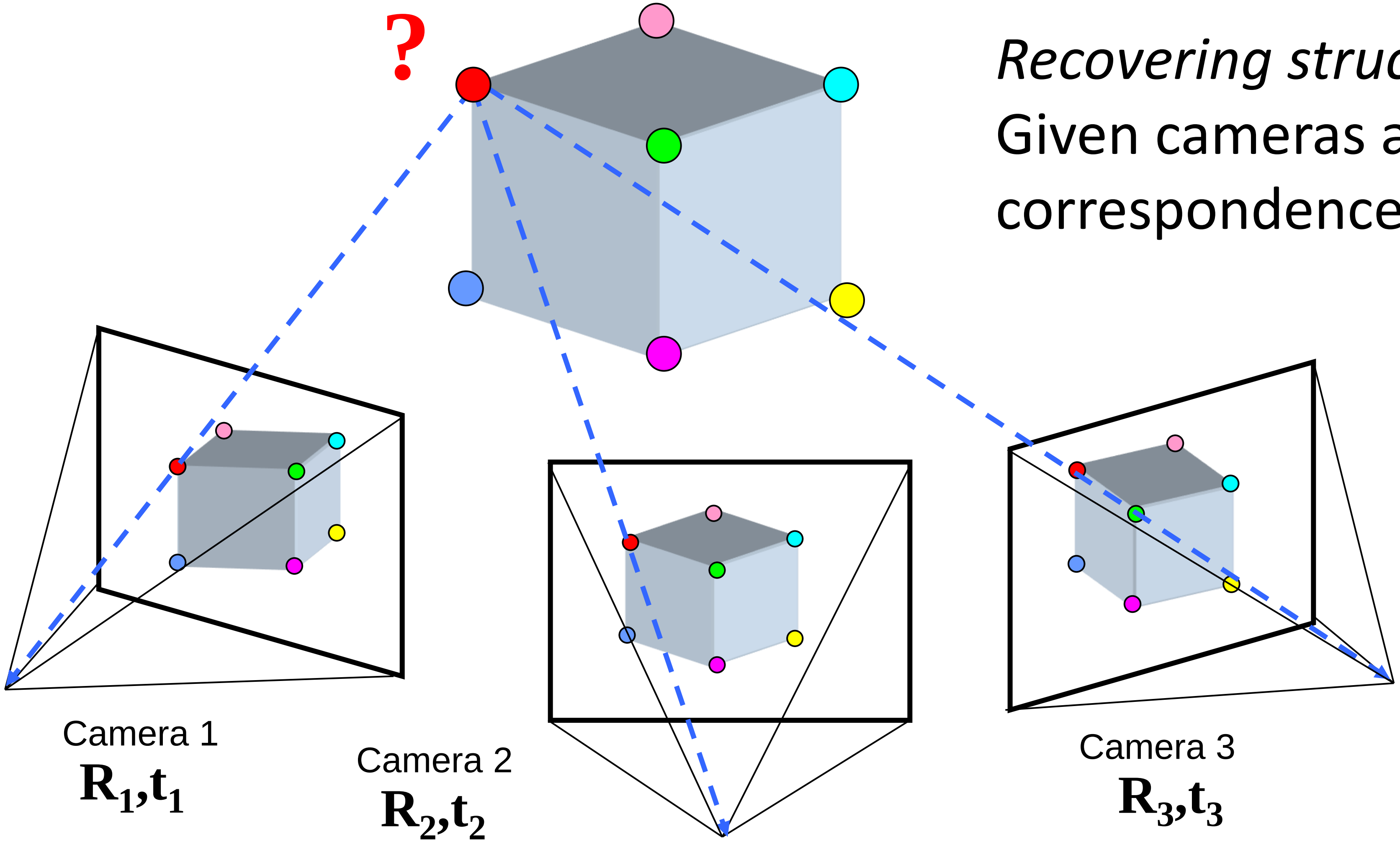
Multiple View Geometry

Computer Vision – Lecture 15

Further Reading

- 7 lectures from S Lazebnik
- Slides from D Fouhey and J Johnson
- Many slides adapted from both sources

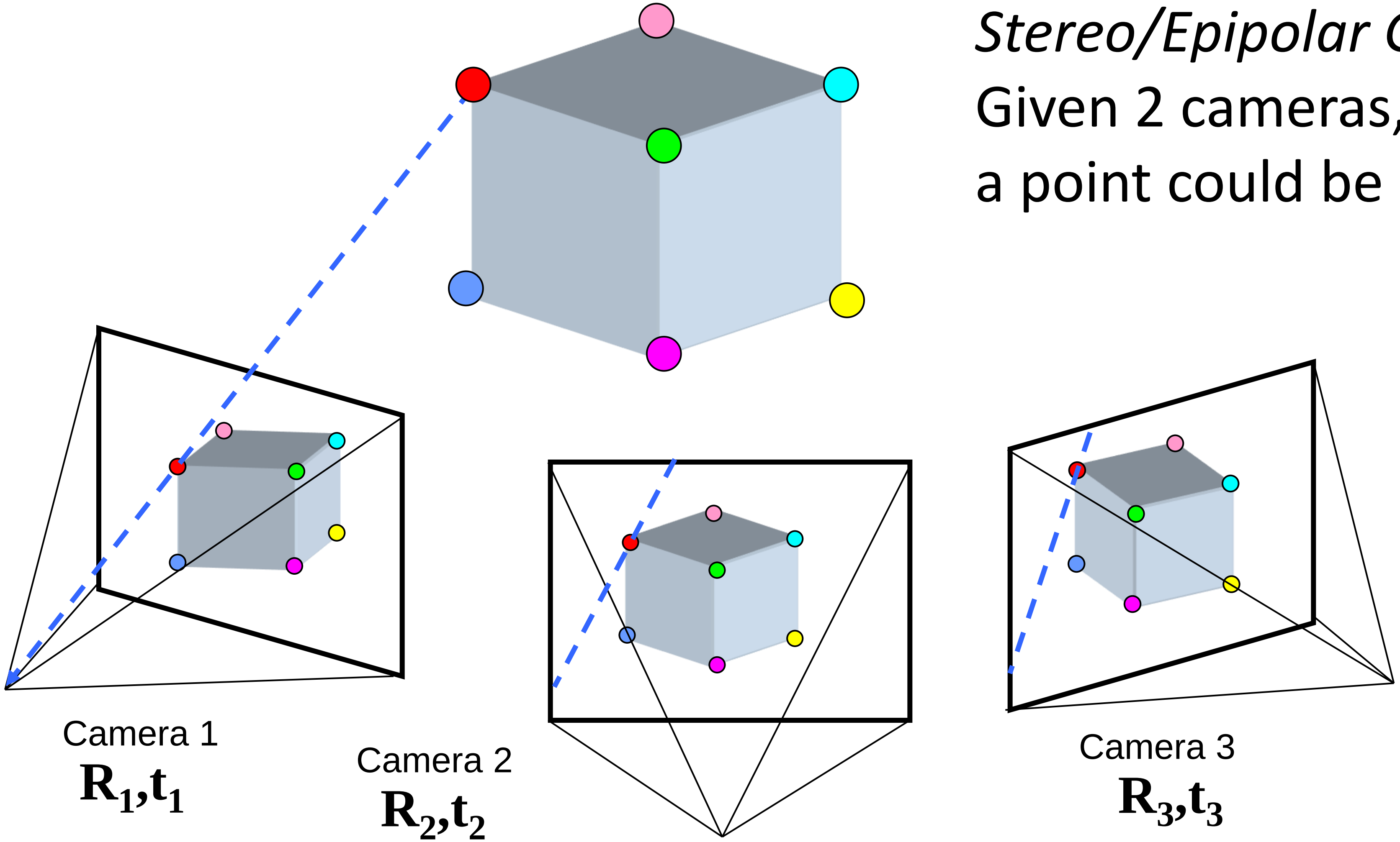
Multi-view geometry problems



Recovering structure: Given cameras and correspondences, find 3D.

Slide credit:
Noah Snavely

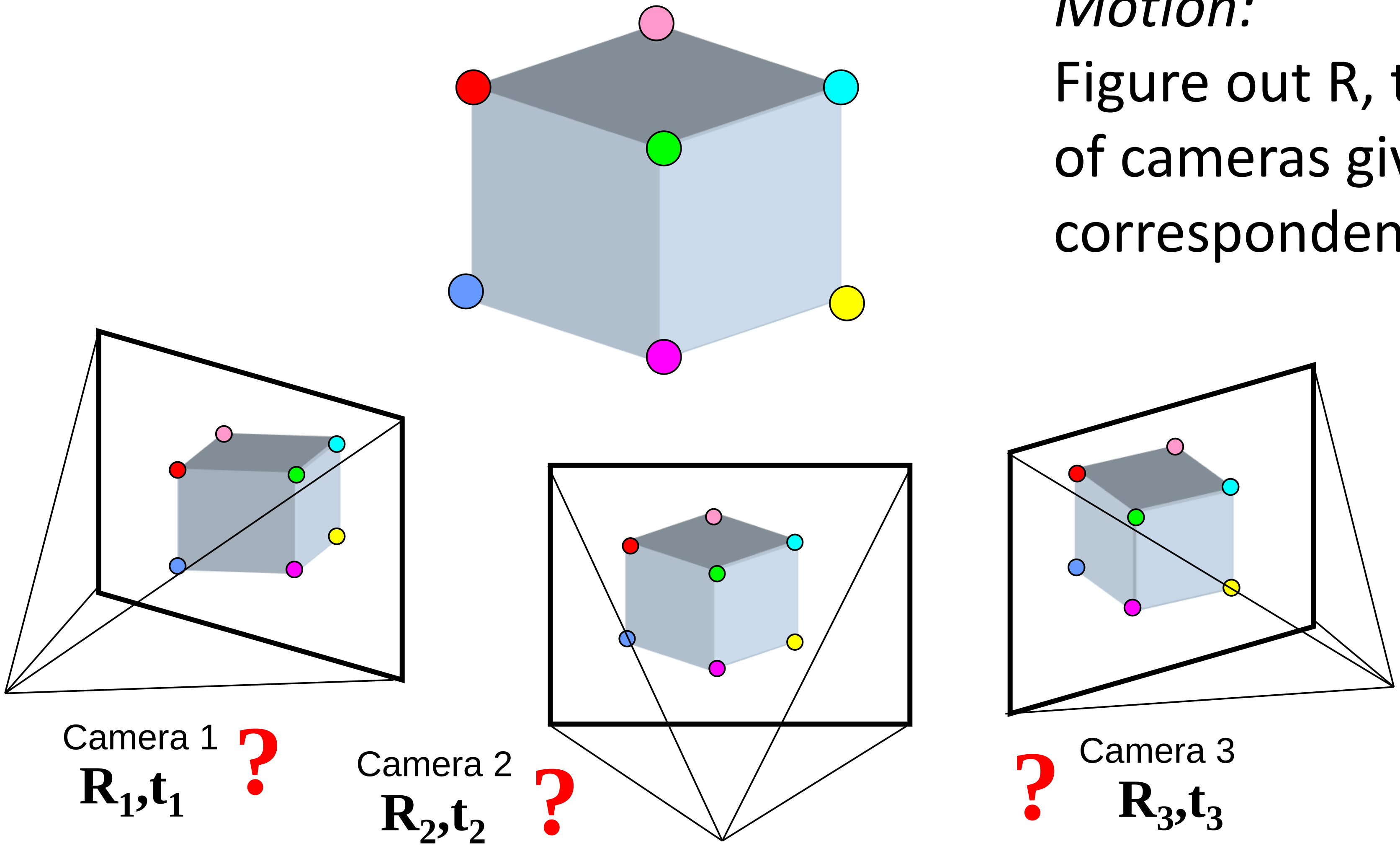
Multi-view geometry problems



Stereo/Epipolar Geomery:
Given 2 cameras, find where a point could be

Slide credit:
Noah Snavely

Multi-view geometry problems



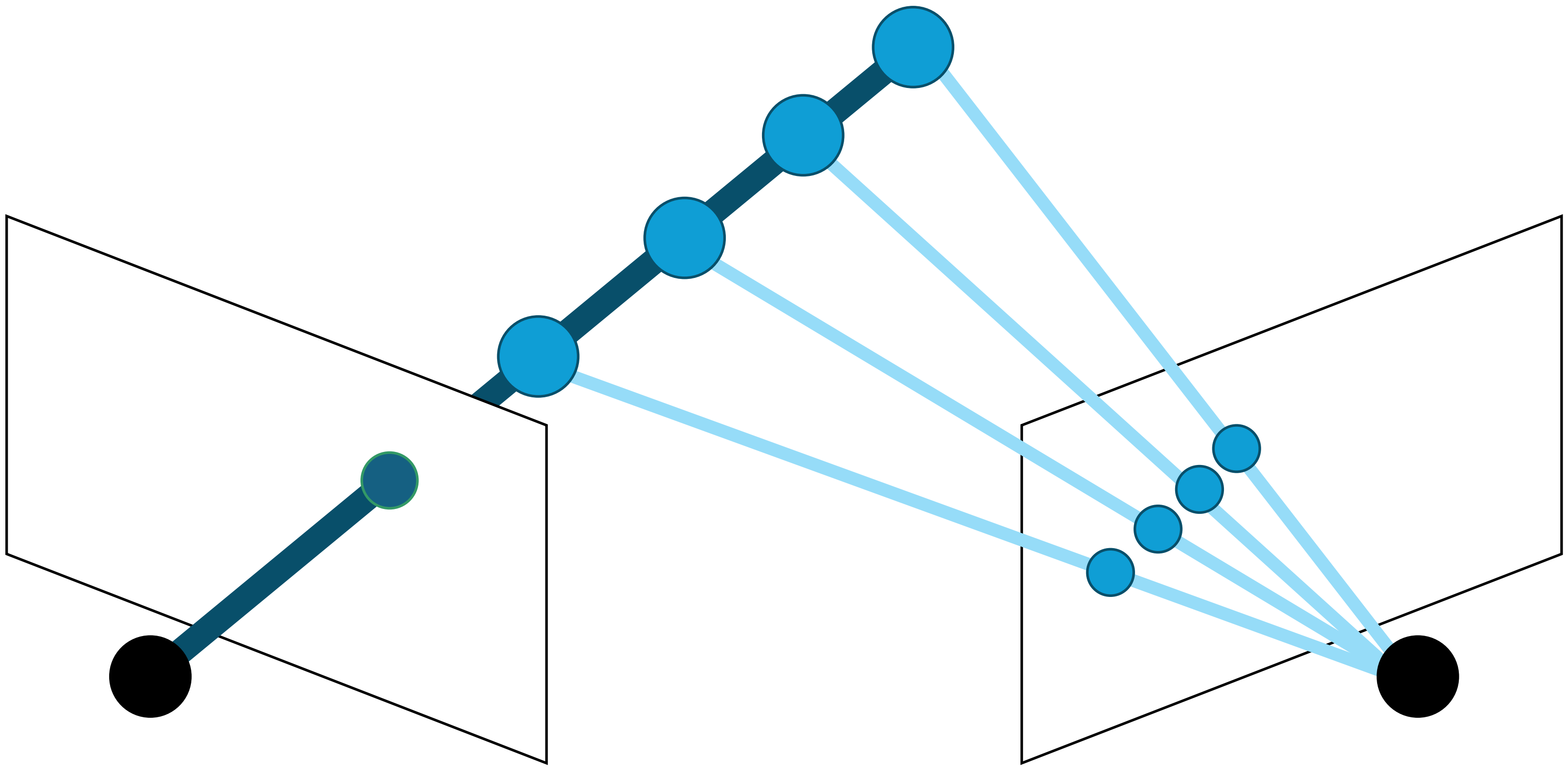
Motion:

Figure out R, t for a set of cameras given correspondences

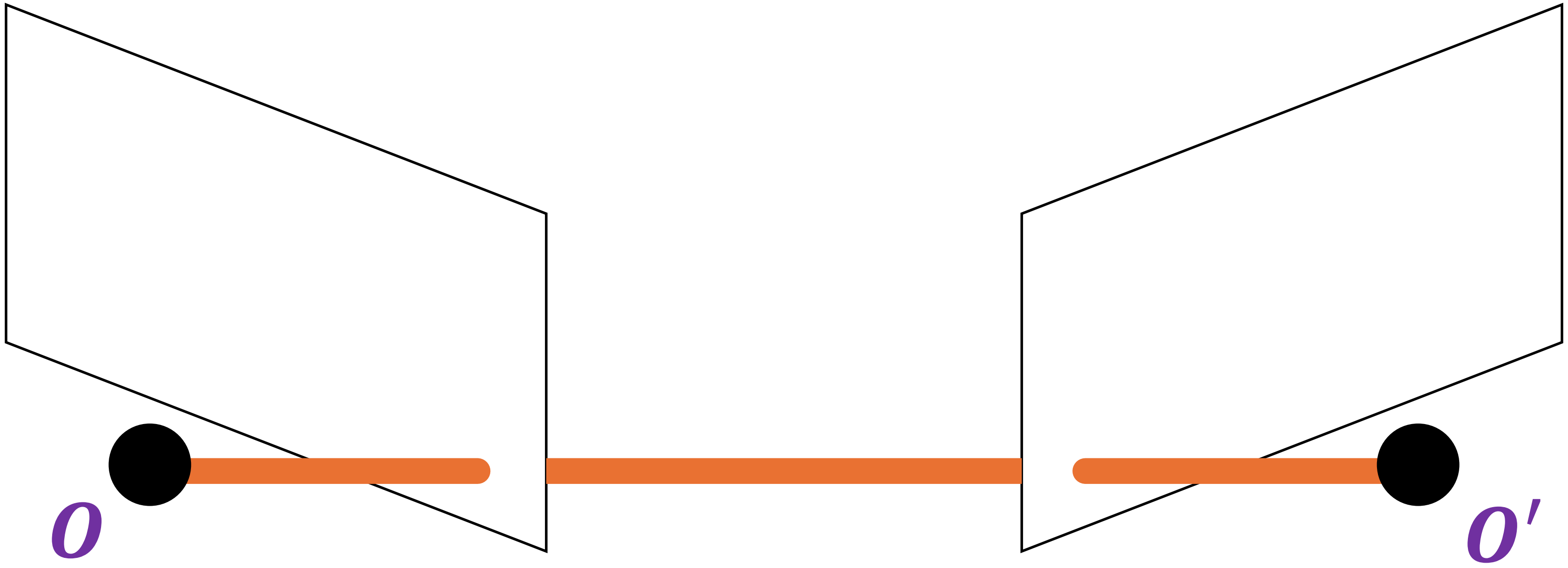
Slide credit: Noah Snavely

Epipolar constraint

Where can we find the x' corresponding to x in the other image?

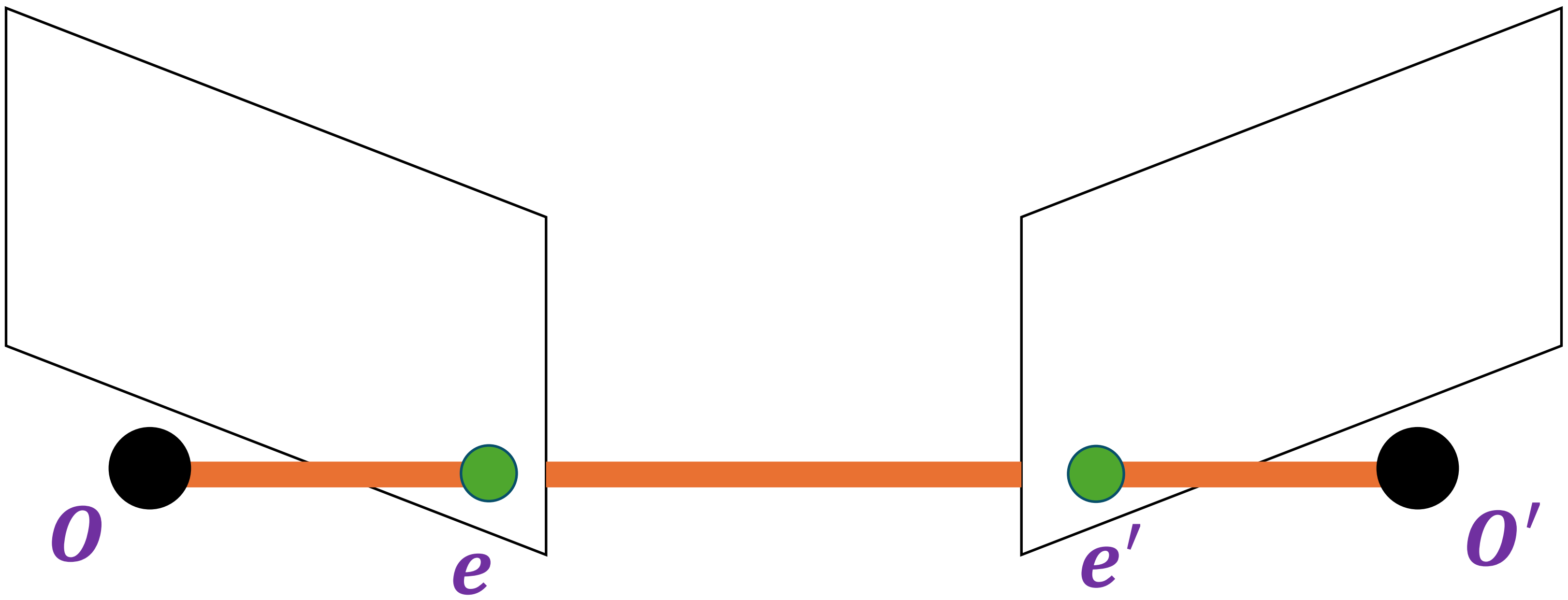


Epipolar geometry setup



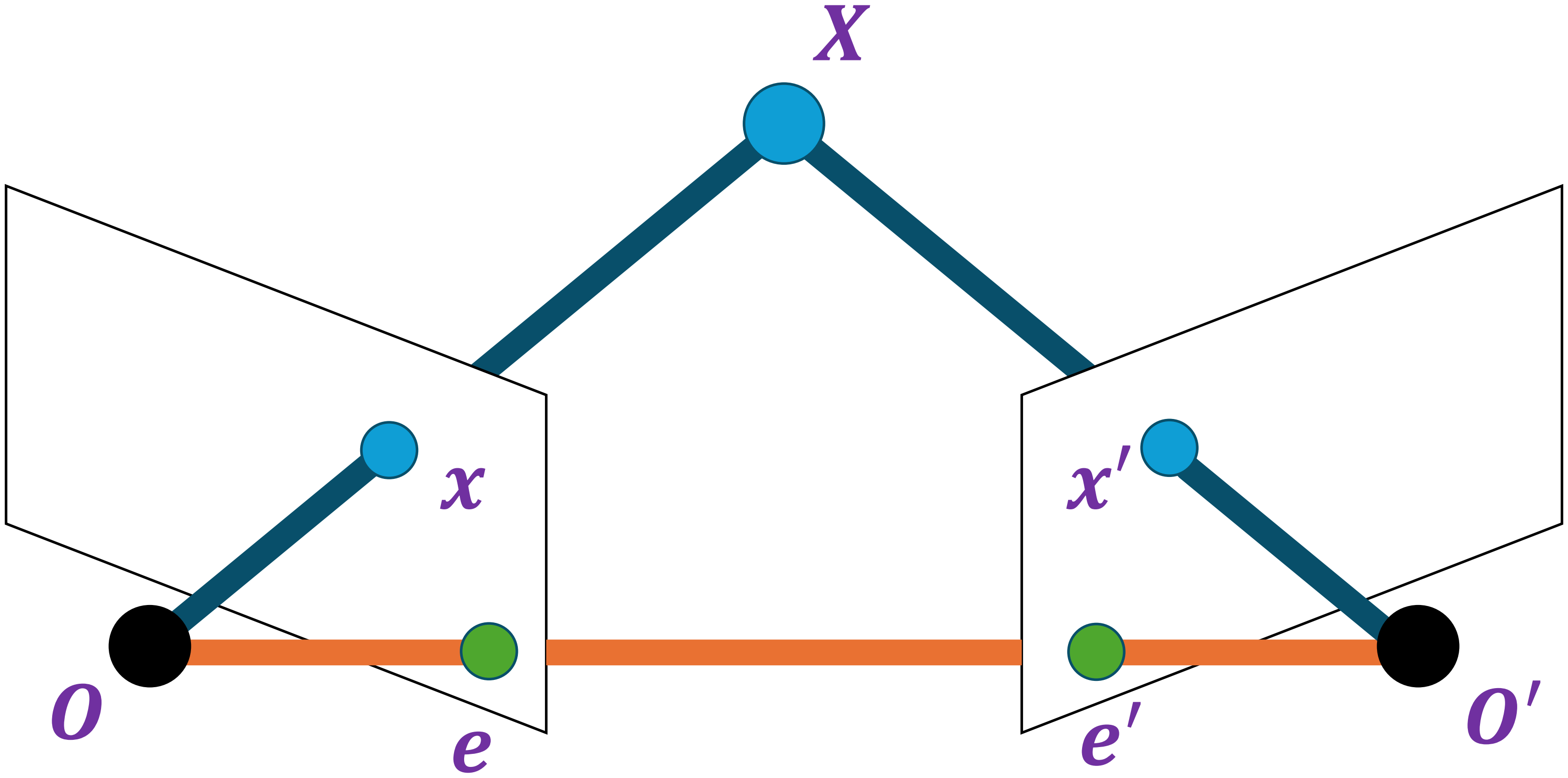
- Suppose we have two cameras with centers $\boldsymbol{O}$, $\boldsymbol{O}'$
- The **baseline** is the line connecting the origins

Epipolar geometry setup



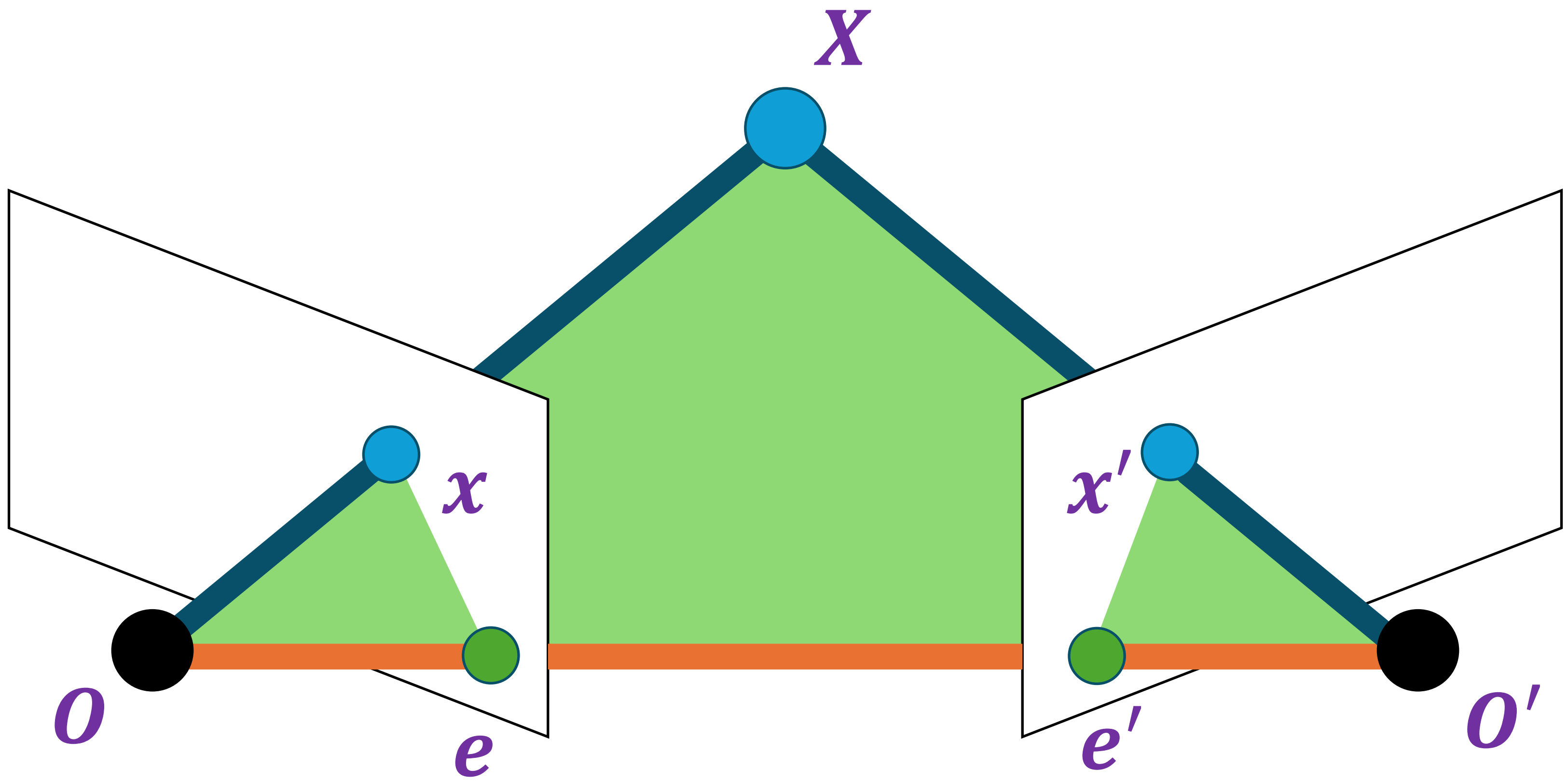
Epipoles e, e' are where the baseline intersects the image planes, or projections of the other camera in each view

Epipolar geometry setup



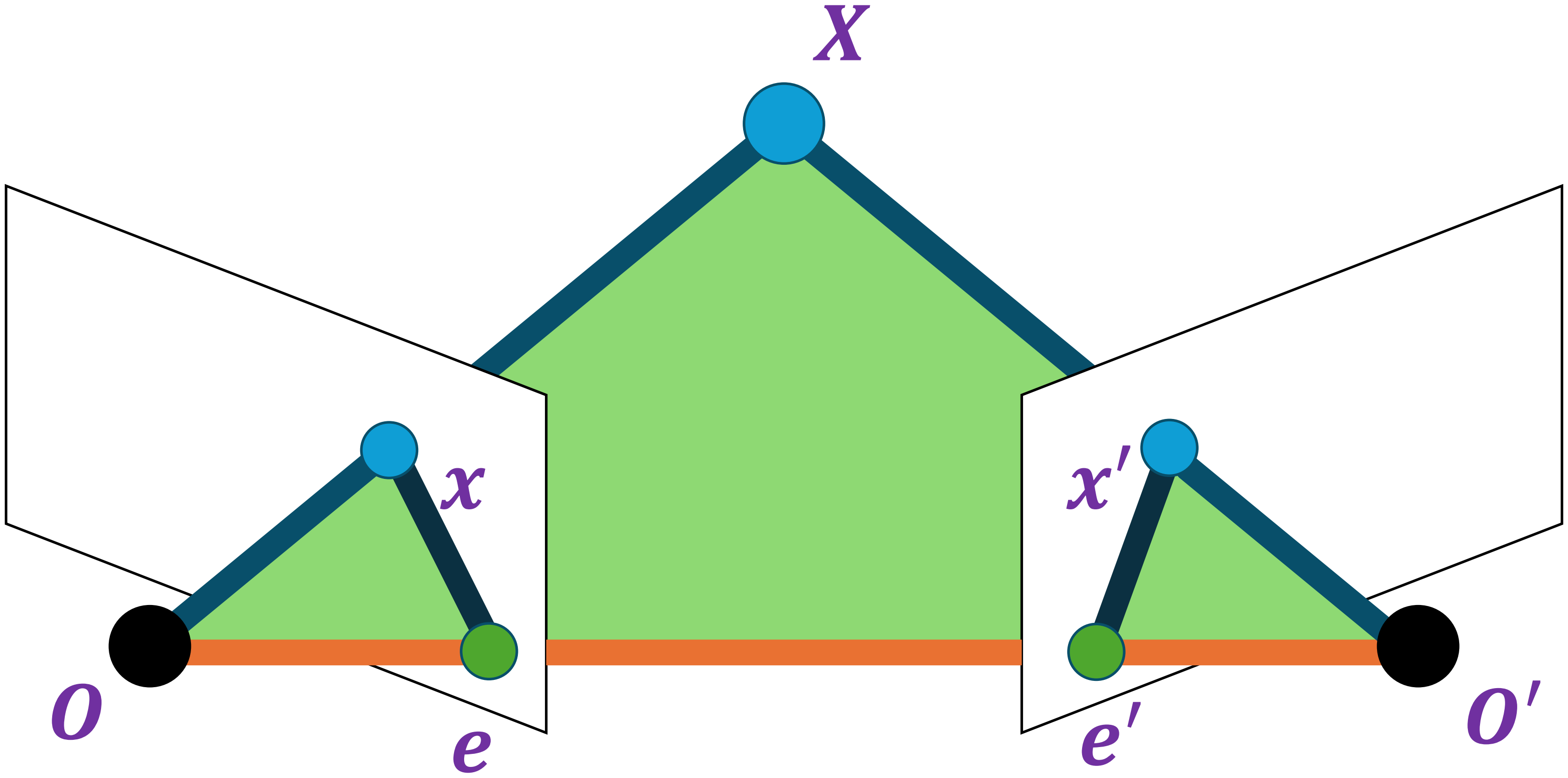
- Consider a **point** $\boldsymbol{X}$, which projects to $\boldsymbol{x}$ and $\boldsymbol{x'}$

Epipolar geometry setup



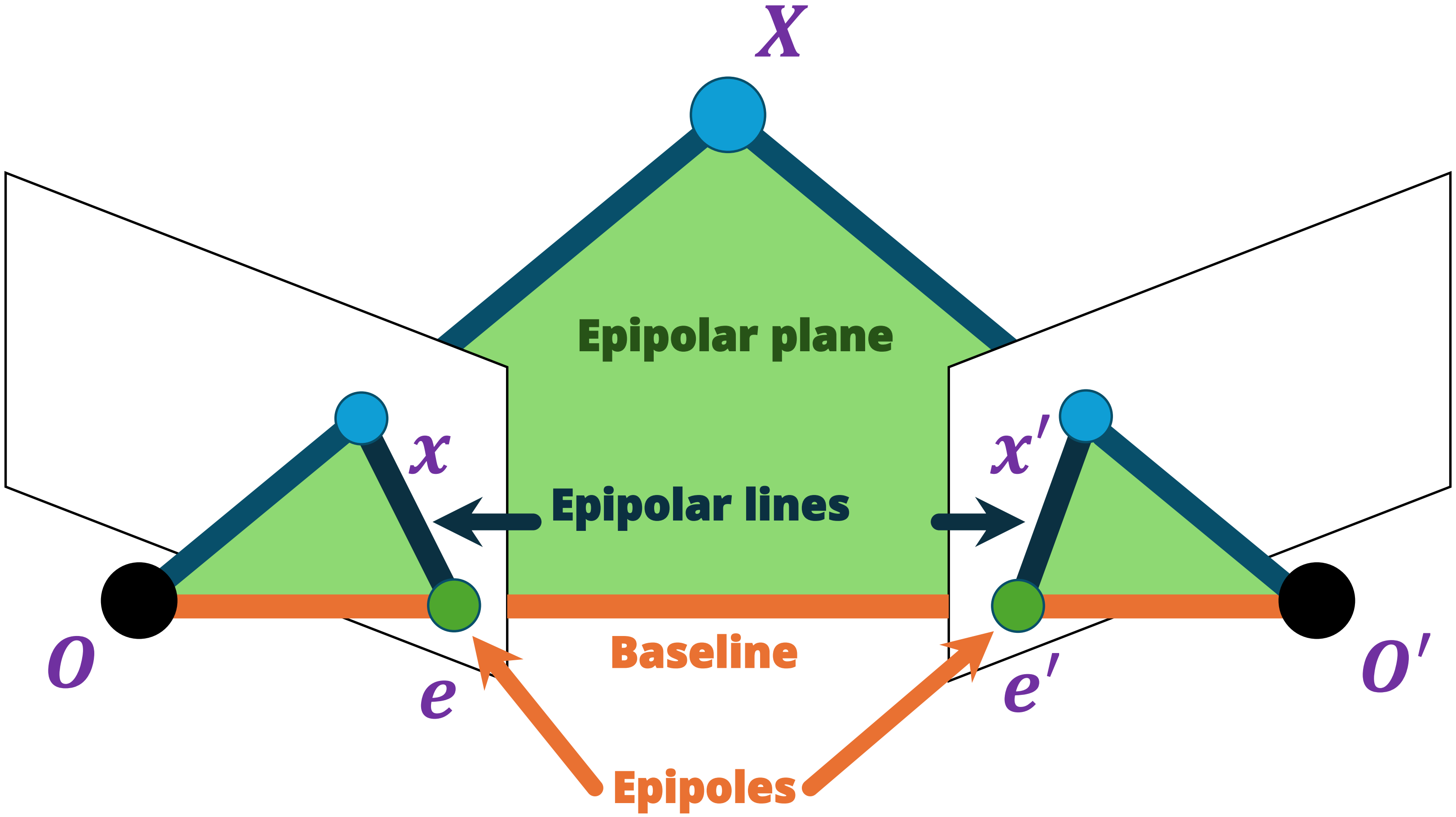
- The plane formed by $\boldsymbol{X}$, $\boldsymbol{O}$, and $\boldsymbol{O'}$ is called an **epipolar plane**
- There is a family of planes passing through $\boldsymbol{O}$ and $\boldsymbol{O'}$

Epipolar geometry setup

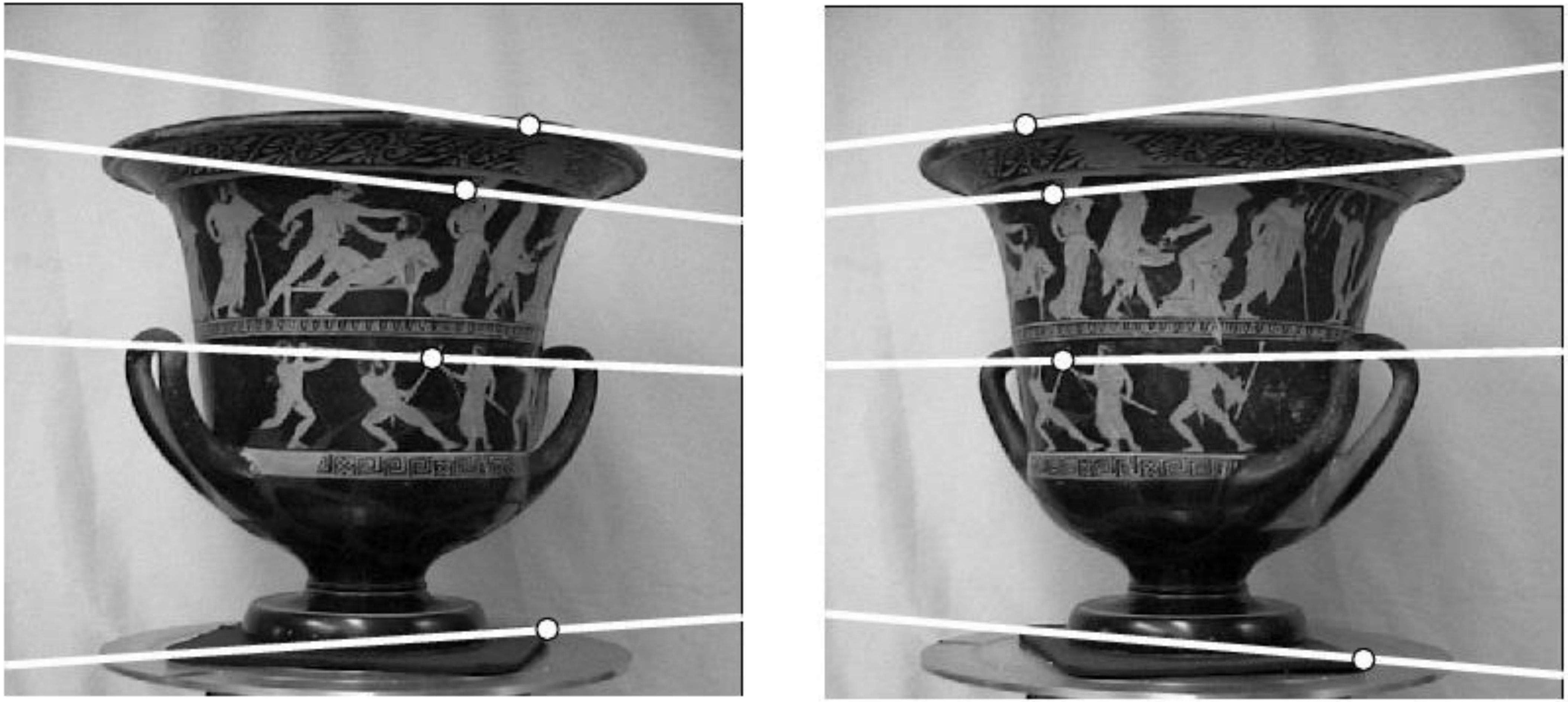


- **Epipolar lines** connect the epipoles to the projections of $\boldsymbol{X}$
- Equivalently, they are intersections of the epipolar plane with the image planes – thus, they come in matching pairs

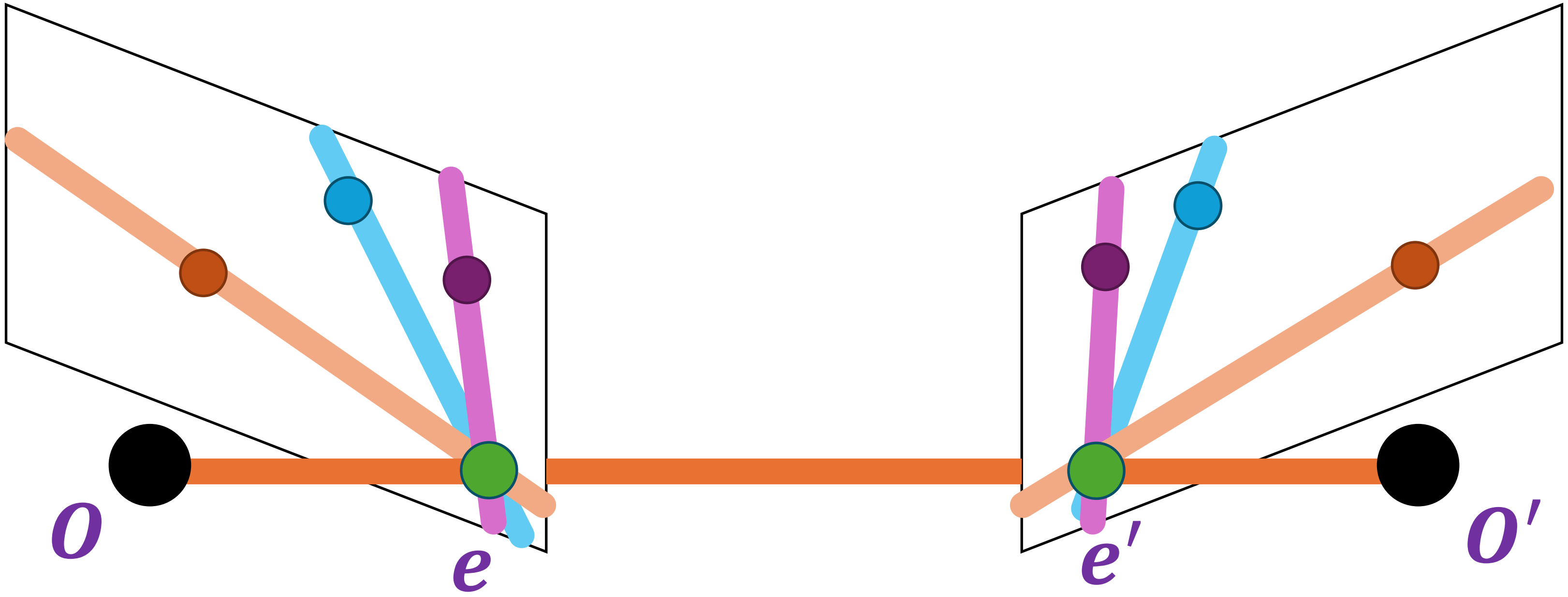
Epipolar geometry setup: Summary



Example: Converging cameras

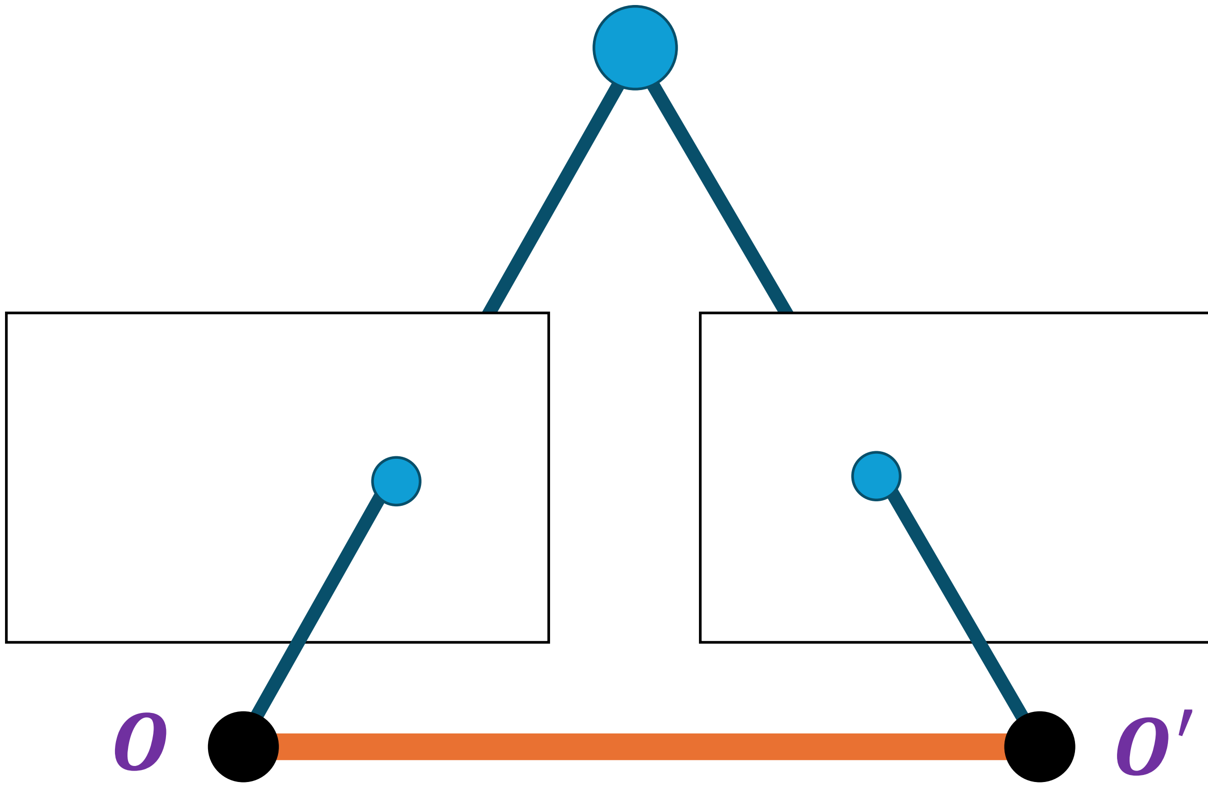


Example: Converging cameras



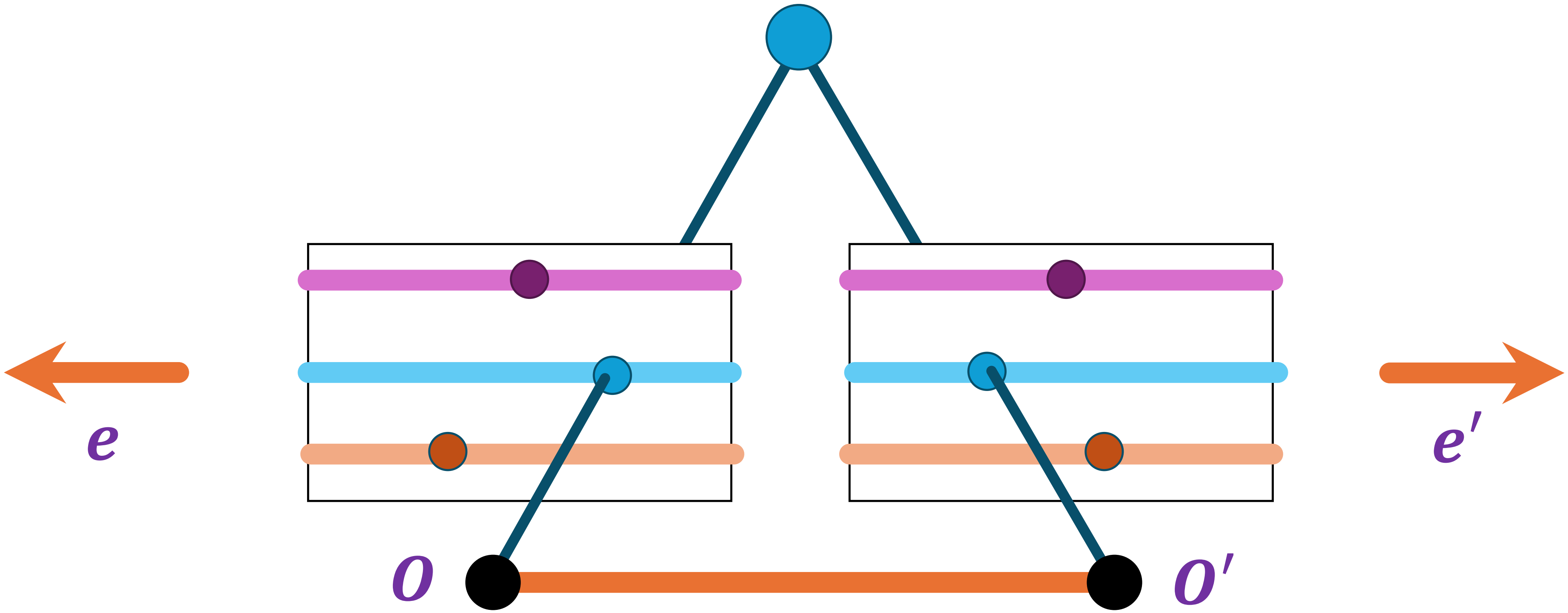
- Epipoles are finite, may be visible in the image

Example: Motion parallel to image plane



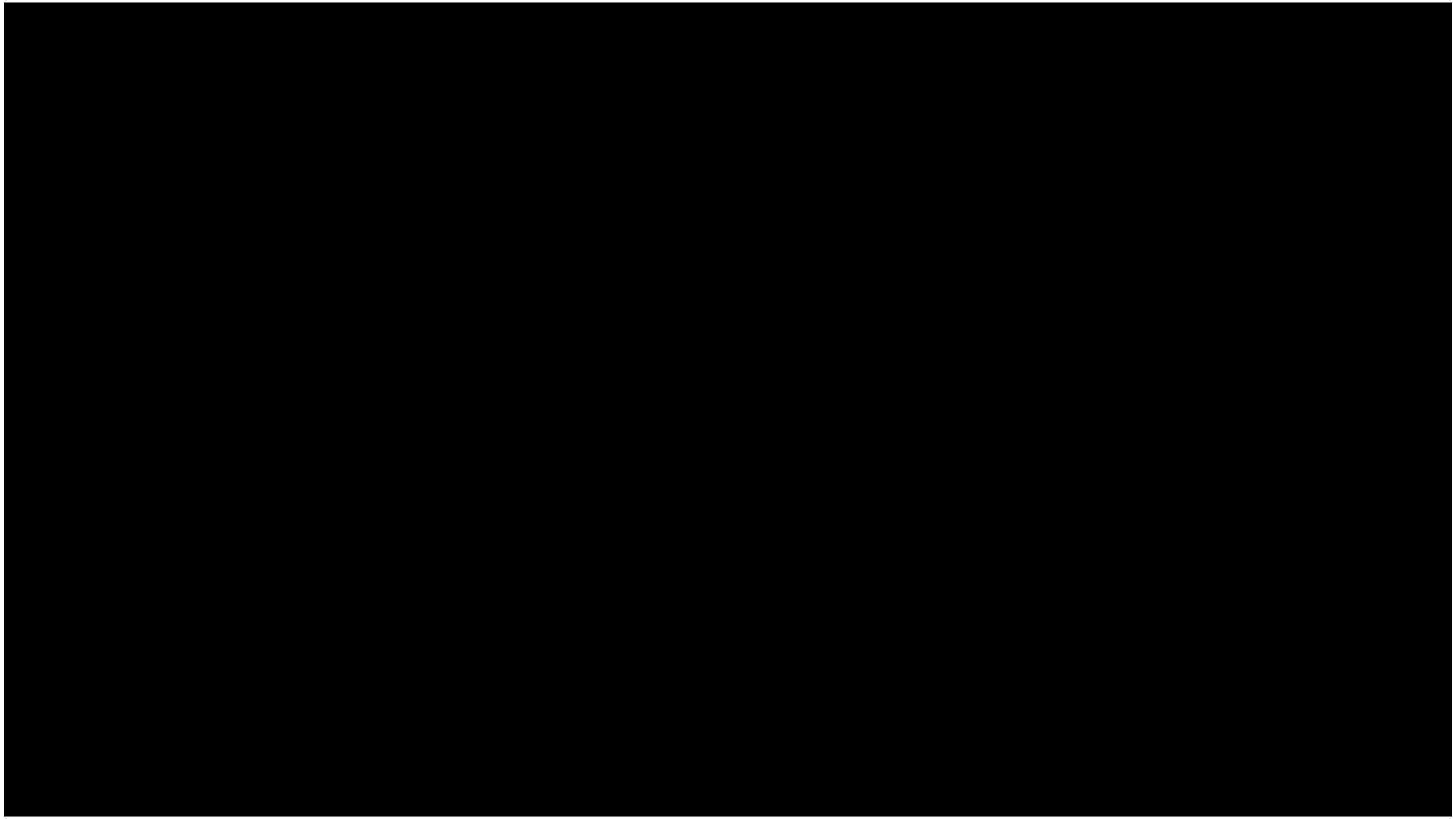
- Where are the epipoles and what do the epipolar lines look like?

Example: Motion parallel to image plane



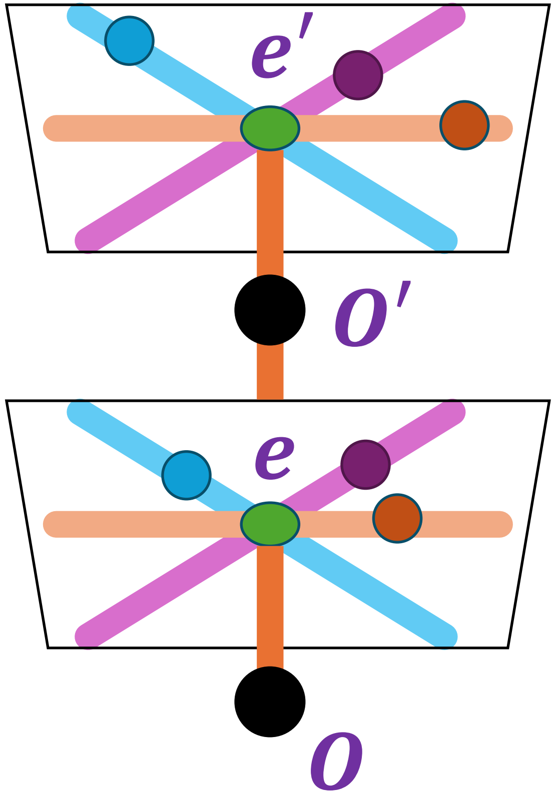
- Epipoles ***infinitely*** far away, epipolar lines parallel

Example: Motion perpendicular to image plane



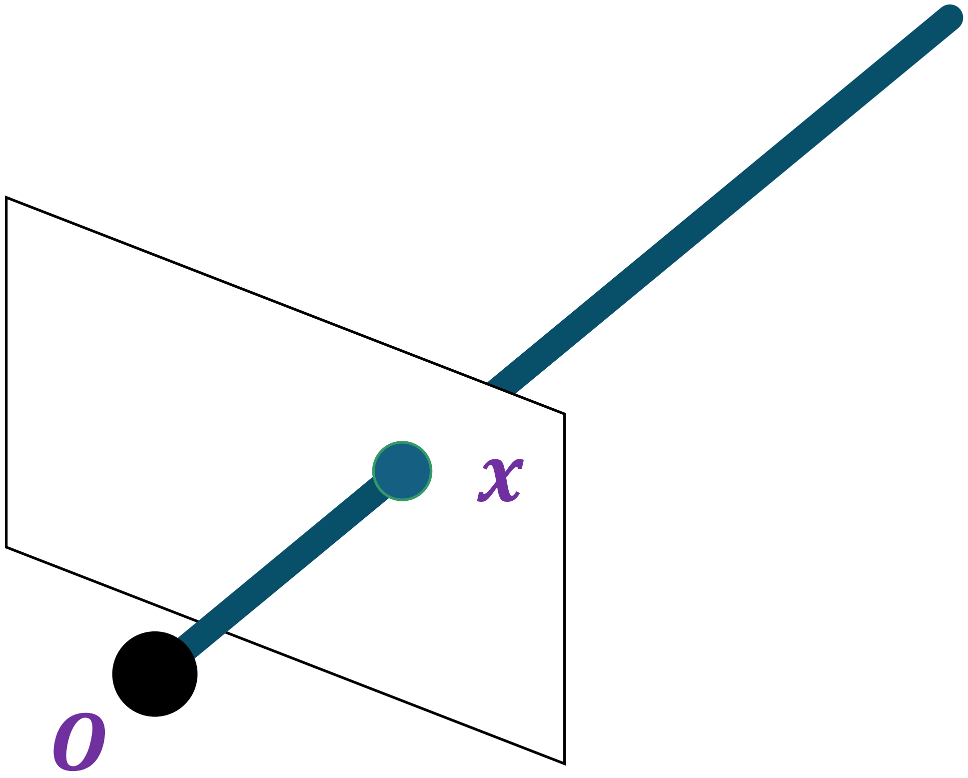
http://vimeo.com/48425421

Example: Motion perpendicular to image plane



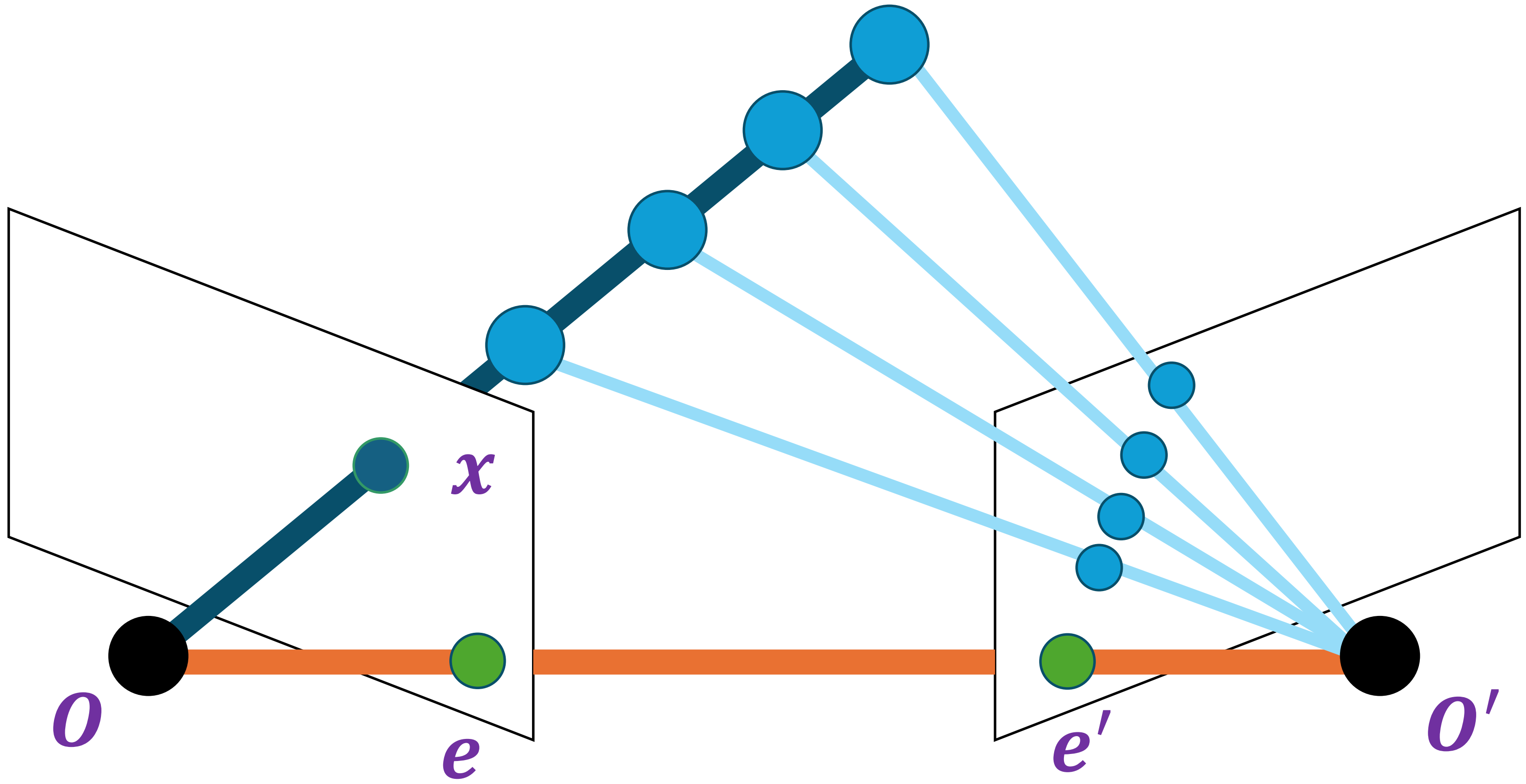
- Epipole is "focus of expansion" and coincides with the principal point of the camera
- Epipolar lines go out from principal point

Epipolar constraint



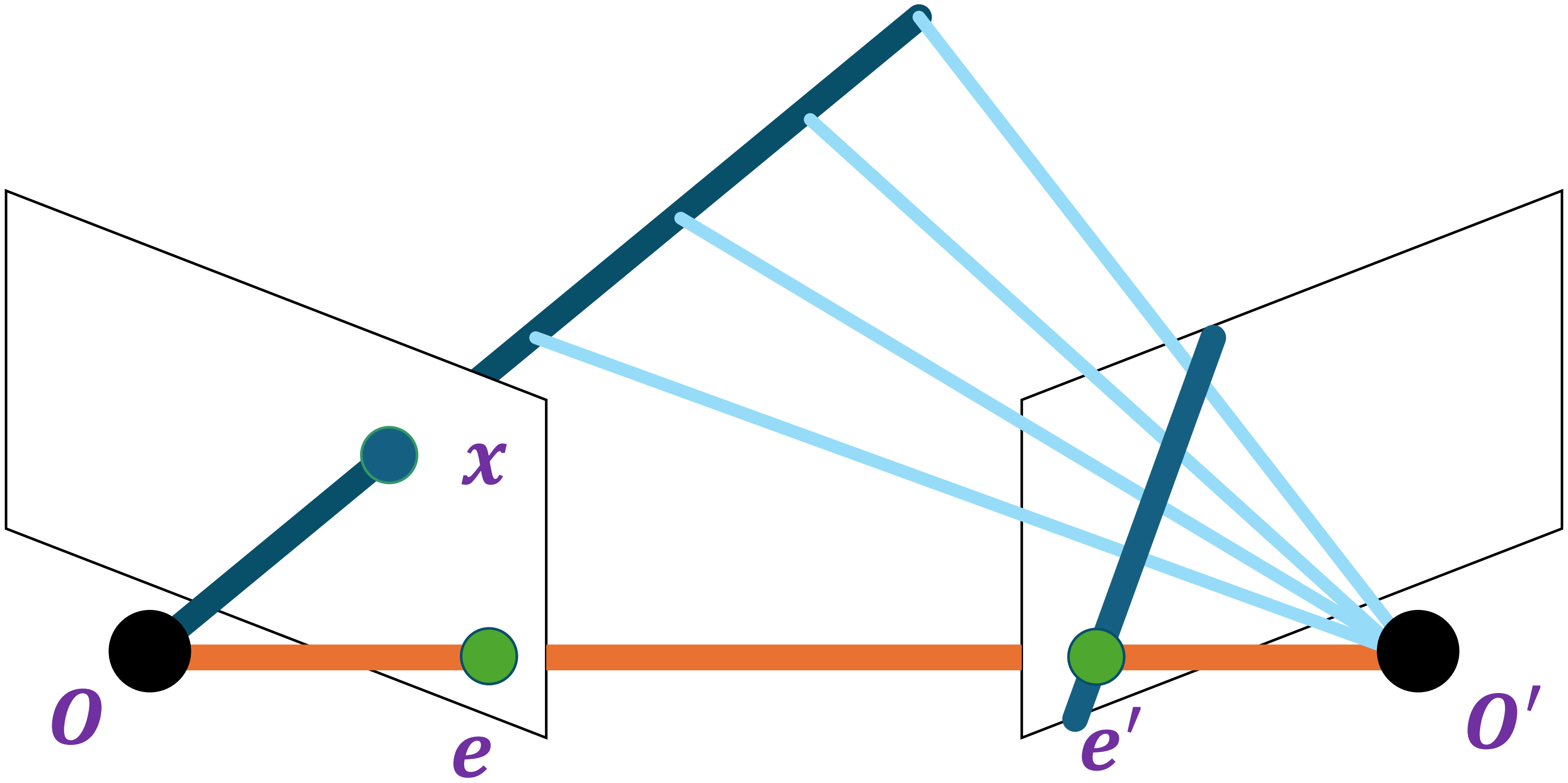
Suppose we observe a single point $\boldsymbol{x}$ in one image

Epipolar constraint



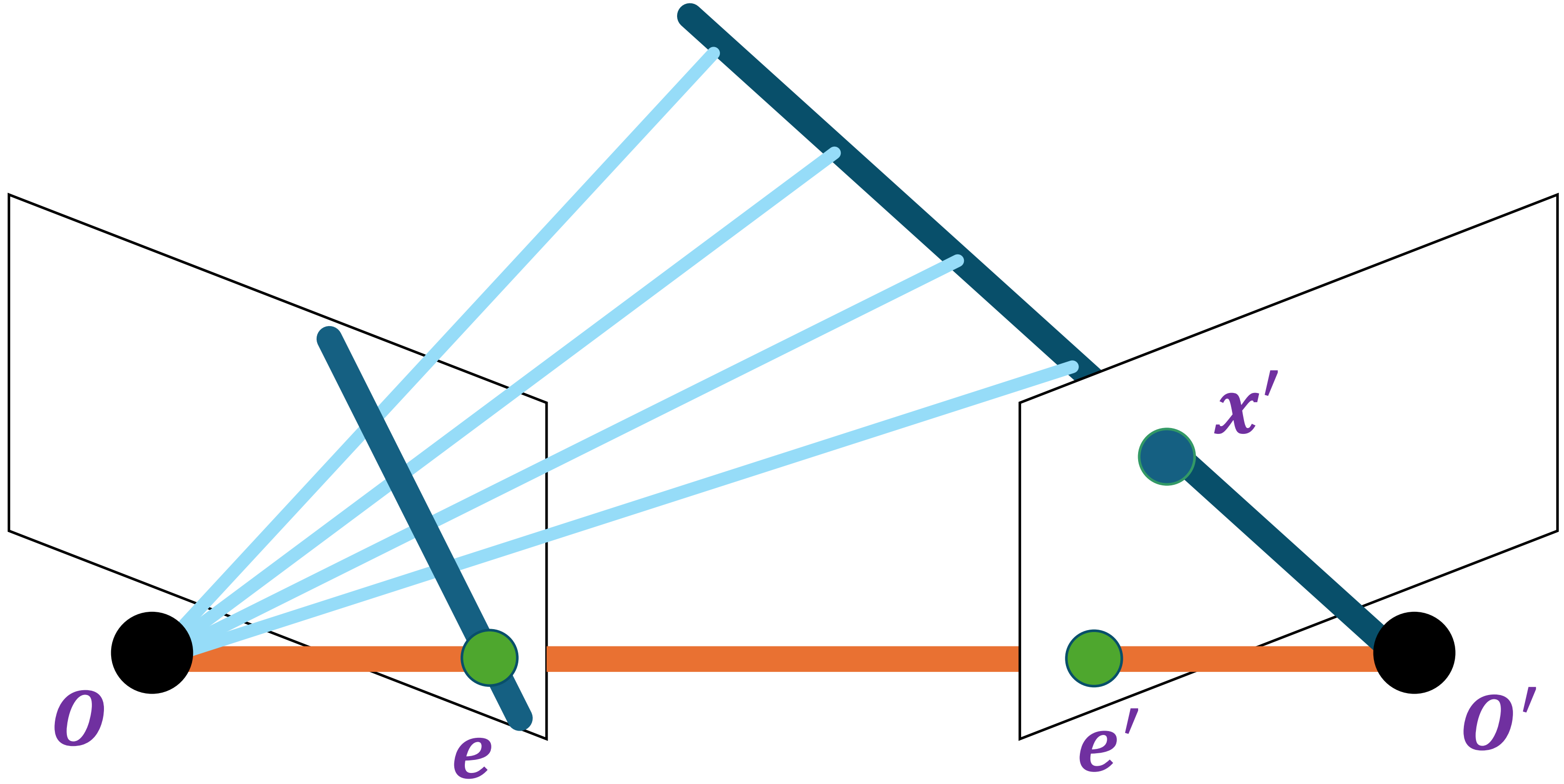
Where can we find the $\boldsymbol{x'}$ corresponding to $\boldsymbol{x}$ in the other image?

Epipolar constraint



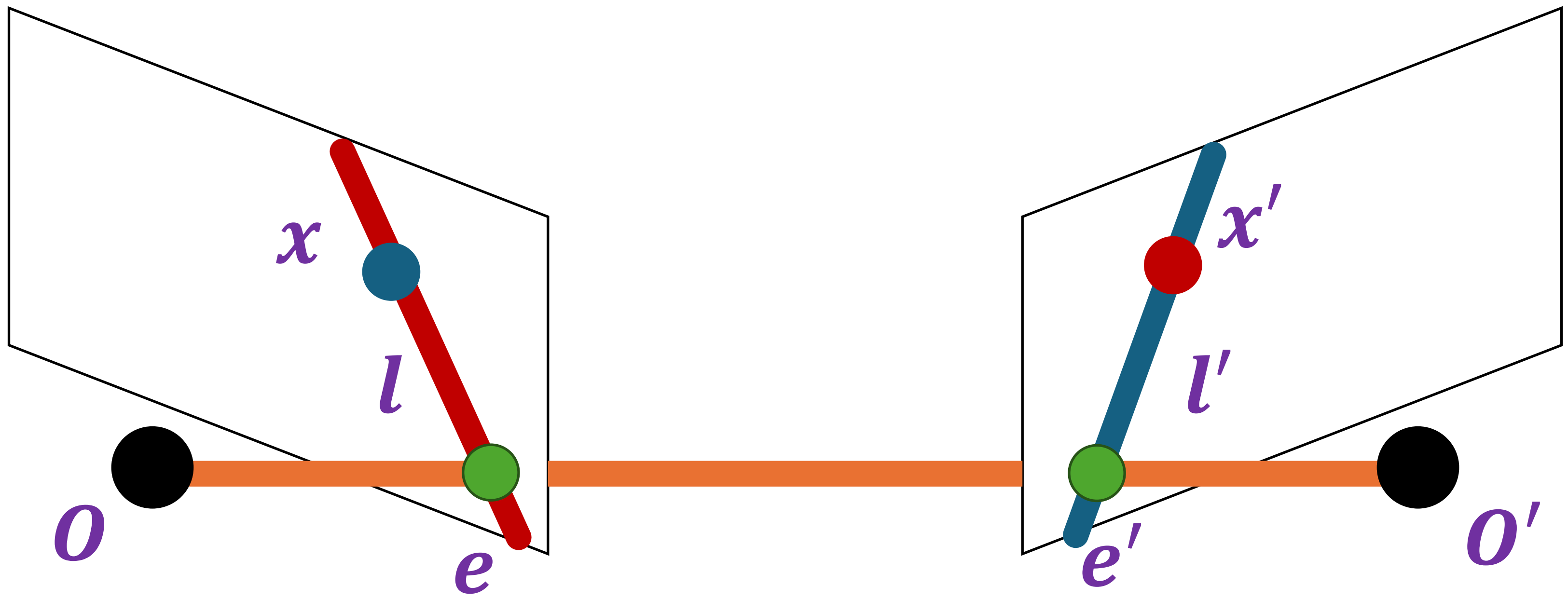
- Where can we find the x' corresponding to x in the other image?
- Along the **epipolar line** corresponding to x (projection of visual ray connecting O with x into the second image plane)

Epipolar constraint



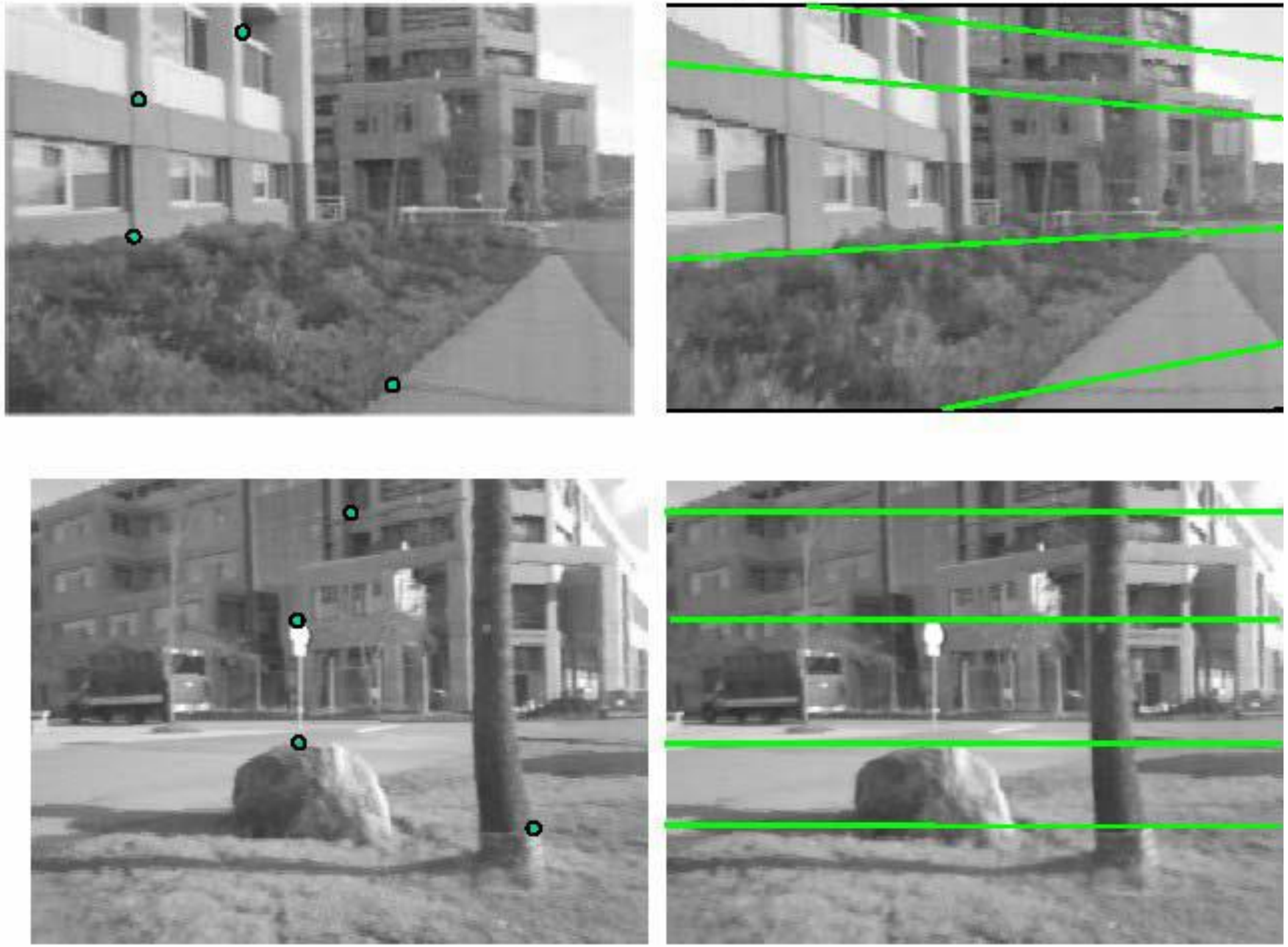
Similarly, all points in the left image corresponding to $\boldsymbol{x}'$ have to lie along the epipolar line corresponding to $\boldsymbol{x}'$

Epipolar constraint

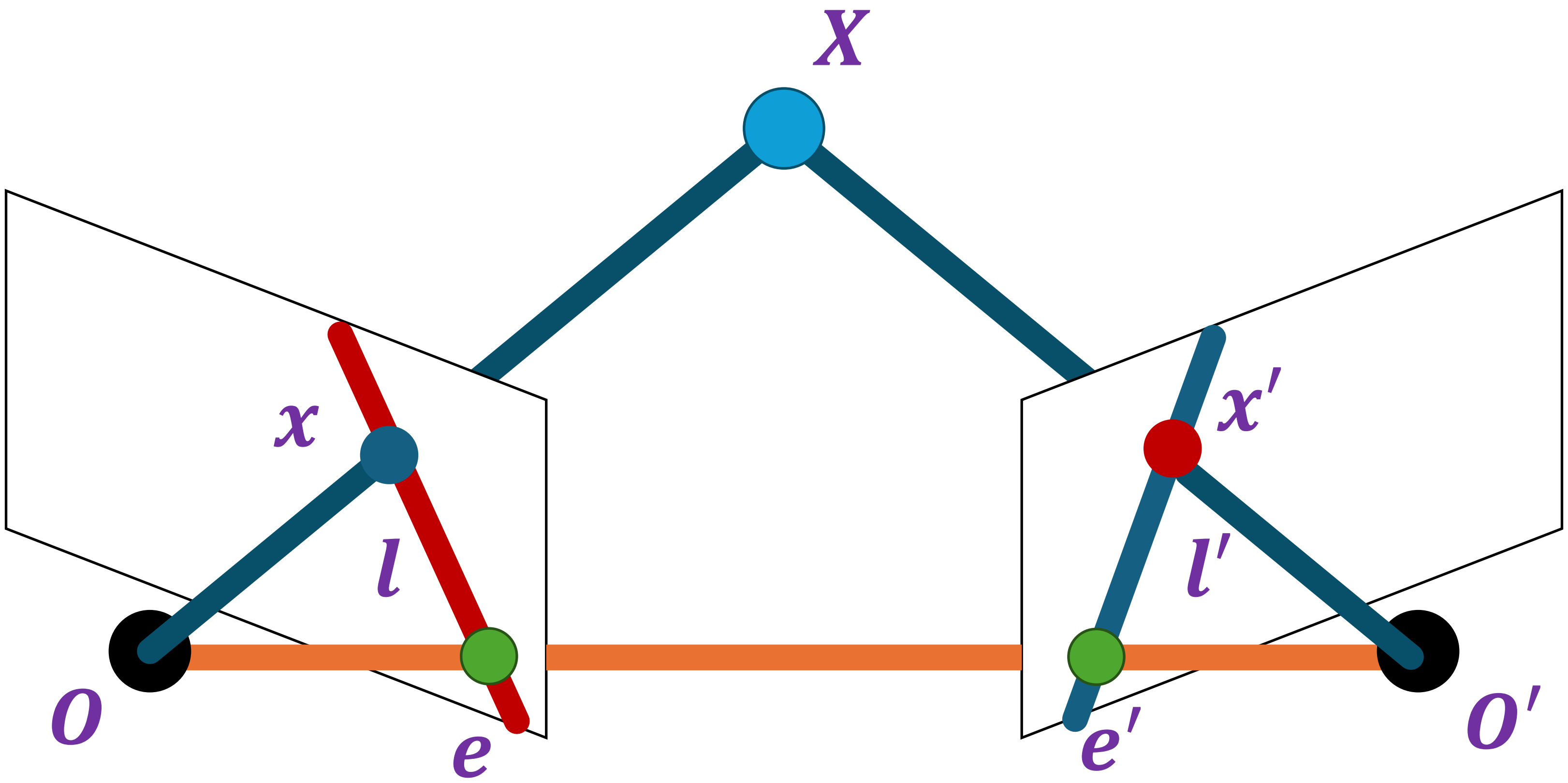


- Potential matches for x have to lie on the matching epipolar line l'
- Potential matches for x' have to lie on the matching epipolar line l

Epipolar constraint: Example

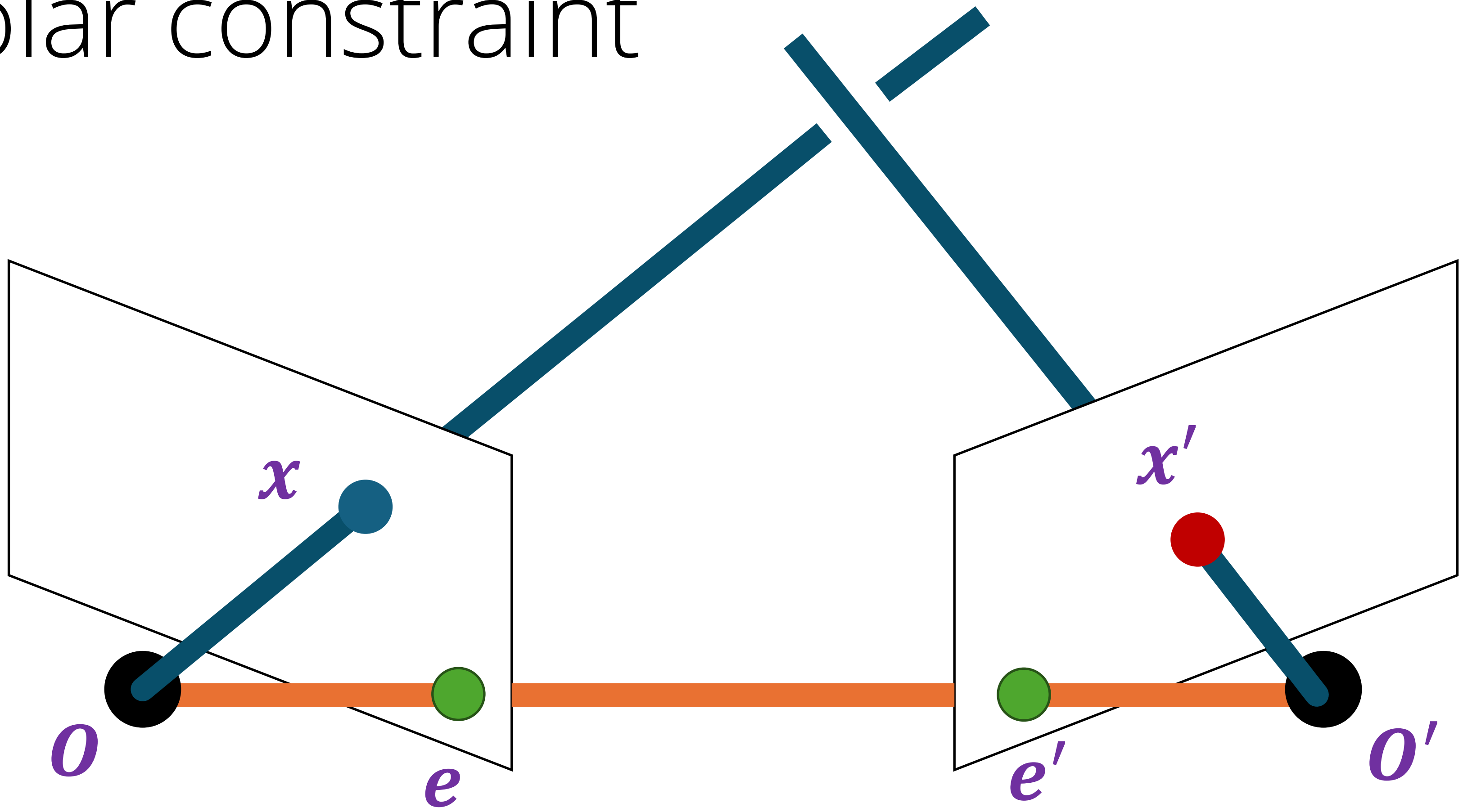


Epipolar constraint



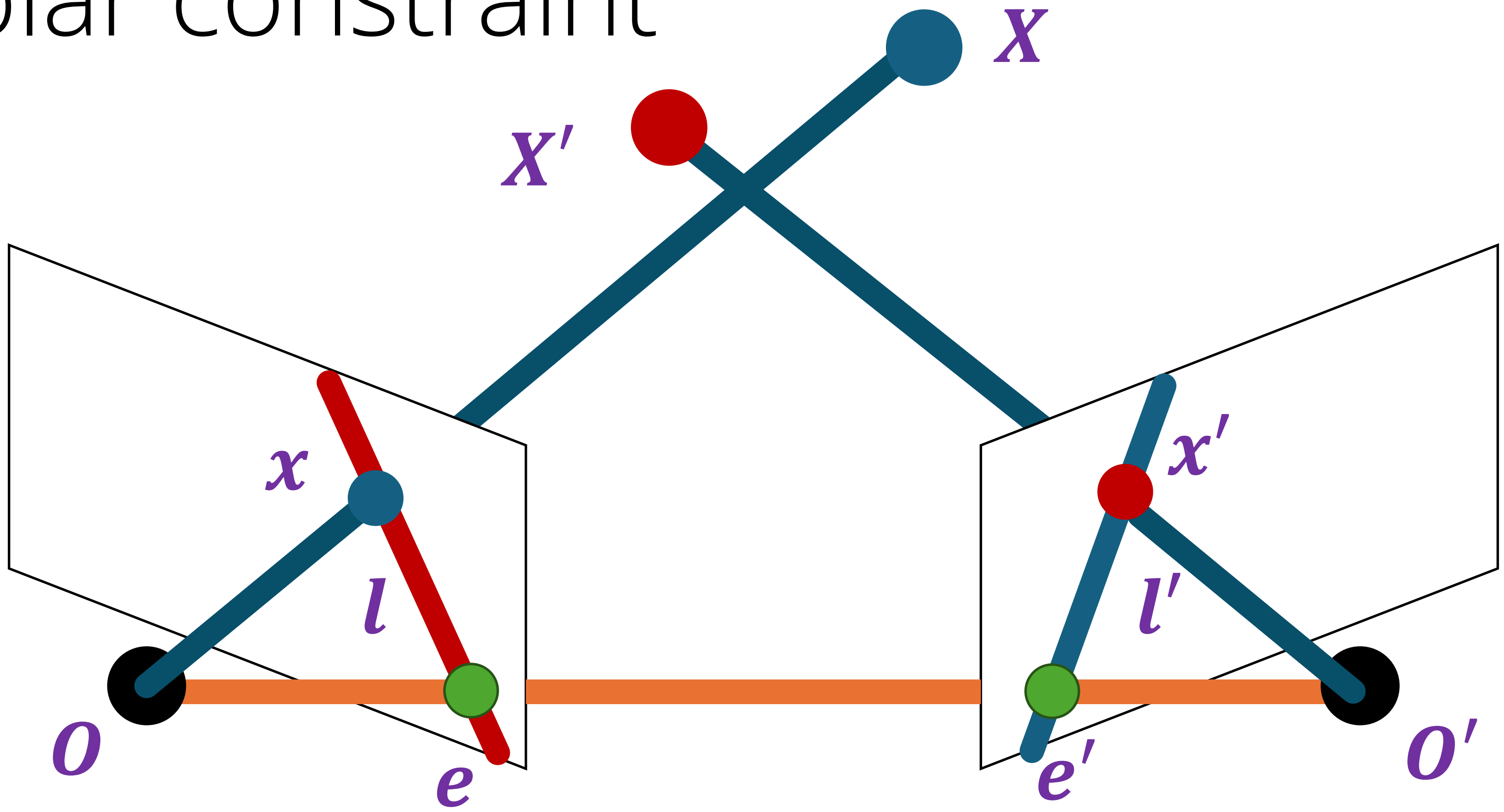
Whenever two points $\boldsymbol{x}$ and $\boldsymbol{x'}$ lie on matching epipolar lines $\boldsymbol{l}$ and $\boldsymbol{l'}$, the visual rays corresponding to them meet in space, i.e., $\boldsymbol{x}$ and $\boldsymbol{x'}$ could be projections of the same 3D point $\boldsymbol{X}$

Epipolar constraint



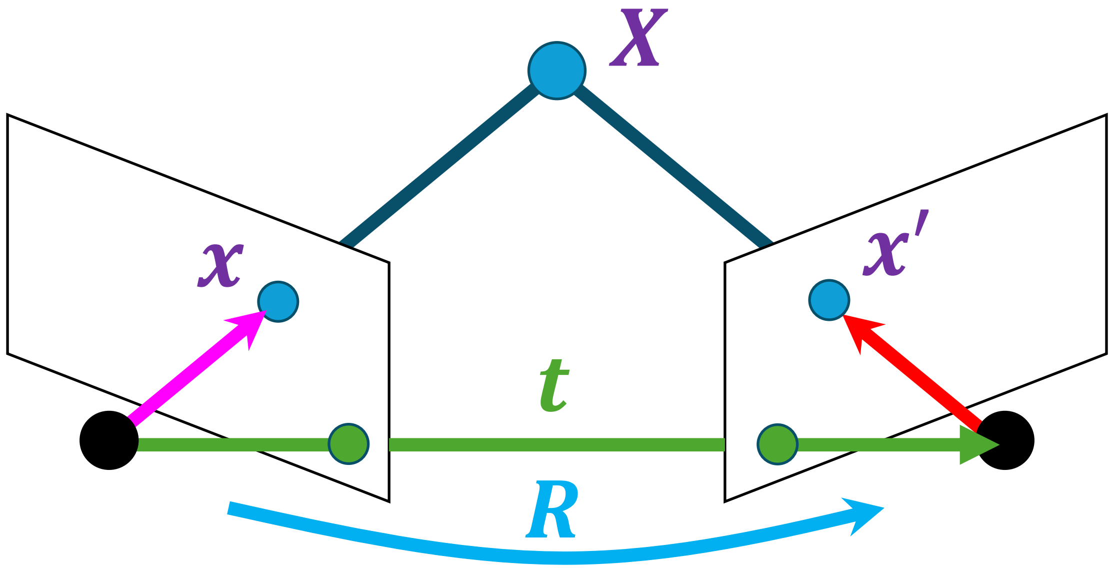
Remember: in general, two rays ***do not*** meet in space!

Epipolar constraint



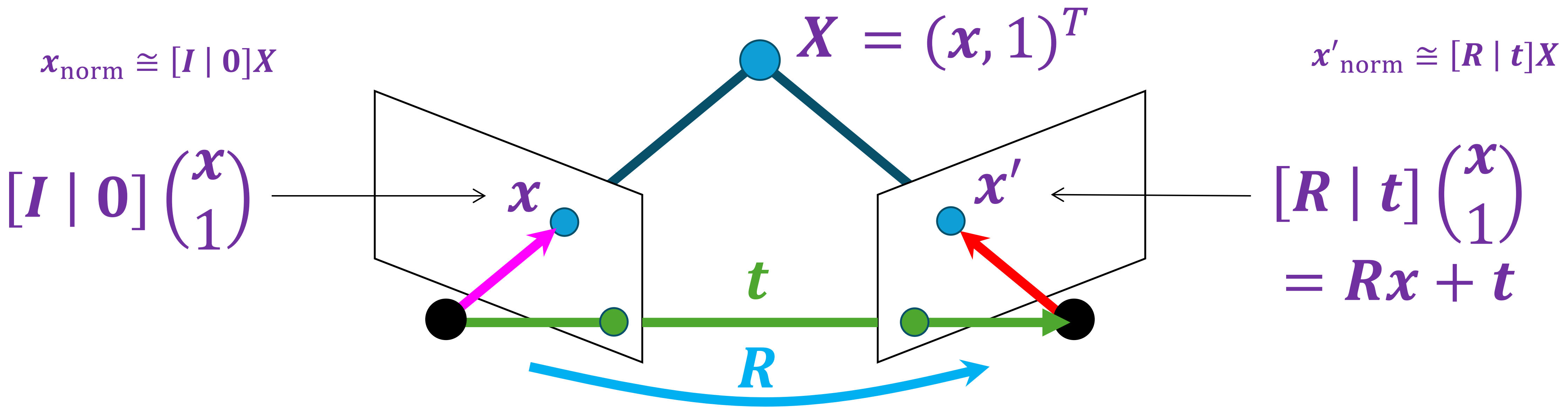
Caveat: if x and x' satisfy the epipolar constraint, this doesn't mean they ***have to be*** projections of the same 3D point

Epipolar Constraint: Calibrated case



- Assume the intrinsic and extrinsic parameters of the cameras are known, world coordinate system is set to that of the first camera
- Then the projection matrices are given by $\boldsymbol{K}[\boldsymbol{I} \mid \boldsymbol{0}]$ and $\boldsymbol{K}'[\boldsymbol{R} \mid \boldsymbol{t}]$
- We can pre-multiply the projection matrices (and the image points) by the inverse calibration matrices to get *normalized* image coordinates:
  - $\boldsymbol{x}_{\text{norm}} = \boldsymbol{K}^{-1}\boldsymbol{x}_{\text{pixel}} \cong [\boldsymbol{I} \mid \boldsymbol{0}]\boldsymbol{X}, \qquad \boldsymbol{x}'_{\text{norm}} = \boldsymbol{K}'^{-1}\boldsymbol{x}'_{\text{pixel}} \cong [\boldsymbol{R} \mid \boldsymbol{t}]\boldsymbol{X}$

Epipolar Constraint: Calibrated case

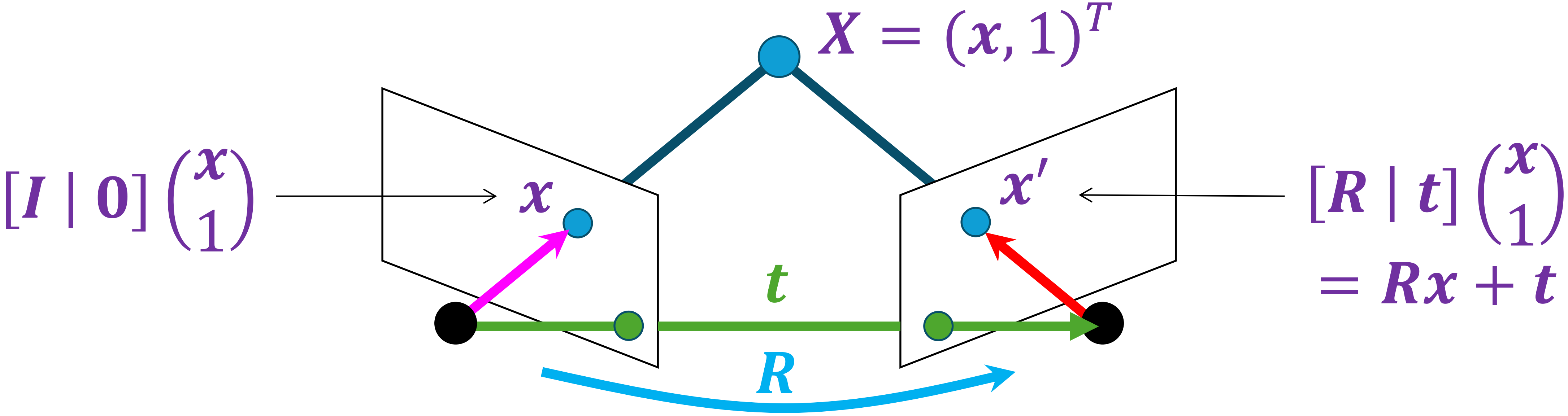


$\boldsymbol{x}' \cong \boldsymbol{R}\boldsymbol{x} + \boldsymbol{t}$

- This means the three vectors $\boldsymbol{x}'$, $\boldsymbol{R}\boldsymbol{x}$, and $\boldsymbol{t}$ are linearly dependent
- This constraint can be written using the *triple product*

• $\boldsymbol{x}' \cdot [\boldsymbol{t} \times (\boldsymbol{R}\boldsymbol{x})] = 0$

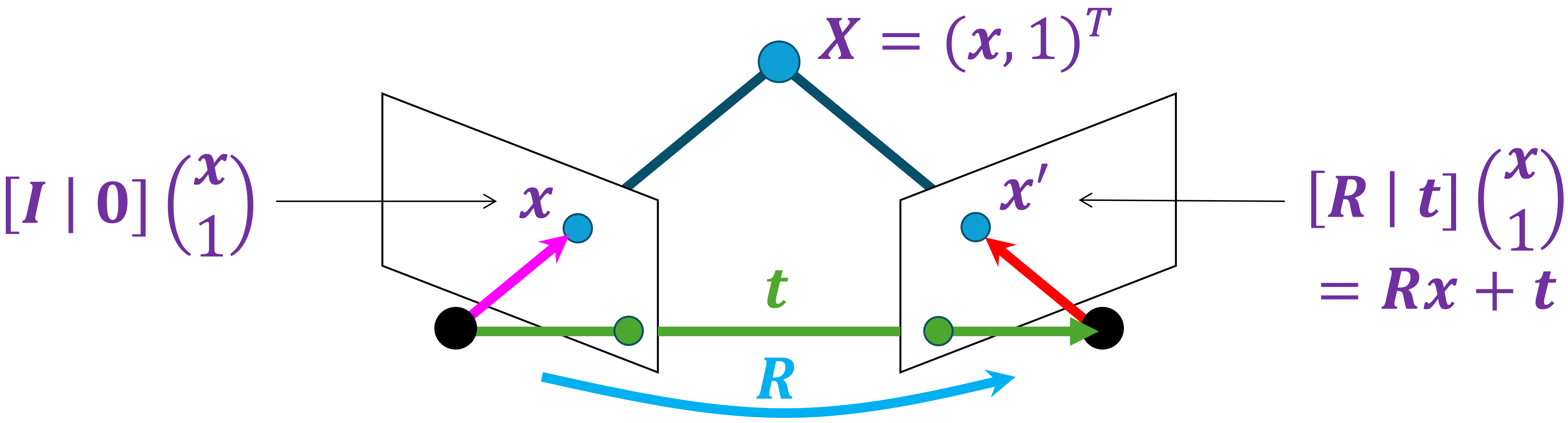
Epipolar Constraint: Calibrated case



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

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

Epipolar Constraint: Calibrated case

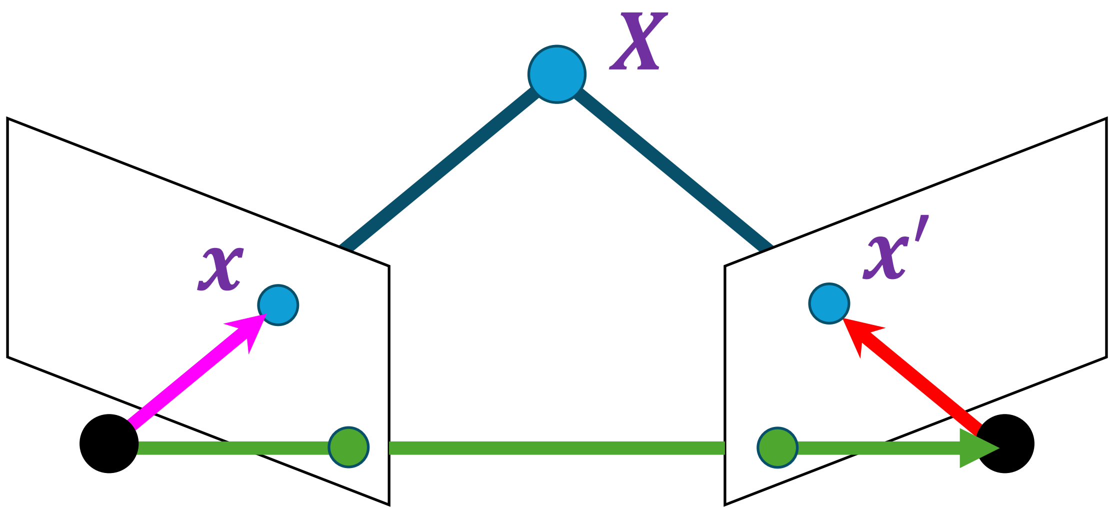


- $\boldsymbol{x}' \cdot [\boldsymbol{t} \times (\boldsymbol{R}\boldsymbol{x})] = 0 \Rightarrow \boldsymbol{x}'^T [\boldsymbol{t}_\times] \boldsymbol{R}\boldsymbol{x} = 0 \Rightarrow \boldsymbol{x}'^T \boldsymbol{E} \boldsymbol{x} = 0$

Essential Matrix

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

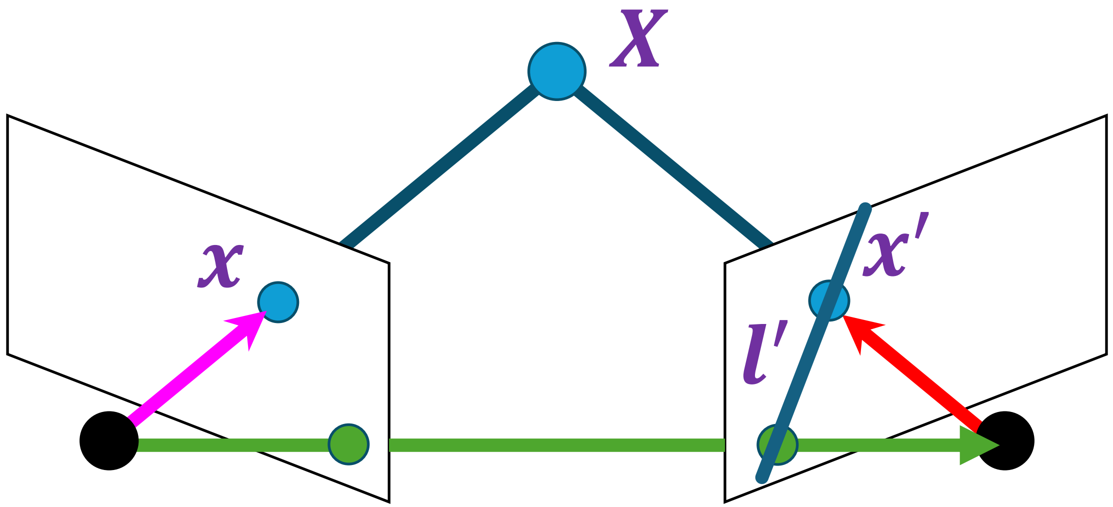
The essential matrix



$$\boldsymbol{x'}^T \boldsymbol{E} \boldsymbol{x} = 0$$

$$(x', y', 1) \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

The essential matrix: Properties

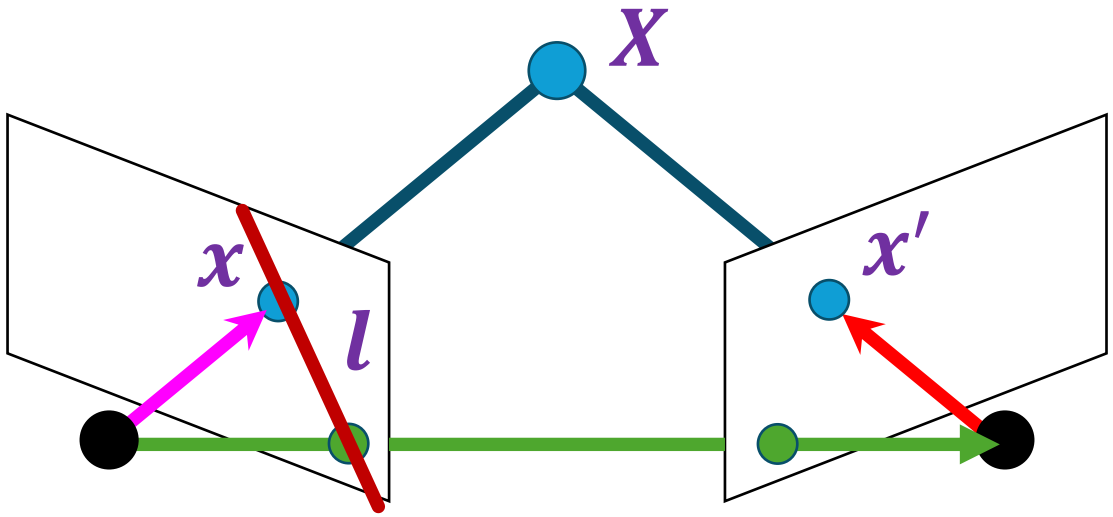


$$\boldsymbol{x}'^T \boldsymbol{E}\boldsymbol{x} = 0$$

$\boldsymbol{Ex}$ is the epipolar line associated with $\boldsymbol{x}$ ($\boldsymbol{l}' = \boldsymbol{Ex}$)

Recall: a line is given by $ax + by + c = 0$ or $\boldsymbol{l}^T\boldsymbol{x} = 0$
where $\boldsymbol{l} = (a, b, c)^T$ and $\boldsymbol{x} = (x, y, 1)^T$

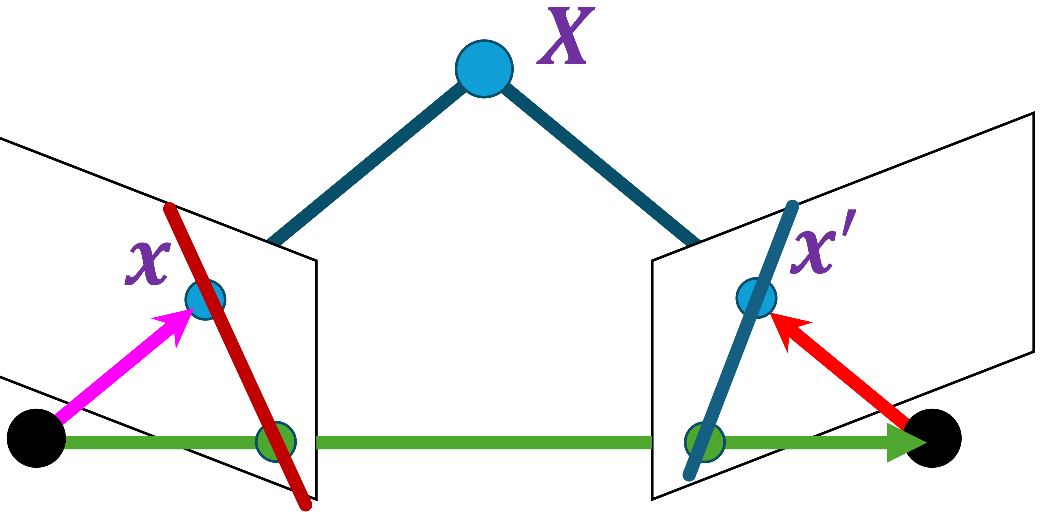
The essential matrix: Properties



$$\boldsymbol{x}'^T \boldsymbol{E}\boldsymbol{x} = 0$$

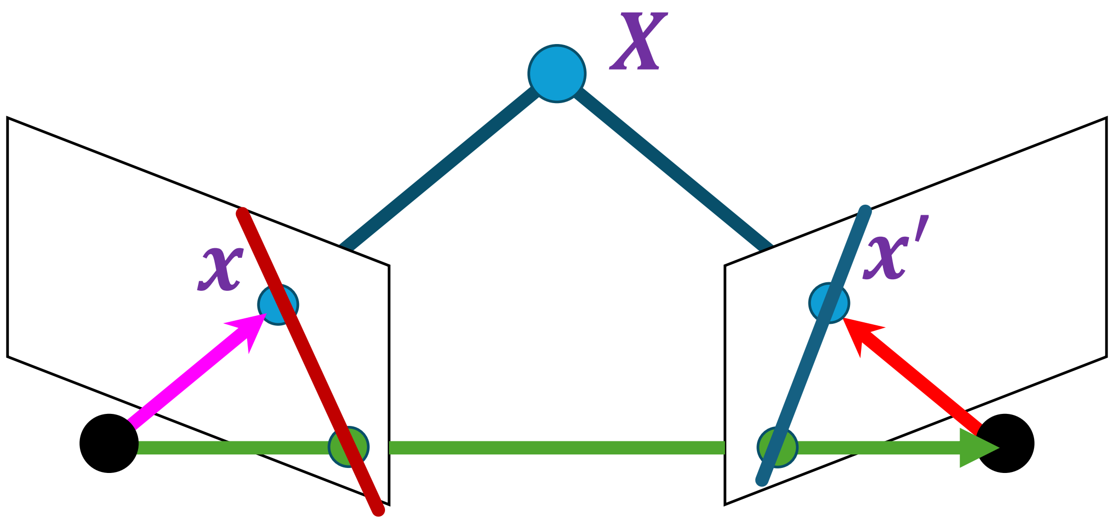
- $\boldsymbol{Ex}$ is the epipolar line associated with $\boldsymbol{x}$ ($\boldsymbol{l}' = \boldsymbol{Ex}$)
- $\boldsymbol{E}^T\boldsymbol{x}'$ is the epipolar line associated with $\boldsymbol{x}'$ ($\boldsymbol{l} = \boldsymbol{E}^T\boldsymbol{x}'$)
- $\boldsymbol{Ee} = \boldsymbol{0}$ and $\boldsymbol{E}^T\boldsymbol{e}' = \boldsymbol{0}$
- $\boldsymbol{E}$ is singular (rank two) and has five degrees of freedom

Epipolar constraint: Uncalibrated case



- The calibration matrices $\boldsymbol{K}$ and $\boldsymbol{K}'$ of the two cameras are unknown
- We can write the epipolar constraint in terms of *unknown* normalized coordinates:
  - $\boldsymbol{x}'^T_{\text{norm}}\boldsymbol{E}\boldsymbol{x}_{\text{norm}} = 0,$
  - where $\boldsymbol{x}_{\text{norm}} = \boldsymbol{K}^{-1}\boldsymbol{x}$, $\boldsymbol{x}'_{\text{norm}} = \boldsymbol{K}'^{-1}\boldsymbol{x}'$

Epipolar constraint: Uncalibrated case



- $x'^{T}_{\text{norm}} E x_{\text{norm}} = 0$ ⇒ $x'^T F x = 0$, where $F = K'^{-T} E K^{-1}$

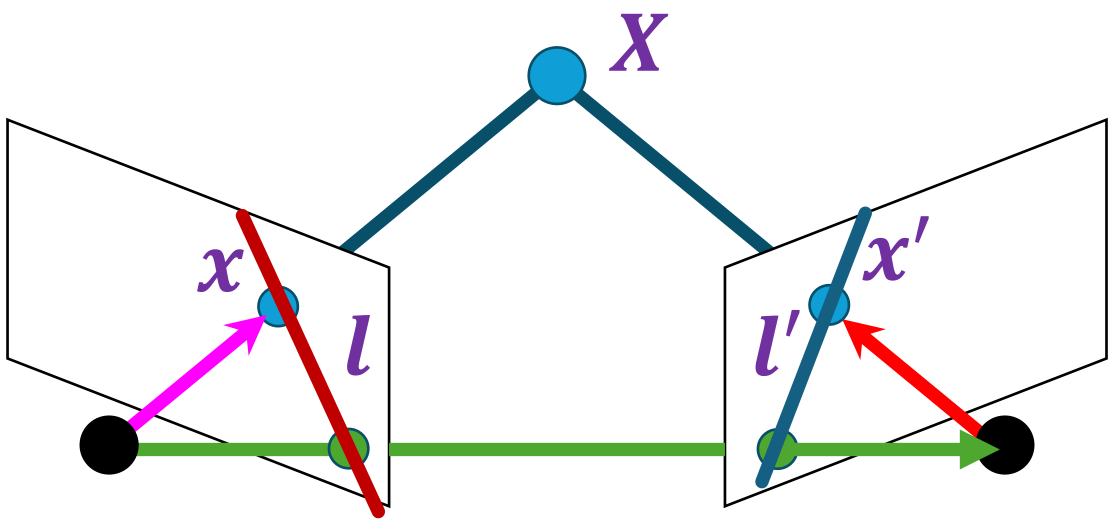
Fundamental Matrix

$x_{\text{norm}} = K^{-1} x$

$x'_{\text{norm}} = K'^{-1} x'$

Faugeras et al., (1992), Hartley (1992)

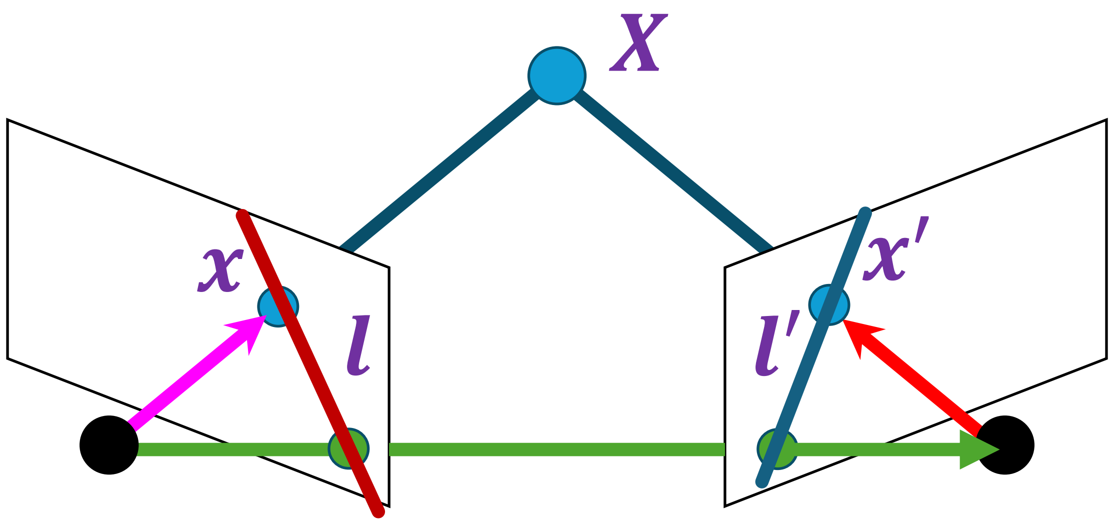
The fundamental matrix



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

$$(x', y', 1) \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

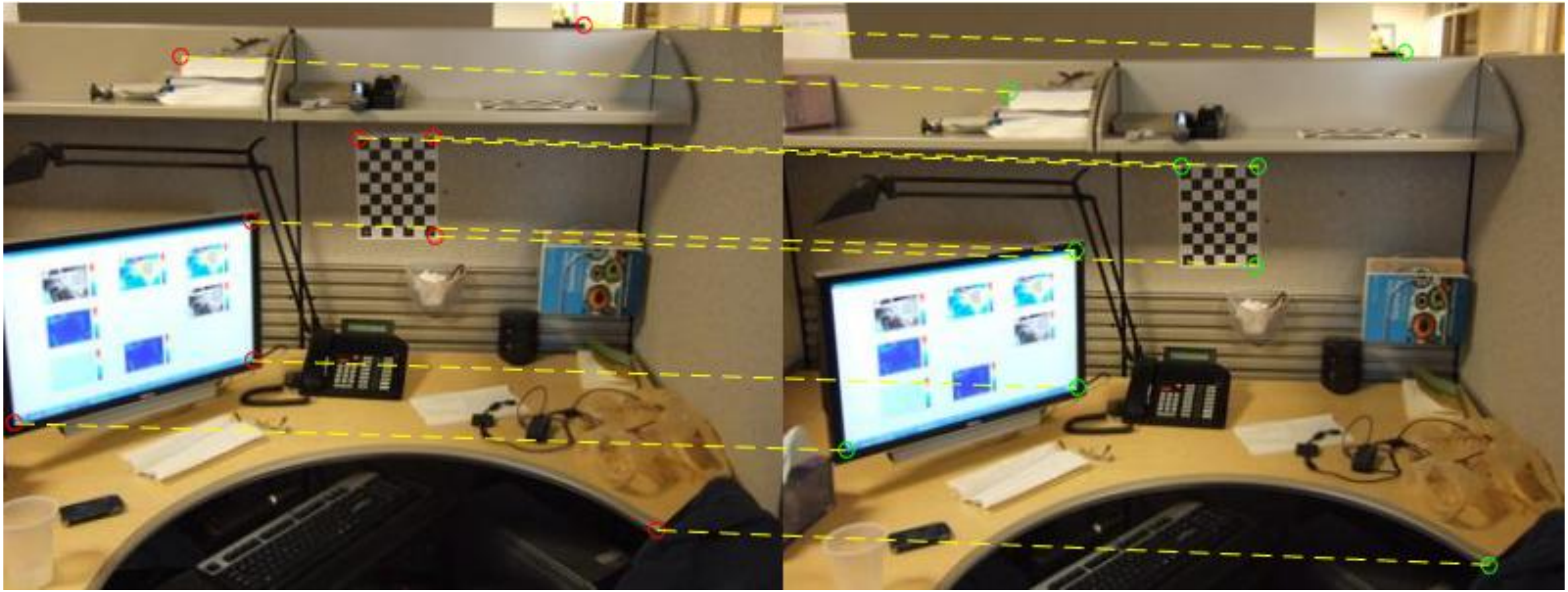
The fundamental matrix: Properties



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

- $\boldsymbol{Fx}$ is the epipolar line associated with $\boldsymbol{x}$ ($\boldsymbol{l}' = \boldsymbol{Fx}$)
- $\boldsymbol{F}^T\boldsymbol{x}'$ is the epipolar line associated with $\boldsymbol{x}'$ ($\boldsymbol{l} = \boldsymbol{F}^T\boldsymbol{x}'$)
- $\boldsymbol{Fe} = \boldsymbol{0}$ and $\boldsymbol{F}^T\boldsymbol{e}' = \boldsymbol{0}$
- $\boldsymbol{F}$ is singular (rank two) and has seven degrees of freedom

Estimating the fundamental matrix



Estimating the fundamental matrix

- Given: correspondences $\boldsymbol{x_i} = (x_i, y_i, 1)^T$ and $\boldsymbol{x'_i} = (x_i{}', y_i{}', 1)^T$
- Constraint: $\boldsymbol{x'}^T \boldsymbol{F} \boldsymbol{x} = 0$

- $(x', y', 1)\begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0 \quad \Rightarrow \quad (x'x, x'y, x', y'x, y'y, y', x, y, 1)\begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = 0$

The eight point algorithm

$$\underbrace{\begin{bmatrix} & & & \vdots & & & & & \\ x'x & x'y & x' & y'x & y'y & y' & x & y & 1 \\ & & & \vdots & & & & & \end{bmatrix}}_{\boldsymbol{U}} \begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = \mathbf{0}$$

Homogeneous least squares to find $\boldsymbol{f}$:

$\underset{\|\boldsymbol{f}\|=1}{\arg\min} \|\boldsymbol{U}\boldsymbol{f}\|_2^2$ ➡ Eigenvector of $\boldsymbol{U}^T\boldsymbol{U}$ with smallest eigenvalue

Enforcing rank-2 constraint

- We know $\boldsymbol{F}$ needs to be singular/rank 2. How do we force it to be singular?
- Solution: take SVD of the initial estimate and throw out the smallest singular value

$$\boldsymbol{F}_{\text{init}} = \boldsymbol{U\Sigma V}^T \rightarrow \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \cancel{\sigma_3} \end{bmatrix} \longrightarrow \boldsymbol{\Sigma}' = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\downarrow$$

$$\boldsymbol{F} = \boldsymbol{U\Sigma' V}^T$$

Enforcing rank-2 constraint

Initial $\boldsymbol{F}$ estimate

Rank-2 estimate

The Fundamental Matrix Song

http://danielwedge.com/fmatrix/

Large-scale SfM

- 2006: Photo Tourism (Snavely et al,, SIGGAPH’06)
  - 3D reconstruction from internet images
  - Large scale compute
- 2009: Building Rome in a Day (Agarwal et al. ICCV’09)
  - Search “rome” on flickr
  - Reconstruction: 150k images, 21h, 500CPUs

Neural Rendering

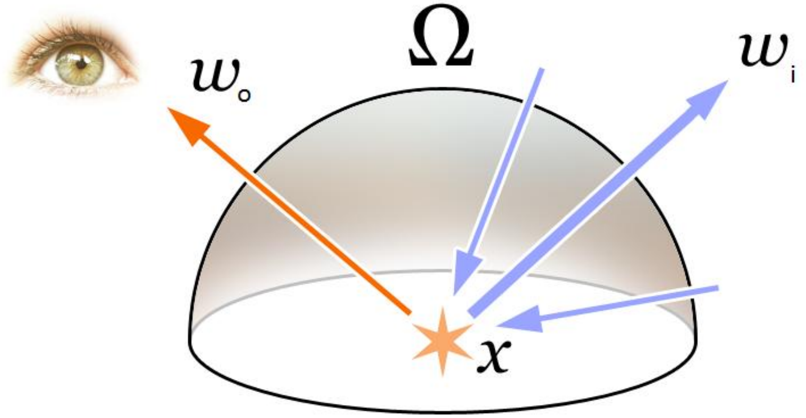
1850: Photosculpture

- 24 photographs of an object/person
- Cut contour from wood
- Assemble radial sculpture

1986: The Rendering Equation

How much light (of wavelength λ) is leaving a point x in the direction of ω_o at time t?

$$L_o(x, \omega_o, \lambda, t) = \underbrace{L_e(x, \omega_o, \lambda, t)}_{\text{emitted radiance (glowing things)}} + \underbrace{L_r(x, \omega_o, \lambda, t)}_{\text{reflected radiance}}$$

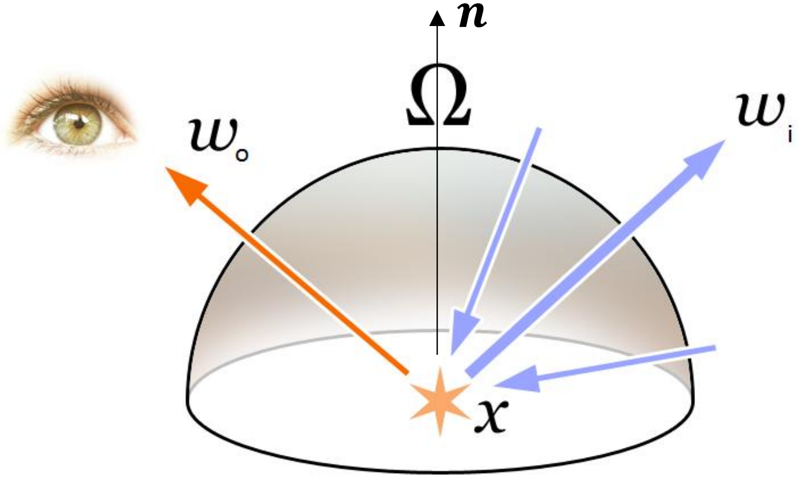


Immel, David S.; Cohen, Michael F.; Greenberg, Donald P. "*A radiosity method for non-diffuse environments*", SIGGRAPH 1986
Kajiya, James T. "*The rendering equation*". Conference on Computer graphics and interactive techniques 1986

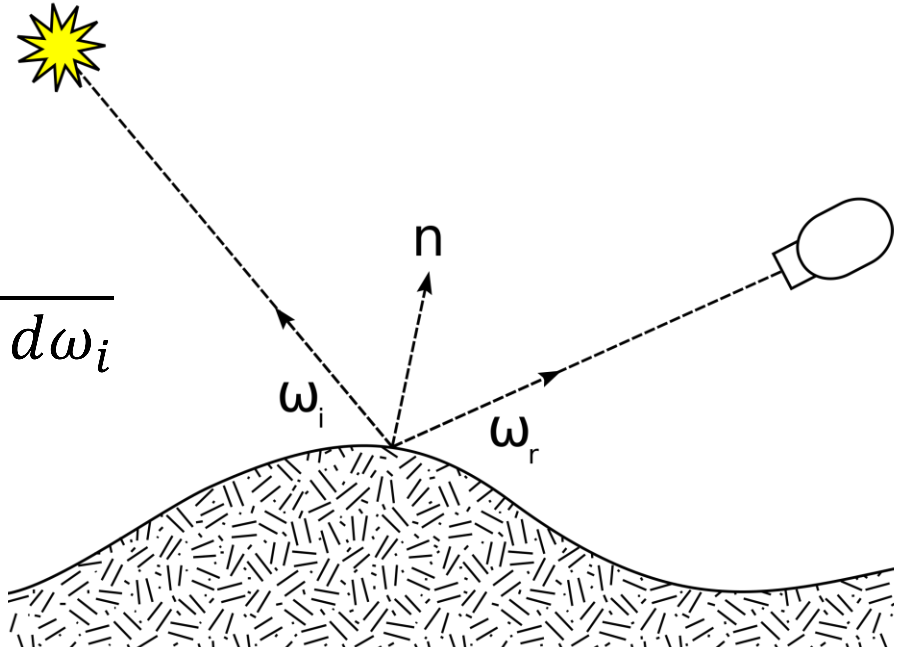
1986: The Rendering Equation

$$L_r(x,\omega_o,\lambda,t) = \int_\Omega \underbrace{f_i(x,\omega_i,\omega_o,\lambda,t)}_{\text{bidirectional reflectance distribution function (BRDF)}} \overbrace{L_i(x,\omega_i,\lambda,t)}^{\text{incoming radiance at } x \text{ from direction } \omega_i}(\omega_i \cdot \boldsymbol{n})d\omega_i$$

surface normal



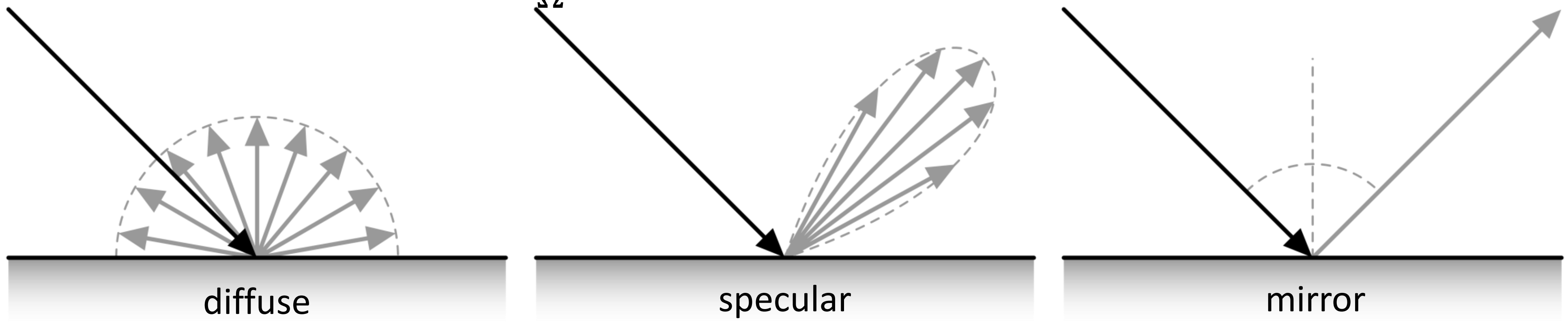
1965: The BRDF



$$f_r(\omega_i, \omega_r) = \frac{dL_r(\omega_r)}{L_i(\omega_i)\,(\omega_i \cdot n)\,d\omega_i}$$

- Positivity: $f_r(\omega_i, \omega_r) > 0$
- Reciprocity: $f_r(\omega_i, \omega_r) = f_r(\omega_r, \omega_i)$
- Energy conservation:

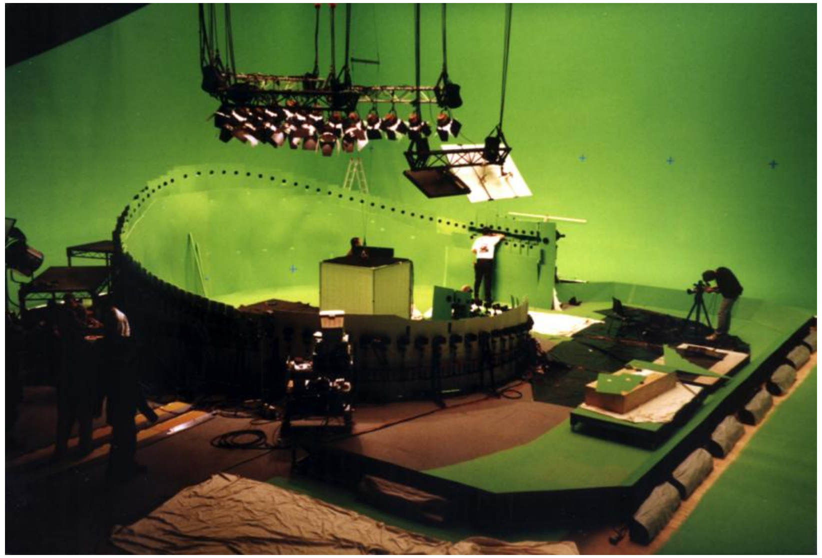
$$\forall \omega_i, \int_\Omega f_r(\omega_i, \omega_r)(\omega_r \cdot n) d\omega_r \leq 1$$



Nicodemus, Fred (1965). "*Directional reflectance and emissivity of an opaque surface*". Applied Optics

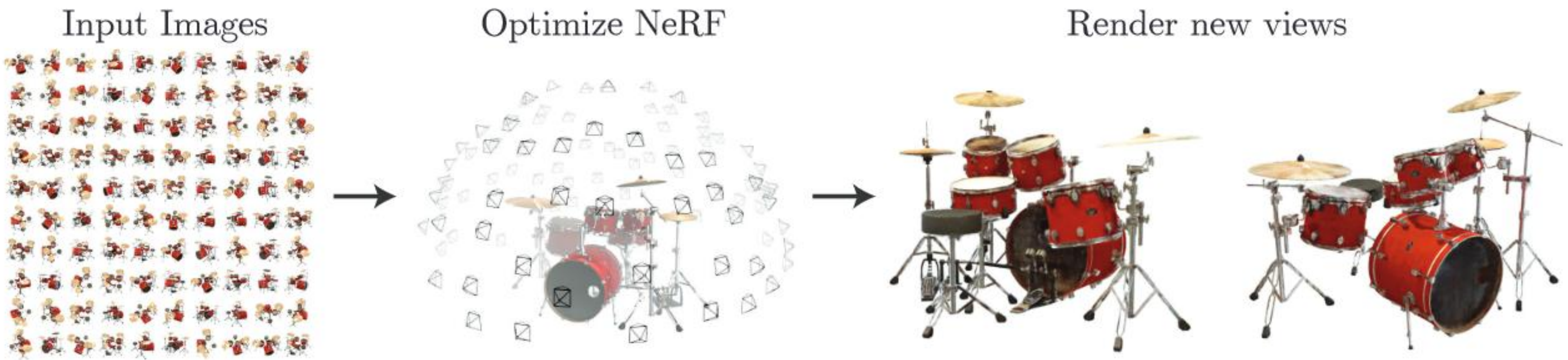
Lightfield camera arrays

- Use many synchronized cameras to capture a scene from many angle simlutaneously
- Film use: The Matrix (1999)



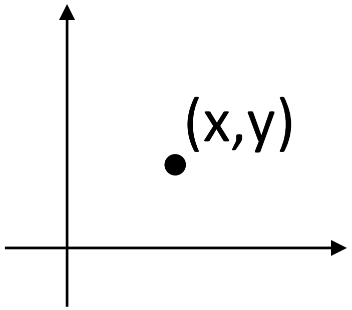
Neural Radiance Fields

- Input: Image collection
- Learning: mapping coordinates (x,y,z) to color and occupancy
- Output: rendering from novel viewpoints

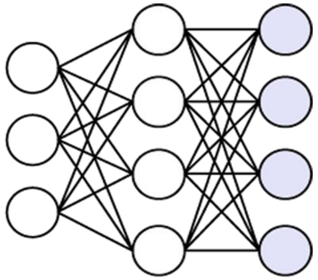


Mildenhall, Ben, et al. "Nerf: Representing scenes as neural radiance fields for view synthesis.“, ECCV 2020

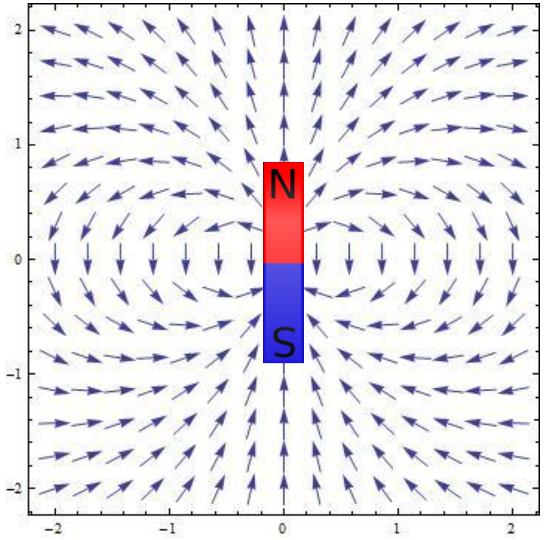
Neural Fields



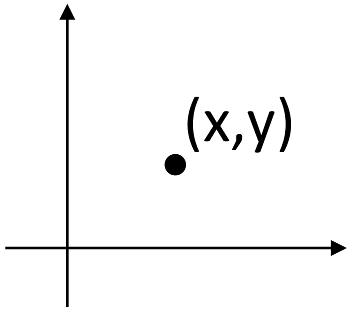
$\Phi: \mathbb{R}^2 \rightarrow \mathbb{R}^2$



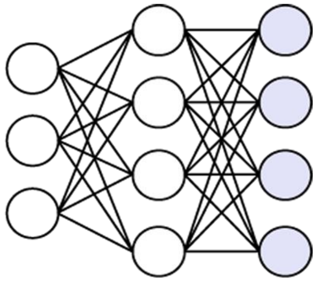
Neural Network
(Φ)



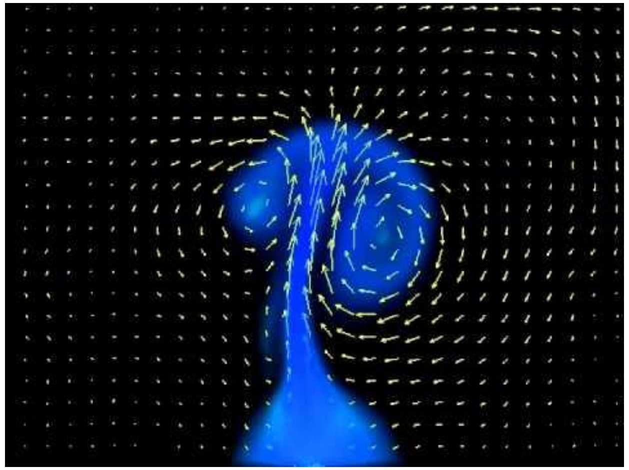
Magnetic Field



$\Phi: \mathbb{R}^2 \rightarrow \mathbb{R}^2$



Neural Network
(Φ)

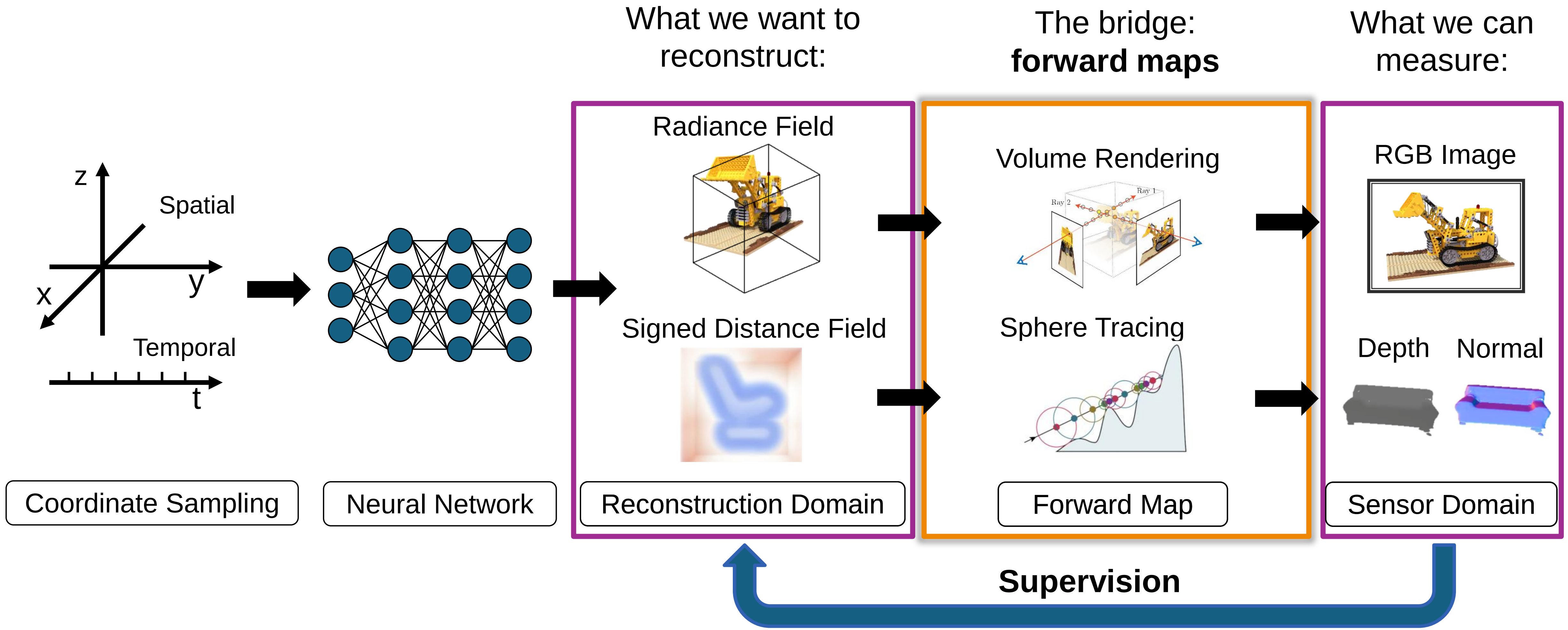


Eulerian Flow Field

[Koldora CC]

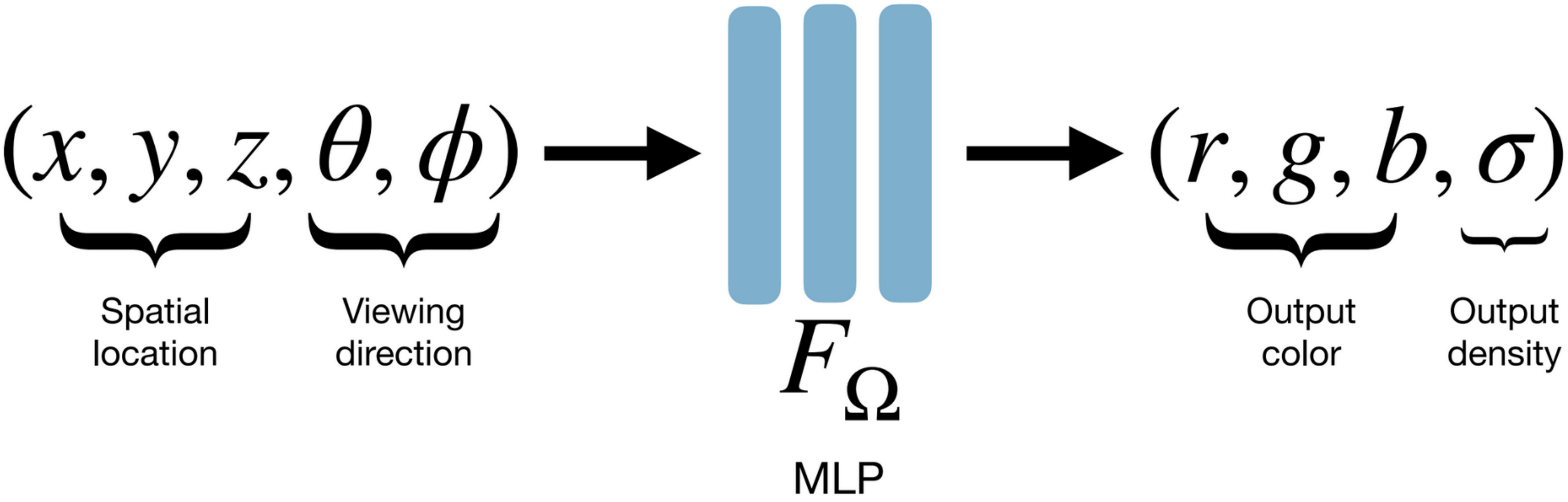
[Slide: Srinath Sridhar, Towaki Takikawa at CVPR ‘22 Tutorial on Neural Fields in Computer Vision]

Neural Fields



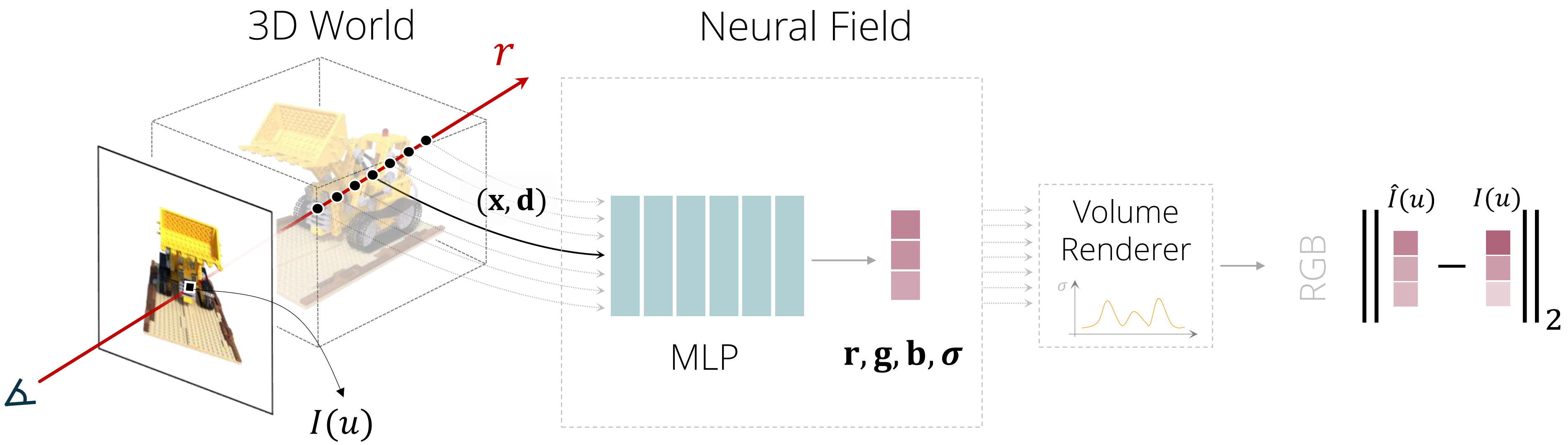
[Slide: Yiheng Xie, at CVPR ‘22 Tutorial on Neural Fields in Computer Vision]

Neural Radiance Fields



Slide credit: Angjoo Kanazawa, ECCV 2022 Tutorial Neural Volumetric Rendering for Computer Vision

Neural Radiance Fields



(Direct) Volume Rendering

- Rendering a ray $r(t) = o + td$

$$C \approx \sum_{i=1}^{N} T_i \alpha_i c_i$$

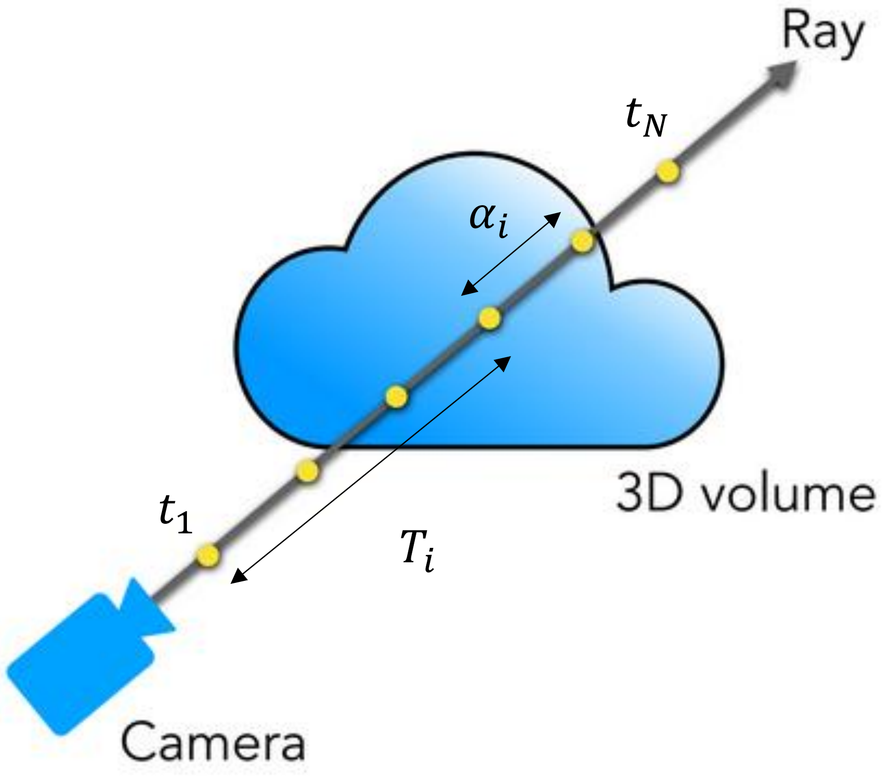
visibility opacity colour

- Visibility depends on opacity before

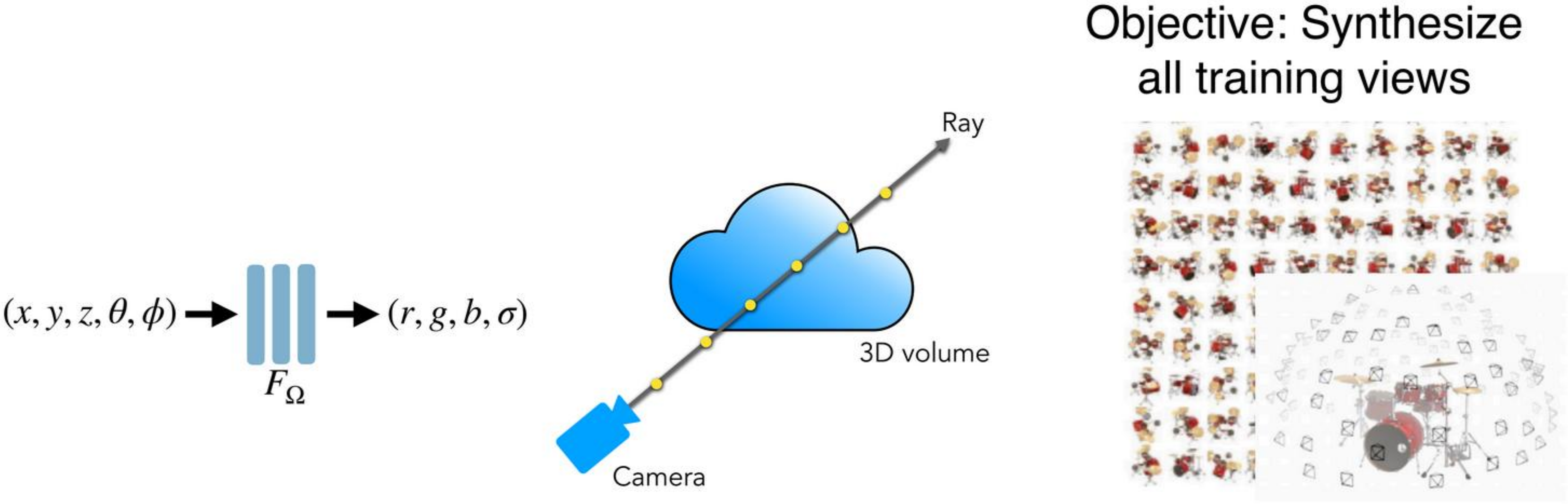
$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_i)$$

- Opacity is a function of density

$$\alpha_i = 1 - e^{-\sigma_i \Delta t}$$



Neural Radiance Fields



Neural Volumetric 3D Scene Representation

Differentiable Volumetric Rendering Function

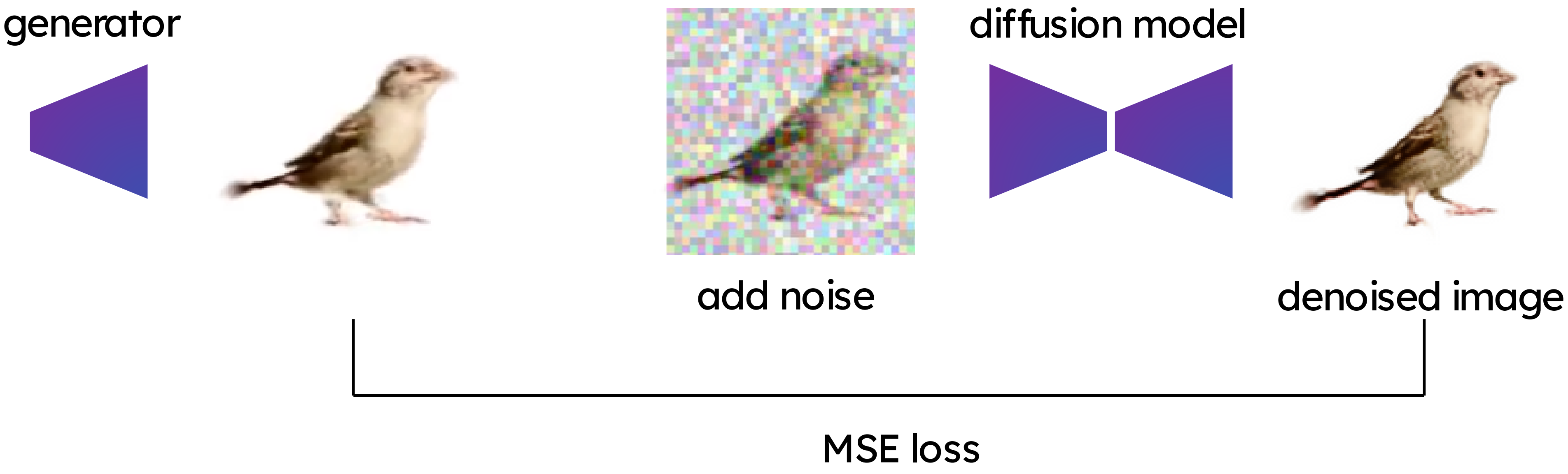
Optimization via Analysis-by-Synthesis

Slide credit: Angjoo Kanazawa, ECCV 2022 Tutorial Neural Volumetric Rendering for Computer Vision

NeRF Examples

NeRF Song

Diffusion Model as a Prior



- No need to compute gradients from the diffusion model
- Any generator works

Poole, A Jain, JT Barron, B Mildenhall.
"DreamFusion: Text-to-3D using 2D Diffusion" ICLR 2023.

Textual Inversion

- Learn a *new word* “<e>” for a specific concept

- Freeze everything and train only the embedding of <e>

R Gal, Y Alaluf, Y Atzmon, O Patashnik, AH Bermano, G Chechik, D Cohen-Or.
"An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion." arxiv 2022.

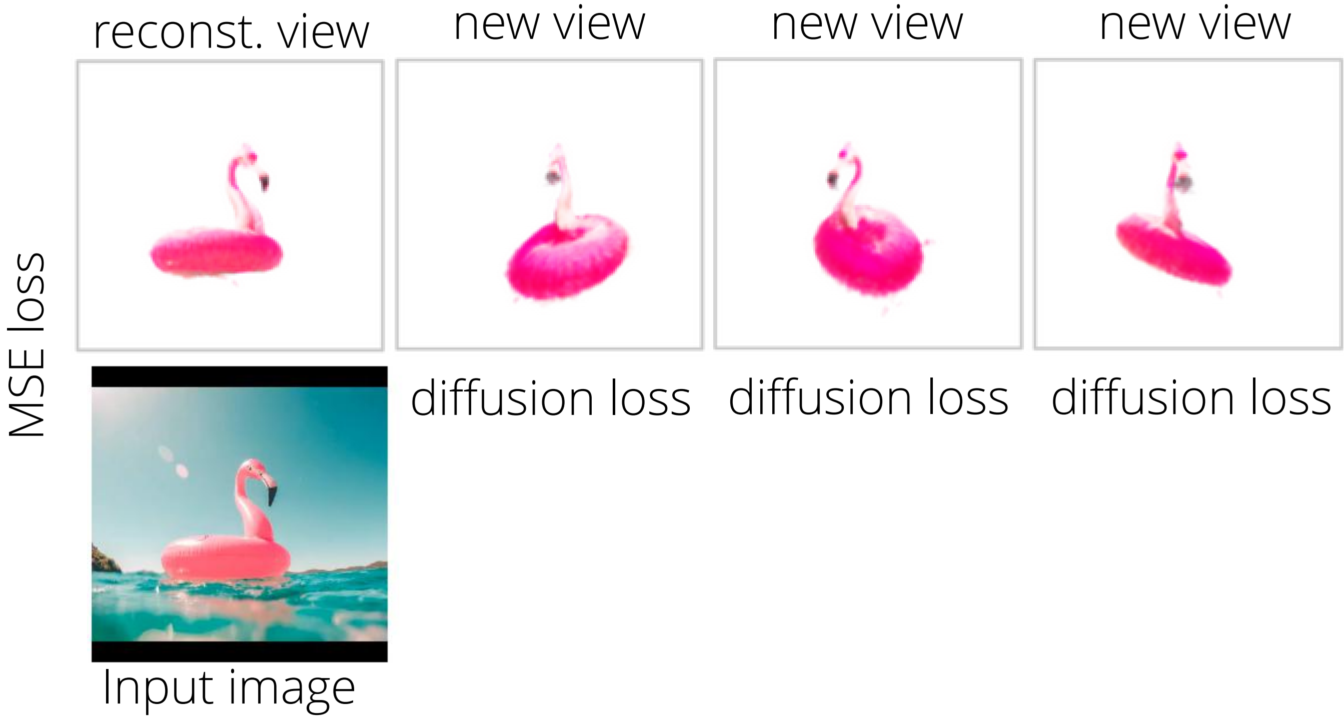
Textual Inversion

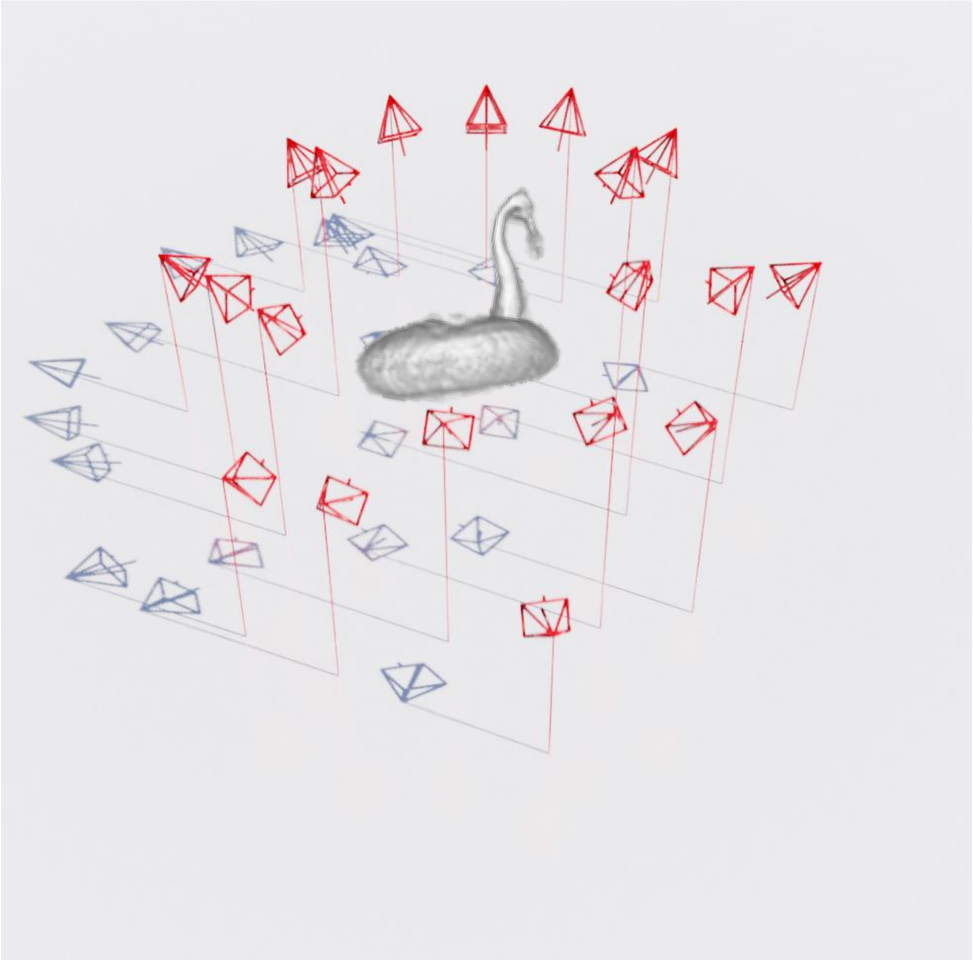
Original image

Generated image:
“an image of a <e>”

Learning 3D Shapes

- Compute <e>
- Sample camera poses
- Render views (NeRF)





3D from large models

- Optimization only: no learning required
- Needs several tricks to converge (shading, regularization, “<e> from the side”, …)

- Photographer bias reflected in the reconstructions

The “Janus Problem”