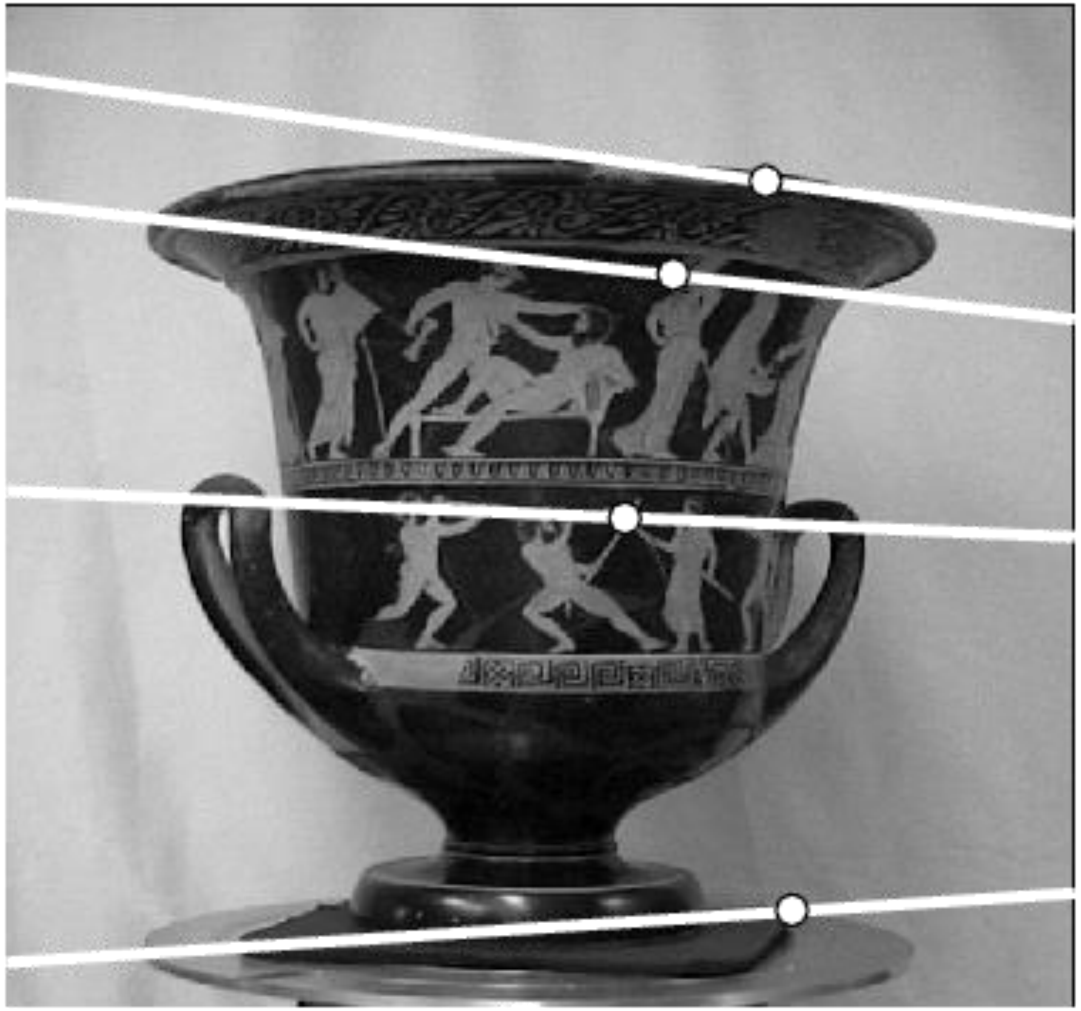
Epipolar geometry

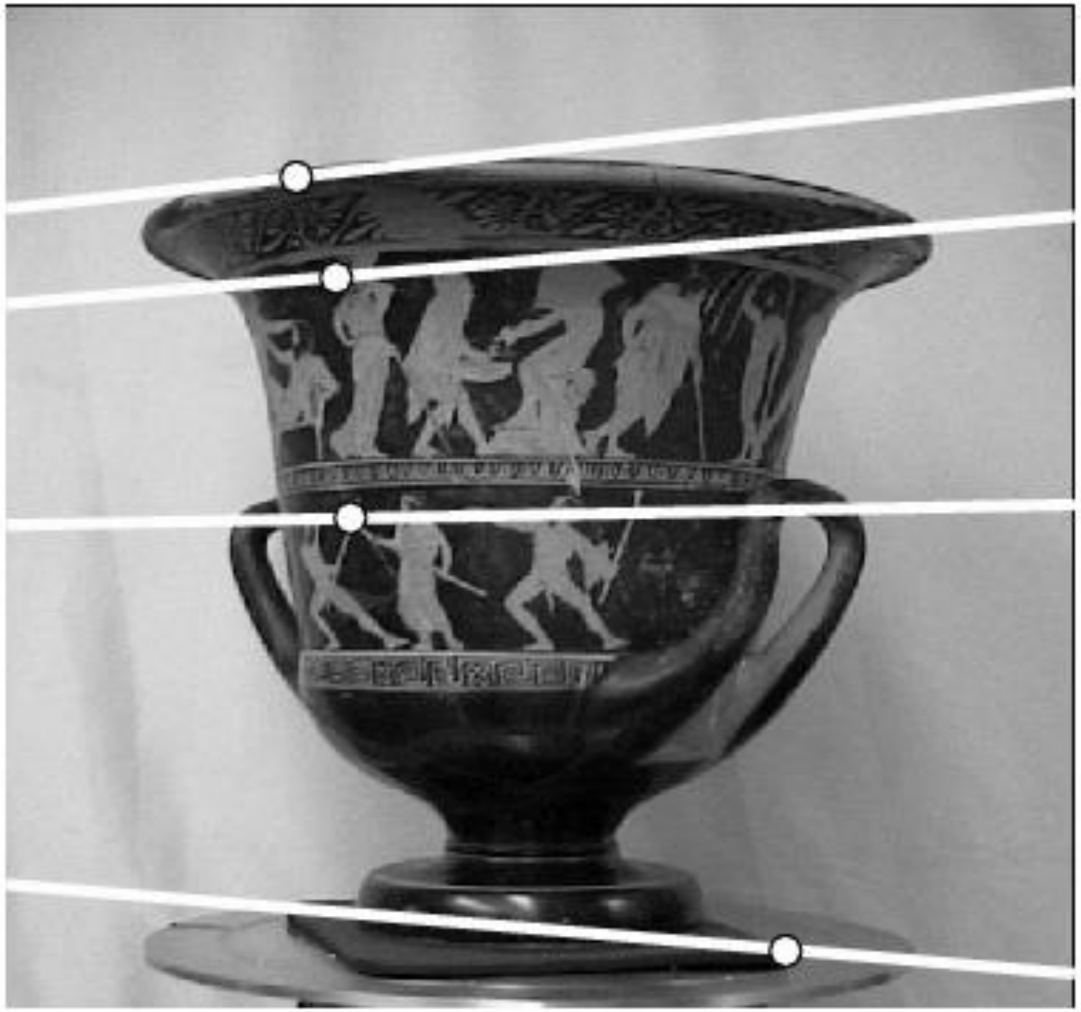
- Note: all epipolar lines pass through the epipole.



Epipolar geometry

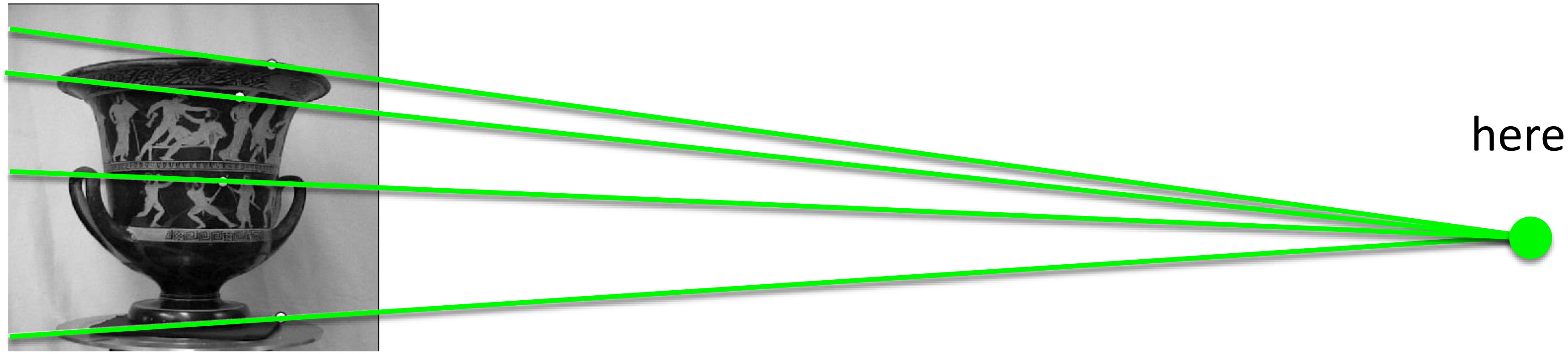
- Where is the epipole in this images?





Epipolar geometry

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



Epipolar geometry

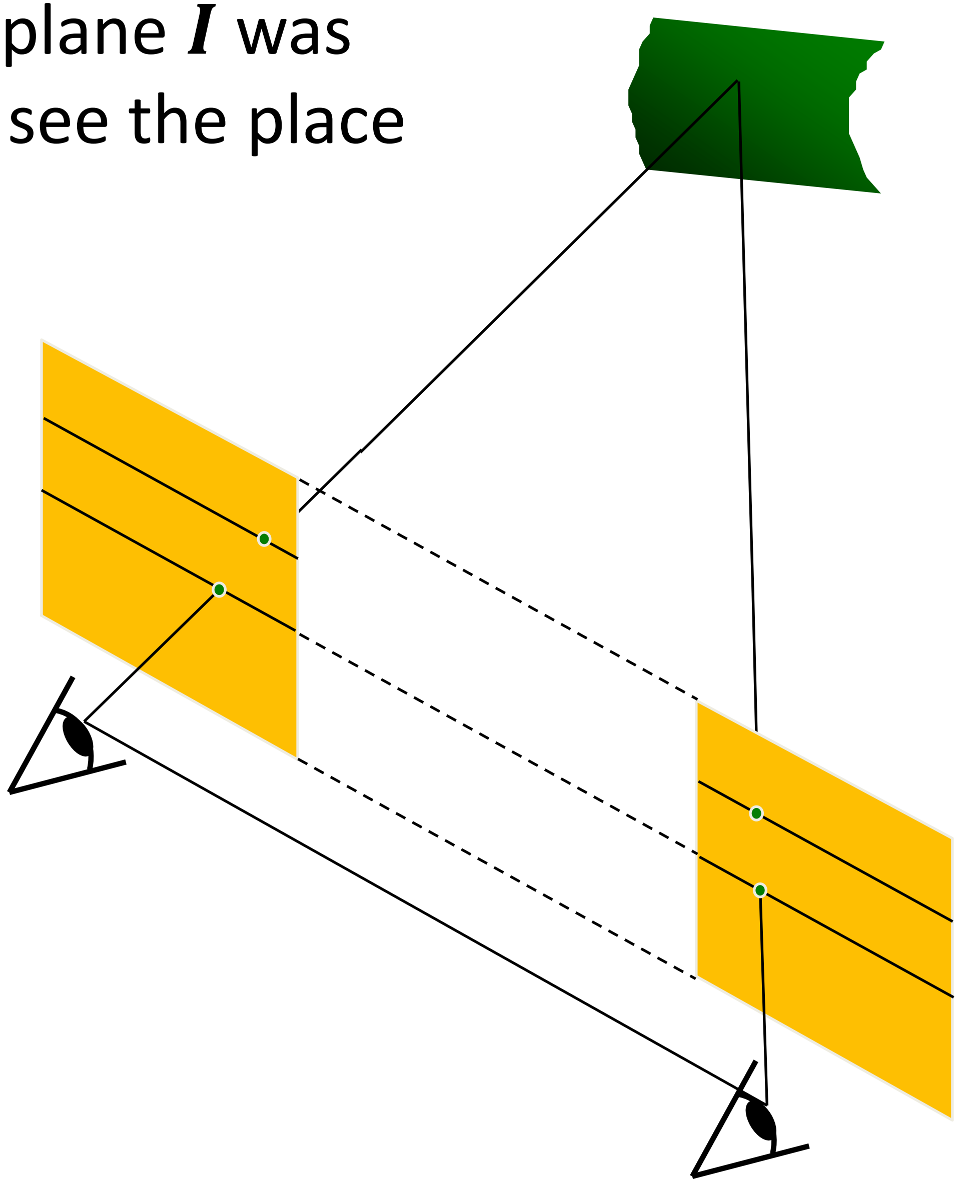
- Where is the epipole in this image?

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

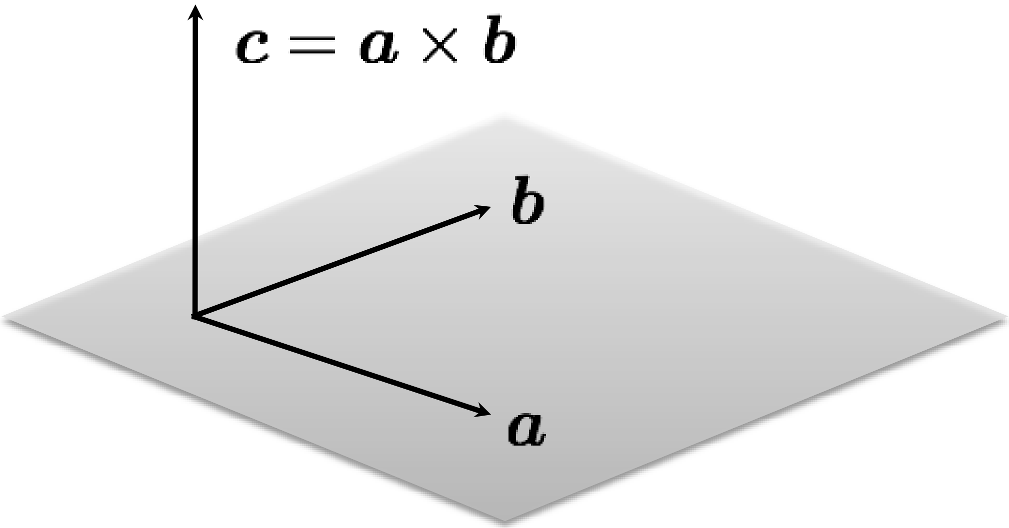
- Even if the image plane $\boldsymbol{I}$ was infinite, you can't see the place of $\boldsymbol{o}'$.



Contents

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

Recall: Dot Product



$$c \cdot a = 0 \qquad c \cdot b = 0$$

Dot product of two orthogonal vectors is zero.

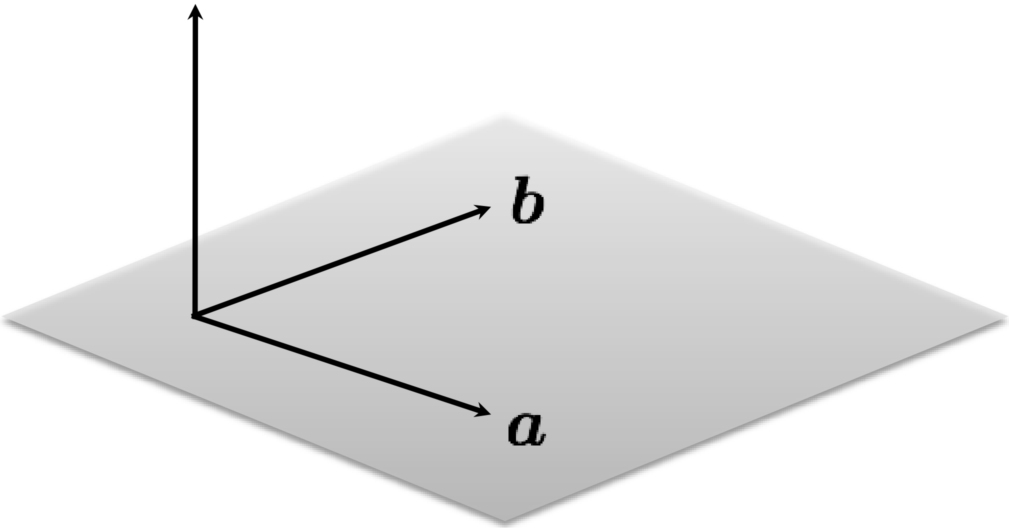
Dot product can also be written as vector multiplication [a, b are size $3X1$]:

$$a \cdot b = 0 \leftrightarrow a^T b = 0$$

Recall: Cross Product

Vector (cross) product
takes two vectors and returns a vector perpendicular to both

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



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

Recall: Cross Product

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

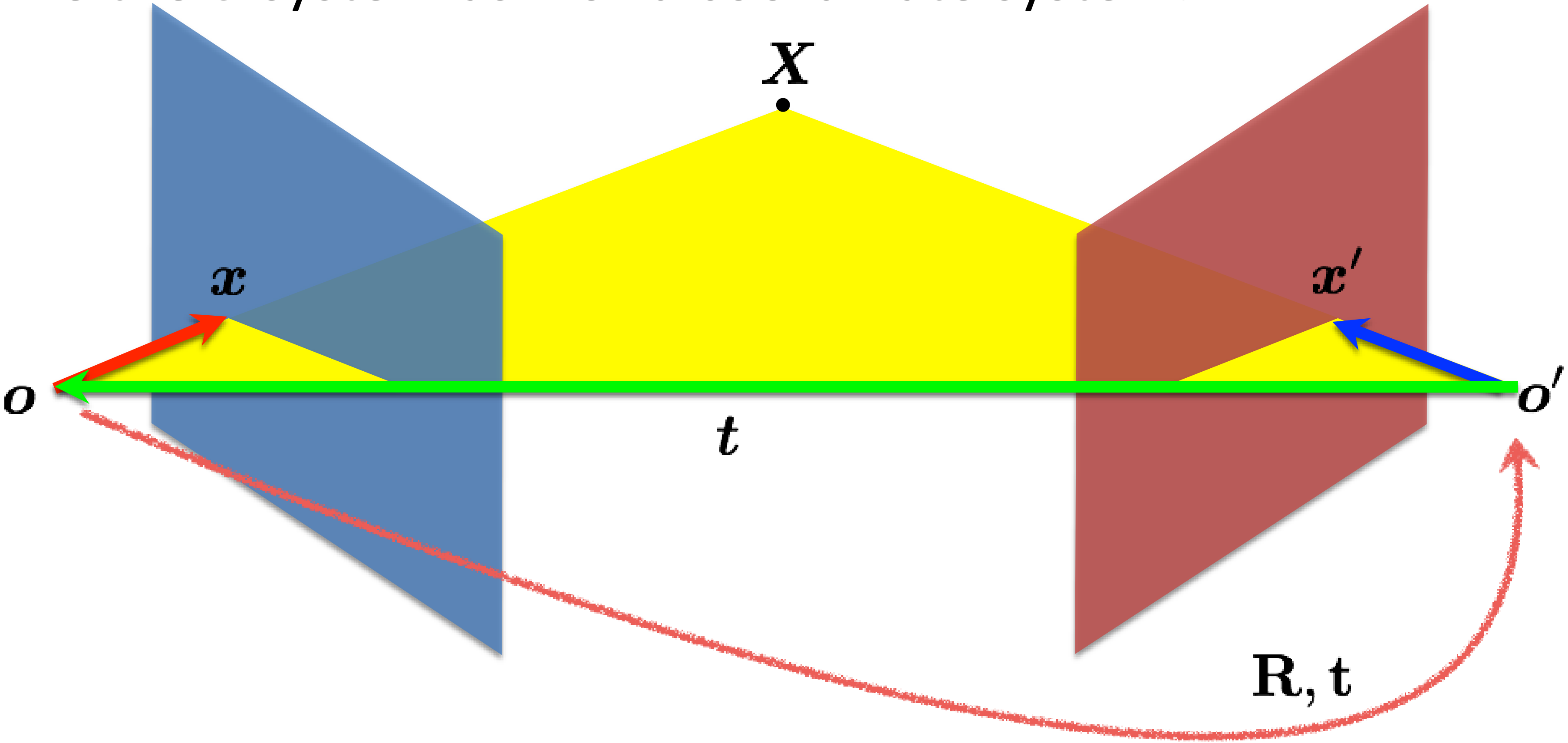
Can also be written as a matrix multiplication

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

Skew symmetric

Building the essential matrix

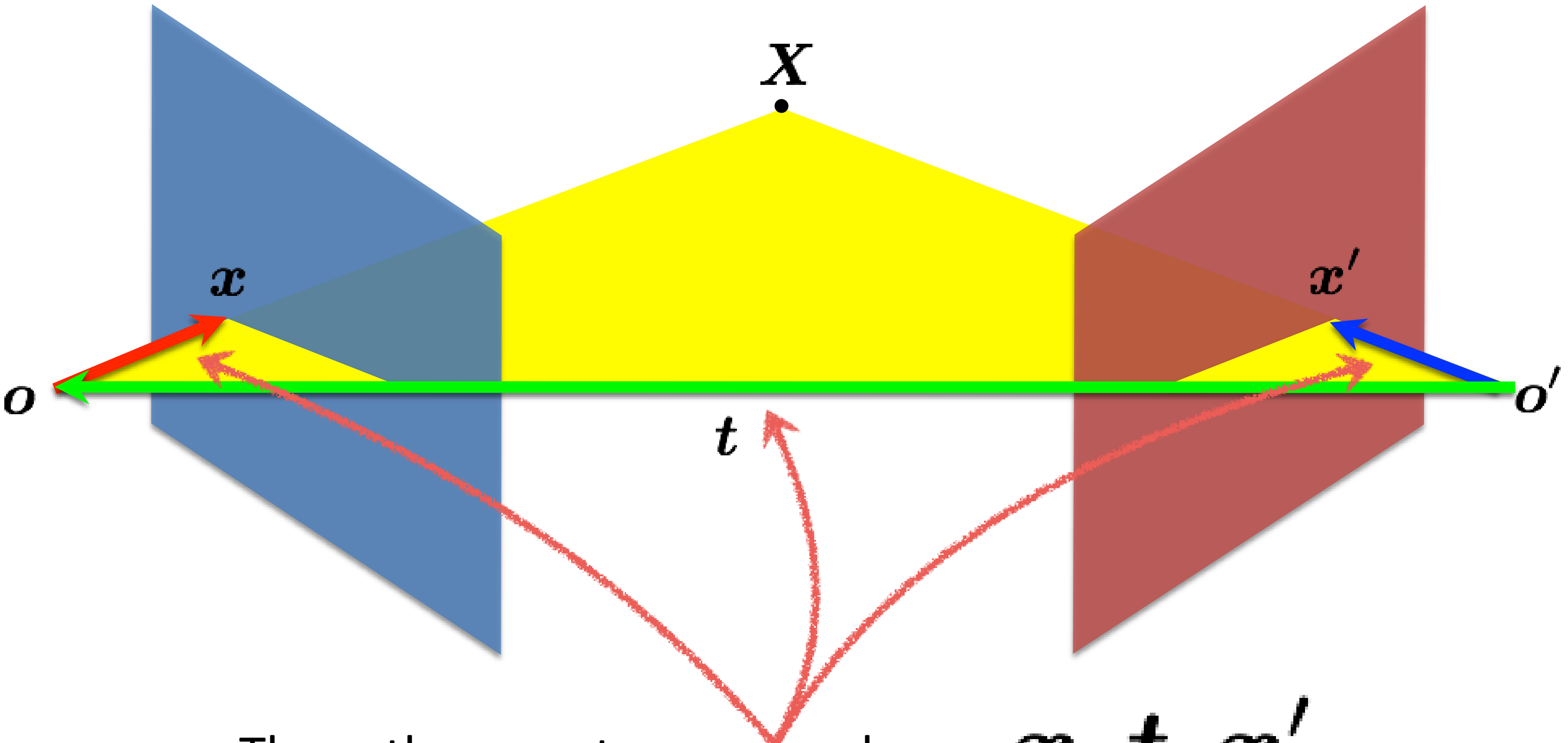
- Let's define the $\boldsymbol{o}$ system as world coordinate system.



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

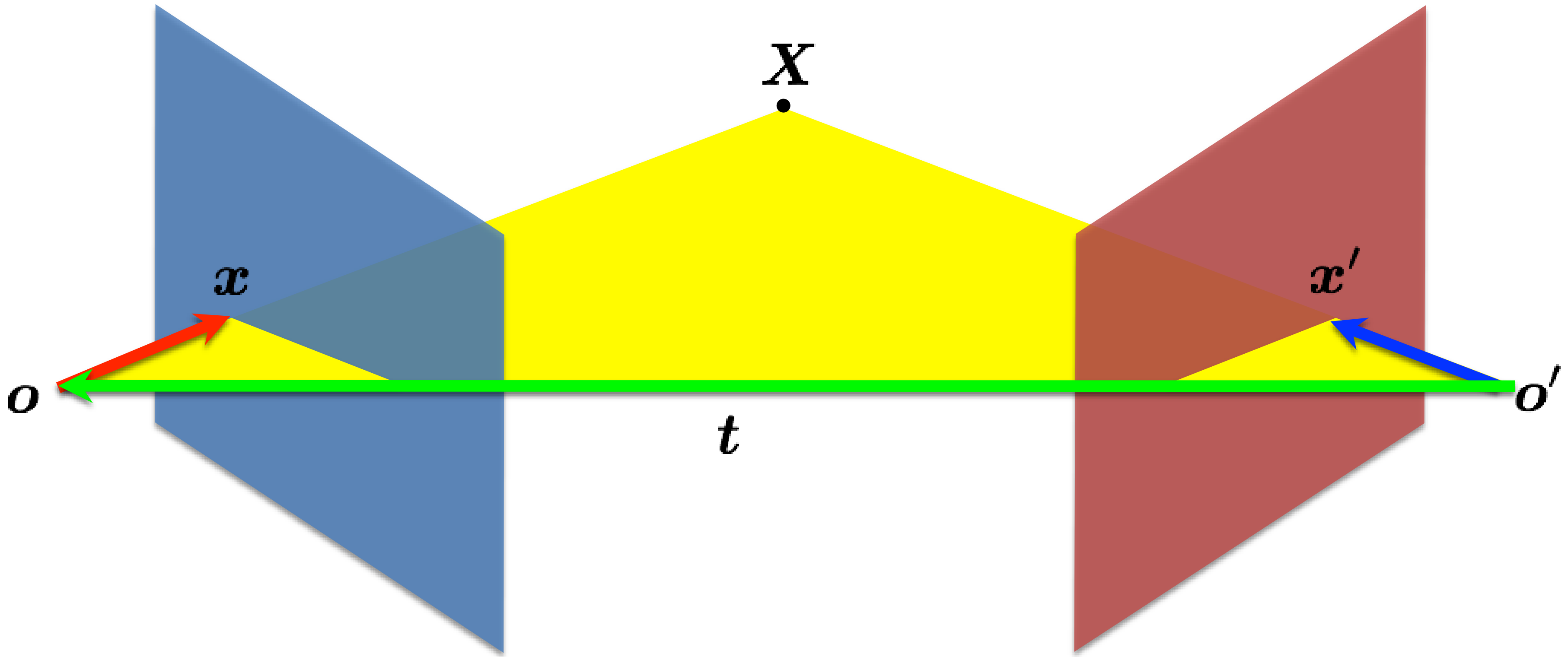
*t here is c from camera calibration class

Building the essential matrix



These three vectors are coplanar x, t, x'

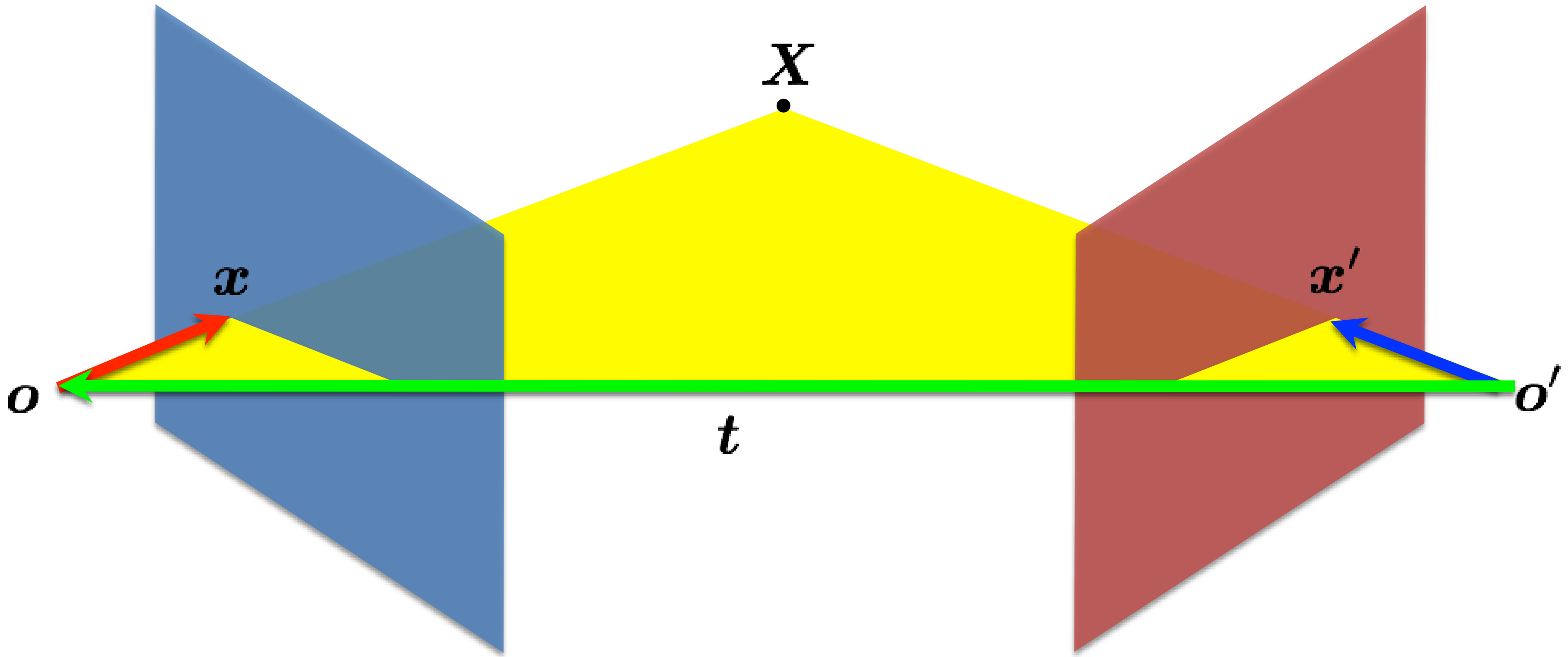
Building the essential matrix



These three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$

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

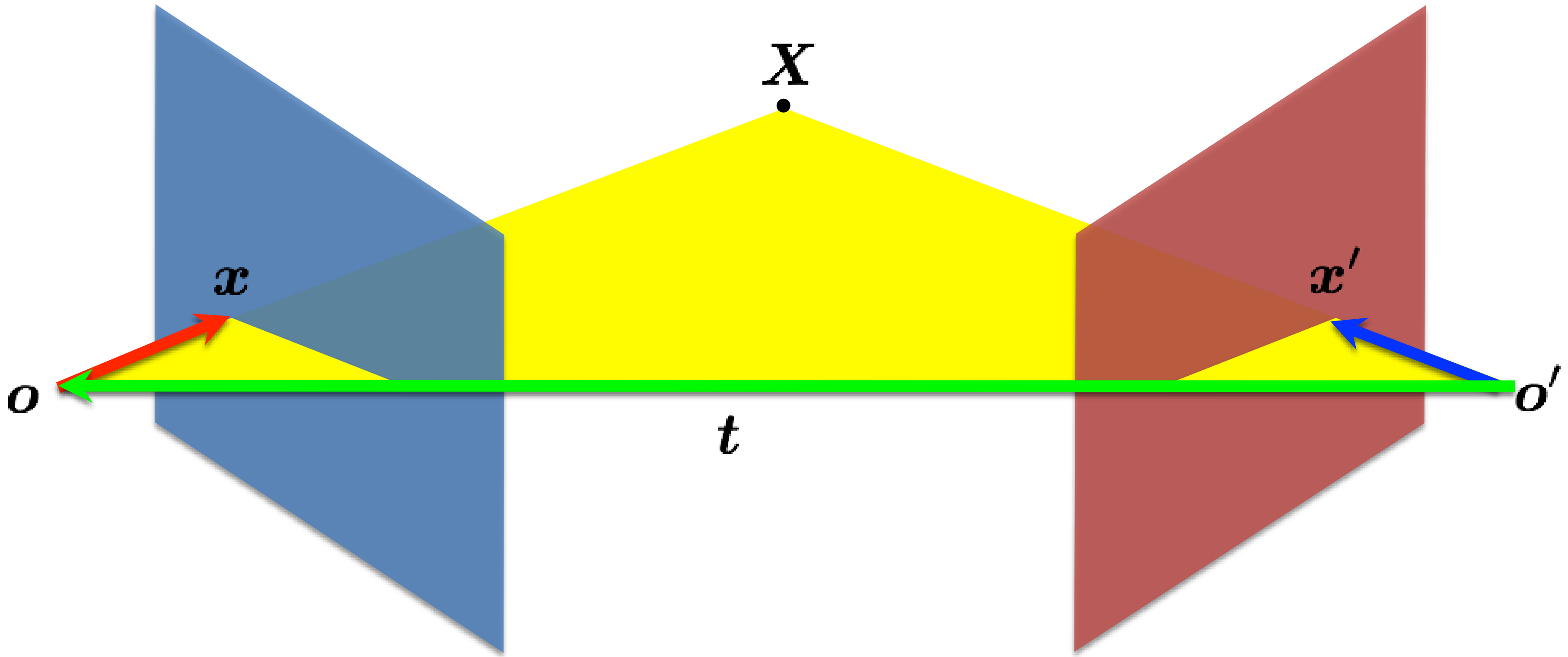
Building the essential matrix



These three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$

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

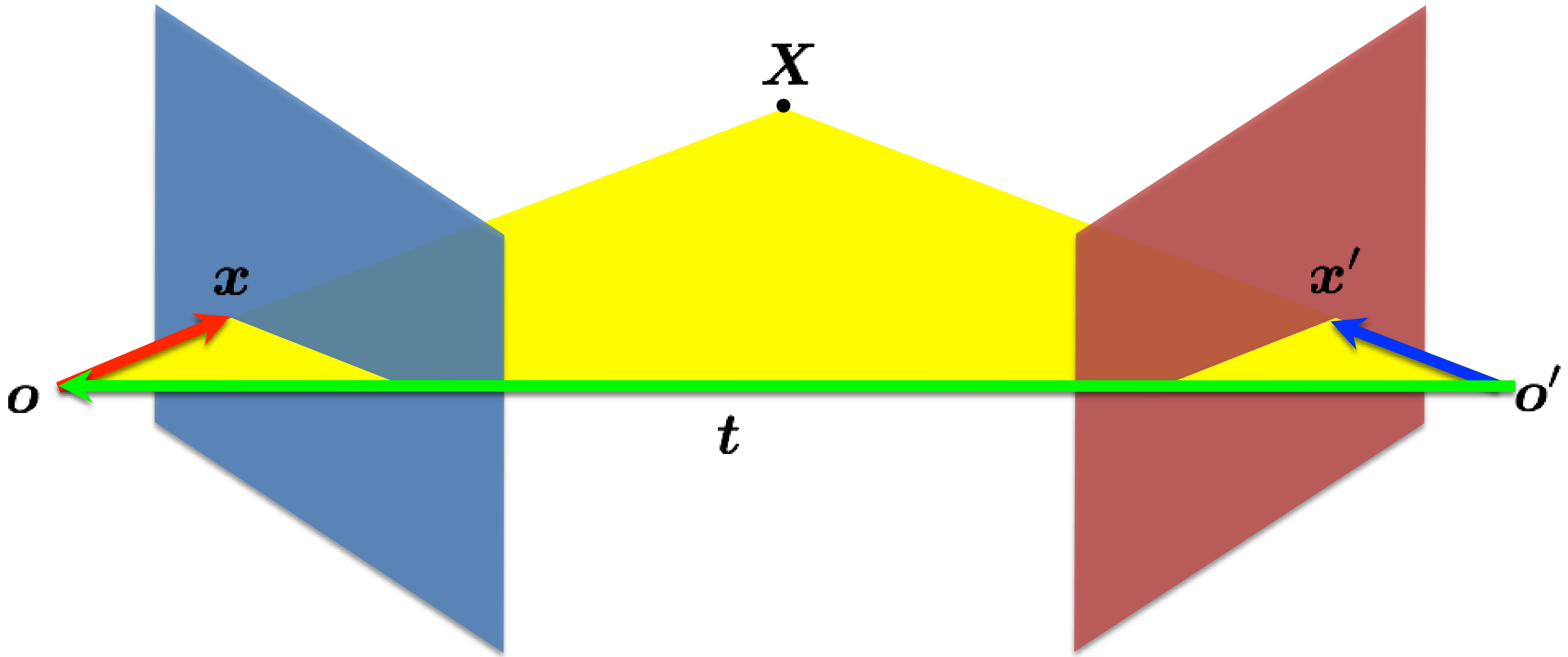
Building the essential matrix



These three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$

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

Building the essential matrix



These three vectors are coplanar $\boldsymbol{x}, \boldsymbol{t}, \boldsymbol{x}'$

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

Building the essential matrix

rigid motion

$$R^T = R^{-1}$$

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

$$\boldsymbol{R}^T \boldsymbol{x}' = \boldsymbol{x} - \boldsymbol{t}$$

$$\boldsymbol{x}'^T \boldsymbol{R} = (\boldsymbol{x} - \boldsymbol{t})^T$$

coplanarity

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

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

Building the essential matrix

rigid motion

$$R^T = R^{-1}$$

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

$$\boldsymbol{R}^T\boldsymbol{x}' = \boldsymbol{x} - \boldsymbol{t}$$

$$\boldsymbol{x}'^T\boldsymbol{R} = (\boldsymbol{x} - \boldsymbol{t})^T$$

coplanarity

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

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

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

Building the essential matrix

rigid motion

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

$$\boldsymbol{R}^T = \boldsymbol{R}^{-1}$$

$$\boldsymbol{R}^T \boldsymbol{x}' = \boldsymbol{x} - \boldsymbol{t}$$

$$\boldsymbol{x}'^T \boldsymbol{R} = (\boldsymbol{x} - \boldsymbol{t})^T$$

coplanarity

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

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

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

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

Building the essential matrix

rigid motion

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

$$\boldsymbol{R}^T = \boldsymbol{R}^{-1}$$

$$\boldsymbol{R}^T \boldsymbol{x}' = \boldsymbol{x} - \boldsymbol{t}$$

$$\boldsymbol{x}'^T \boldsymbol{R} = (\boldsymbol{x} - \boldsymbol{t})^T$$

coplanarity

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

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

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

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

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

Building the essential matrix

rigid motion

$$R^T = R^{-1}$$

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

$$\boldsymbol{R}^T\boldsymbol{x}' = \boldsymbol{x} - \boldsymbol{t}$$

$$\boldsymbol{x}'^T\boldsymbol{R} = (\boldsymbol{x} - \boldsymbol{t})^T$$

coplanarity

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

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

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

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

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

Essential Matrix

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

Contents

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

Fundamental matrix

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

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

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

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

Fundamental matrix

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

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

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

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

camera point

image point

Writing out the epipolar constraint in terms of image coordinates

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

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

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

Fundamental Matrix

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

Contents

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

Estimating F

- Given enough correspondence point between the two images, one can reconstruct the fundamental matrix $\boldsymbol{F}$.
- If $\boldsymbol{K_1}, \boldsymbol{K_2}$ are known, we can find $\boldsymbol{E}$.
  - We can then decompose $\boldsymbol{E}$ to $\boldsymbol{R}, t$ between the two images (This part is out of scope for this lecture).
  - t is found up to a scale in the estimation but it's easy to get a good measure of it with a ruler.

Estimating F – 8-point algorithm

- 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 x=$(u,v,1)^{\mathrm{T}}$ and 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

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

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 $\mathbf{A}^{\mathrm{T}}\mathbf{A}$.

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

8-point algorithm – Problem?

- **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

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

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

8-point algorithm

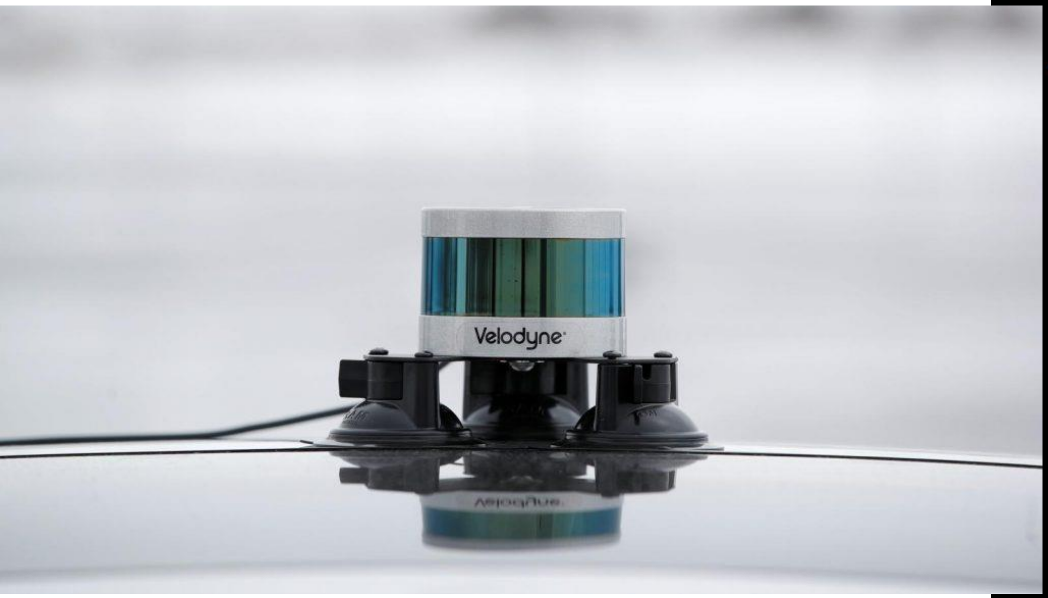
- Pros: it is linear, easy to implement and fast
- Cons: susceptible to noise.
  - Solutions: (all out of scope)
    - normalized 8 points algorithm.
    - 7 points algorithm.
    - Finding K,K’ with single camera intrinsics calibration and then search for E (only 5 DOFs instead of 8/7).

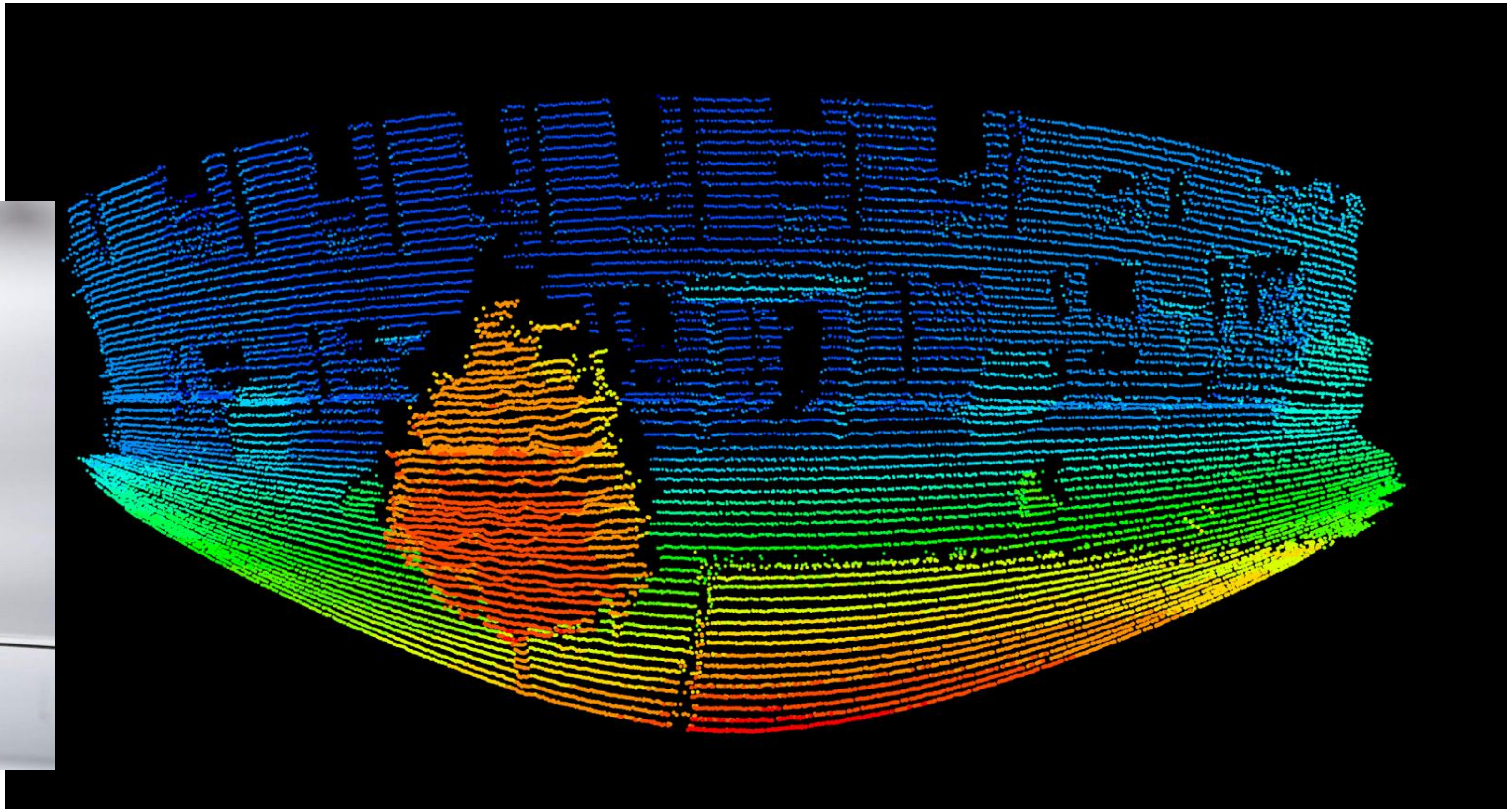
Contents

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

Other types of 3D sensors

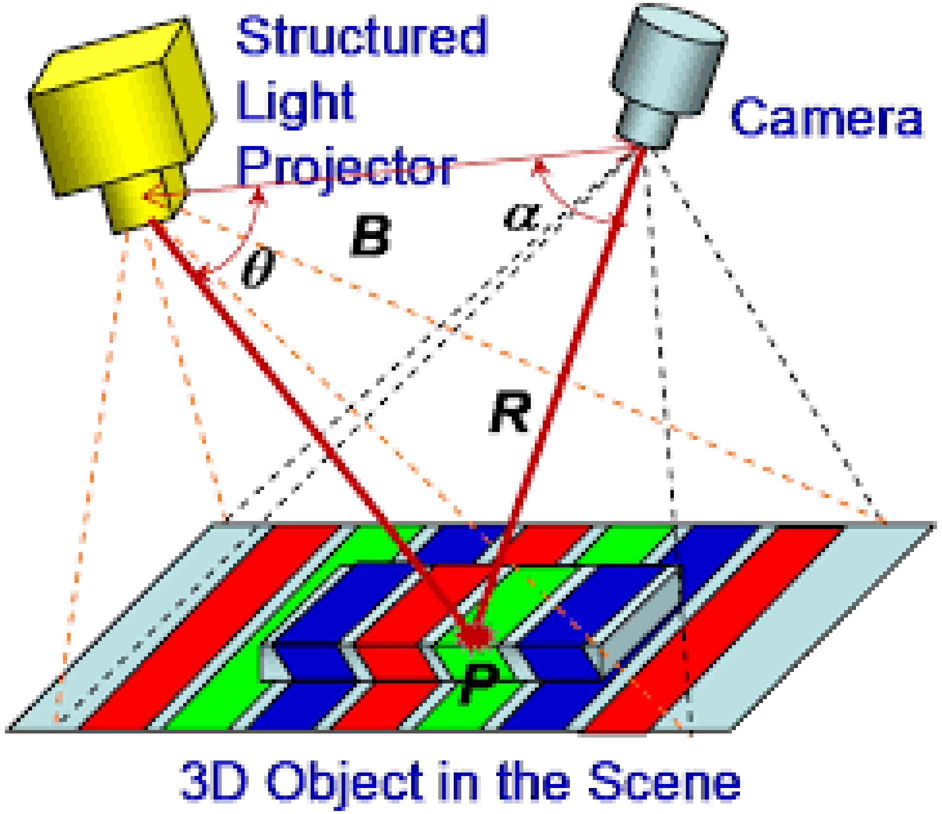
- **LIDAR**, which stands for **Light Detection and Ranging (or light radar),** is a remote sensing method that uses light in the form of a pulsed laser to measure ranges.
- Most known: velodyne projector.

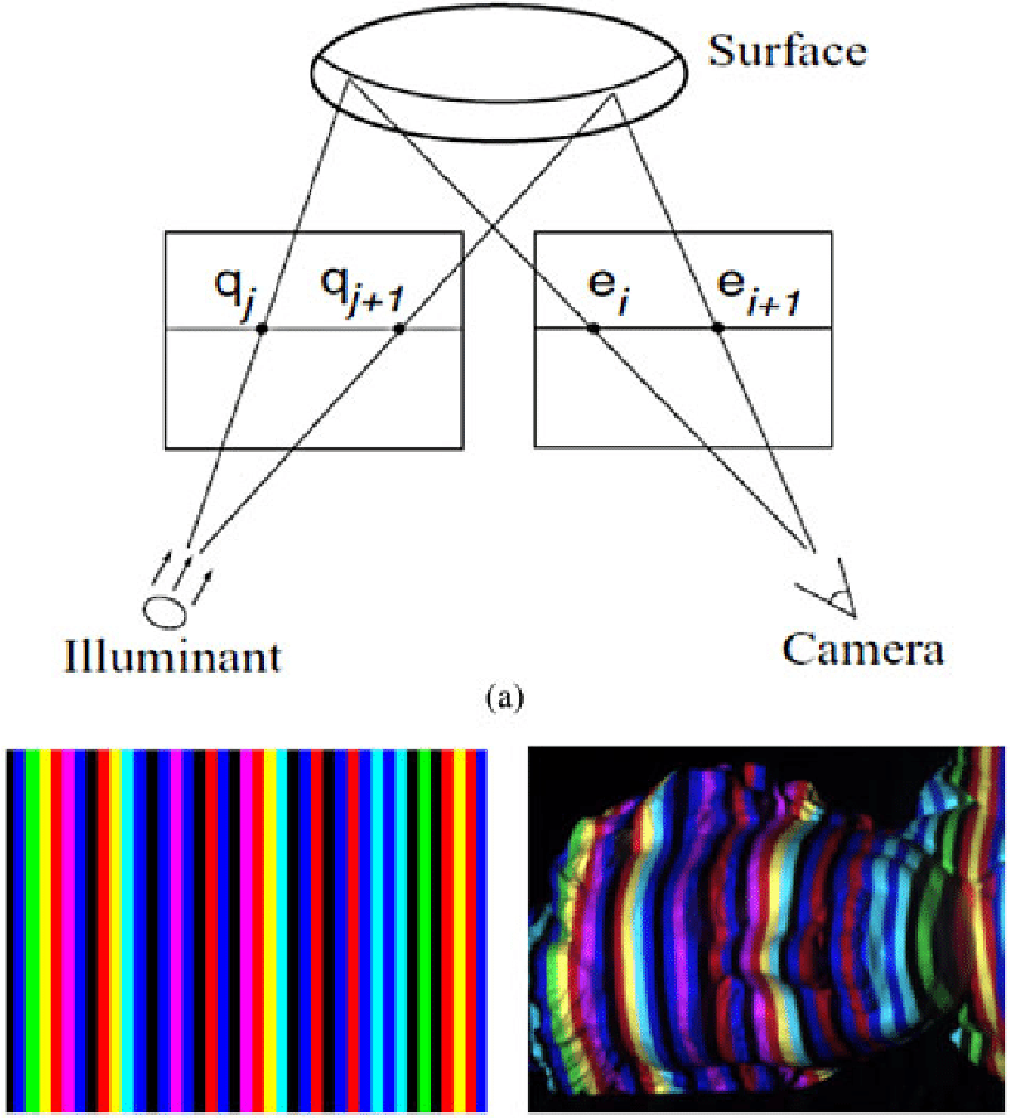




Other types of 3D sensors

- Structured light





(a)

(b)

(c)

Other types of 3D sensors

- Coded light
- Realsense SR305
- https://www.youtube.com/watch?v=PluL7WTlKrM

Other types of 3D sensors

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