CS131: Computer Vision: Foundations and Applications

Multi-view Geometry

Juan Carlos Niebles and Adrien Gaidon

Reference: Szeliski 11.3 & 12.1, H&Z ch. 9

Most slides adapted from S. Lazebnik, J. Johnson, D. Fouhey

The story so far

Model the 3D-to-2D camera projection: $\tilde{\mathbf{x}}^{\mathrm{I}} \sim \mathbf{P}\tilde{\mathbf{X}}^{\mathrm{W}}$ with $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$

Calibrate cameras (get $\mathbf{K}[\mathbf{R}|\mathbf{t}]$) from N 2D-**3D** correspondences $(\tilde{\mathbf{x}}_i^{\mathrm{I}},\ \tilde{\mathbf{X}}_i^{\mathrm{W}})$

- cast constraints (2D-3D correspondences) as a linear system $\mathbf{A}\,\mathbf{p} = \mathbf{0}$

- total least squares ($argmin_{\mathbf{x}}\ \|\mathbf{A}\,\mathbf{p}\|^2 s.t.\ \|\mathbf{p}\|^2 = 1$) gives the best approximation

- closed-form solution via the SVD of $\mathbf{A}$ (its last right singular vector) & Cholesky

- refine by minimizing the 2D reprojection error $\sum_i \|\mathrm{proj}(\mathbf{K}[\mathbf{R} \mid \mathbf{t}]\mathbf{X_i}; \boldsymbol{\kappa}) - \boldsymbol{x}_i\|^2$

In general, we don’t have **3D** measurements…

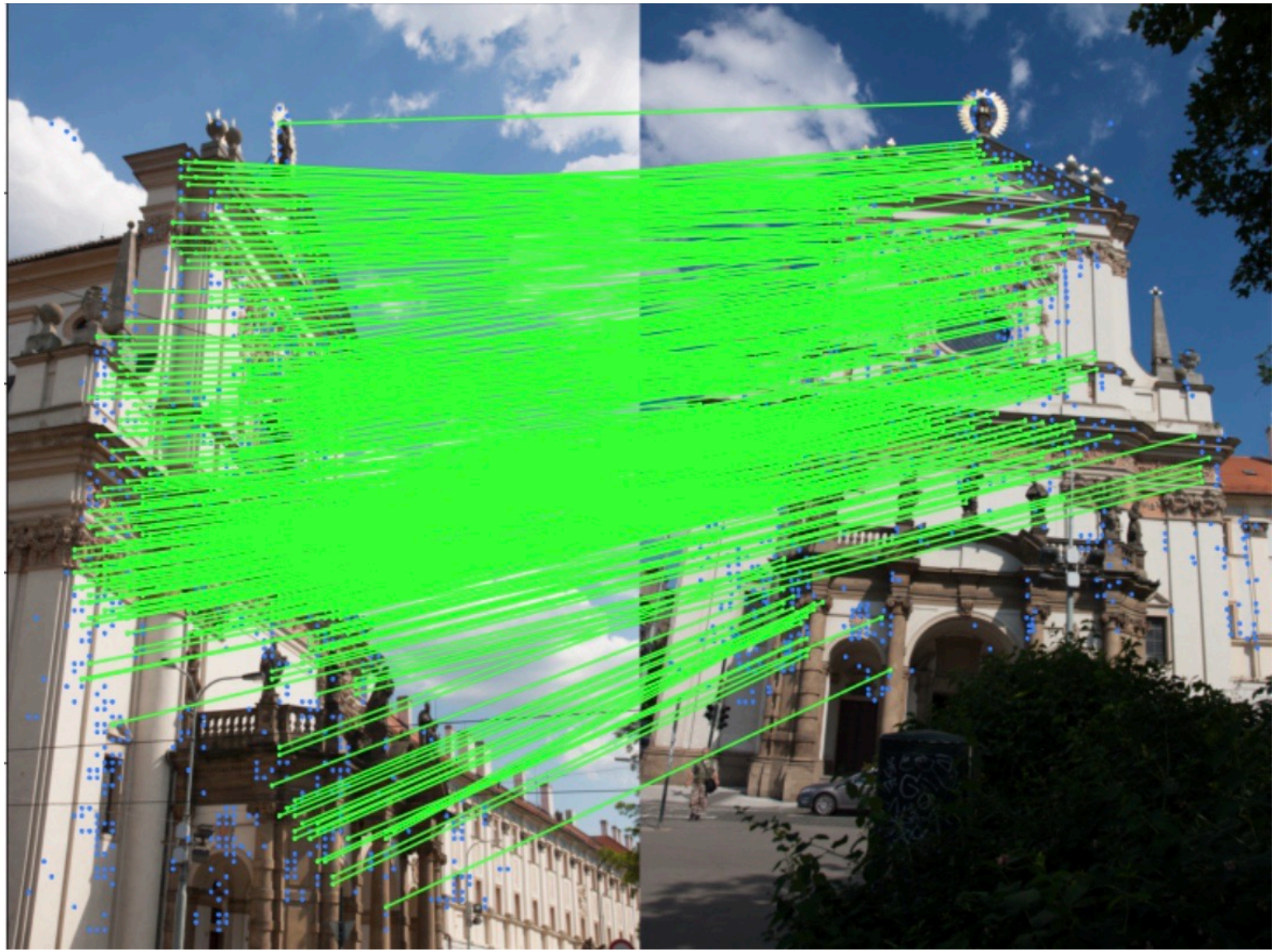
… but more than 1 image!

What are the geometric constraints governing

multiple views of the same scene?

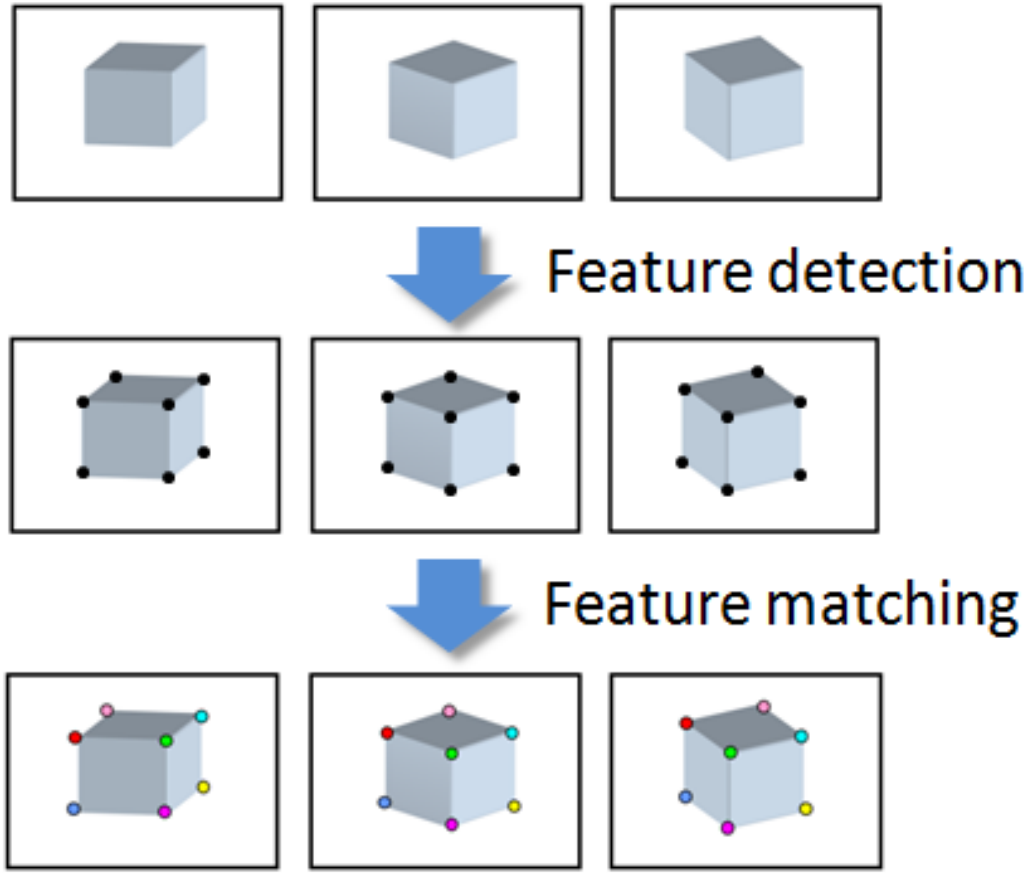
→ **2D correspondences!**

Get 3D structure & motion from 2D correspondences



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

Correspondence estimation



Can use a wide range of features (cf. example in Project 1).

More details in upcoming lectures by Juan Carlos

Get 3D structure & motion from 2D correspondences

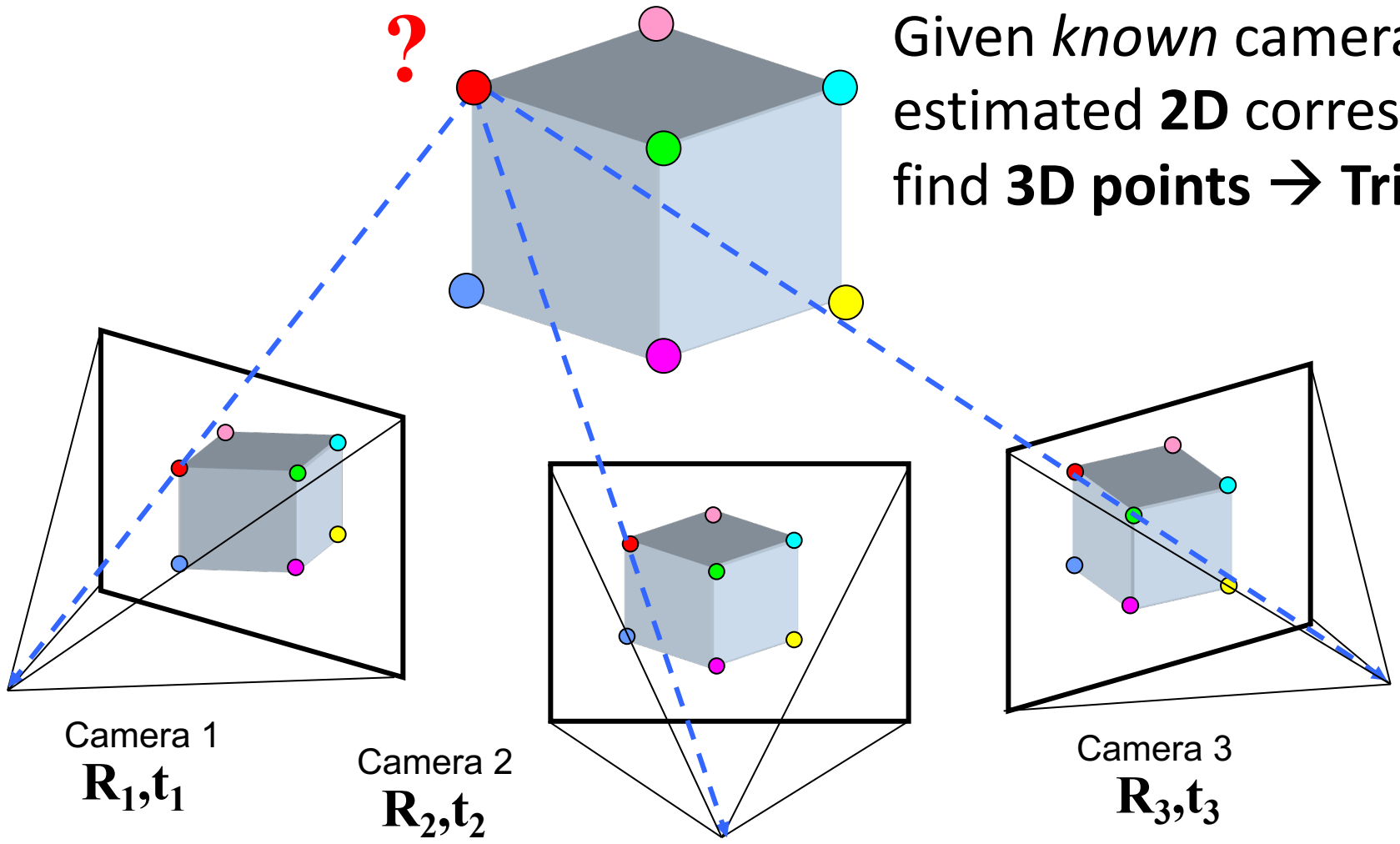
Dubrovnik, Croatia. 4,619 images (out of an initial 57,845 downloaded from Flickr). 3.5M points!
Total reconstruction time: 17.5 hours on 352 cores

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

Why do we care about 3D reconstruction?

- Mapping, Localization, Navigation for Robots, Drones, Cars (cf. also visual SLAM)
- AR (e.g., Hololens) and VR (e.g., Oculus)
- Movies (special FX), Digital Preservation, "Photo Tourism", ...
- Software: COLMAP (SfM), orb-slam2 / g2o / gtsam (SLAM)
- Hot topic in industry & academia (top category at CVPR)

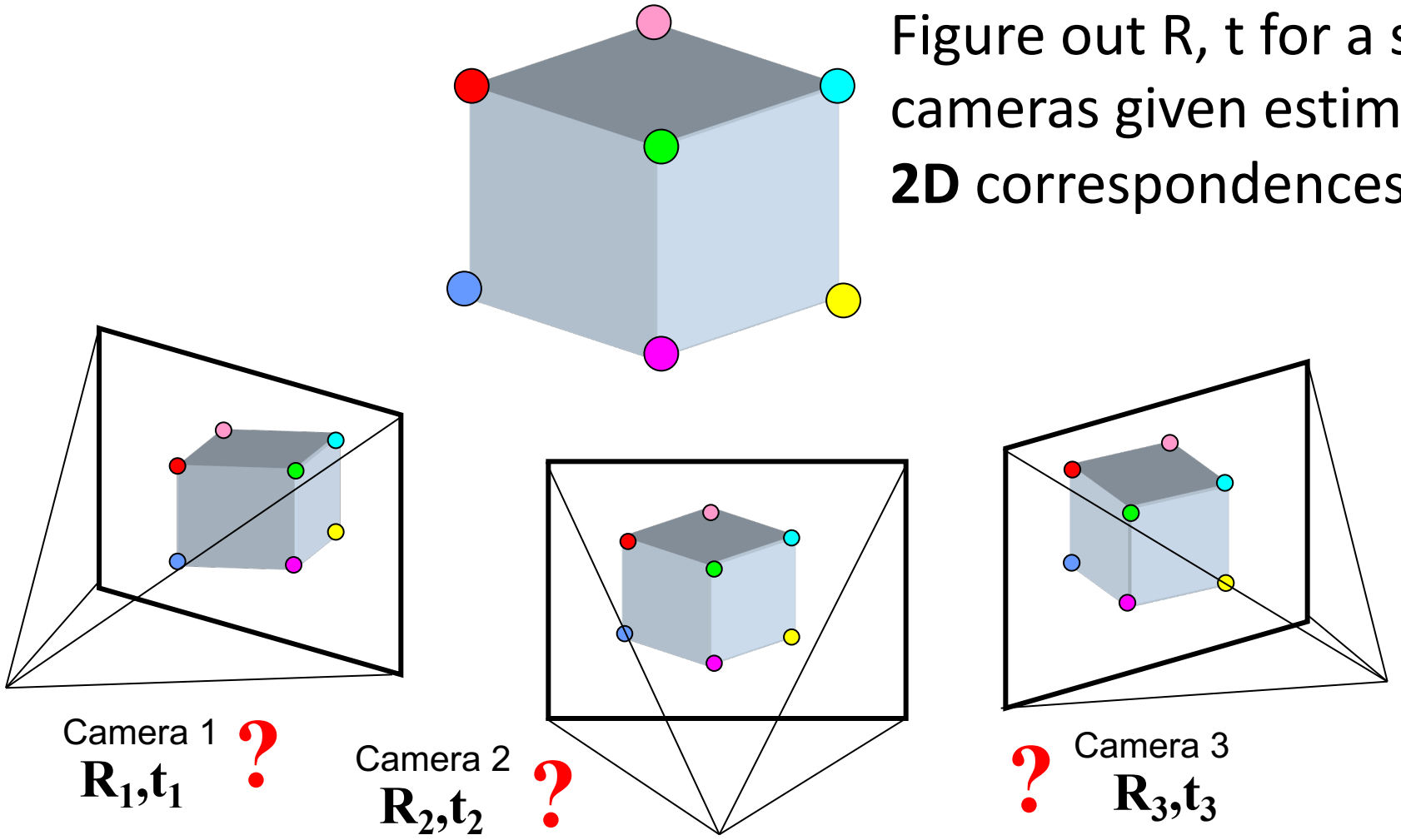
Multi-view geometry problems



Recovering structure:
Given *known* cameras and estimated **2D** correspondences, find **3D points** → **Triangulation**

Slide credit:
Noah Snavely

Multi-view geometry problems



Motion:

Figure out R, t for a set of cameras given estimated **2D** correspondences

Slide credit:
Noah Snavely

What will we learn today?

Triangulation

Epipolar geometry

Stereo

Structure-from-Motion (SfM)

What will we learn today?

Triangulation

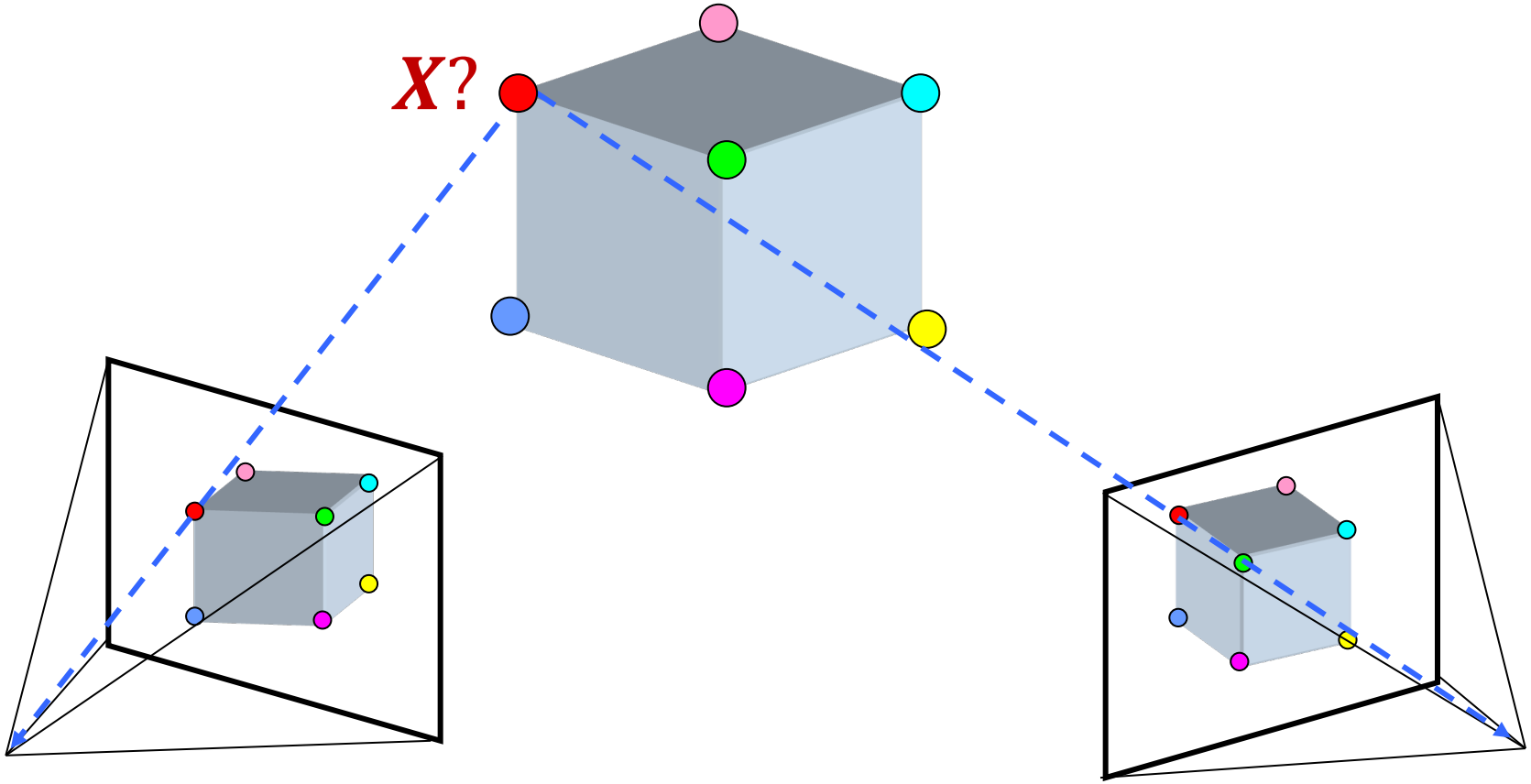
Epipolar geometry

Stereo

Structure-from-Motion (SfM)

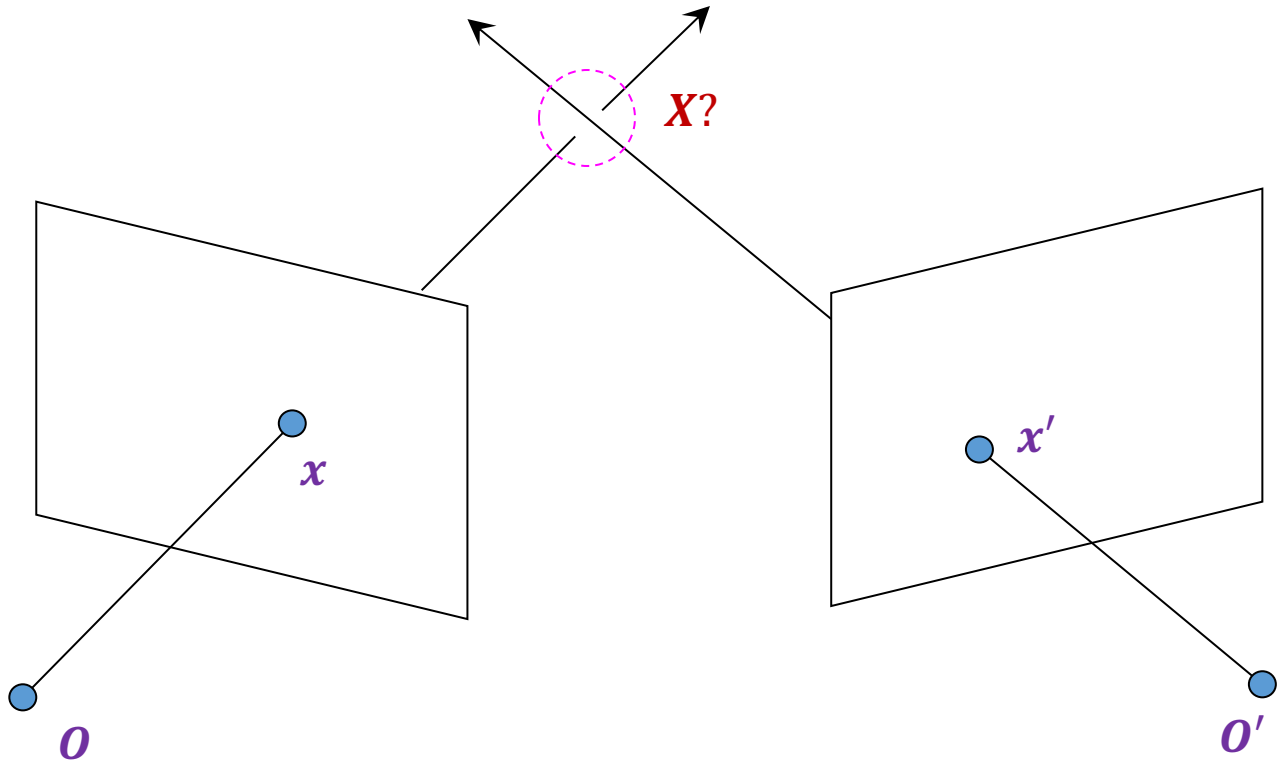
Triangulation

Given projections of a 3D point in two or more images (with known camera matrices), find the coordinates of the point



Triangulation

Given projections of a 3D point in two or more images (with known camera matrices), find the coordinates of the point

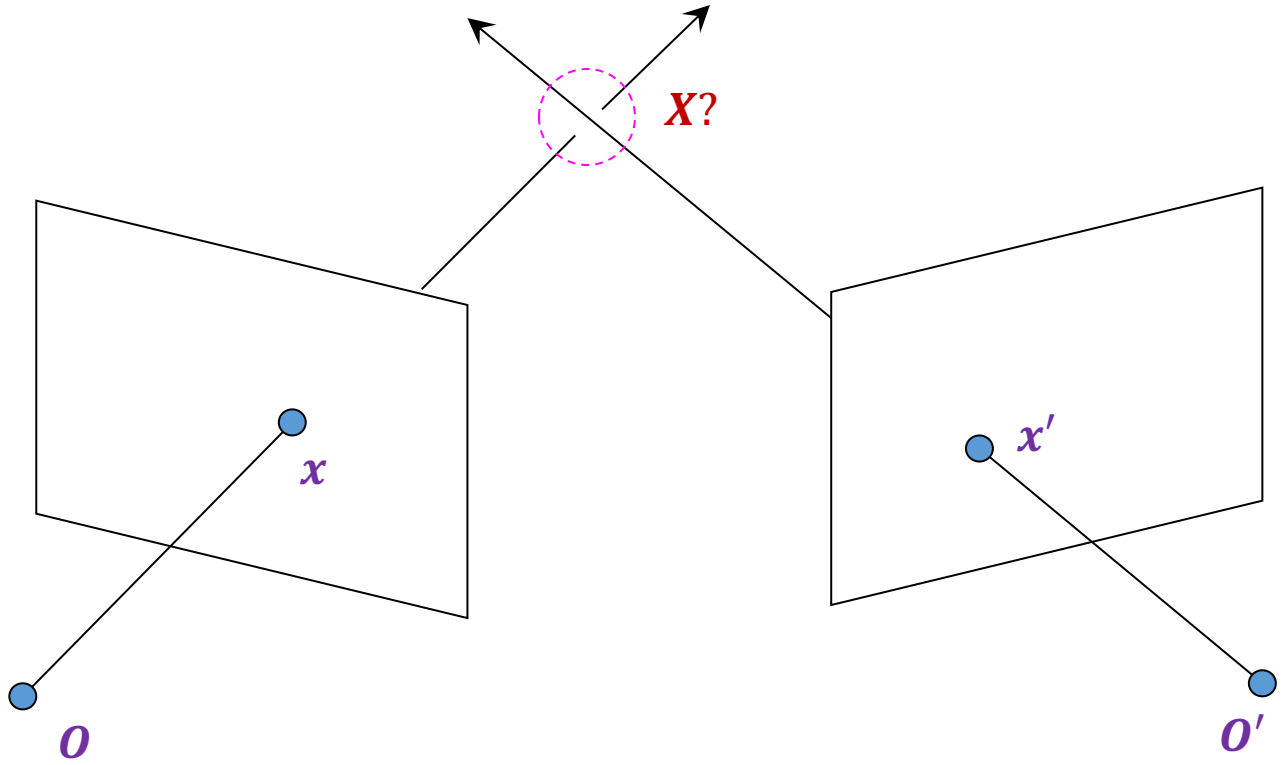


Triangulation

We want to intersect the two visual rays corresponding to $\boldsymbol{x}$ and $\boldsymbol{x'}$

But do they always intersect *exactly*?

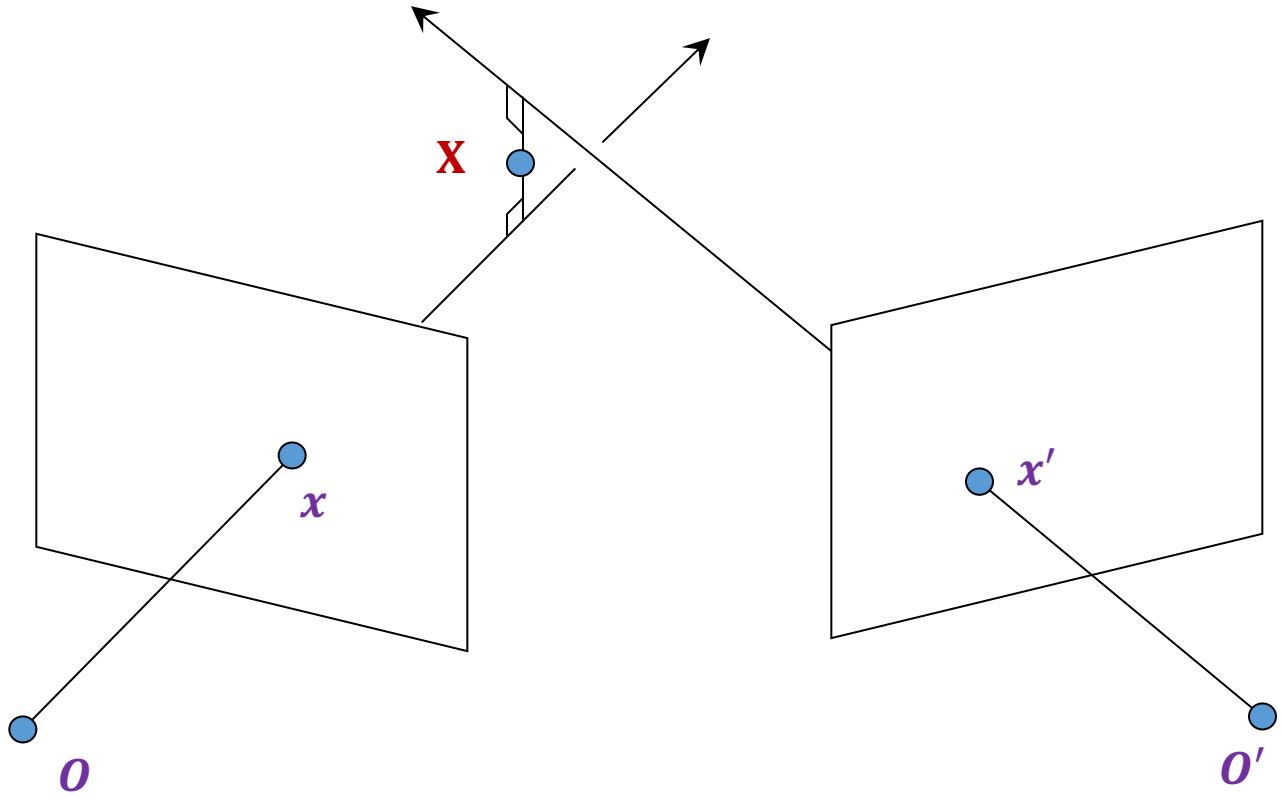
No! Noise in 2D matching or numerical errors



Triangulation: linear approach

Find the shortest segment connecting the two viewing rays

Let $\mathbf{X}$ be the midpoint of that segment: solve for $\mathbf{X}$!



what is $\mathbf{A}$?

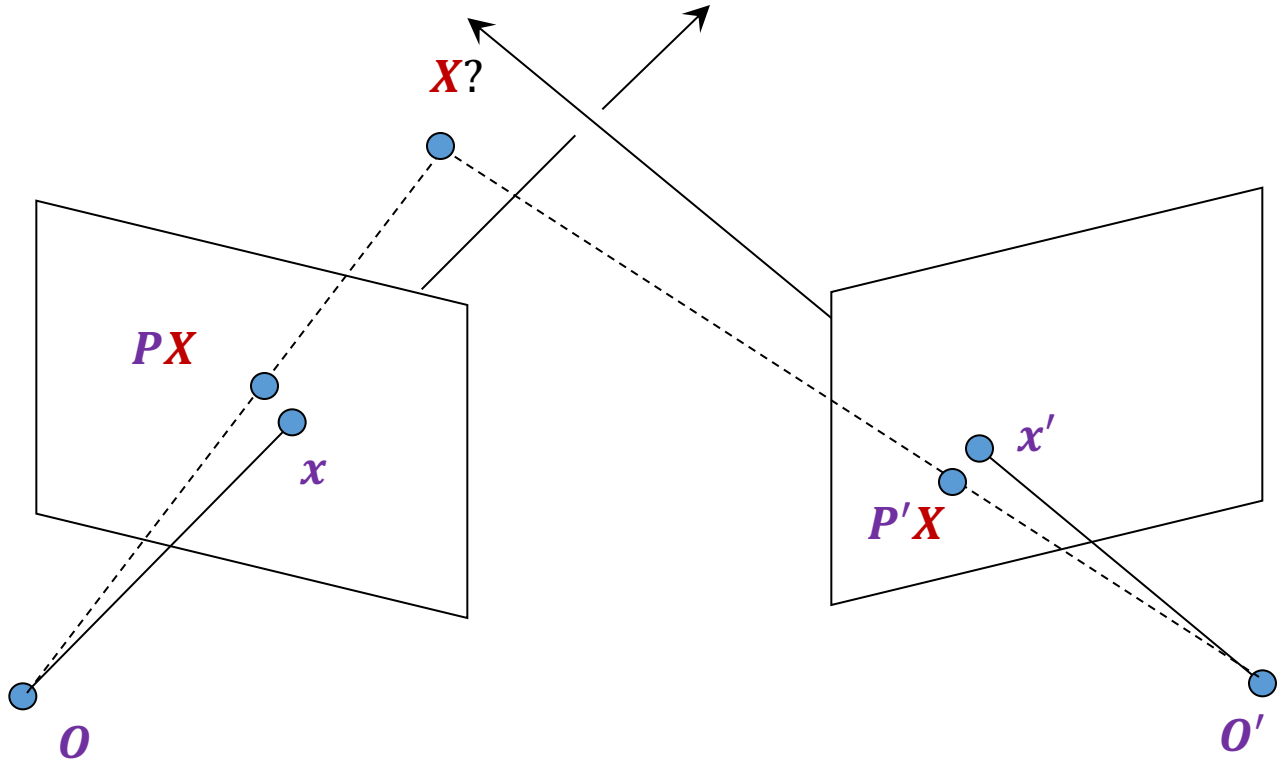
(answer in appendix)

As for calibration: constraints ($\mathbf{x} \sim \mathbf{P}\mathbf{X}$, $\mathbf{x}' \sim \mathbf{P}'\mathbf{X}$) → $\mathbf{A}\mathbf{X} = \mathbf{0}$ → SVD of $\mathbf{A}$

Triangulation: *non-linear* approach

Find $\boldsymbol{X}$ that minimizes the 2D reprojection errors

$$\|\text{proj}(\boldsymbol{P}\boldsymbol{X}) - \boldsymbol{x}\|^2 + \|\text{proj}(\boldsymbol{P}'\boldsymbol{X}) - \boldsymbol{x}'\|^2$$



What will we learn today?

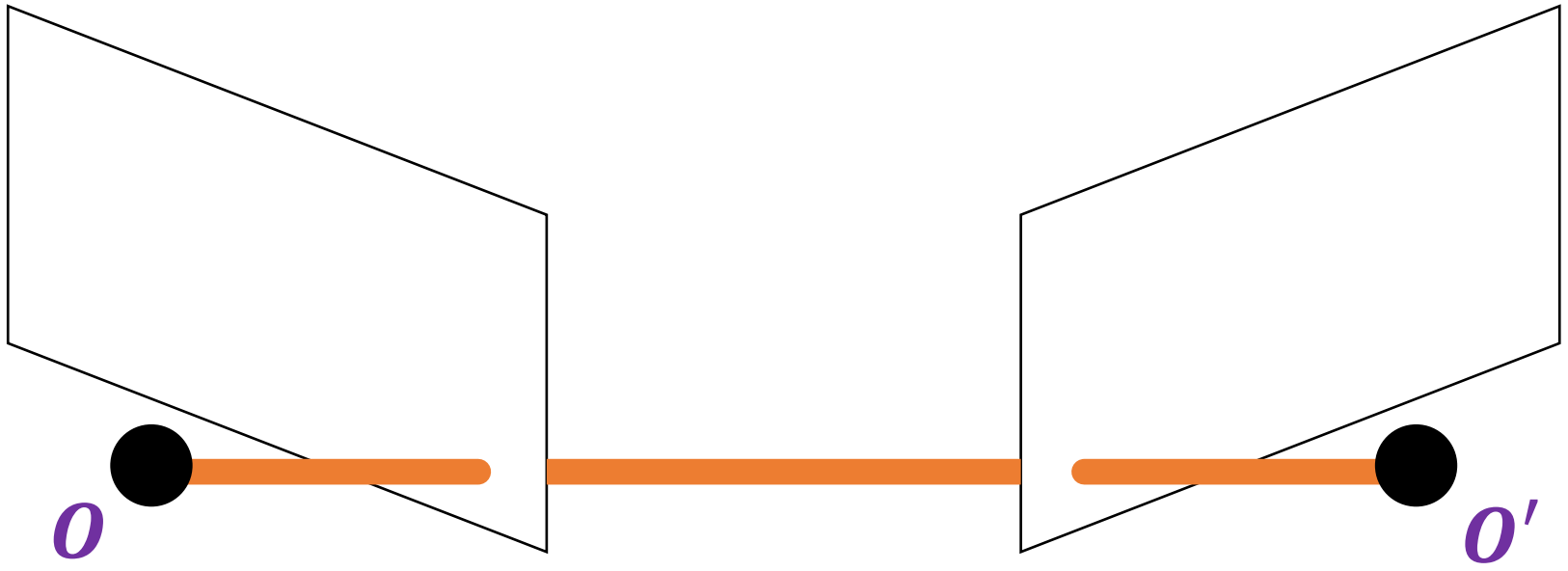
Triangulation

Epipolar geometry

Stereo

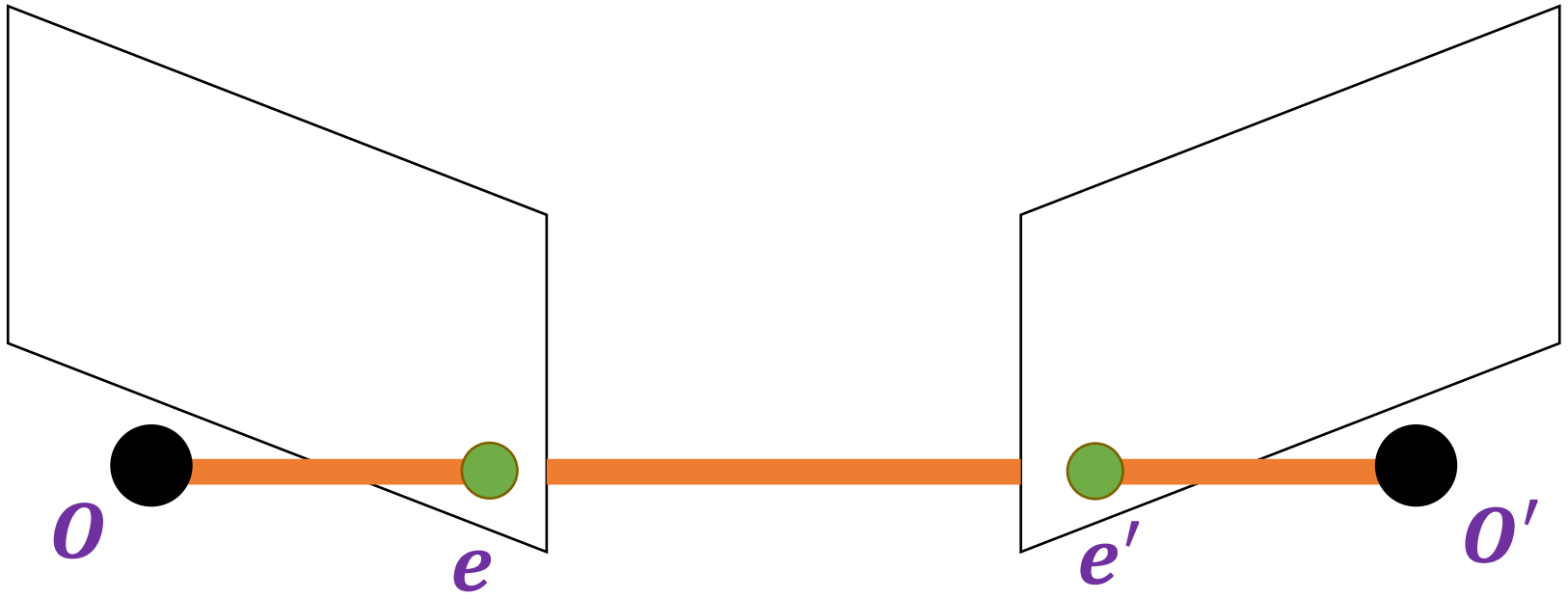
Structure-from-Motion (SfM)

Epipolar geometry setup



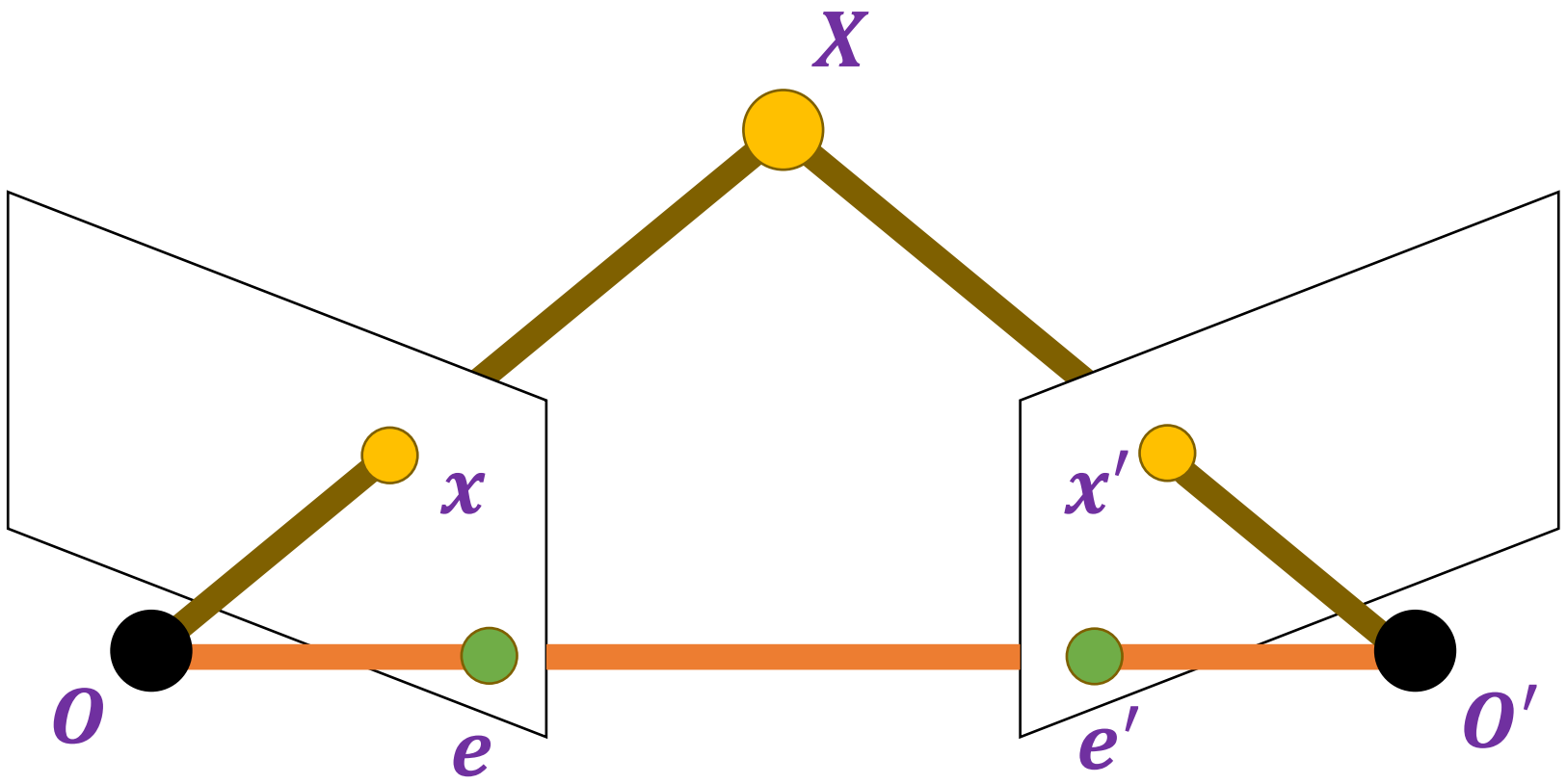
- Suppose we have two cameras with centers O, O'
- The **baseline** is the line connecting the origins

Epipolar geometry setup



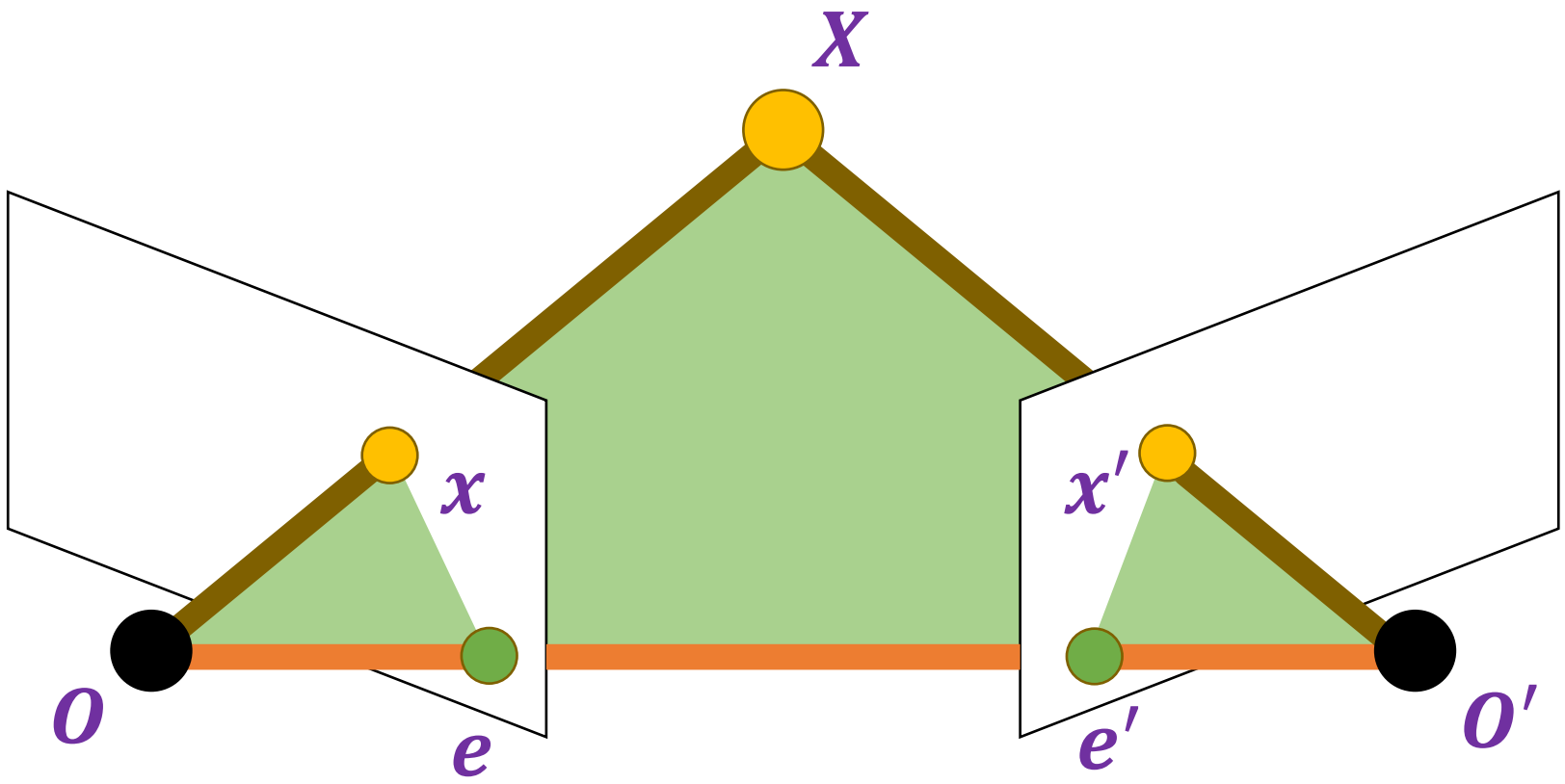
- **Epipoles e, e'** are where the baseline intersects the image planes
- Equivalently: epipoles are projections of the other camera in each view

Epipolar geometry setup



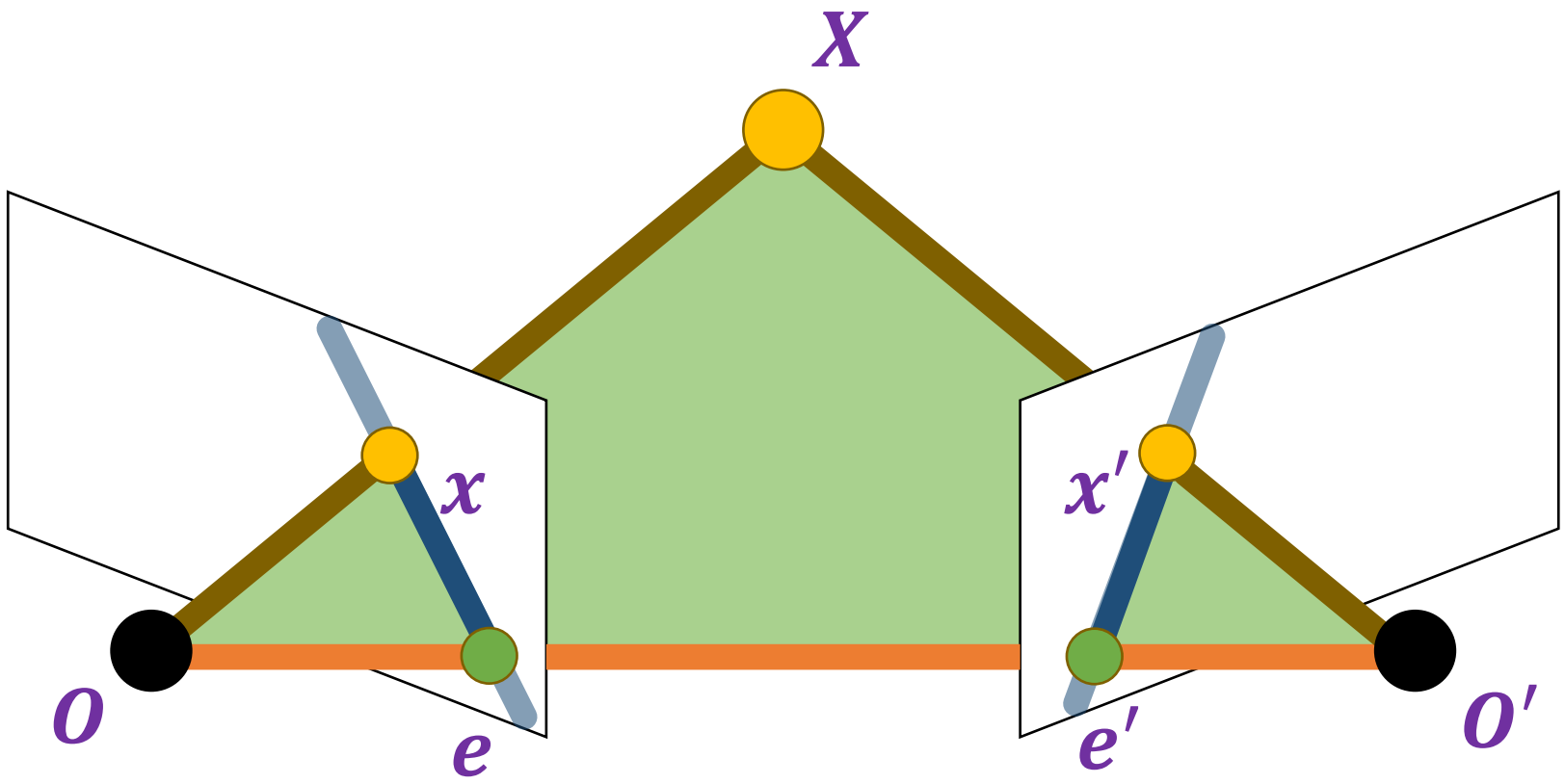
- Consider a **point** X, which projects to x and x'

Epipolar geometry setup



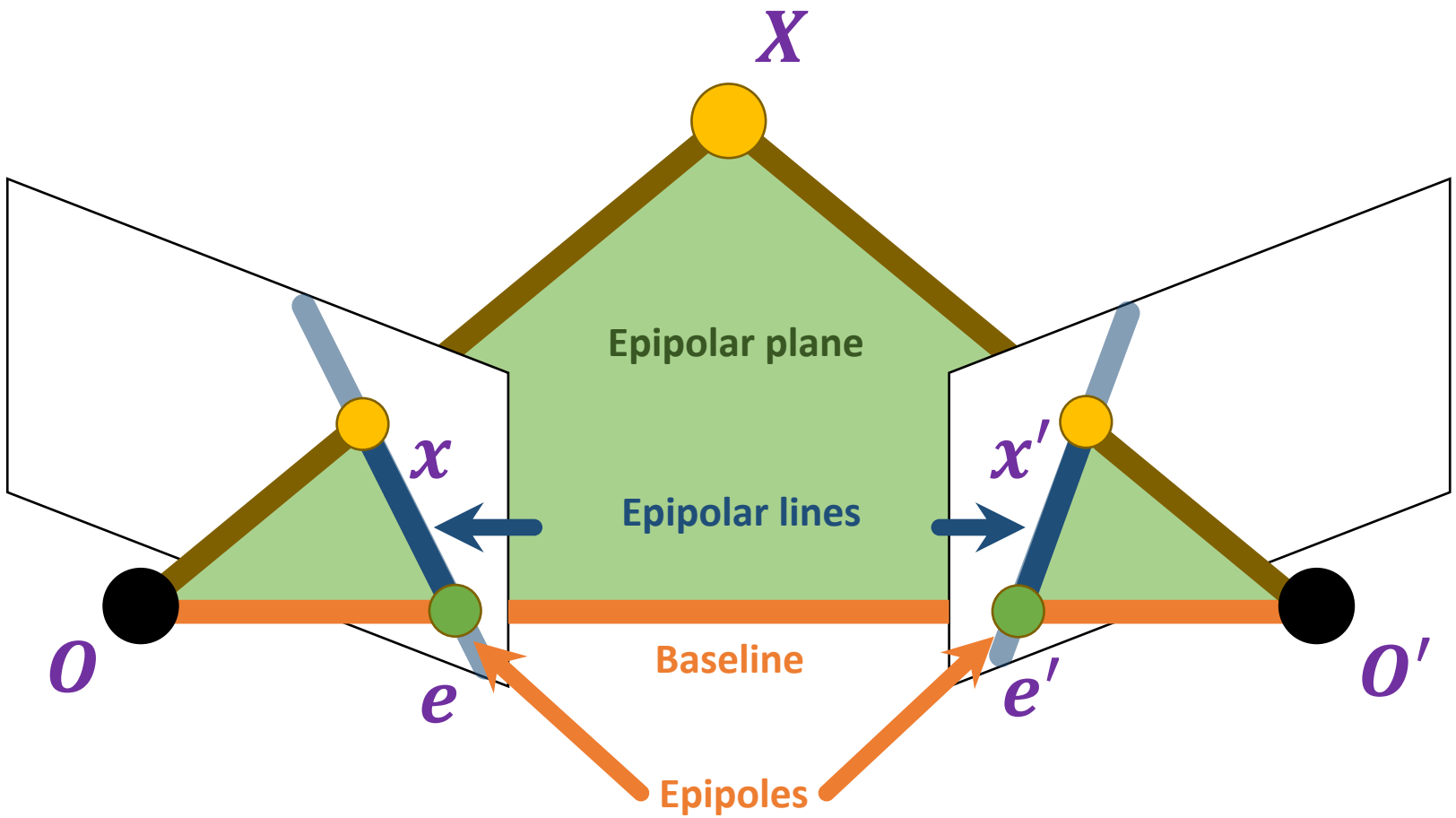
- The plane formed by X, O, and O' is called an epipolar plane

Epipolar geometry setup

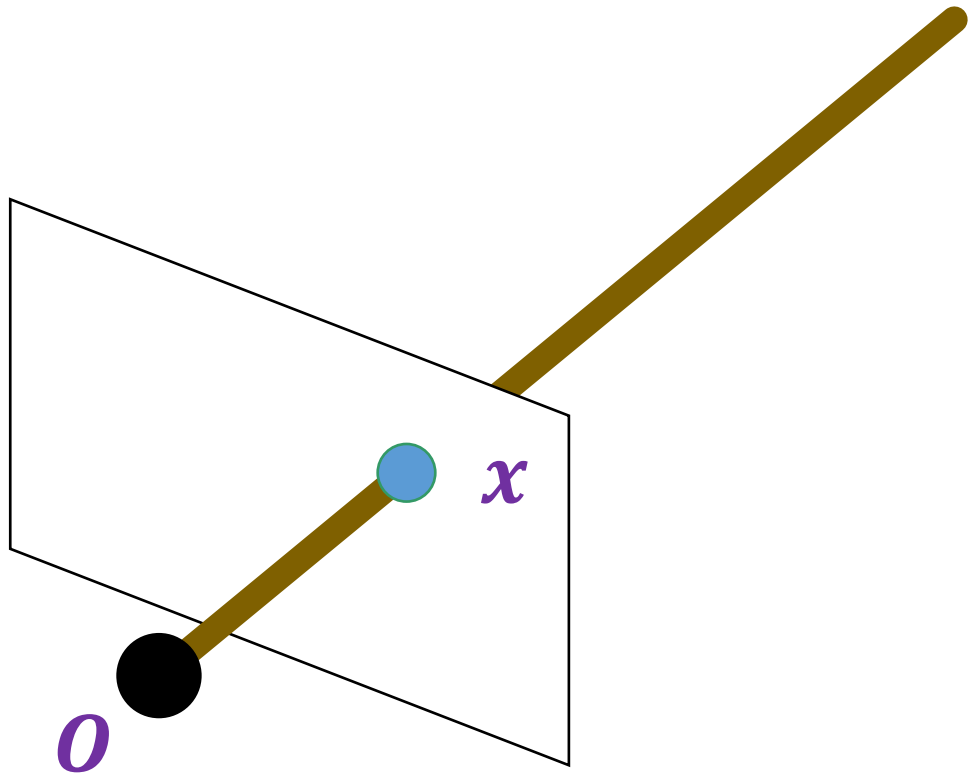


- **Epipolar lines** connect the epipoles to the projections of X
- Equivalently, they are intersections of the epipolar plane with the image planes, come in pairs (for x and x’)

Epipolar geometry setup: Summary

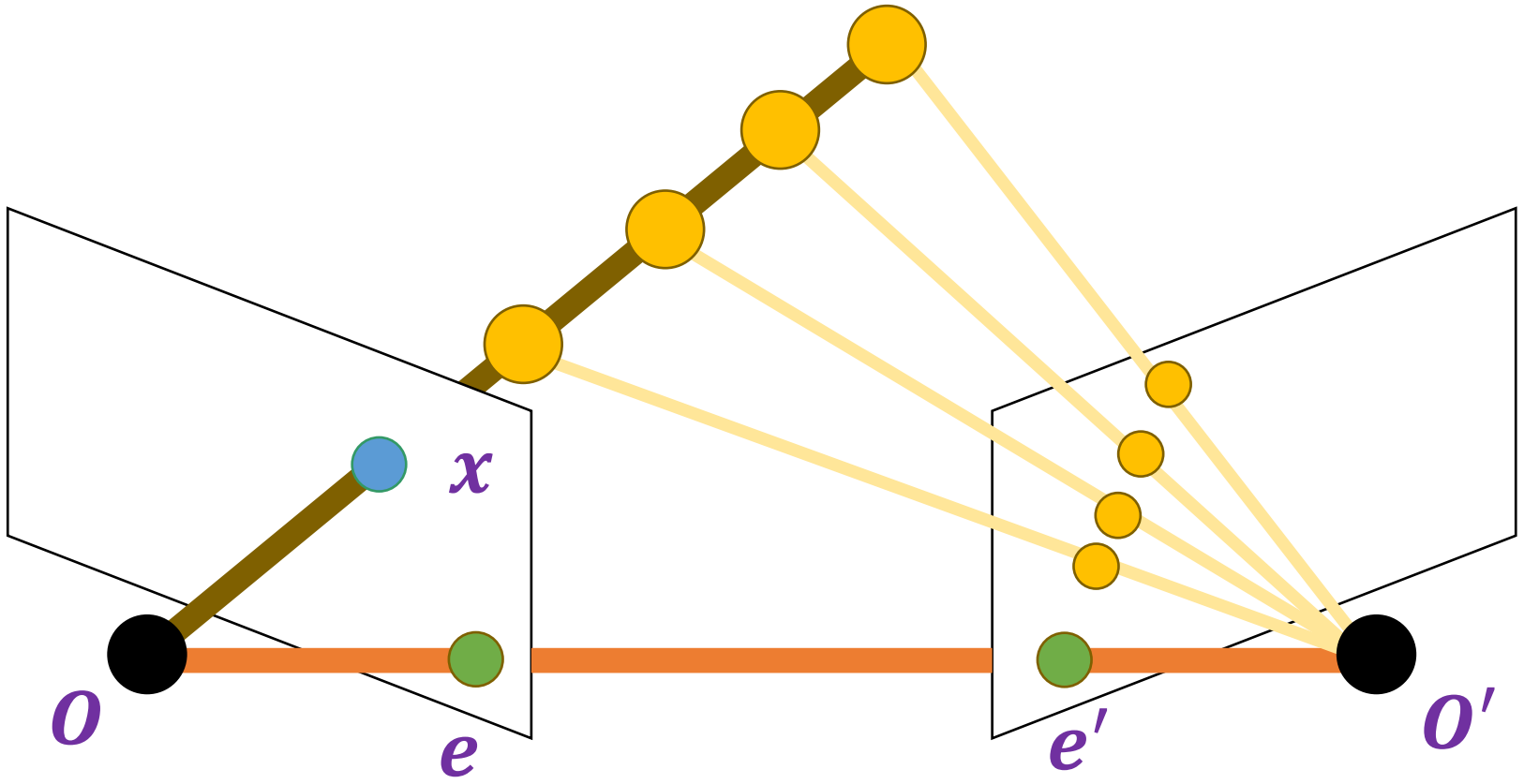


Epipolar constraint



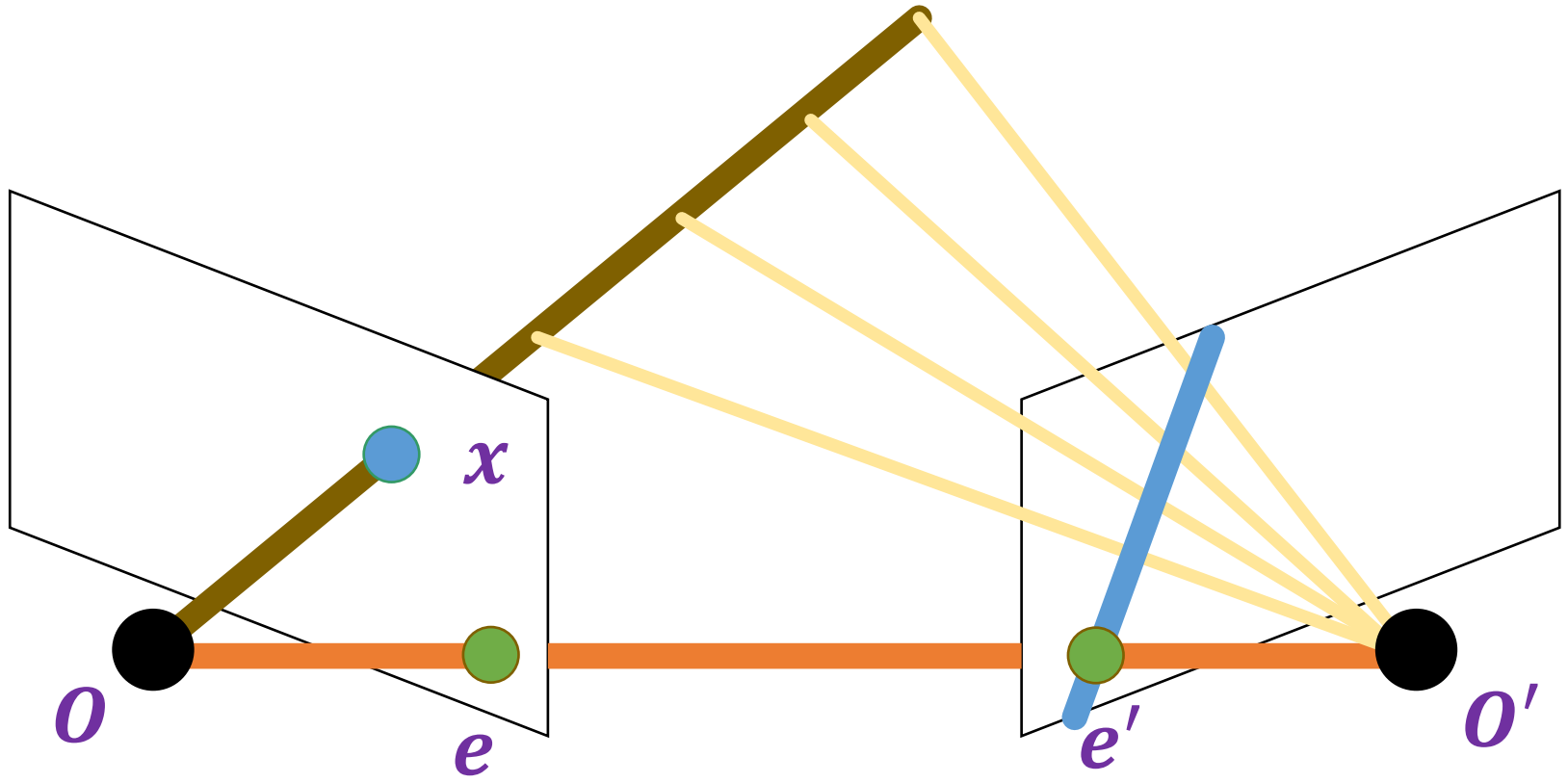
- Suppose we observe a single point x in one image

Epipolar constraint



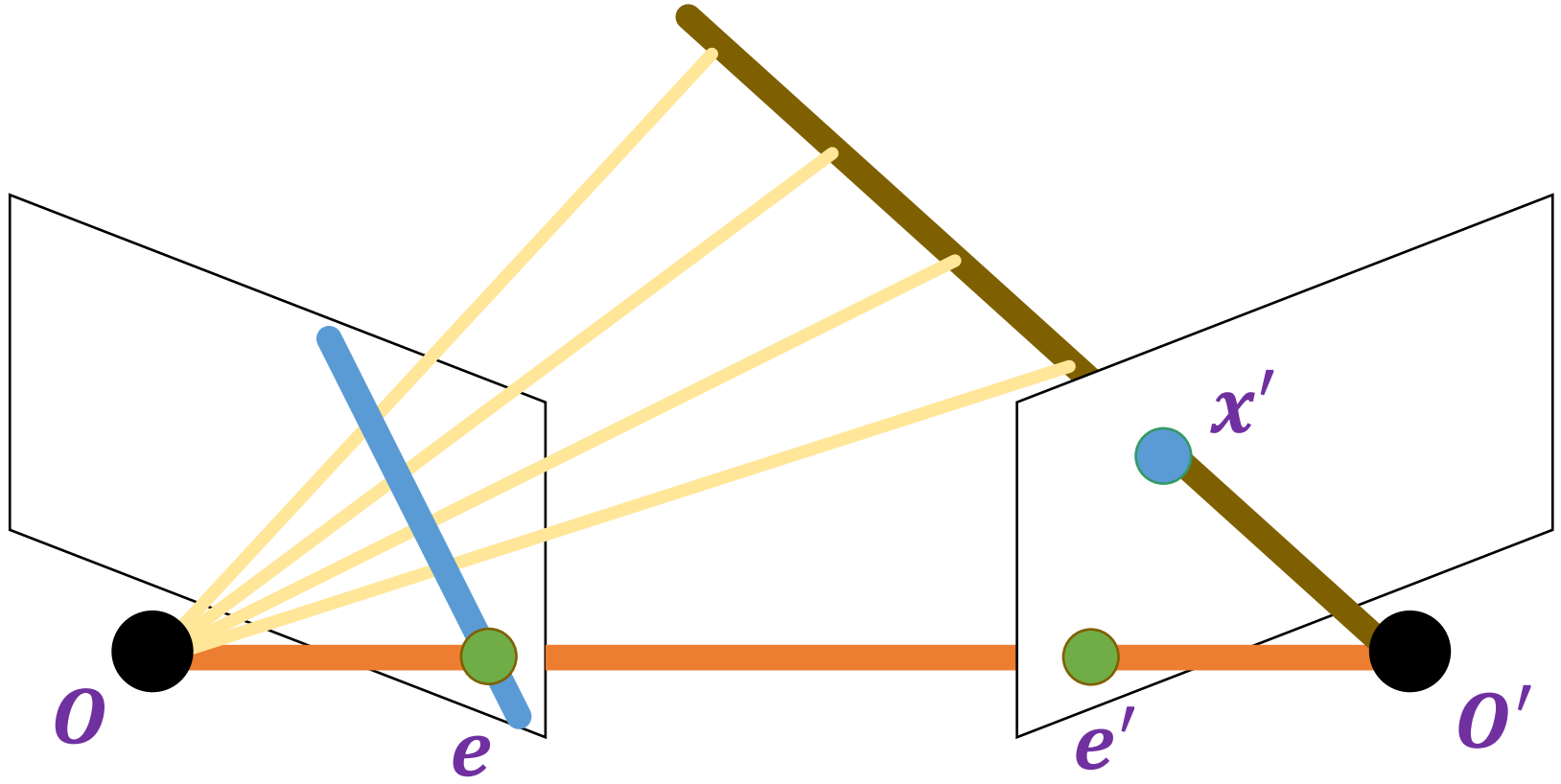
- Where can we find the x' corresponding to x in the other image?

Epipolar constraint



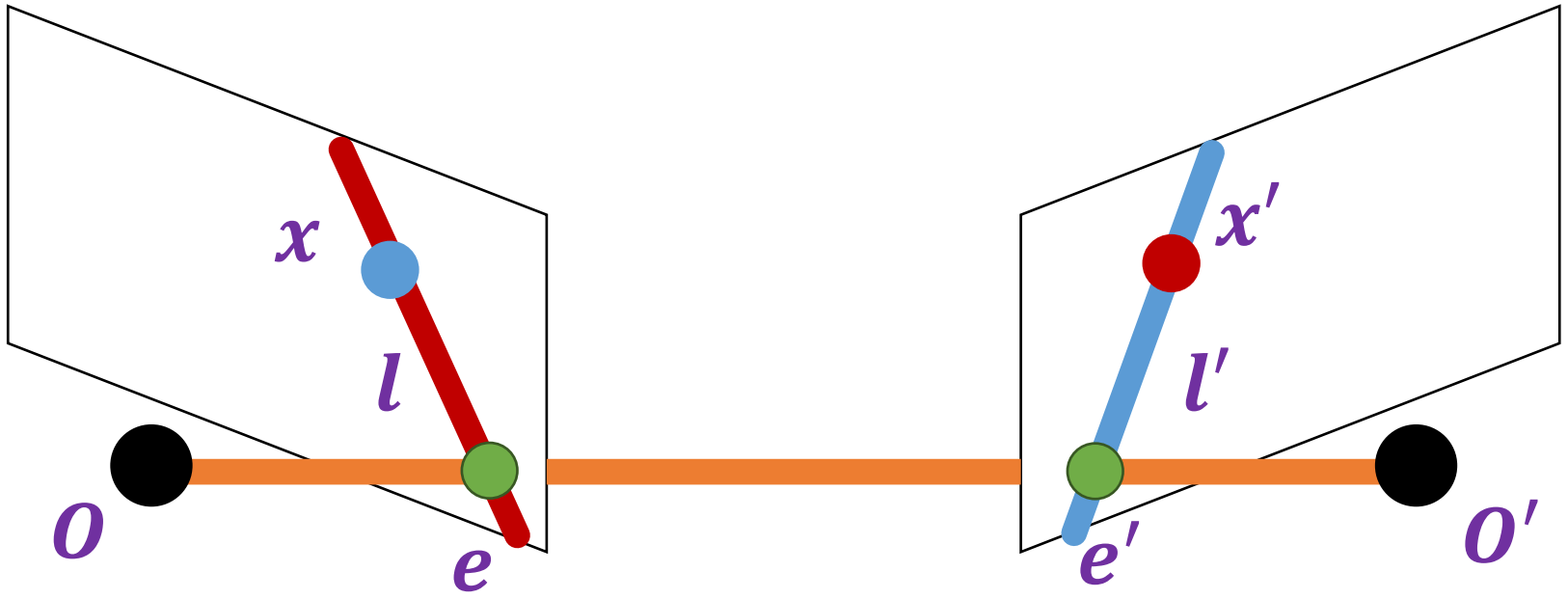
- Where can we find the x' corresponding to x in the other image?
- Along the **epipolar line** corresponding to x (projection of visual ray connecting O with x into the second image plane)

Epipolar constraint



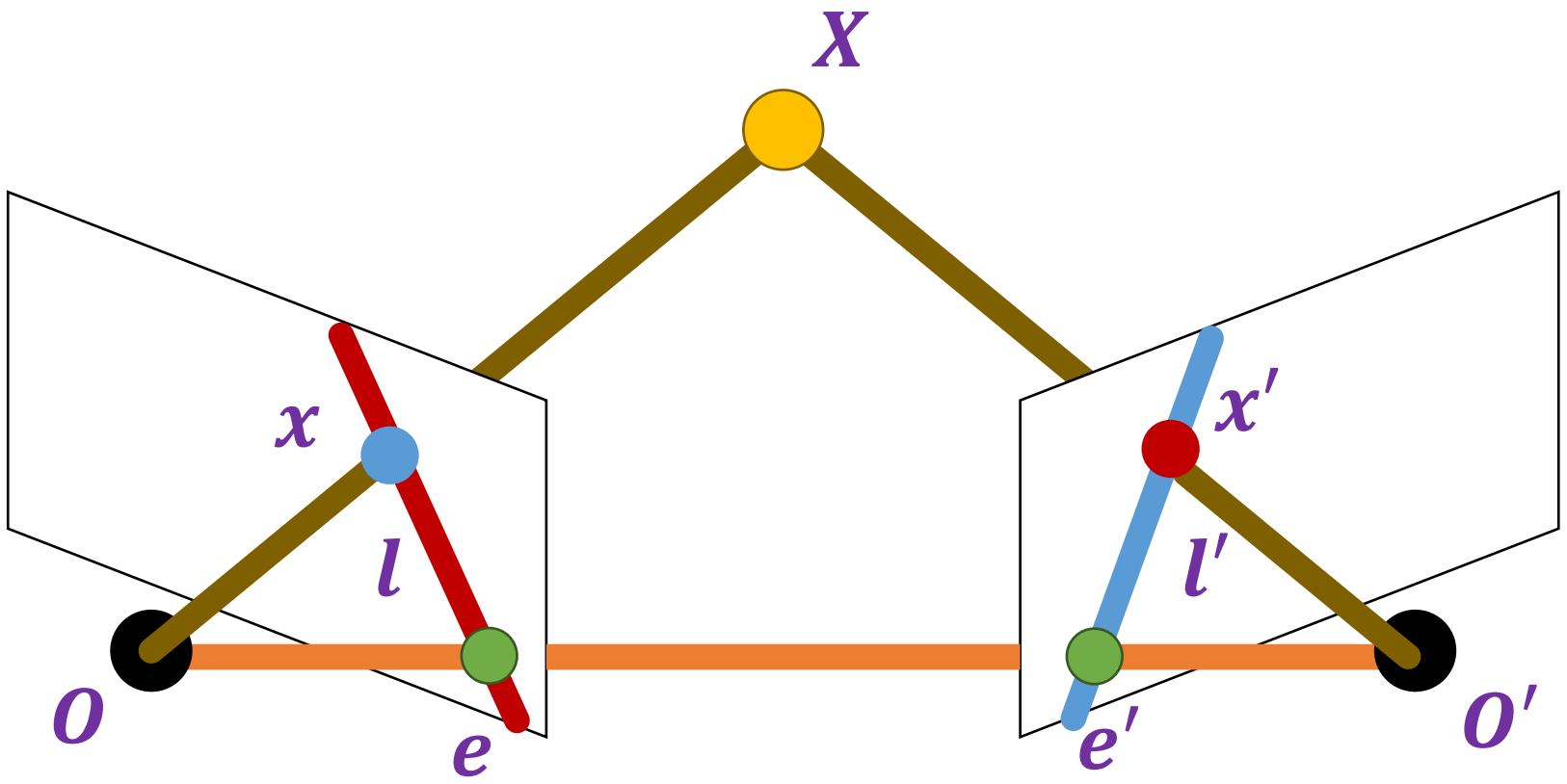
- Similarly, all points in the left image corresponding to x' have to lie along the epipolar line corresponding to x'

Epipolar constraint



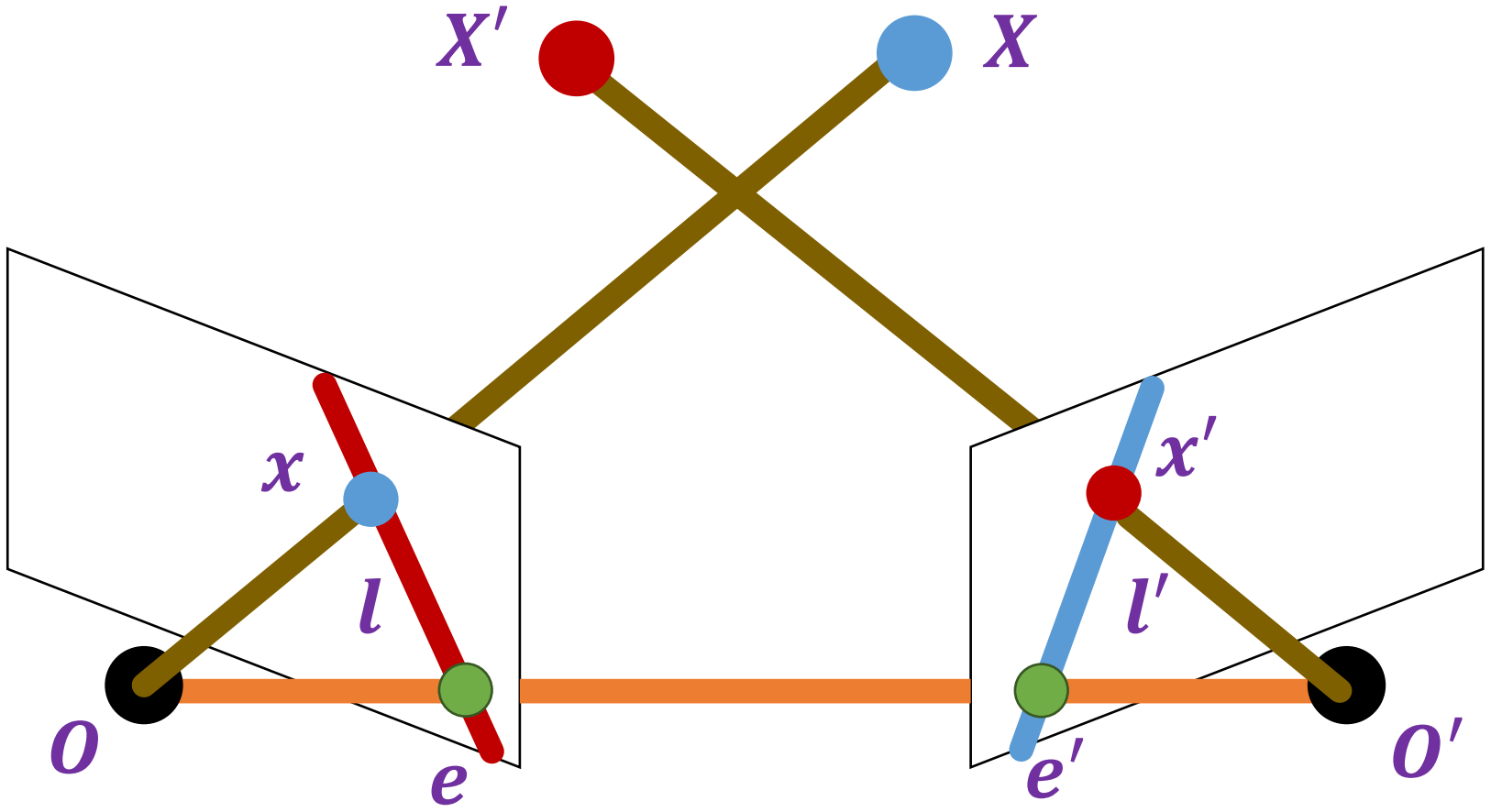
- Potential matches for x have to lie on the matching epipolar line l' and vice-versa → need only to search along 1D epipolar line for matching!

Epipolar constraint



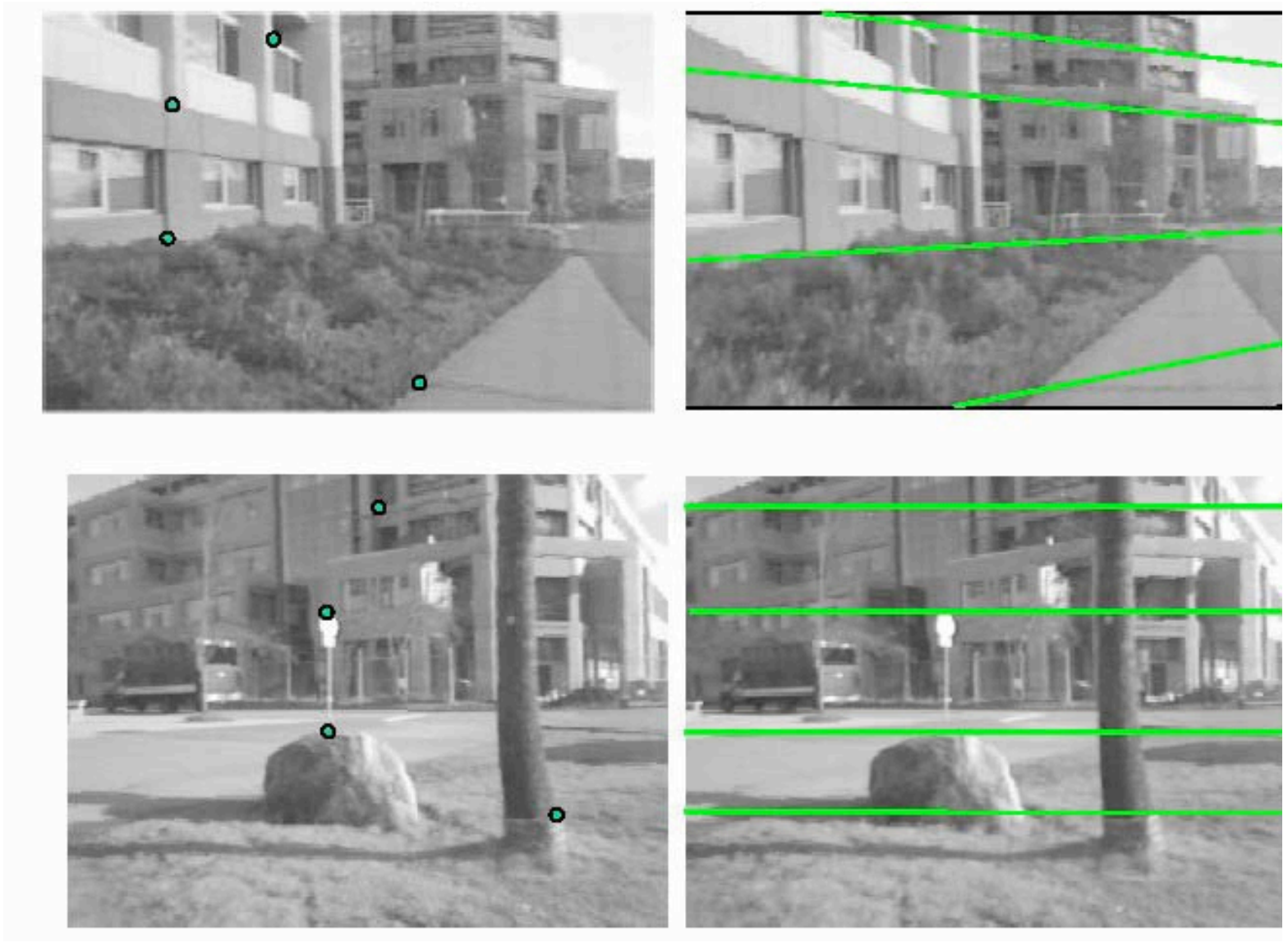
- Whenever two points x and x' lie on matching epipolar lines l and l', the ***visual rays*** corresponding to them meet in space, i.e., x and x' ***could be*** projections of the same 3D point X

Epipolar constraint

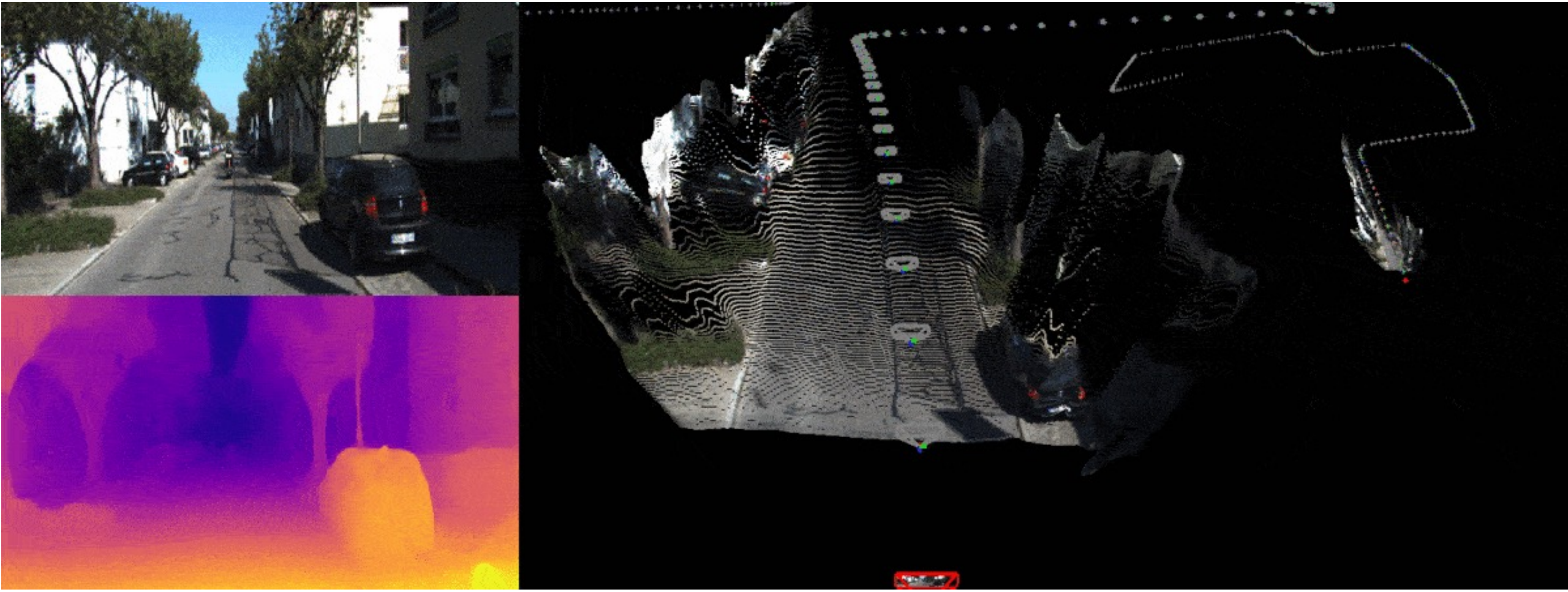


- Caveat: if x and x' satisfy the epipolar constraint, this doesn't mean they *have to be* projections of the same 3D point

Epipolar constraint: Example



Epipolar Geometry & Deep Learning



Multi-Frame Self-Supervised Depth Estimation with Transformers (CVPR 2022)

Vitor Guizilini, Rares Ambrus, Dian Chen, Sergey Zakharov, Adrien Gaidon

Epipolar Geometry & Deep Learning

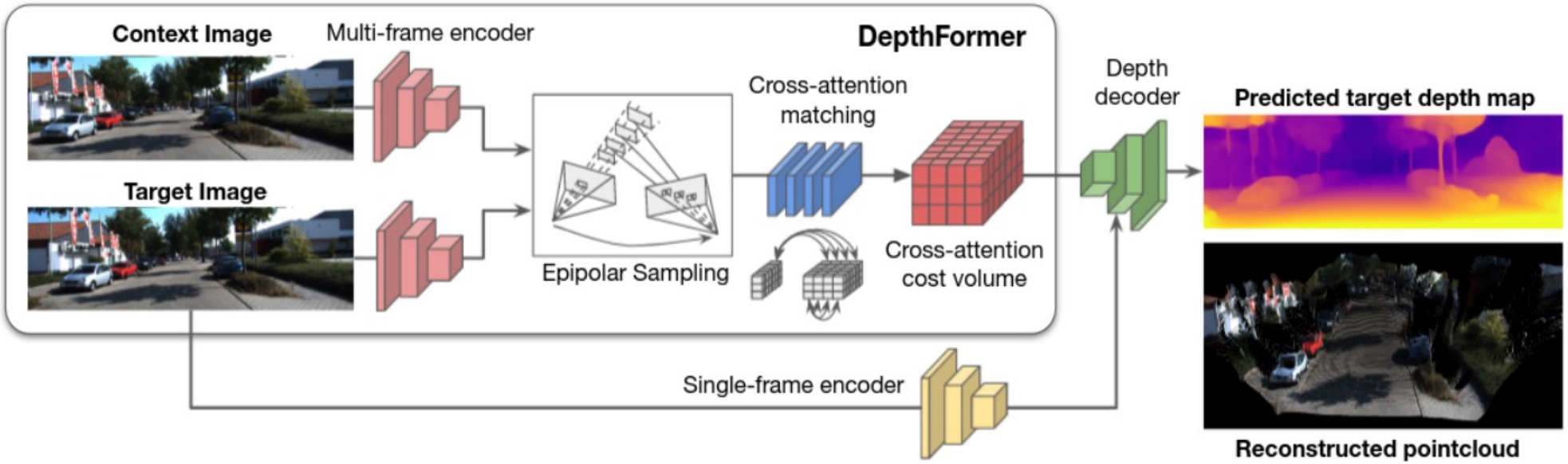
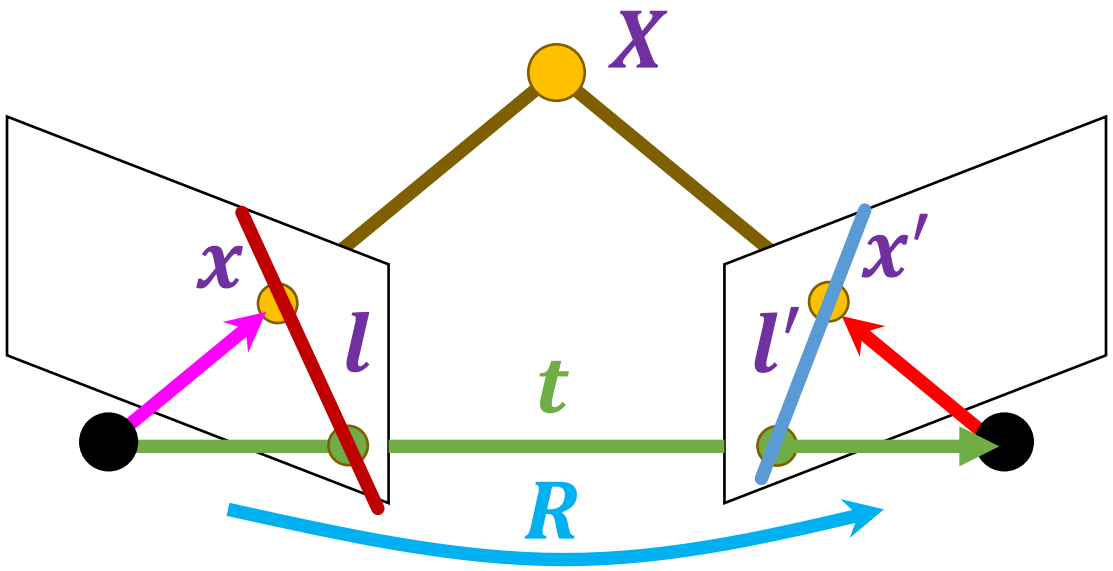


Figure 1. **Our DepthFormer architecture** achieves state-of-the-art multi-frame self-supervised monocular depth estimation by **improving feature matching** across images during cost volume generation.

Multi-Frame Self-Supervised Depth Estimation with Transformers (CVPR 2022)

Vitor Guizilini, Rares Ambrus, Dian Chen, Sergey Zakharov, Adrien Gaidon

The Epipolar Constraint as an Equation



$x'^T F x = 0$ where $F = K'^{-T} E K^{-1}$ is called the ✨**Fundamental Matrix**✨ [1]

and $E = [t]_\times R$ is the **Essential Matrix** [2]

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

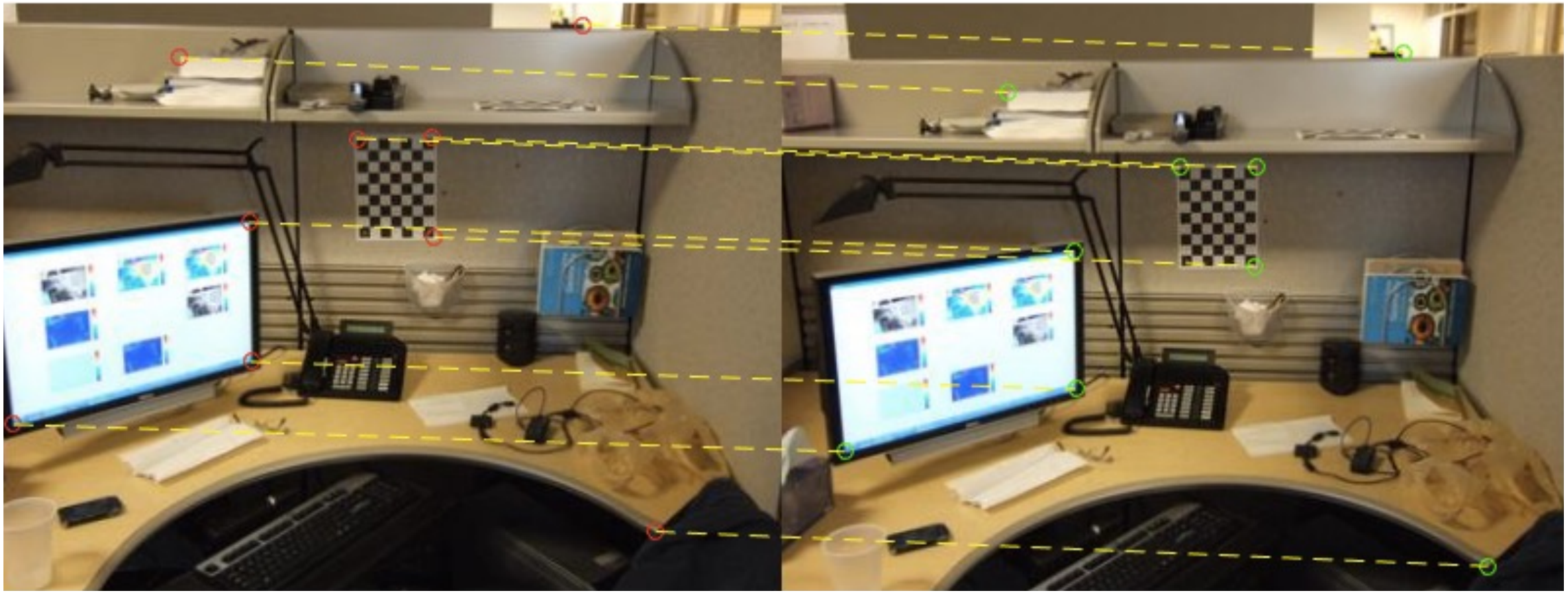
(sketch of proof in appendix)

[1] Faugeras et al., (1992), Hartley (1992)

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

Estimating the fundamental matrix - teaser

- Given: correspondences $\boldsymbol{x} = (x, y, 1)^T$ and $\boldsymbol{x}' = (x', y', 1)^T$



Estimating the fundamental matrix - teaser

- Given: 2D correspondences $\boldsymbol{x} = (x, y, 1)^T$ and $\boldsymbol{x}' = (x', y', 1)^T$
- Constraints: $\boldsymbol{x}'^T \boldsymbol{F} \boldsymbol{x} = 0$ (1 per correspondence, how many needed?)
- Boils down to another homogeneous linear equation $\boldsymbol{AX} = \boldsymbol{0}$
- Recast once more into total least squares (Sz.A.2.1) due to noise
- SVD gives the solution as usual + enables enforcing rank 2 constraint by replacing smallest singular value with 0
- This "algebraic" algorithm is called "[normalized] 8-point algorithm" (R. Hartley. In defense of the eight-point algorithm. TPAMI 1997)
- As in calibration and homography fitting: non-linear "geometric" optimization (of reprojected distances) is more precise
- Can be made robust to outliers via the RANSAC algorithm
- See appendix, H&Z ch. 9, Szeliski 11.3, or take ***CS231A*** for more!

From epipolar geometry to camera calibration

Estimating the fundamental matrix is known as “weak calibration”

If we know the calibration matrices (K, K') of the two cameras, we can estimate the ***essential matrix***: $\boldsymbol{E = K'^{T} F K}$

The essential matrix gives us the relative rotation and translation between the cameras, or their ***extrinsic parameters*** ($\mathbf{E} = [\boldsymbol{t}]_{\times}\boldsymbol{R}$)

Alternatively, if the calibration matrices are known (or in practice, if good initial guesses of the intrinsics are available), the five-point algorithm can be used to estimate relative camera pose

D. Nister. An efficient solution to the five-point relative pose problem. IEEE Trans. PAMI, 2004

What will we learn today?

Triangulation

Epipolar geometry

Stereo

Structure-from-Motion (SfM)

Example configuration: Converging cameras

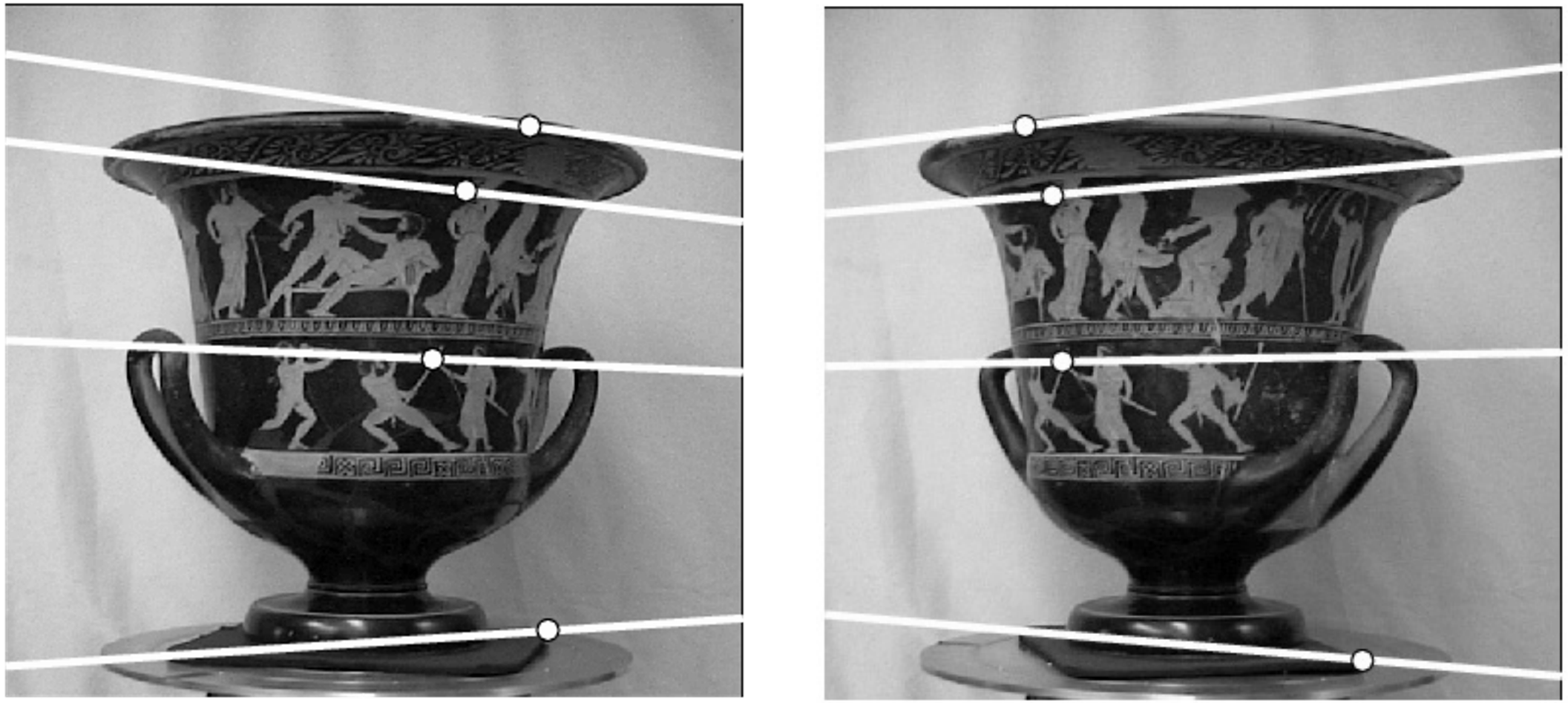
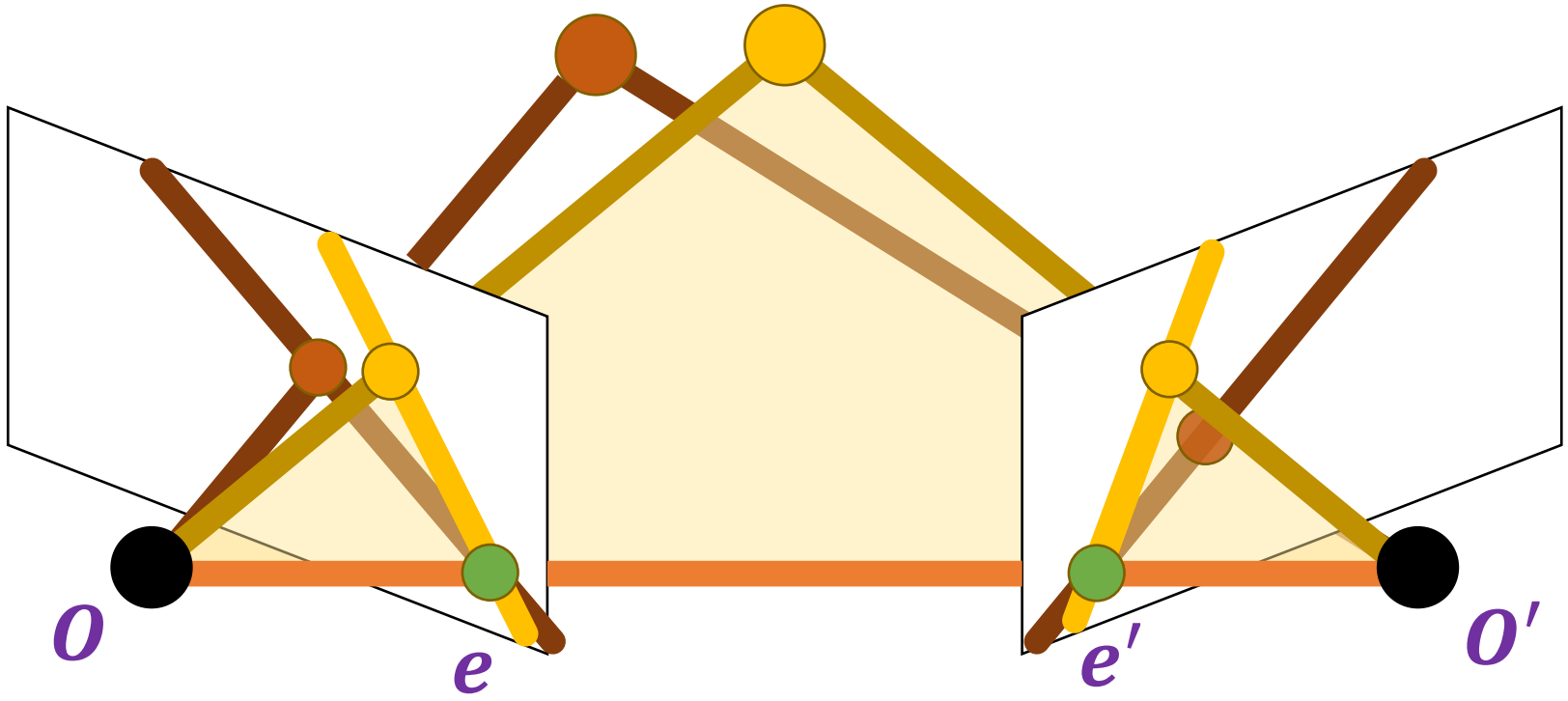


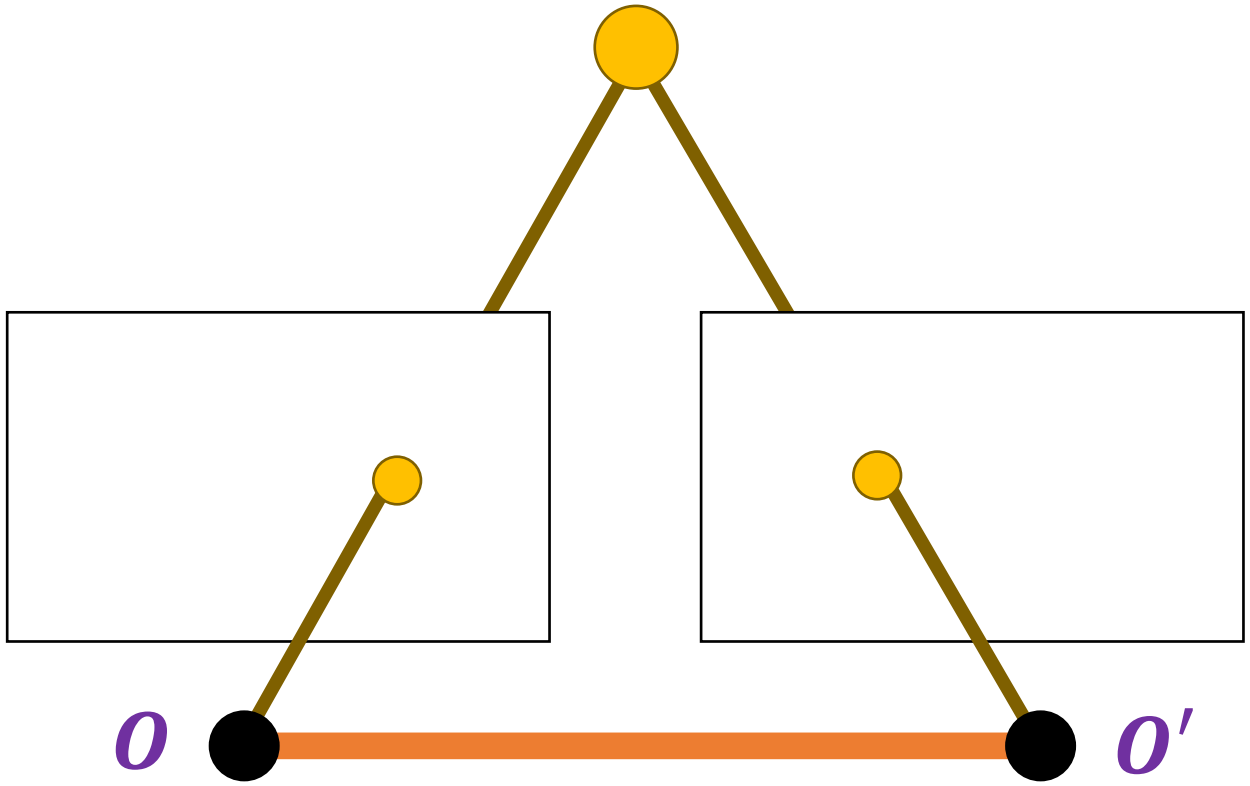
Image Credit: Hartley & Zisserman

Example configuration: Converging cameras



- Epipoles are finite, may be visible in the image

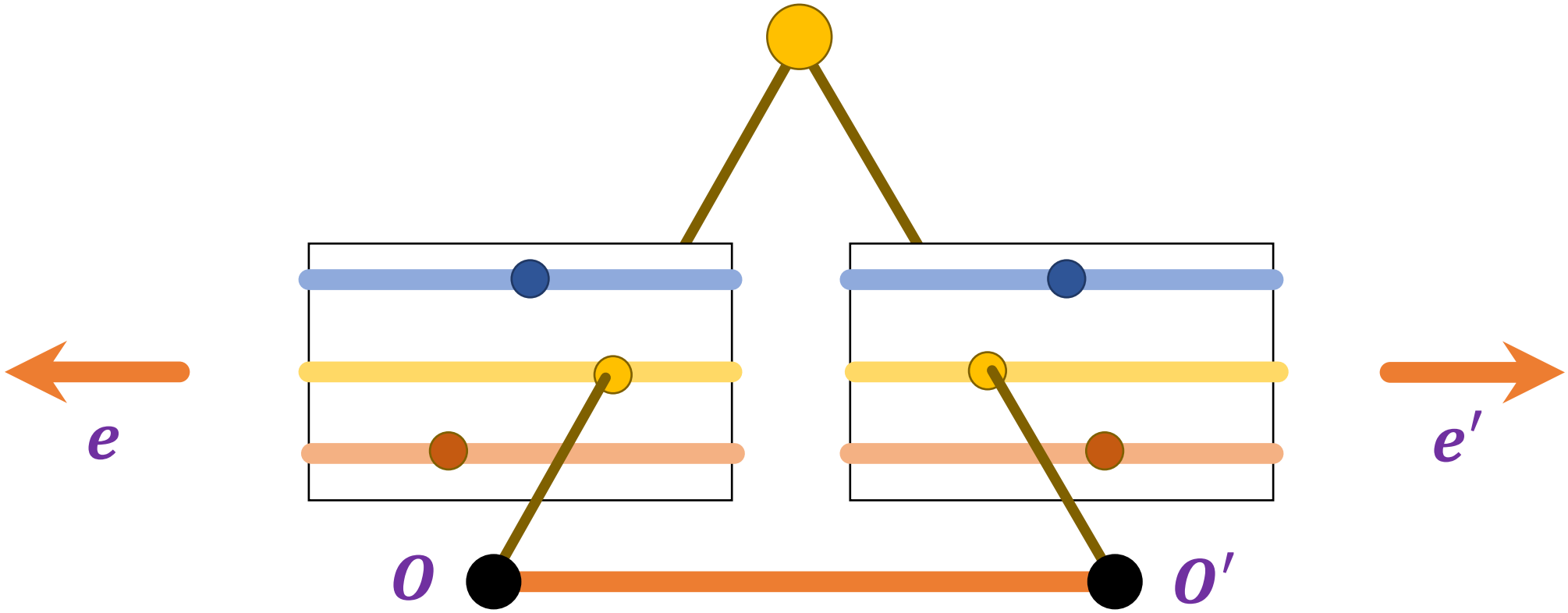
Example configuration: Motion parallel to image plane



Where are the epipoles?

What do the epipolar lines look like?

Example configuration: Motion parallel to image plane



Epipoles *infinitely* far away!

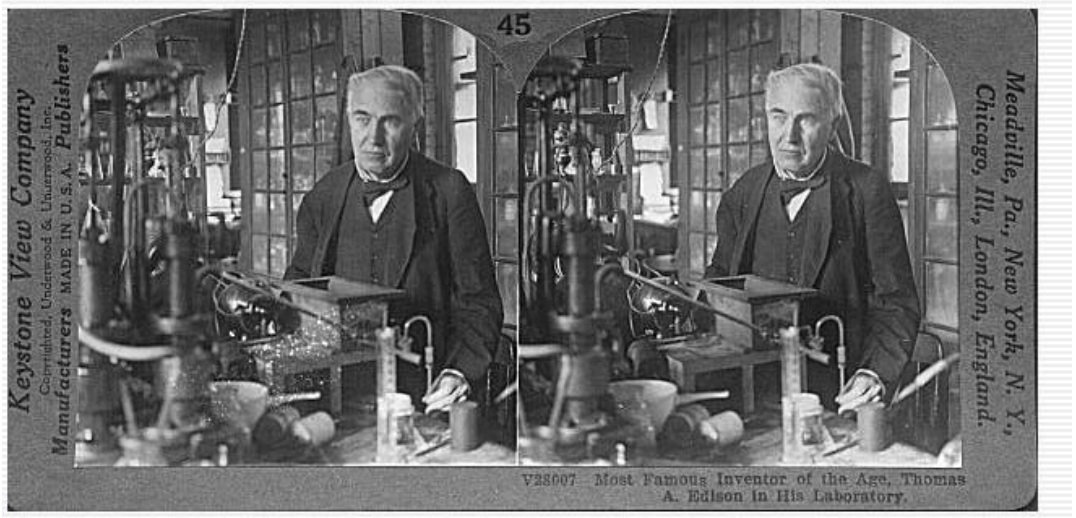
Epipolar lines parallel: "scan lines"

→ Stereo = easier fronto-parallel special case!

History: Stereograms

Humans can fuse pairs of images to get a sensation of depth

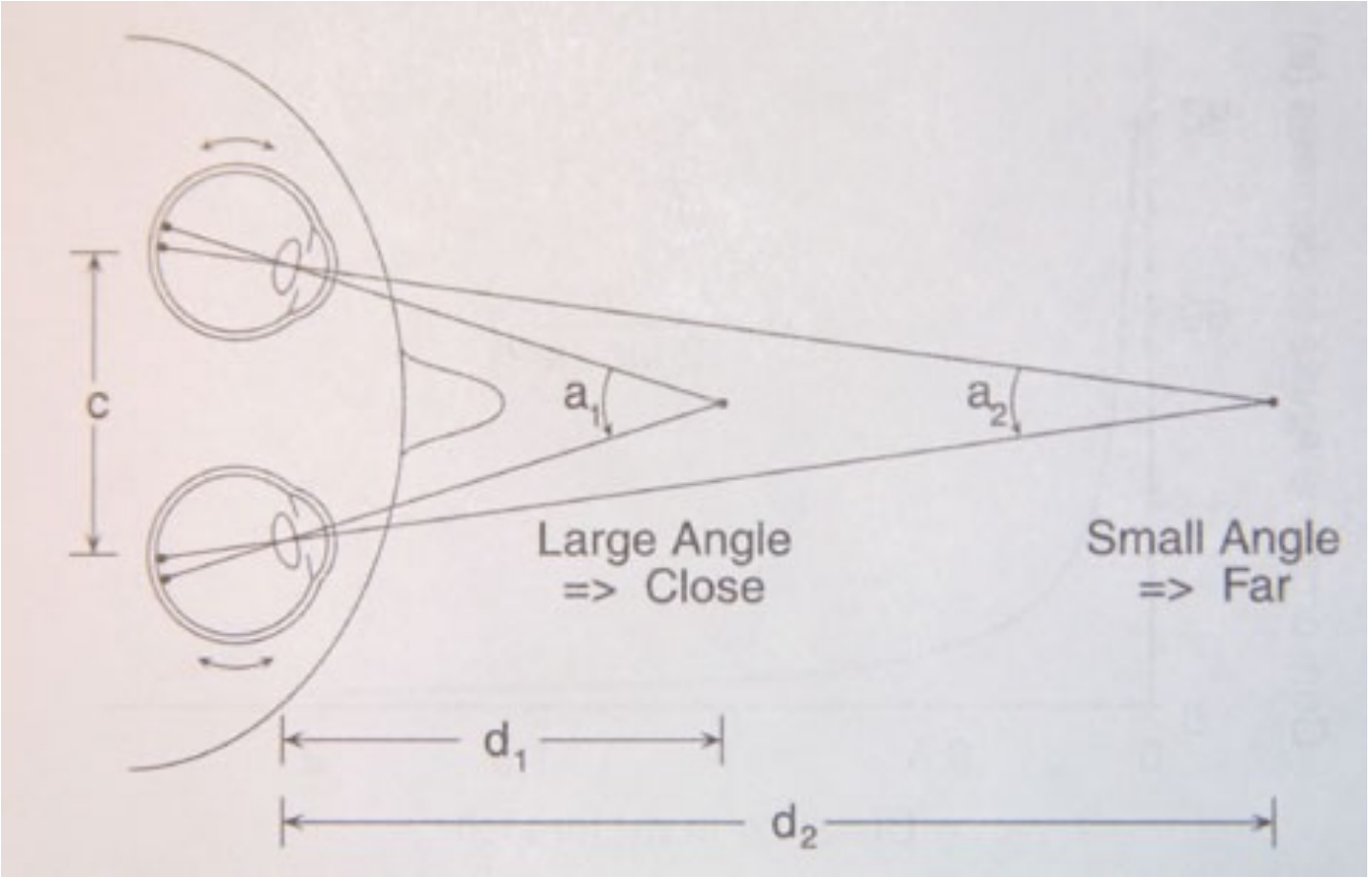
Vérascope 40



Stereograms: Invented by Sir Charles Wheatstone, 1838

https://en.wikipedia.org/wiki/Stereoscopy

Depth from convergence



$$d = \frac{c}{2tan(a/2)}$$

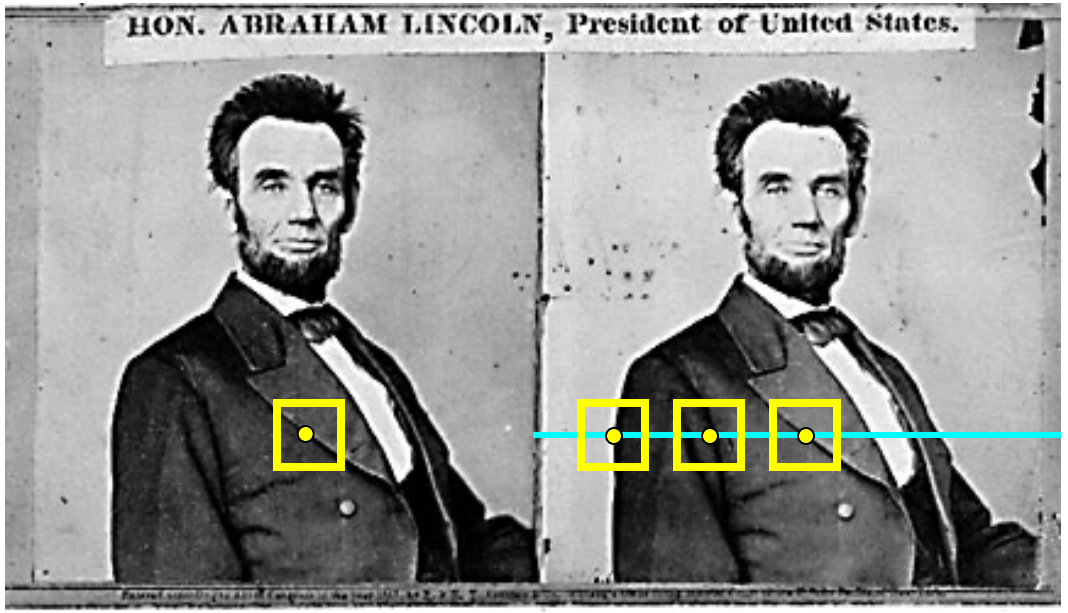
Slide credit: J. Hayes

Stereo Matching for Depth Estimation

Given: stereo pair (assumed calibrated)

Wanted: dense depth map

Basic stereo matching algorithm

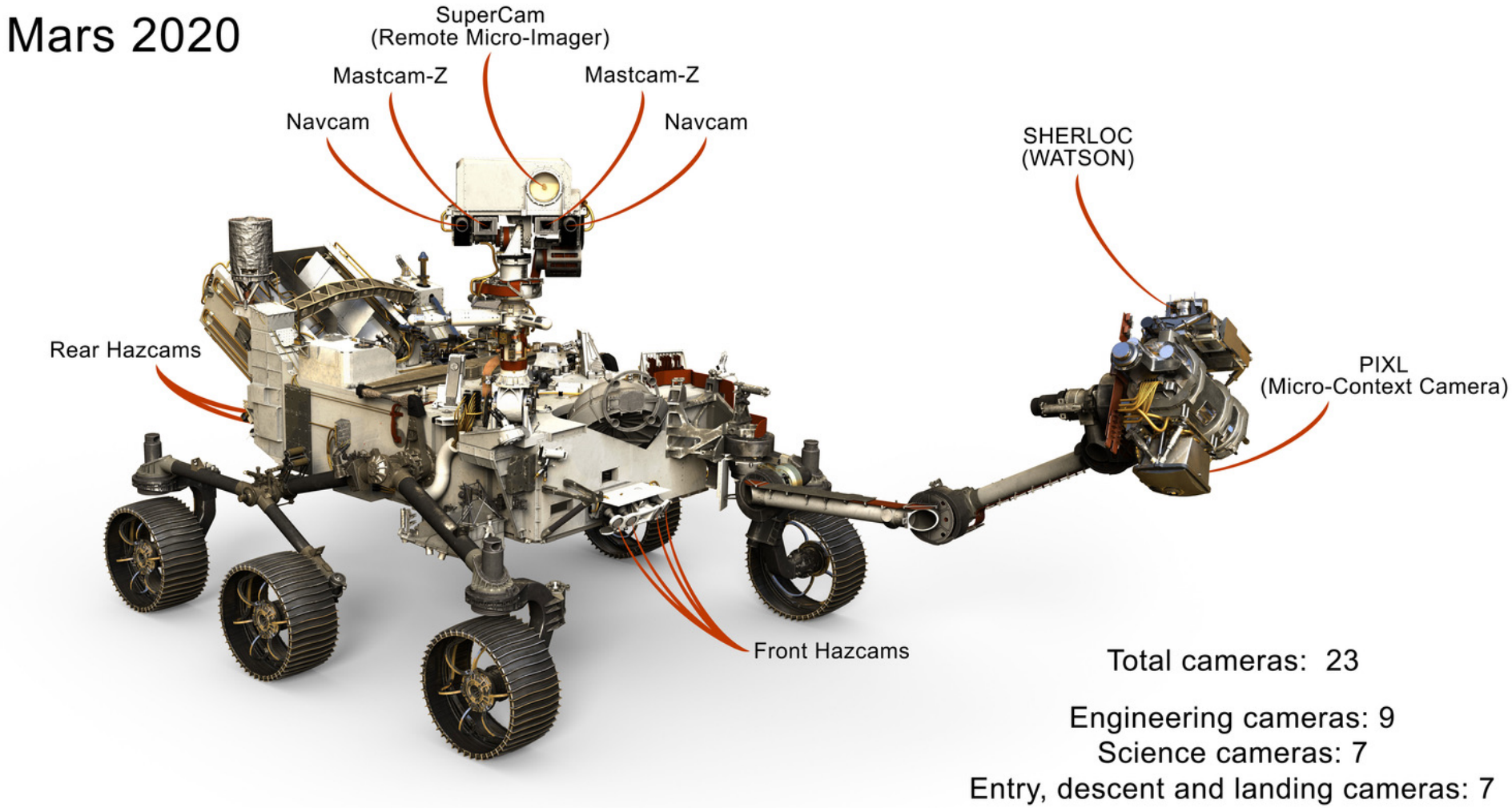


For each pixel in the first image

- Find corresponding epipolar line in the right image: same row!
- Examine all pixels on the epipolar line and pick the best match
- Triangulate the matches to get depth information

More details in appendix: rectification, matching, depth from disparity, etc

Stereo on the Perseverance Mars Rover



https://mars.nasa.gov/mars2020/spacecraft/rover/cameras/

What will we learn today?

Triangulation

Epipolar geometry

Stereo

Structure-from-Motion (SfM)

Reference: Szeliski 11, H&Z ch. 9

Most slides adapted from N. Snavely & S. Lazebnik

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/

Geometry of more than two views?

2 views: governed by the 3x3 ***Fundamental Matrix*** (how to go from one point in an image to the epipolar line in the 2nd image)

3 views: governed by the 3 x 3 x 3 ***Trifocal Tensor***

4 views: governed by the 3 x 3 x 3 x 3 ***Quadrifocal Tensor***

After this it starts to get complicated...

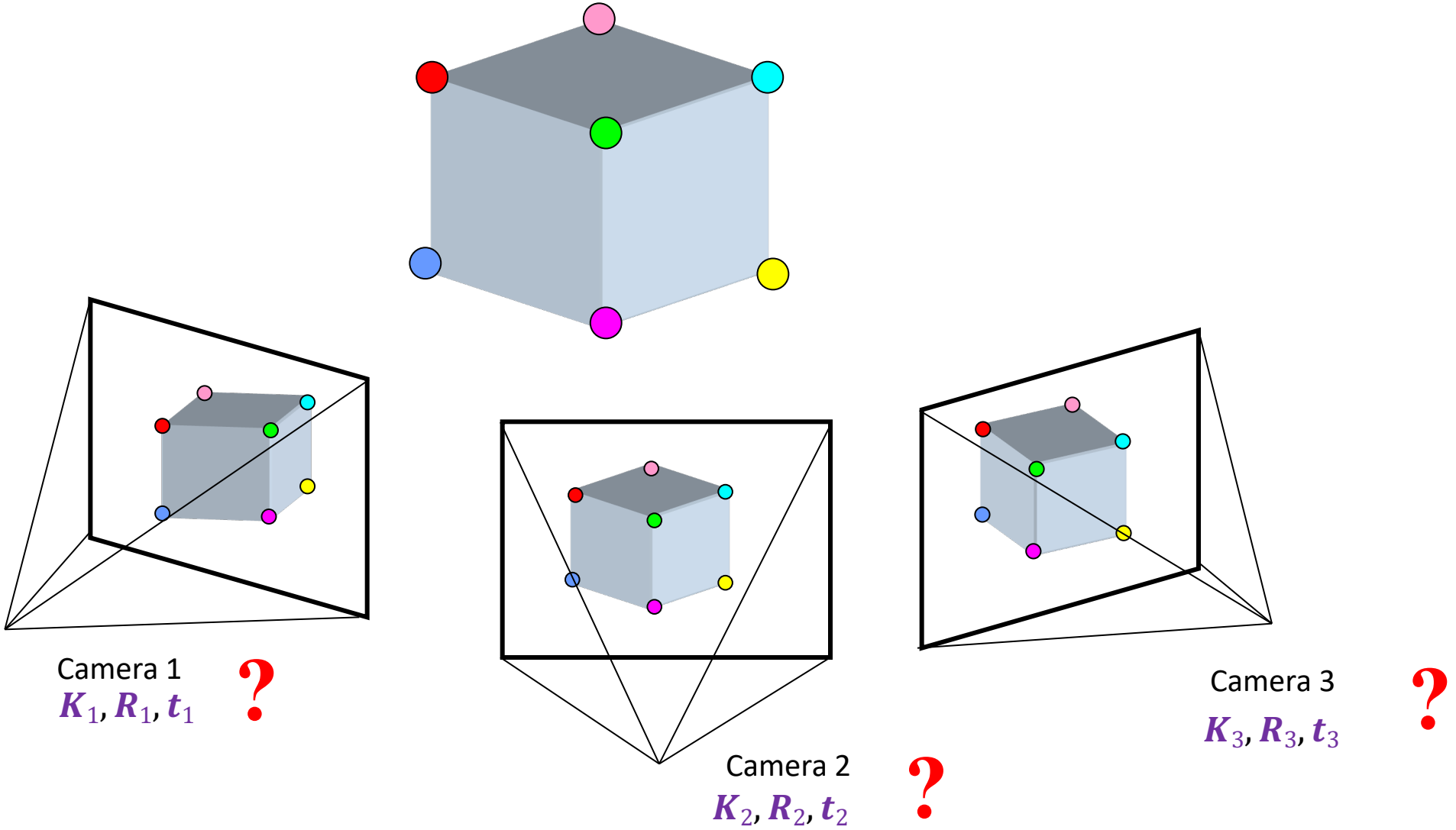
→ ***explicitly*** solve for camera poses ***and*** scene geometry

Large-scale structure-from-motion

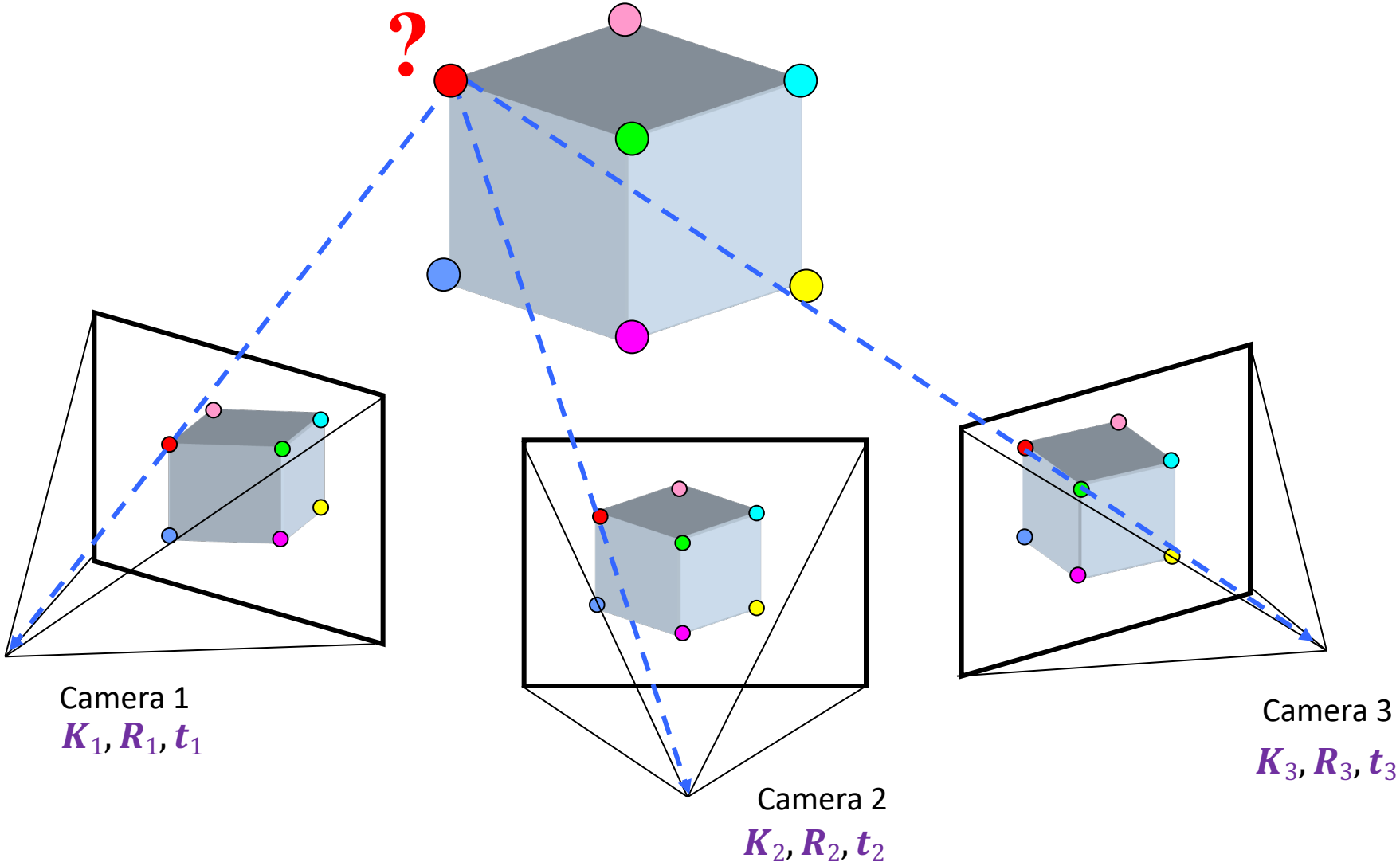
Dubrovnik, Croatia. 4,619 images (out of an initial 57,845 downloaded from Flickr). 3.5M points!
Total reconstruction time: 17.5 hours on 352 cores

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

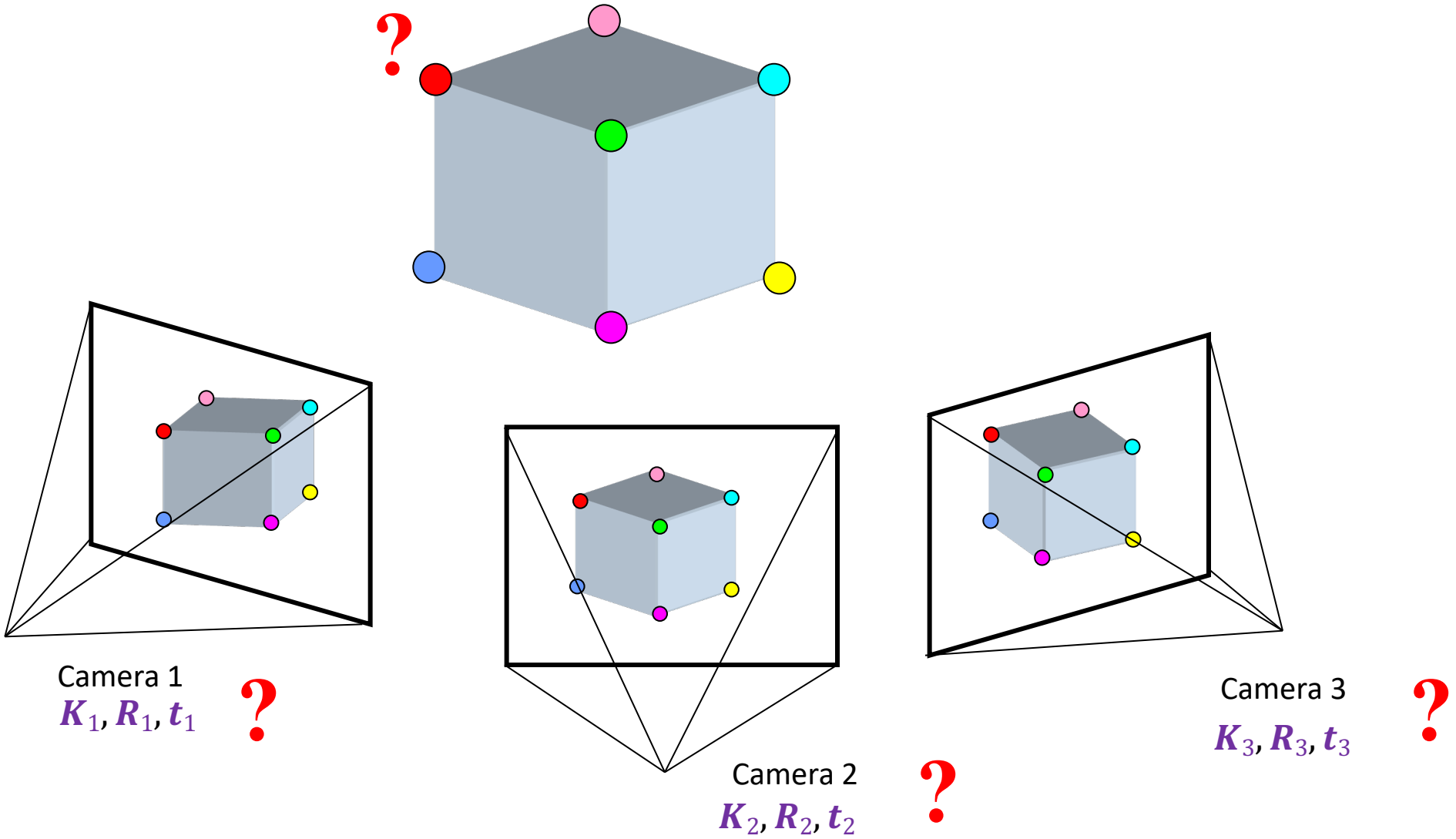
Recall: Calibration



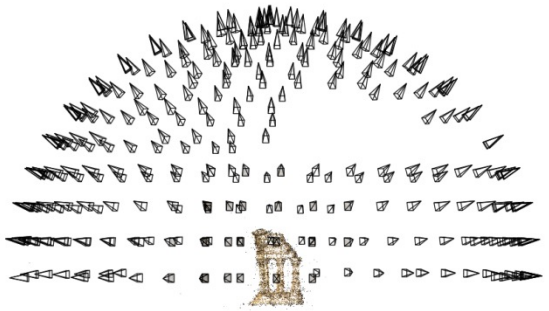
Recall: Triangulation / Multi-view Stereo



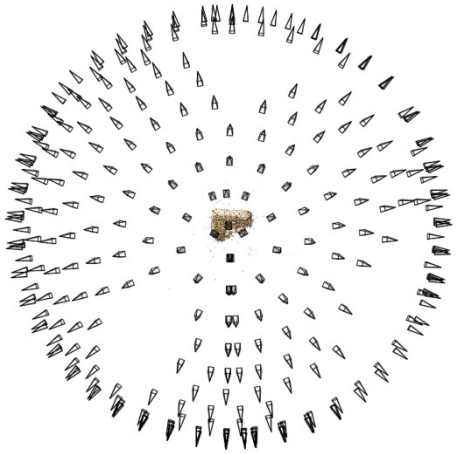
Structure-from-Motion



Structure-from-Motion



Reconstruction (side)



(top)

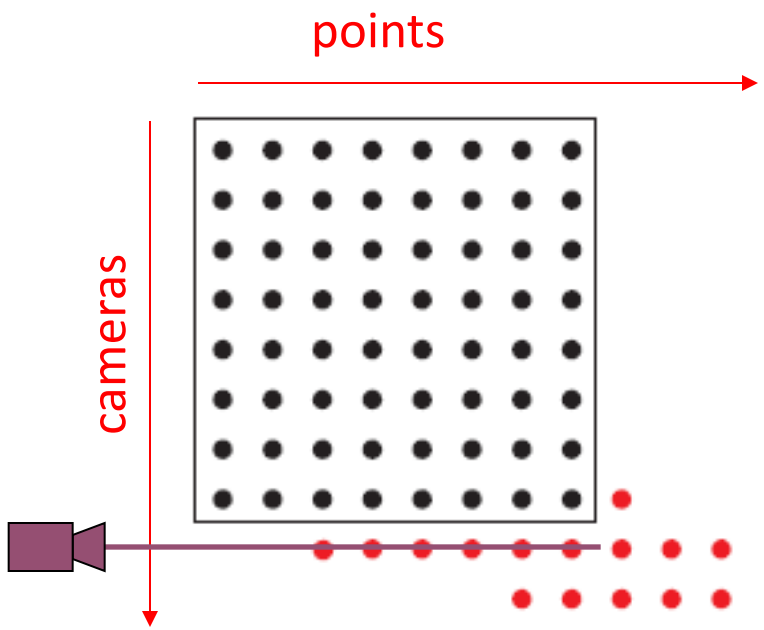
- Input: images with 2D points $\boldsymbol{x}_{ij}$ in correspondence

- Output (solved simultaneously now!)
  - structure: 3D location $\boldsymbol{X}_j$ for each point $\boldsymbol{x}_{ij}$
  - motion: camera parameters $\boldsymbol{R}_i$, $\boldsymbol{t}_i$ & possibly $\boldsymbol{K}_i$

- Objective function: minimize *reprojection error in 2D*

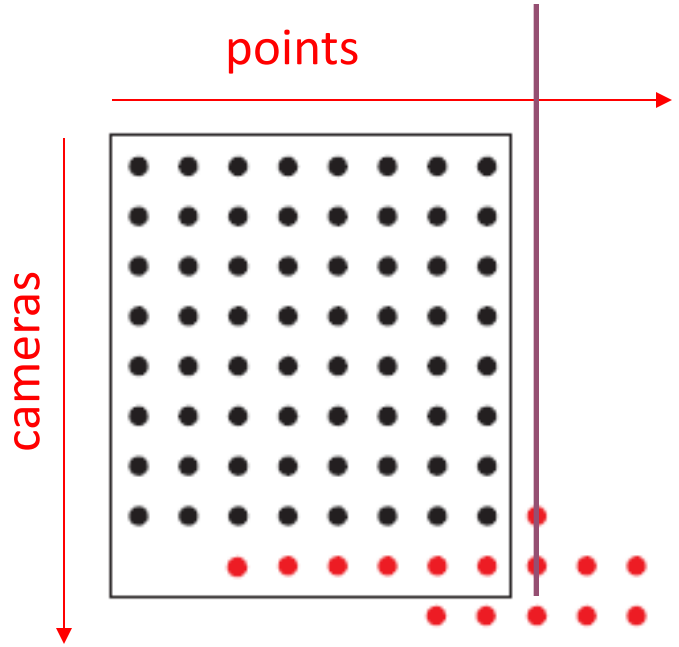
Incremental Structure-from-Motion

- Initialize motion from two images using the fundamental matrix
- Initialize structure by triangulation
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image – calibration



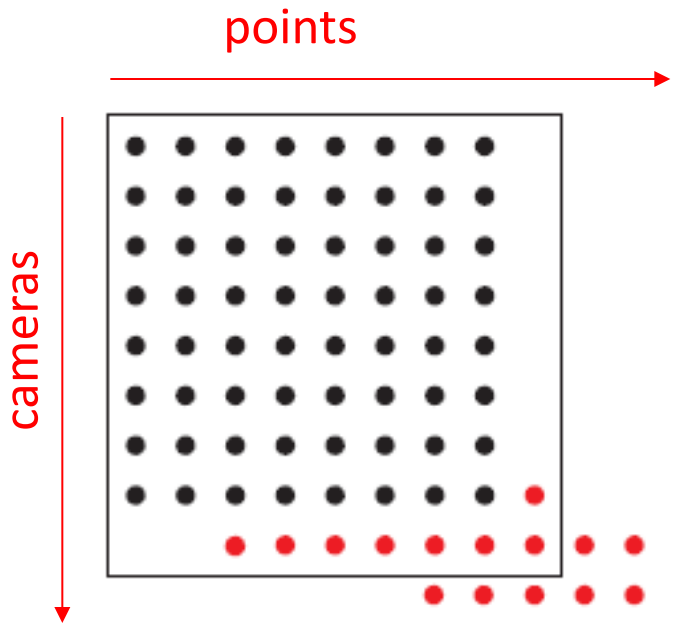
Incremental Structure-from-Motion

- Initialize motion from two images using the fundamental matrix
- Initialize structure by triangulation
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image – calibration
  - Refine and extend structure: compute newly visible 3D points, re-optimize existing points that are also seen by this camera – triangulation



Incremental Structure-from-Motion

- Initialize motion from two images using the fundamental matrix
- Initialize structure by triangulation
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image – calibration
  - Refine and extend structure: compute newly visible 3D points, re-optimize existing points that are also seen by this camera – triangulation



- Refine all cameras & points *jointly*: ***bundle adjustment***

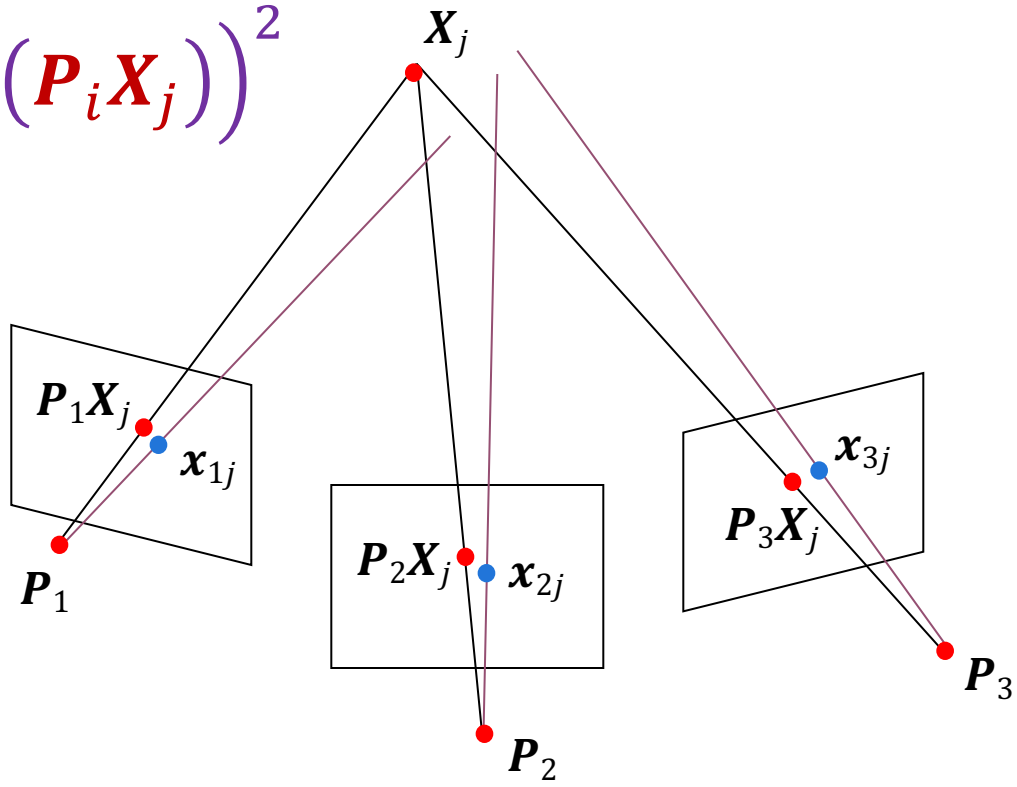
Bundle Adjustment

Non-linear method for refining structure ($\boldsymbol{X}_j$) and motion ($\boldsymbol{P}_i$)

Minimize reprojection error (with lots of bells and whistles):

$$\sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}\, d\left(\boldsymbol{x}_{ij} - \text{proj}\left(\boldsymbol{P}_i \boldsymbol{X}_j\right)\right)^2$$

visibility flag: is point j visible in view i?

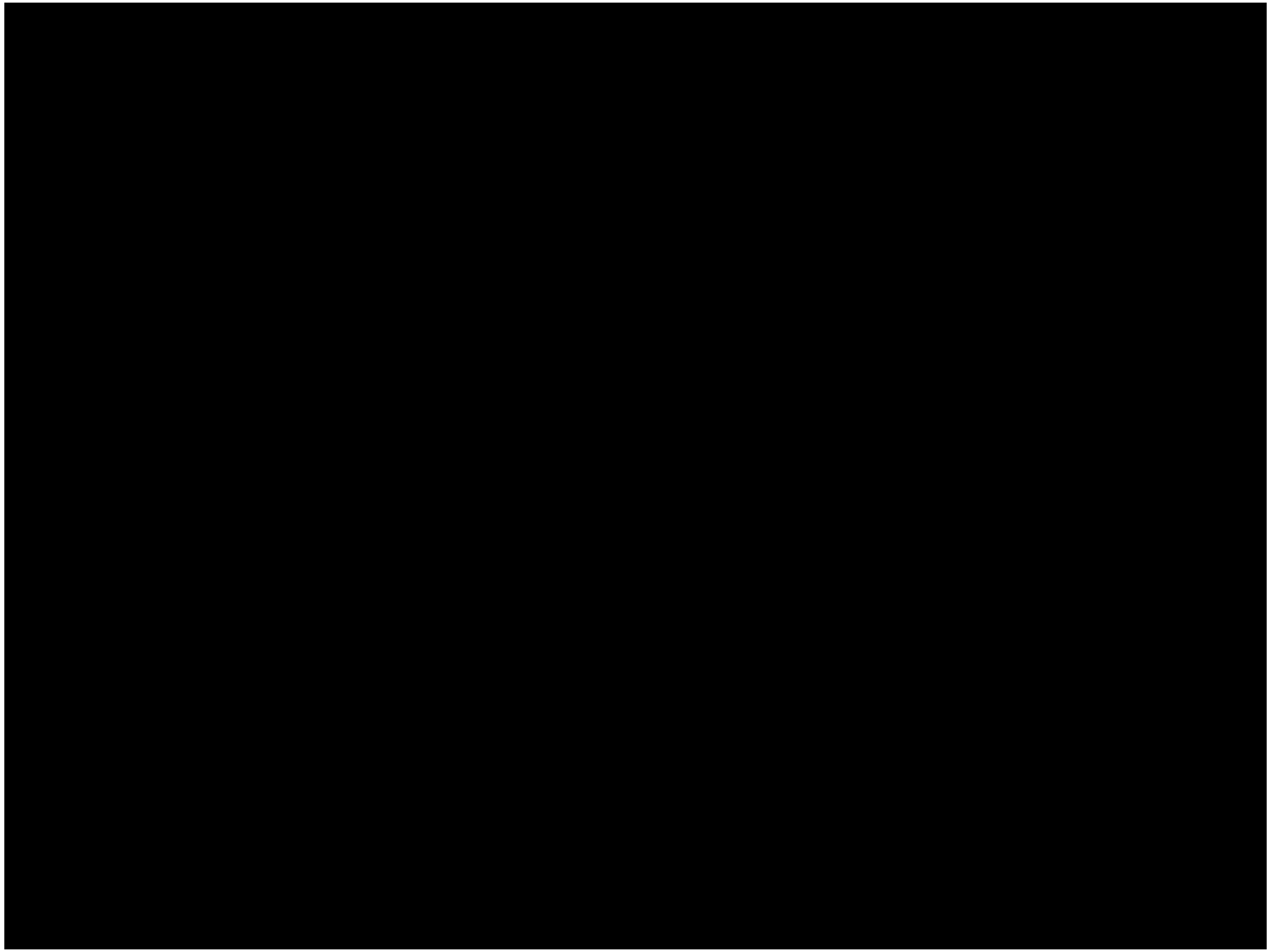


B. Triggs et al. Bundle adjustment – A modern synthesis. International Workshop on Vision Algorithms, 1999

Incremental SfM in Practice

- Pick a pair of images with lots of inliers (and good EXIF data)
  - Initialize intrinsic parameters (focal length, principal point) from EXIF
  - Estimate extrinsic parameters ($\boldsymbol{R}$ and $\boldsymbol{t}$) using five-point algorithm
  - Use triangulation to initialize model points
- While remaining images exist
  - Find an image with many feature matches with images in the model
  - Run RANSAC on feature matches to register new image to model
  - Triangulate new points
  - Perform bundle adjustment to re-optimize everything
  - Optionally, align with GPS from EXIF data or ground control points

Incremental structure from motion



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

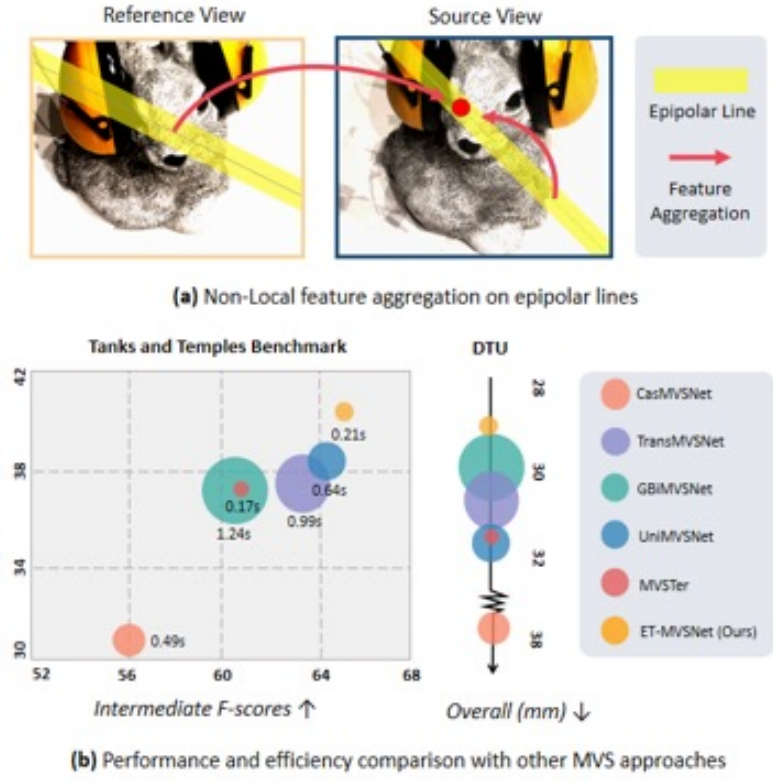
COLMAP

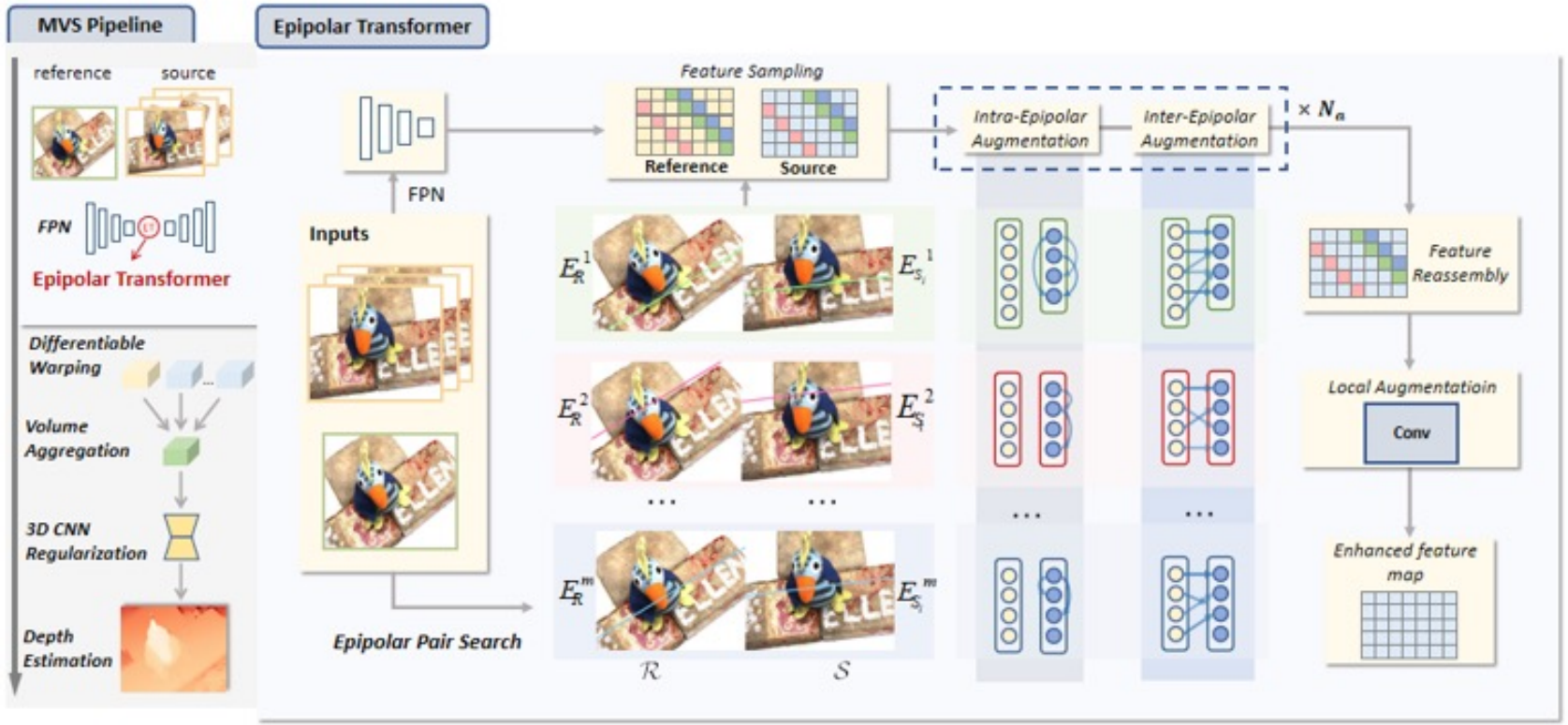
Sparse model of central Rome using 21K photos produced by COLMAP's SfM pipeline.

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)]

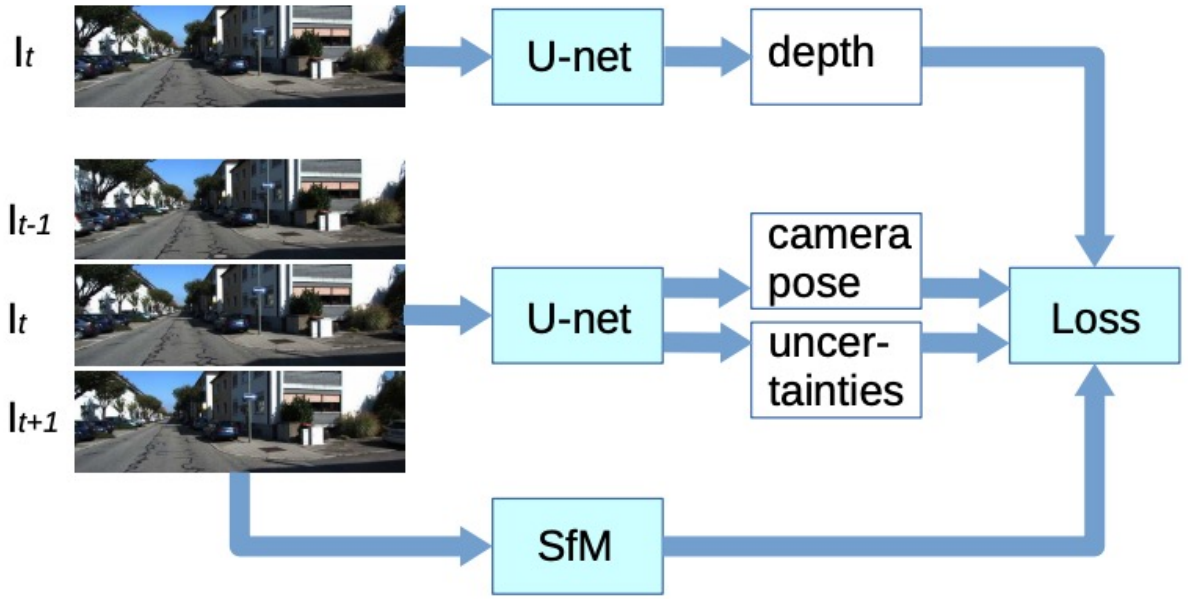
See also [MVSFormer: Multi-View Stereo by Learning Robust Image Features and Temperature-based Depth] (TMLR'23)

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

What did we learn today?

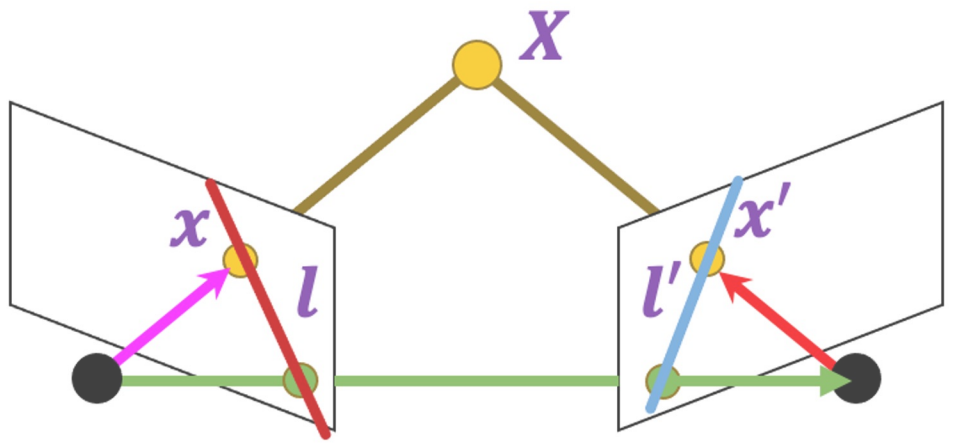
Triangulation

from calibrated cameras and 2D correspondences to 3D points

Epipolar geometry

$$x'^T F x = 0$$

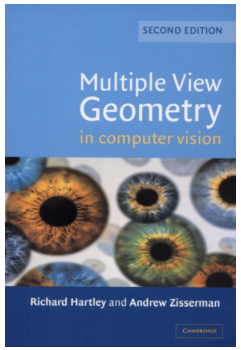
Epipolar constraint,
Essential & Fundamental matrices



Stereo (fronto-parallel special case)

Structure-from-Motion (SfM) (many images / video)

Note: this is just an introduction → for more, take CS231a or get into HZ



Wrapping up Geometric Vision

Homogeneous Coordinates & Projective Space

2D & 3D Transforms as Matrix Multiplication

Pinhole Camera Model P=K[R|t]

Calibration from known 3D-to-2D correspondences

Multi-view geom: fundamental matrix, stereo, SfM

Appendix

Triangulation: Linear approach

$$\lambda \boldsymbol{x} = \boldsymbol{P}\boldsymbol{X} \qquad \boldsymbol{x} \times \boldsymbol{P}\boldsymbol{X} = \boldsymbol{0} \qquad [\boldsymbol{x}]_\times \boldsymbol{P}\boldsymbol{X} = \boldsymbol{0}$$
$$\lambda' \boldsymbol{x}' = \boldsymbol{P}'\boldsymbol{X} \qquad \boldsymbol{x}' \times \boldsymbol{P}'\boldsymbol{X} = \boldsymbol{0} \qquad [\boldsymbol{x}']_\times \boldsymbol{P}'\boldsymbol{X} = \boldsymbol{0}$$

Rewrite cross product as matrix multiplication:

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

2 ***independent*** equations per 2D point --> 4 equations in matrix form

$$\mathbf{P} = \begin{bmatrix} \boldsymbol{p}_1^\top \\ \boldsymbol{p}_2^\top \\ \boldsymbol{p}_3^\top \end{bmatrix}, \mathbf{A}\mathbf{X} = \begin{bmatrix} y\boldsymbol{p}_3^\top - \boldsymbol{p}_2^\top \\ \boldsymbol{p}_1^\top - x\boldsymbol{p}_3^\top \\ y'\boldsymbol{p}_3'^\top - \boldsymbol{p}_2'^\top \\ \boldsymbol{p}_1'^\top - x'\boldsymbol{p}_3'^\top \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix} = \mathbf{0}$$

Total Least Squares:

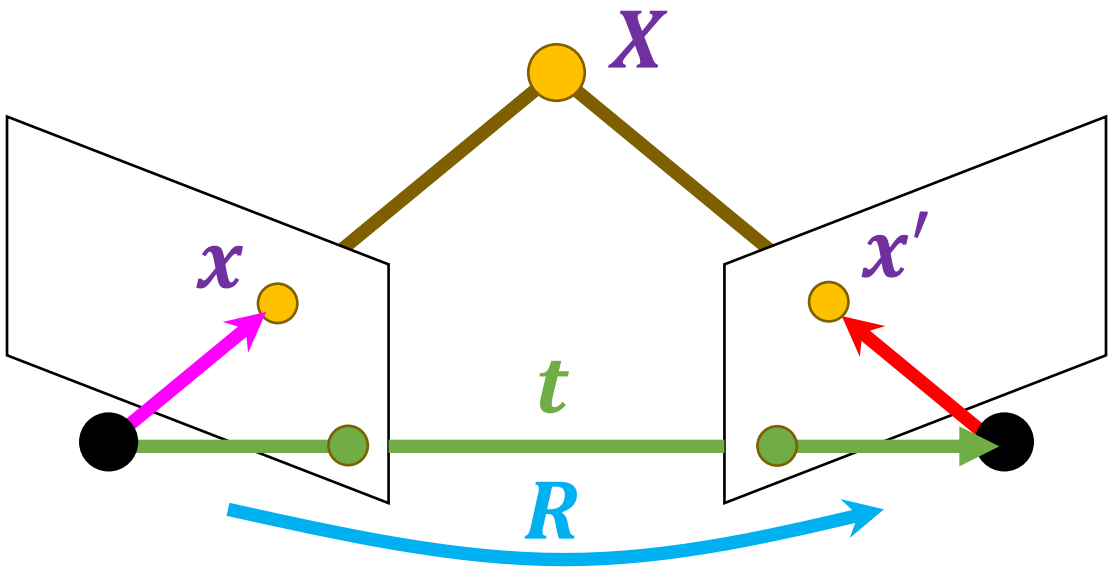
$$\min_{\mathbf{X}} \|\mathbf{A}\mathbf{X}\|^2 \; s.t. \; \|\mathbf{X}\|^2 = 1$$

Solved via SVD! (Sz. A.2.1)

Bonus slides - Math of the Epipolar Constraint

Slides credit: S. Lazebnik

Math of the epipolar constraint: Calibrated case



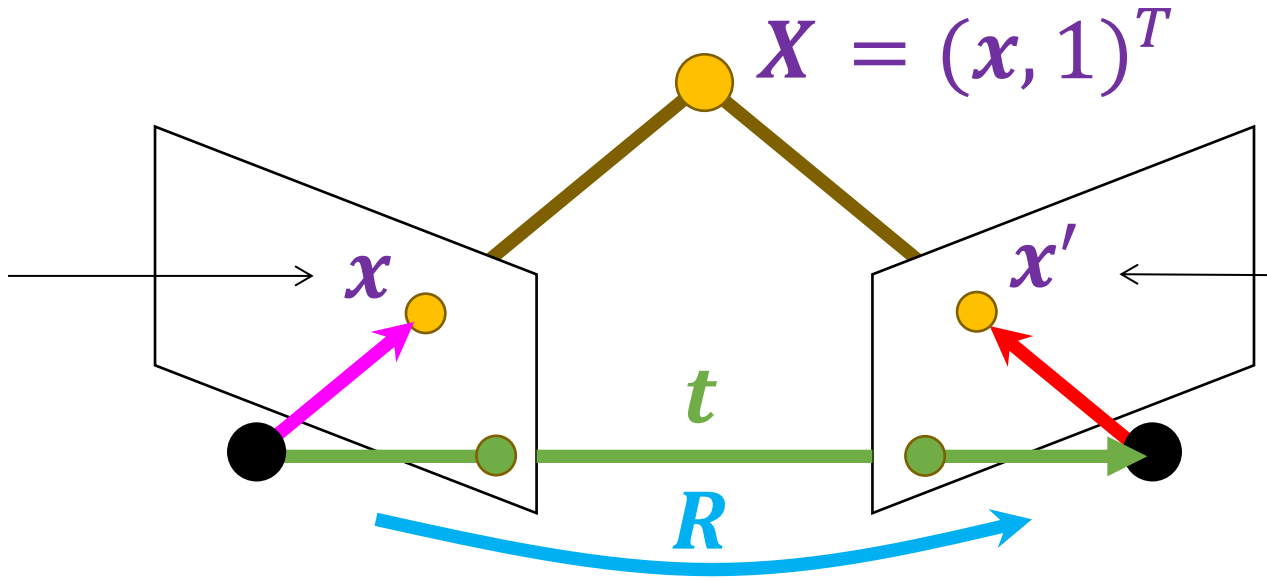
- Assume the intrinsic and extrinsic parameters of the cameras are known, world coordinate system is set to that of the first camera
- Then the projection matrices are given by $\boldsymbol{P = K[I \mid 0]}$ and $\boldsymbol{P' = K'[R \mid t]}$
- We can pre-multiply the projection matrices (and the image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

$$\boldsymbol{x}_{\text{norm}} = \boldsymbol{K}^{-1}\boldsymbol{x}_{\text{pixel}} \sim [\boldsymbol{I} \mid \boldsymbol{0}]\boldsymbol{X}, \qquad \boldsymbol{x'}_{\text{norm}} = \boldsymbol{K'}^{-1}\boldsymbol{x'}_{\text{pixel}} \sim [\boldsymbol{R} \mid \boldsymbol{t}]\boldsymbol{X}$$

Math of the epipolar constraint: Calibrated case

$x_{\text{norm}} \sim [I \mid 0]X$

$[I \mid 0]\begin{pmatrix} x \\ 1 \end{pmatrix}$

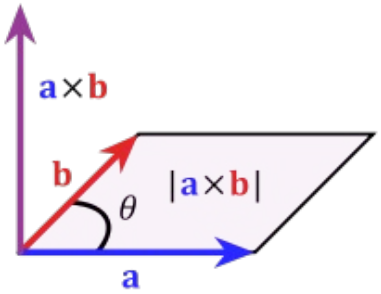


$x'_{\text{norm}} \sim [R \mid t]X$

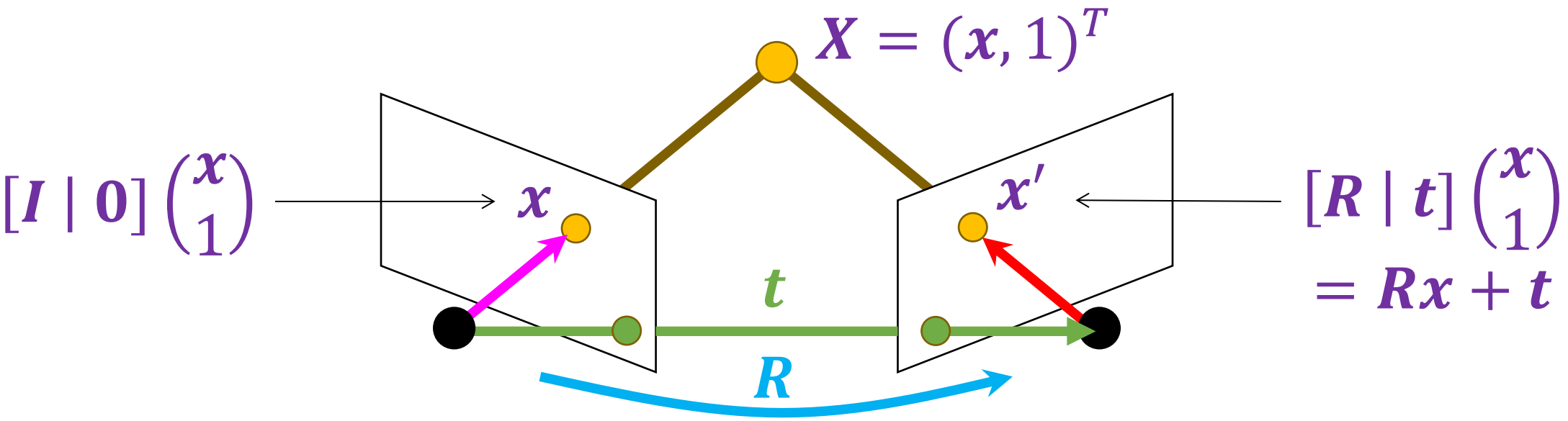
$[R \mid t]\begin{pmatrix} x \\ 1 \end{pmatrix} = Rx + t$

- We have $x' \sim Rx + t$
- This means the three vectors x', Rx, and t are linearly dependent
- This constraint can be written using the *triple product*

$$x' \cdot [t \times (Rx)] = 0$$



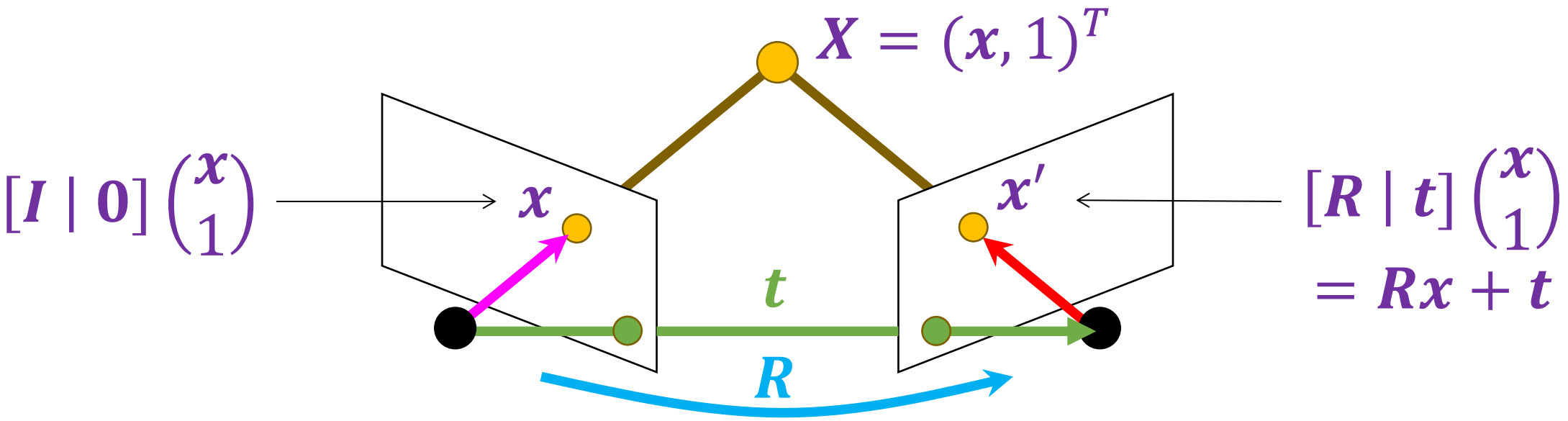
Math of the epipolar constraint: Calibrated case



$$x' \cdot [t \times (Rx)] = 0 \quad \Rightarrow \quad x'^T [t]_\times R x = 0$$

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

Math of the epipolar constraint: Calibrated case



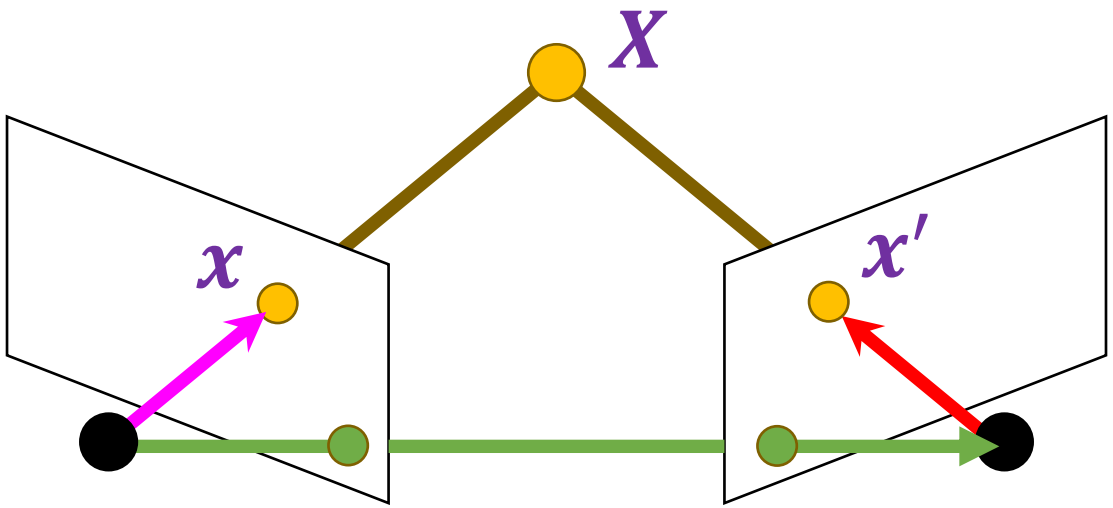
$$x' \cdot [t \times (Rx)] = 0 \quad \Rightarrow \quad x'^T [t]_\times R x = 0 \quad \Rightarrow \quad x'^T E x = 0$$

$$E = [t]_\times R$$

Essential Matrix

H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. Nature 293 (5828): 133–135, September 1981

The essential matrix

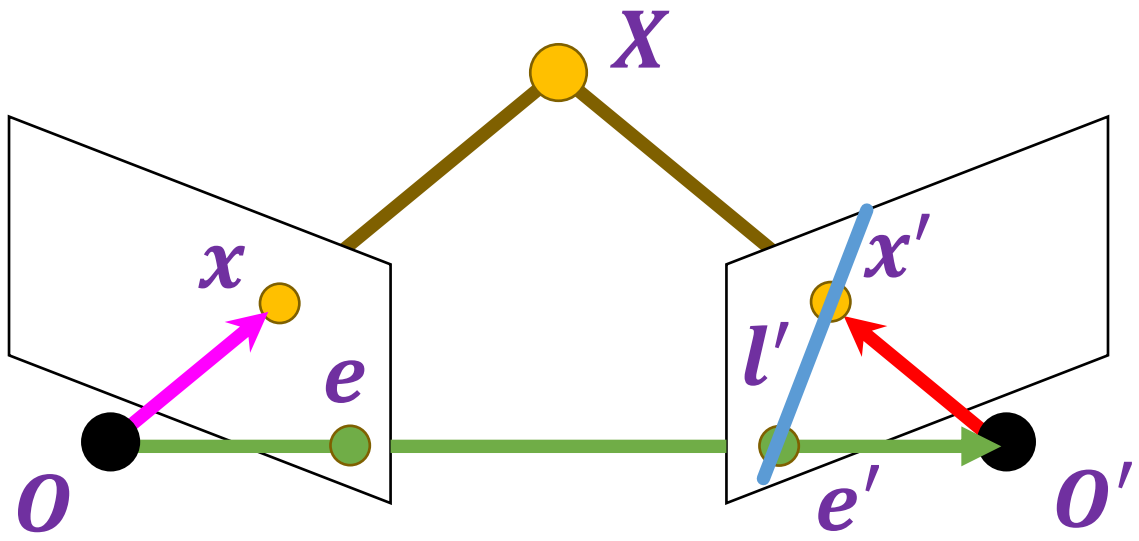


$$\boldsymbol{x'^T E x} = 0$$

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

Essential essential matrix properties

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

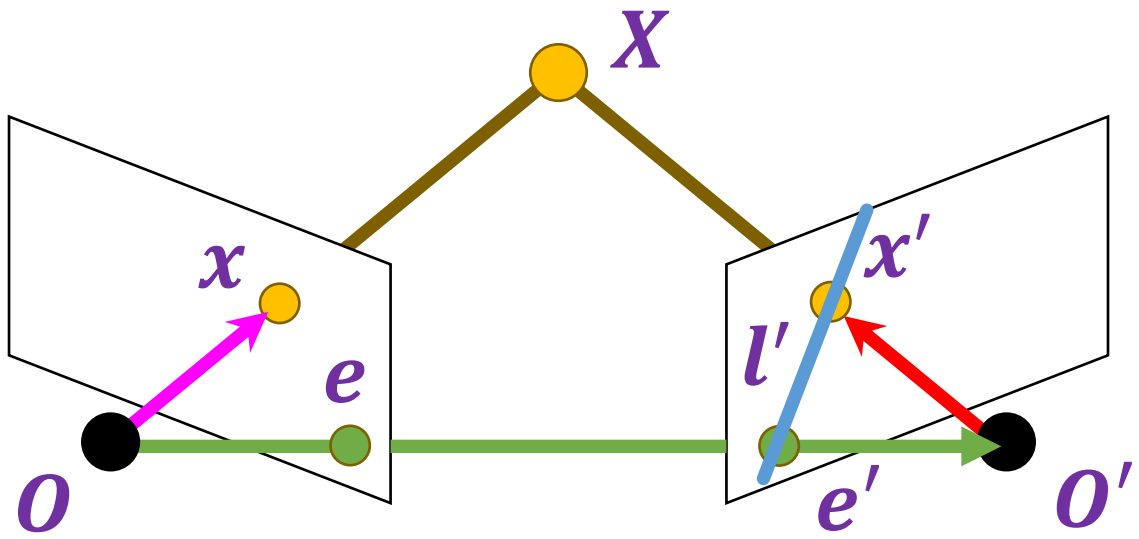


Ex is the epipolar line associated with x ($l' = Ex$)

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

Essential essential matrix properties

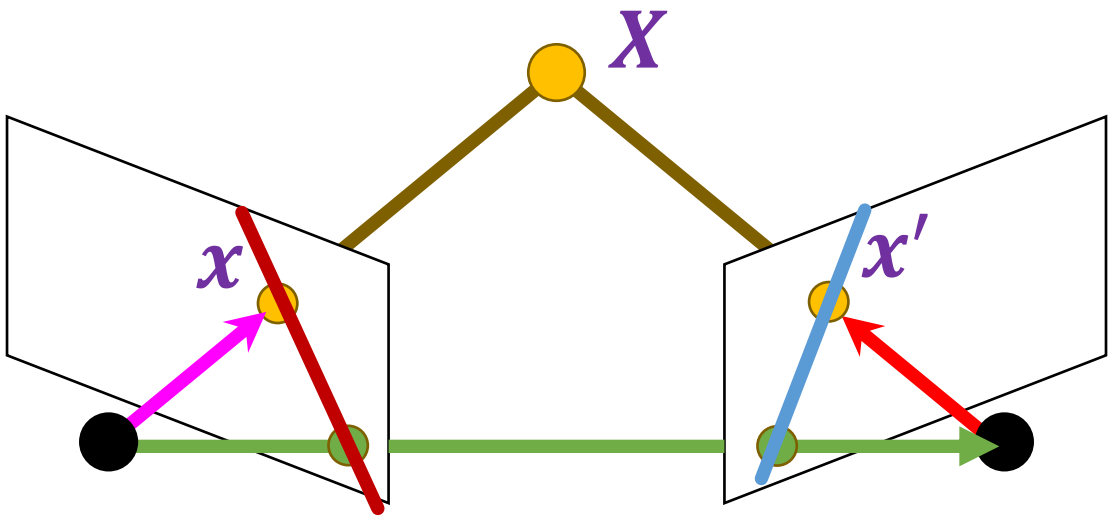
$$x'^T E x = 0$$



- Ex is the epipolar line associated with x ($l' = Ex$)
- $E^T x'$ is the epipolar line associated with x' ($l = E^T x'$)
- Epipoles are the null–spaces of E: $Ee = 0$ and $E^\top e' = 0$
  (all epipolar lines pass through epipoles: $\forall x,\ l'^\top e' = x^\top E^\top e' = 0 \Rightarrow E^\top e' = 0$)
- E is singular: rank 2 with 2 non-zero identical singular values
- E has 5 d.o.f. (3 for R, 3 for t, -1 for scale)

Cf. H&Z ch. 9 for proofs

Epipolar constraint: Uncalibrated case

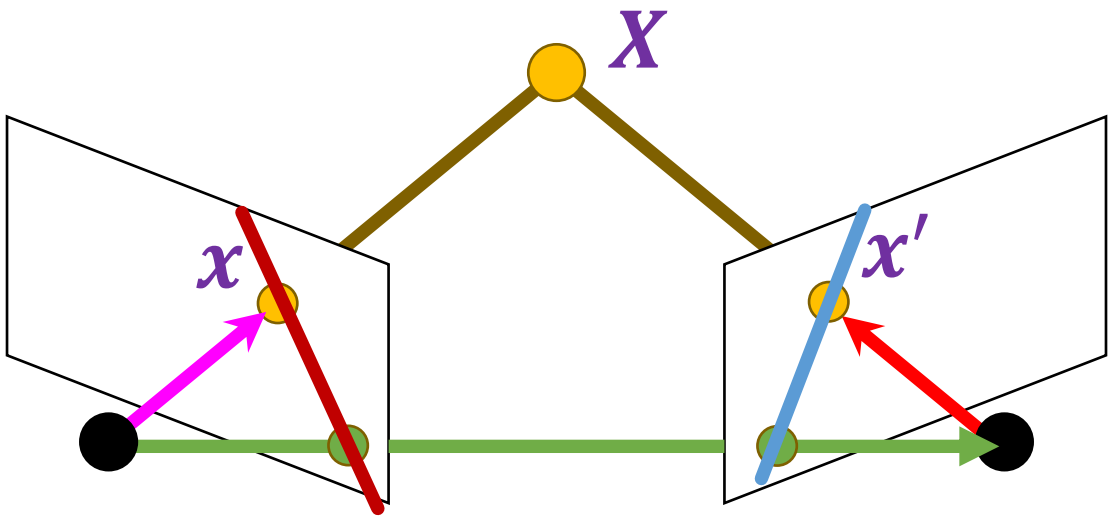


- What if we don't know the camera calibration matrices $\boldsymbol{K}$ and $\boldsymbol{K}'$?
- Epipolar constraint in terms of *unknown* normalized coordinates:

$$\boldsymbol{x}'^{T}_{\text{norm}}\boldsymbol{E}\boldsymbol{x}_{\text{norm}} = 0,$$

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

Epipolar constraint: Uncalibrated case



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

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

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

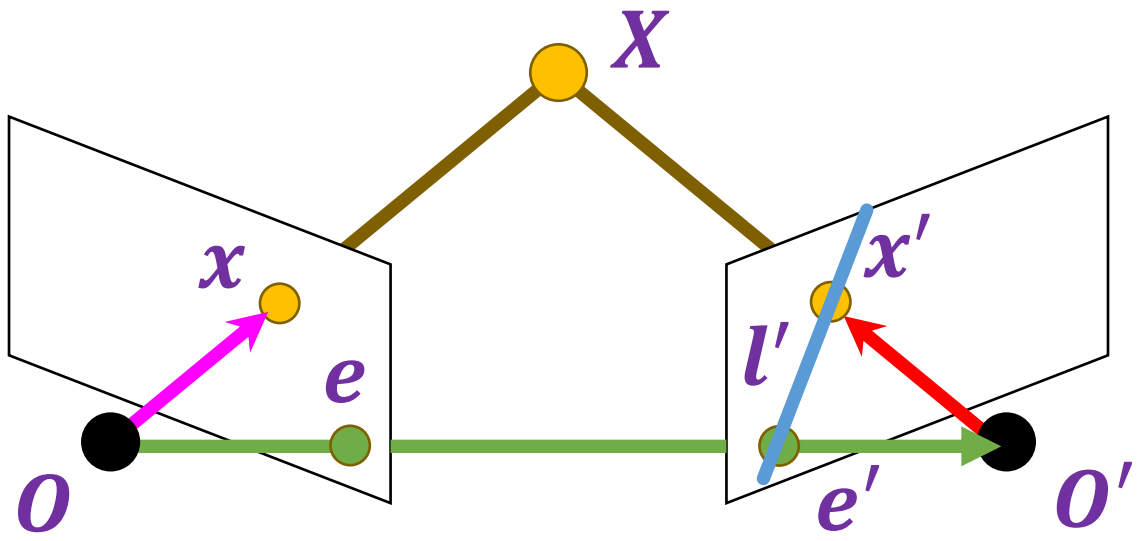
Fundamental Matrix

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

Fundamental fundamental matrix properties

Cf. H&Z ch. 9 for proofs

$$x'^T F x = 0$$

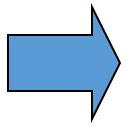


- Fx is the epipolar line associated with x ($l' = Fx$)
- $F^T x'$ is the epipolar line associated with x' ($l = F^T x'$)
- $Fe = 0$ and $F^\top e' = 0$ (can solve for epipoles via… SVD!)
- F is singular (rank 2)
- F has 7 d.o.f. (9 parameters, -1 for scale, -1 for singularity)

Estimating the fundamental matrix

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

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



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

The eight point algorithm

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

$$\boldsymbol{U}$$

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

$$\arg\min_{\|\boldsymbol{f}\|=1} \|\boldsymbol{U}\boldsymbol{f}\|_2^2 \rightarrow$$

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

$$\boldsymbol{F}_{\text{init}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$$

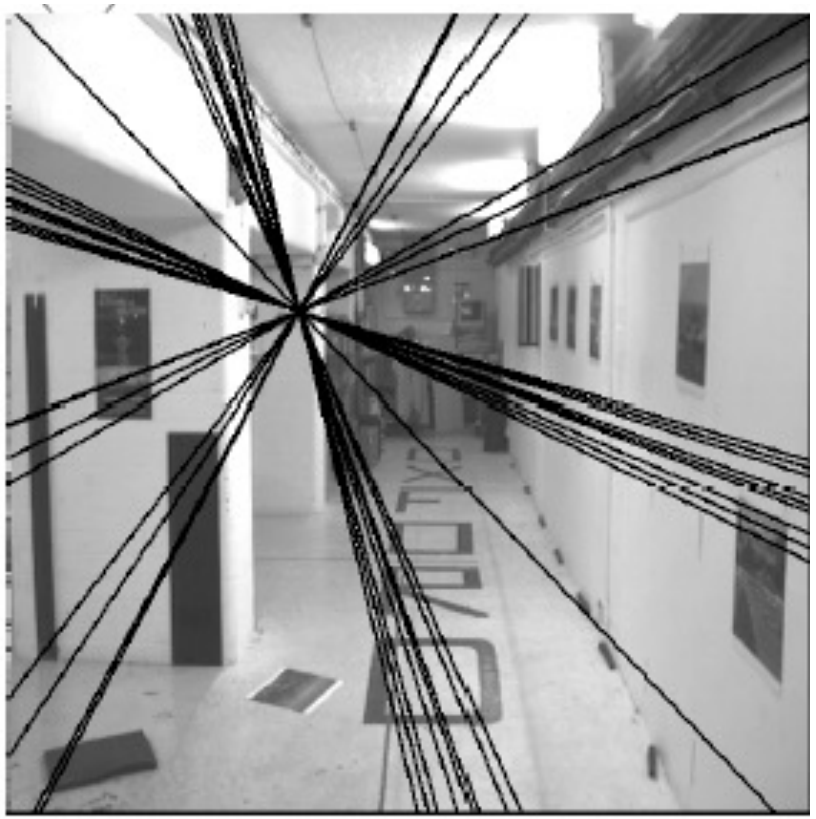
$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \longrightarrow \boldsymbol{\Sigma}' = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

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

Enforcing rank-2 constraint

Initial F estimate

Rank-2 estimate



Normalized eight point algorithm

$$\begin{matrix} 10^6 & 10^6 & 10^3 & 10^6 & 10^6 & 10^3 & 10^3 & 10^3 & 1 \end{matrix}$$

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

$\boldsymbol{U}$

- Recall that x, y, x', y' are pixel coordinates. What might be the order of magnitude of each column of $\boldsymbol{U}$?
- This causes numerical instability!

The normalized eight-point algorithm

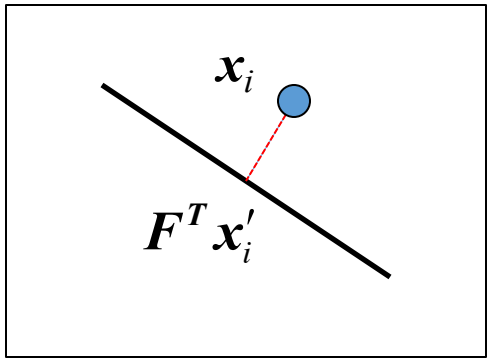
- In each image, center the set of points at the origin, and scale it so the mean squared distance between the origin and the points is 2 pixels
- Use the eight-point algorithm to compute $\boldsymbol{F}$ from the normalized points
- Enforce the rank-2 constraint
- Transform fundamental matrix back to original units: if $\boldsymbol{T}$ and $\boldsymbol{T'}$ are the normalizing transformations in the two images, then the fundamental matrix in original coordinates is $\boldsymbol{T'^{T}FT}$

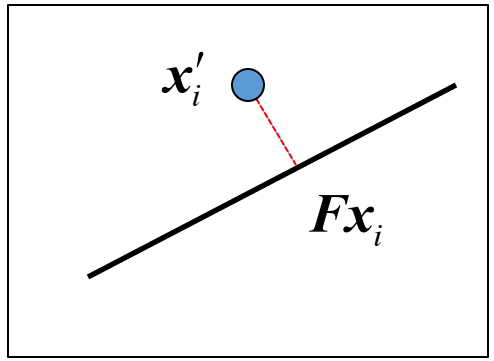
R. Hartley. In defense of the eight-point algorithm. TPAMI 1997

Nonlinear estimation

- Linear estimation minimizes the sum of squared *algebraic* distances between points $\boldsymbol{x}_i'$ and epipolar lines $\boldsymbol{F}\boldsymbol{x}_i$ (or points $\boldsymbol{x}_i$ and epipolar lines $\boldsymbol{F}^T\boldsymbol{x}_i'$):

- $\sum_i \left(\boldsymbol{x}_i'^T \boldsymbol{F}\boldsymbol{x}_i\right)^2$
- Nonlinear approach: minimize sum of squared *geometric* distances
- $\sum_i [\mathrm{dist}(\boldsymbol{x}_i', \boldsymbol{F}\boldsymbol{x}_i)^2 + \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{F}^T\boldsymbol{x}_i')^2]$





Comparison of estimation algorithms

	8-point	Normalized 8-point	Nonlinear least squares
Av. Dist. 1	2.33 pixels	0.92 pixel	0.86 pixel
Av. Dist. 2	2.18 pixels	0.85 pixel	0.80 pixel

Seven-point algorithm

- Set up least squares system with seven pairs of matches and solve for null space (two vectors) using SVD
- Solve for polynomial equation to get coefficients of linear combination of null space vectors that satisfies $\det(\boldsymbol{F}) = 0$

Source: e.g., M. Pollefeys tutorial (2000)

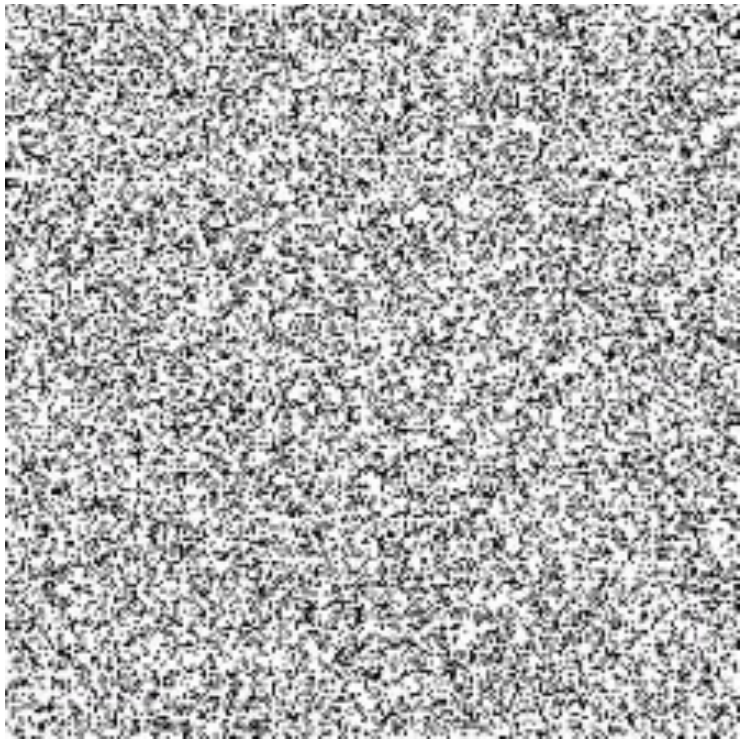
The Fundamental Matrix Song

http://danielwedge.com/fmatrix/

Bonus slides – More on Stereo

History: Random dot stereograms

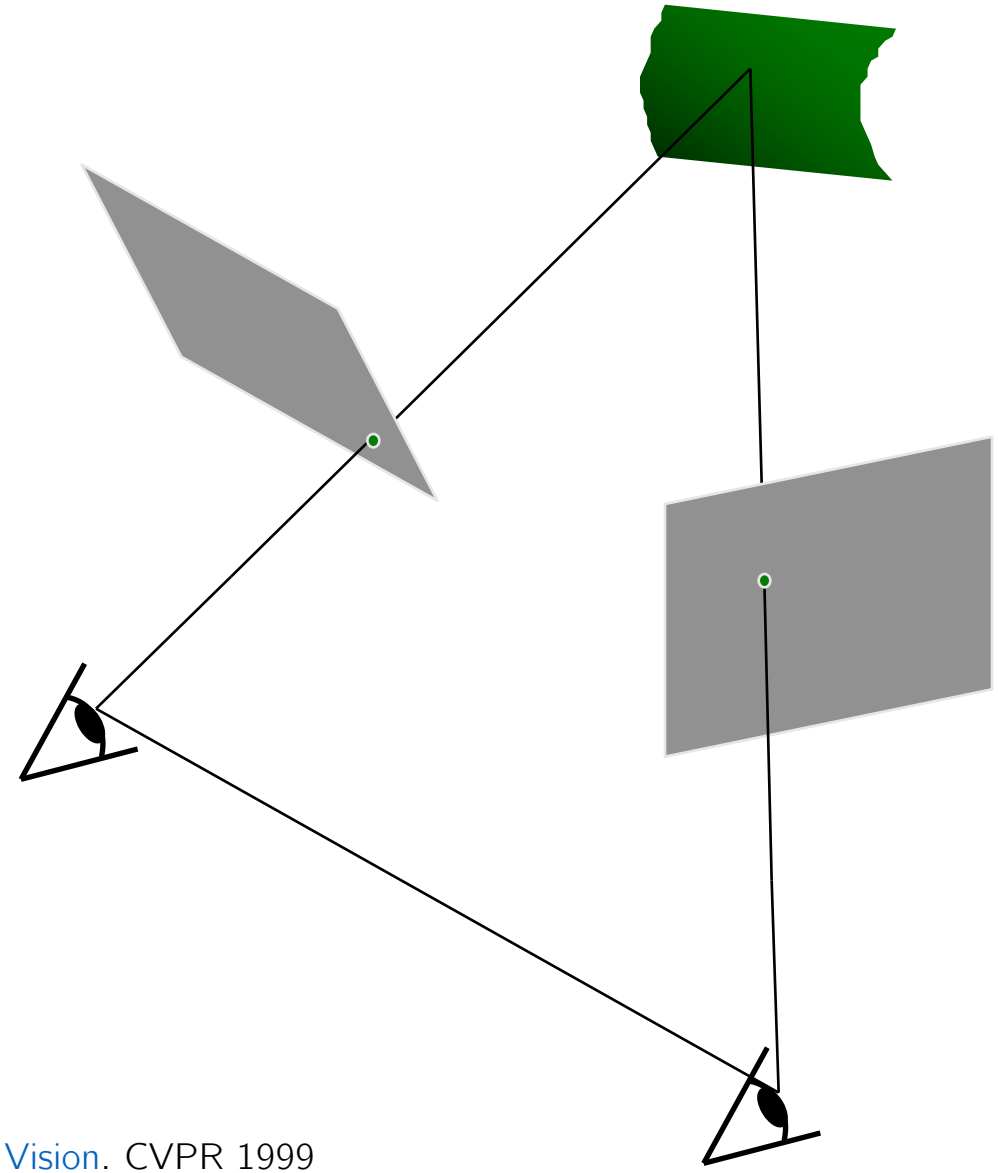
- Invented by Bela Julesz in the mid-20th century
- Demonstration that stereo perception can happen without any monocular cues



https://en.wikipedia.org/wiki/Random_dot_stereogram

Stereo image rectification

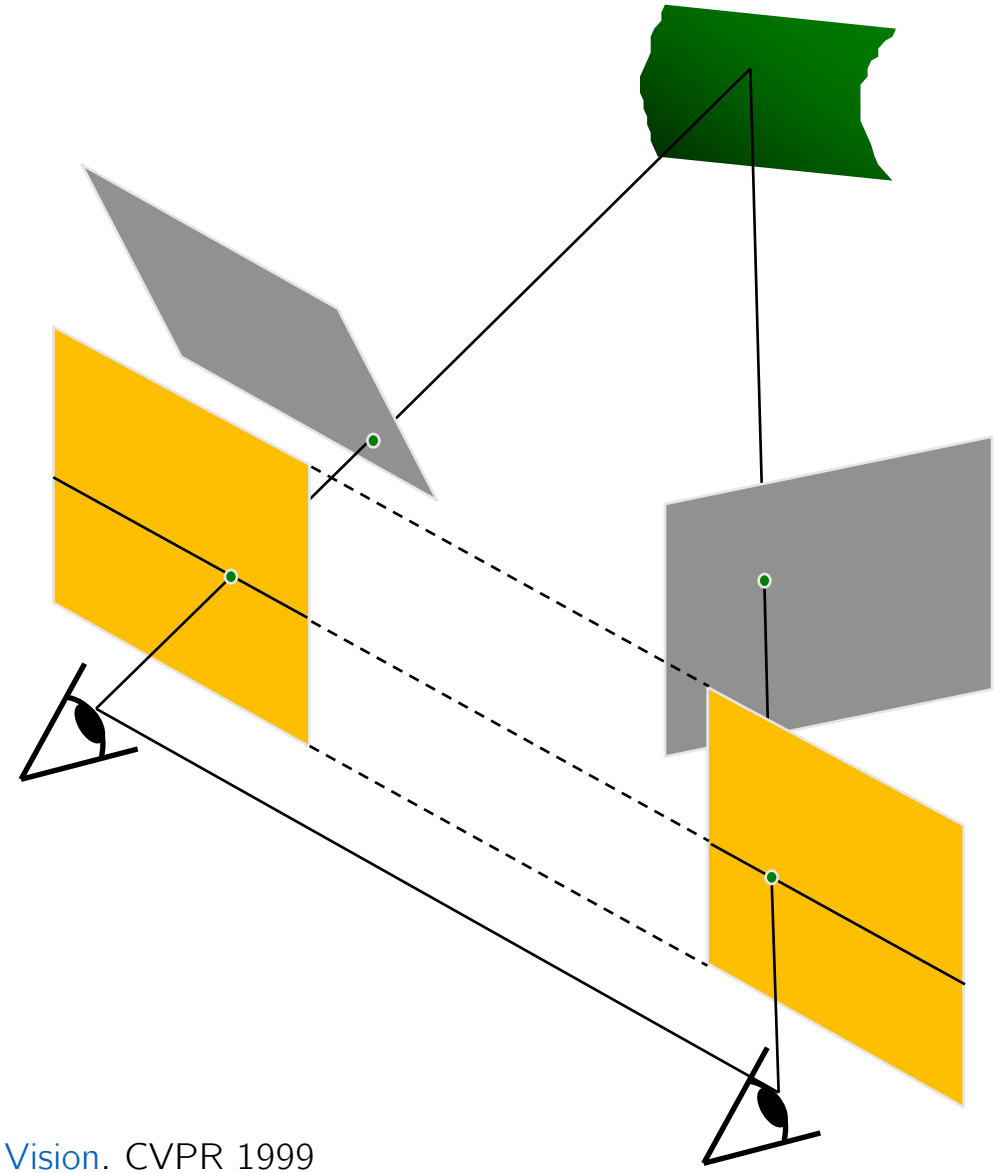
If the image planes are not parallel, we can find homographies to project each view onto a common plane parallel to the baseline



C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. CVPR 1999

Stereo image rectification

If the image planes are not parallel, we can find homographies to project each view onto a common plane parallel to the baseline



C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision. CVPR 1999

Stereo image rectification

Before rectification:

Image source

Stereo image rectification

After rectification:

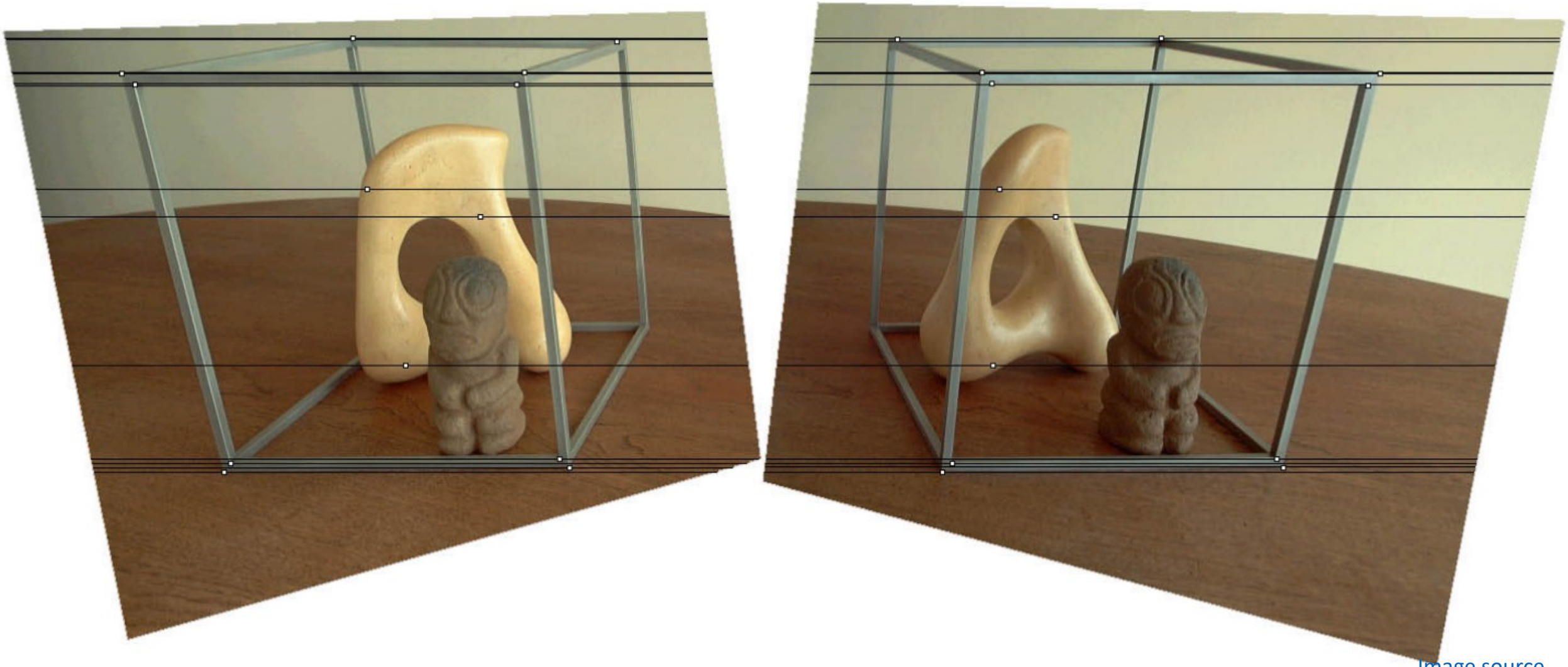
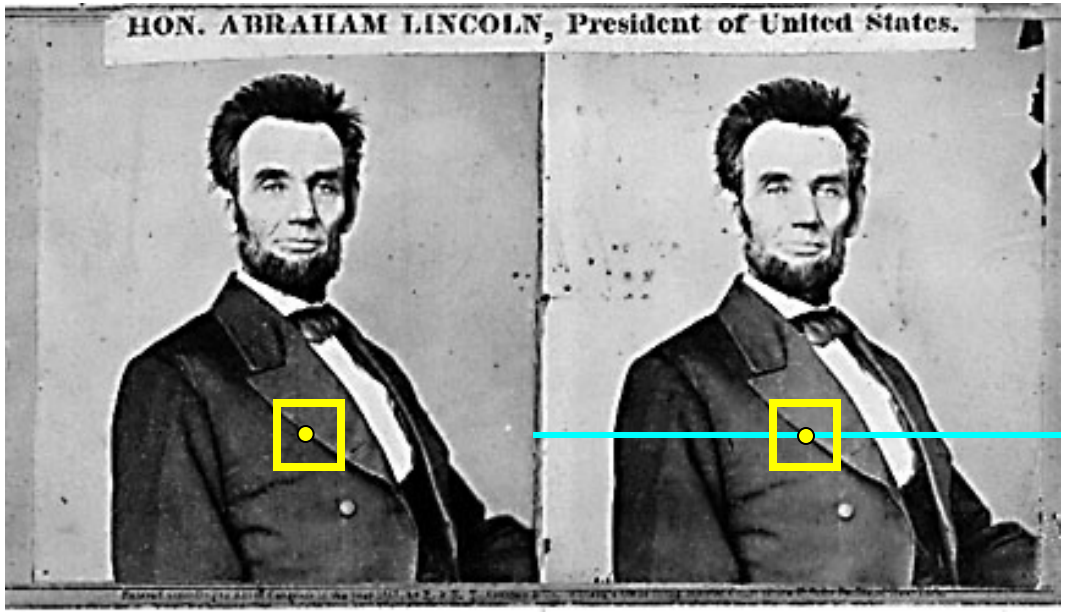


Image source

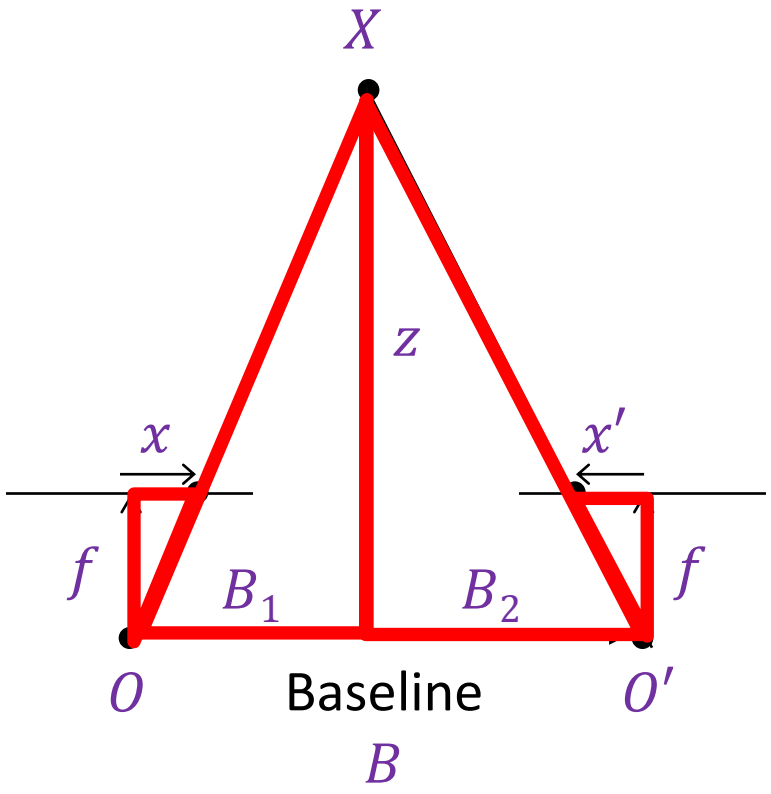
Another rectification example

Basic stereo matching algorithm



- If necessary, rectify the two stereo images to transform epipolar lines into horizontal scanlines
- For each pixel x in the first image
  - Find corresponding epipolar scanline in the right image
  - Examine all pixels on the scanline and pick the best match x'
  - Triangulate the matches to get depth information

Depth from disparity



$$\frac{x}{f} = \frac{B_1}{z} \qquad \frac{-x'}{f} = \frac{B_2}{z}$$

$$\frac{x - x'}{f} = \frac{B_1 + B_2}{z}$$

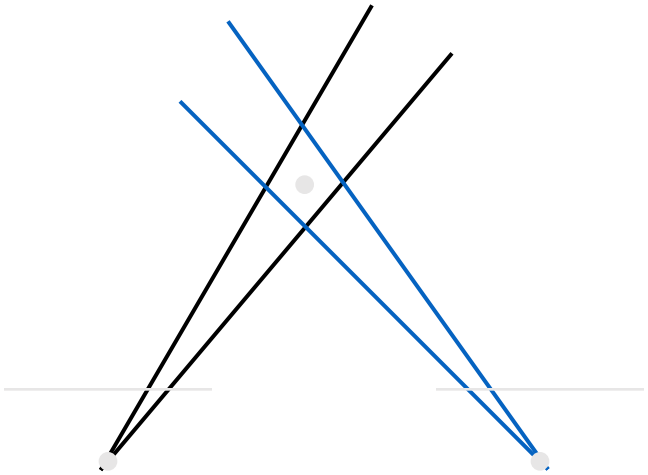
$$\underbrace{x - x'} = \frac{fB}{z}$$

Disparity

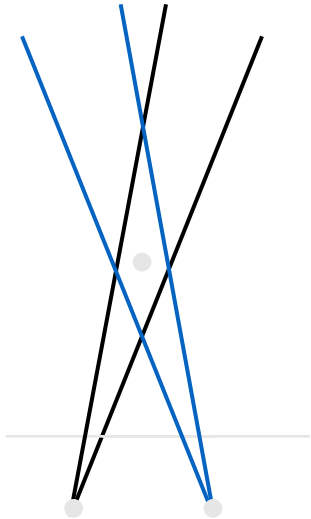
Disparity is inversely proportional to depth!

$$z = \frac{fB}{x - x'}$$

Effect of baseline on stereo results

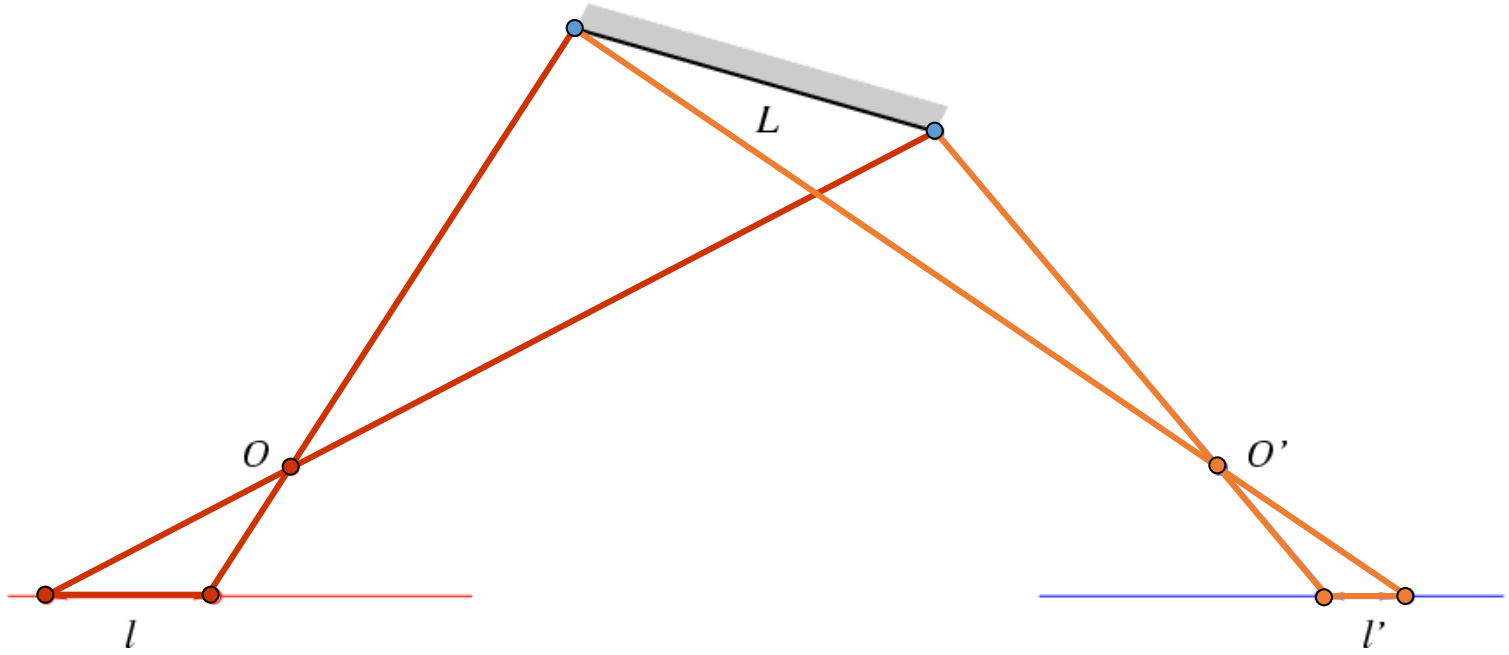


- Larger baseline
  - \+ Smaller triangulation error
  - – Matching is more difficult



- Smaller baseline
  - – Higher triangulation error
  - \+ Matching is easier

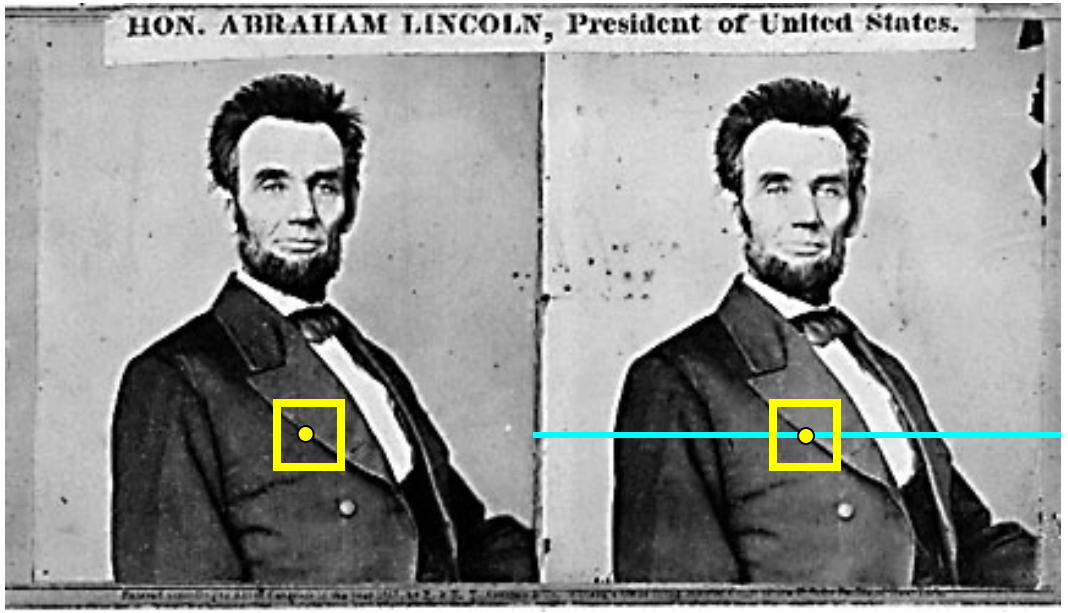
Problem for wide baselines: Foreshortening



- Matching with fixed-size windows will fail!
- Possible solution: adaptively vary window size
- Another solution: *model-based stereo* (CS231a)

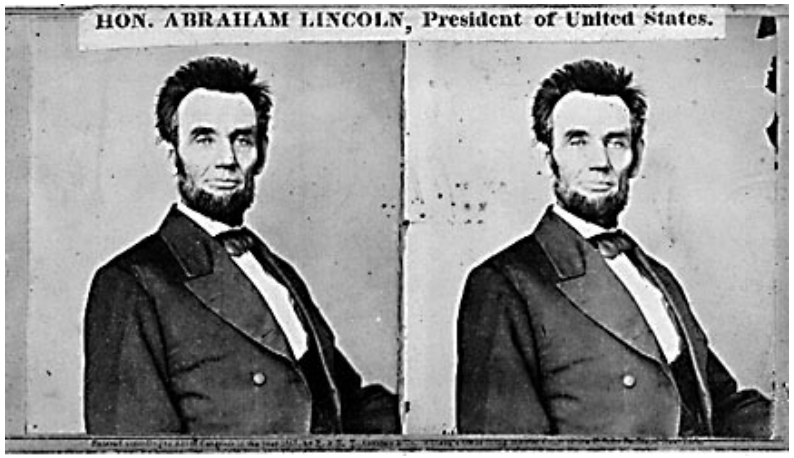
Slide credit: J. Hayes

Basic stereo matching algorithm



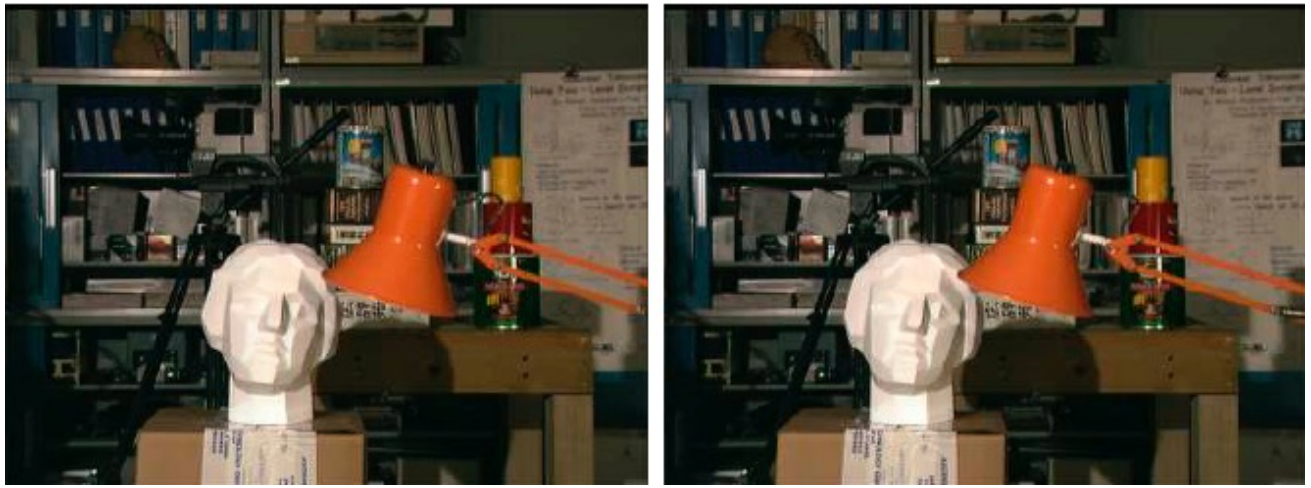
- If necessary, rectify the two stereo images to transform epipolar lines into scanlines
- For each pixel x in the first image
  - Find corresponding epipolar scanline in the right image
  - Examine all pixels on the scanline and pick the best match x'
  - Compute disparity $x - x'$ and set $\text{depth}(x) = Bf/(x - x')$

The correspondence problem



Textureless surfaces

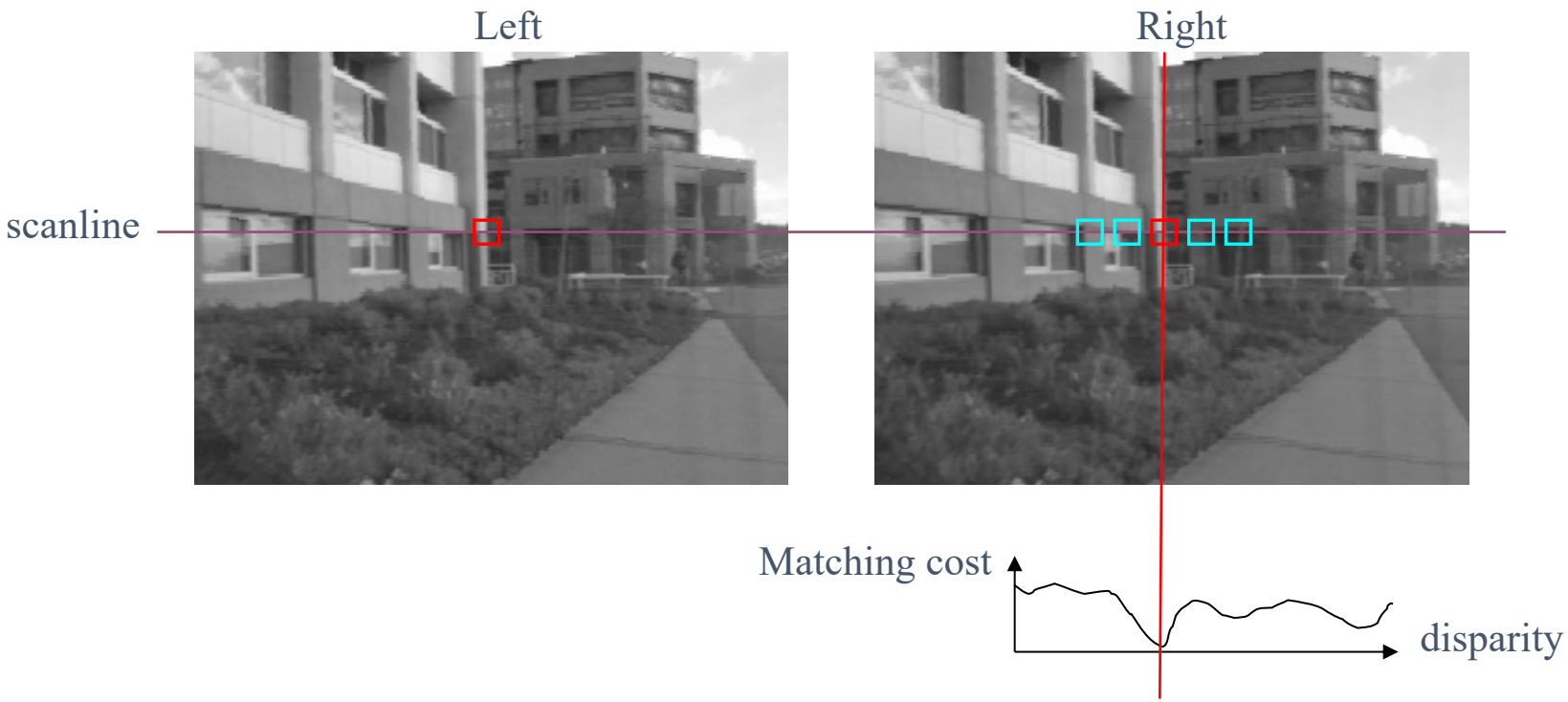
Occlusions, repetition



Non-Lambertian surfaces, specularities

How to find the best match?

- Compare pixels: sliding window + SSD or normalized cross-correlation



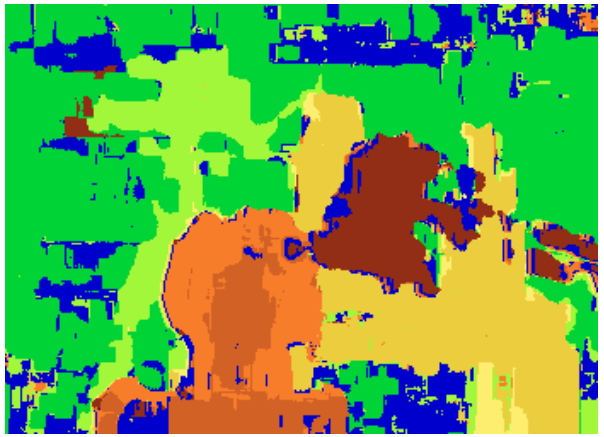
How to find the best match?

- Compare pixels: sliding window + SSD or normalized cross-correlation
- Global optimization (Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001)

Data

Ground truth

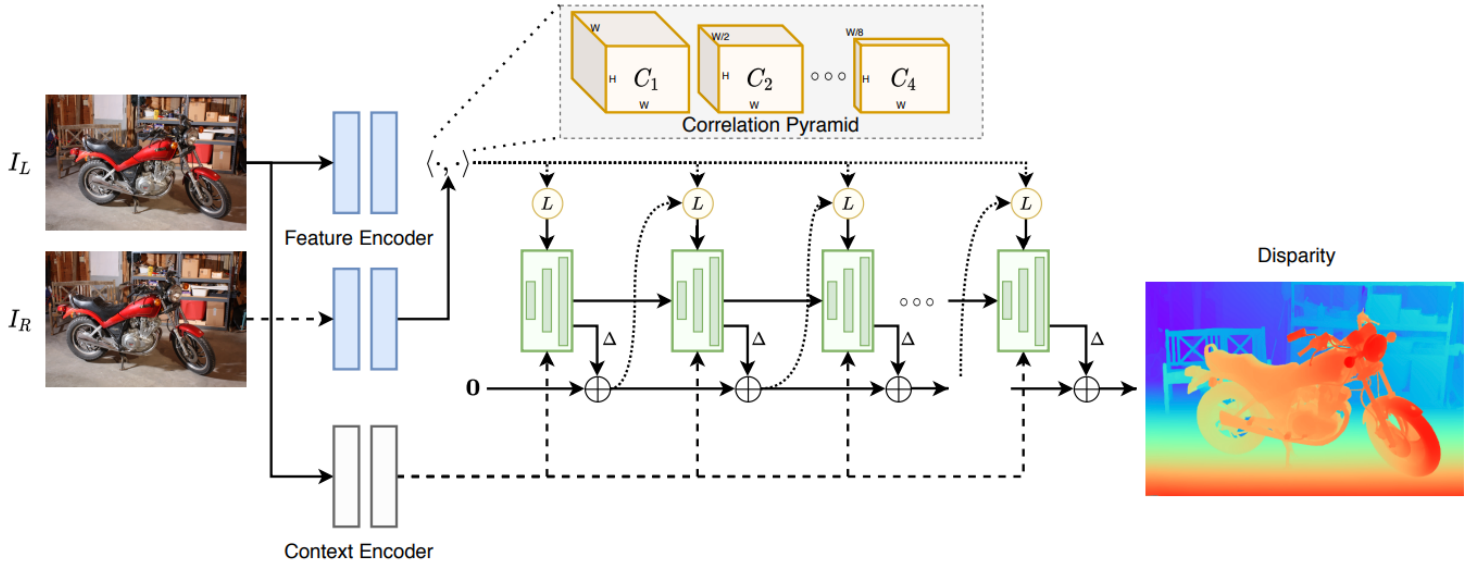
Window-based matching



Global optimization method (graph cuts)

How to find the best match?

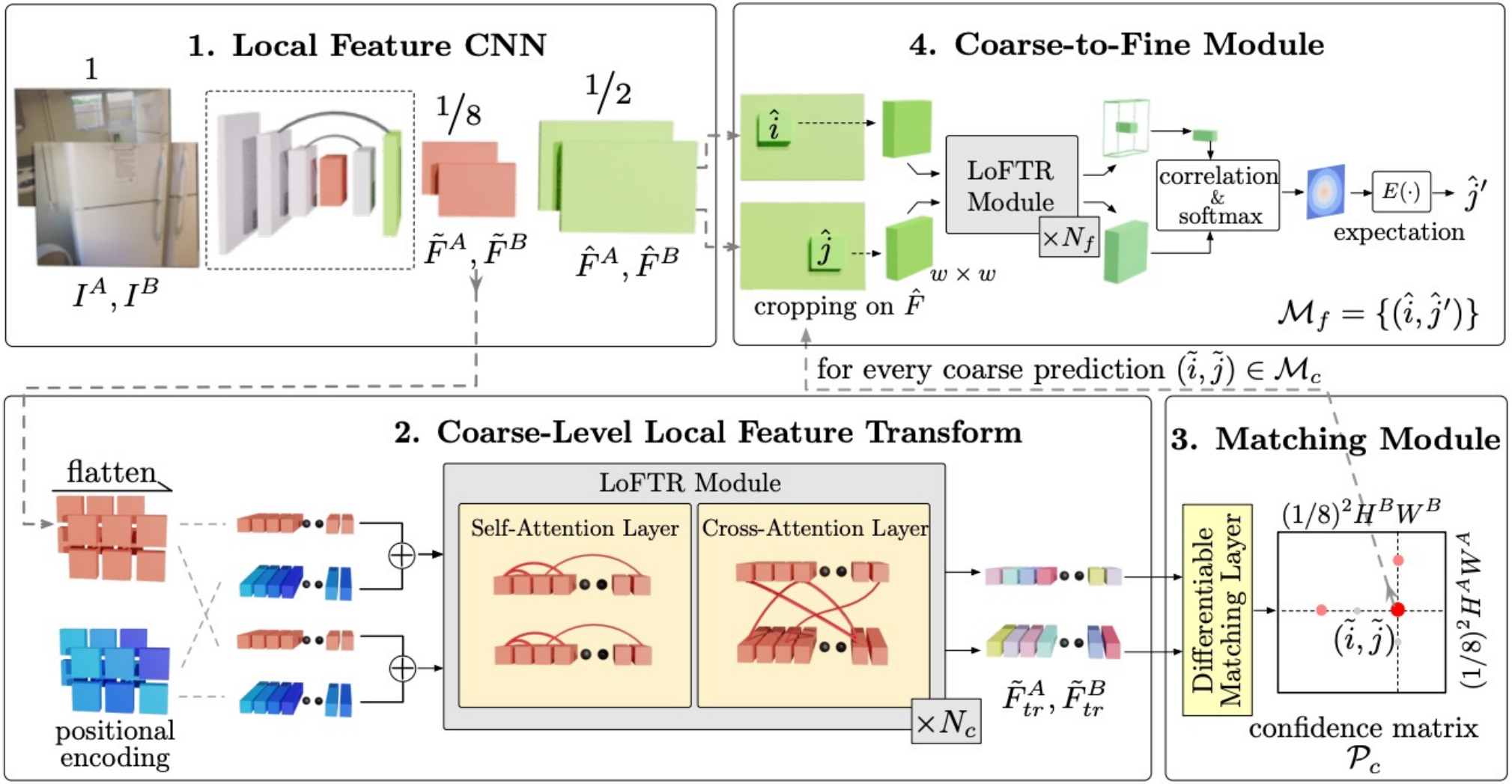
- Compare pixels: sliding window + SSD or normalized cross-correlation
- Global optimization (Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001)
- Deep Learning: match learned representations, end-to-end estimation



RAFT-Stereo: Multilevel Recurrent Field Transforms for Stereo Matching, L. Lipson et al, arXiv 2021

SuperDepth: Self-Supervised, Super-Resolved Monocular Depth Estimation, Pillai et al, ICRA'19

Correspondences in the age of Deep Learning

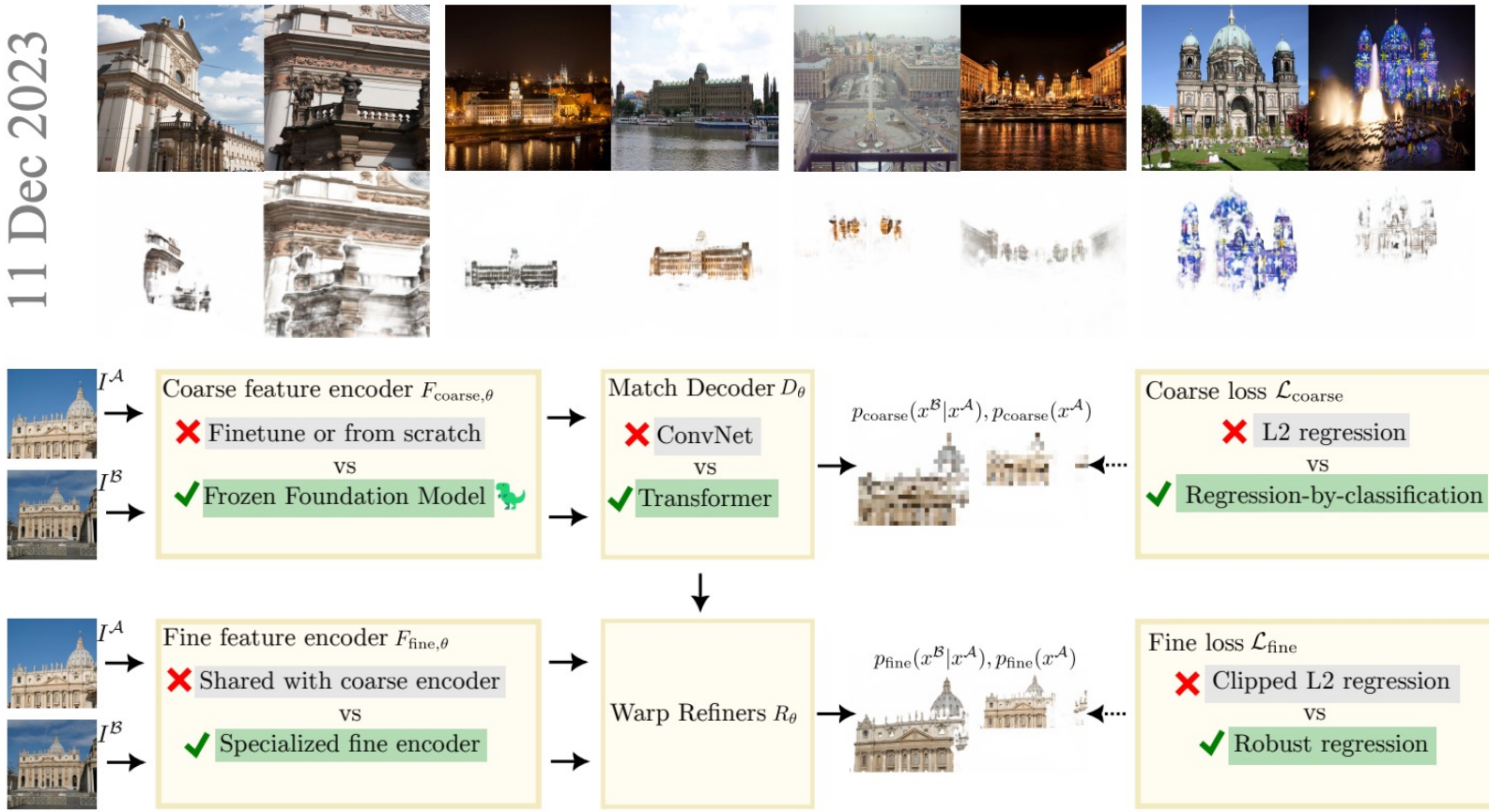


LoFTR: Detector-Free Local Feature Matching with Transformers (CVPR'21) https://arxiv.org/abs/2104.00680v1

Correspondences in the age of Deep Learning

RoMa: Robust Dense Feature Matching

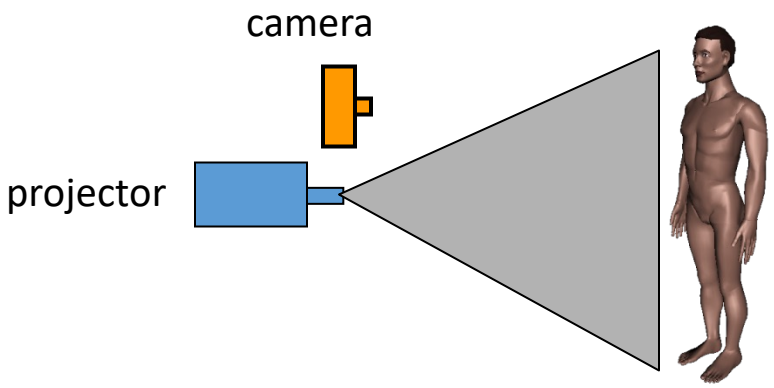
Johan Edstedt[1] Qiyu Sun[2] Georg Bökman[3] Mårten Wadenbäck[1] Michael Felsberg[1]
[1]Linköping University, [2]East China University of Science and Technology, [3]Chalmers University of Technology

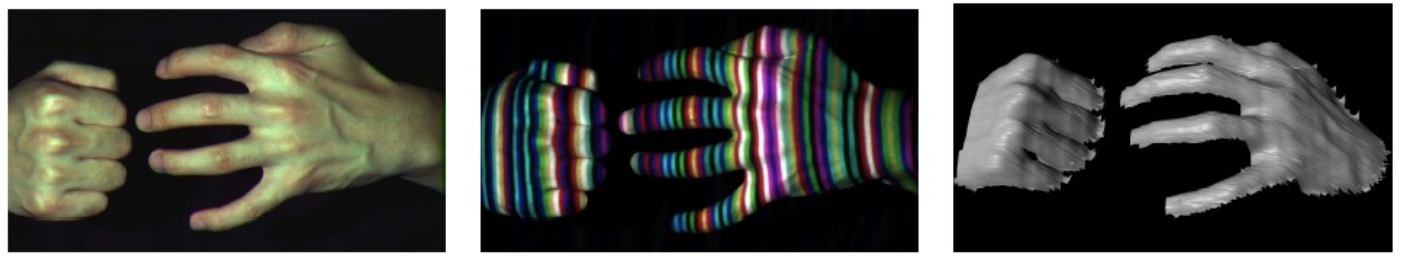


RoMa: Robust Dense Feature Matching https://parskatt.github.io/RoMa/

How to find the best match?

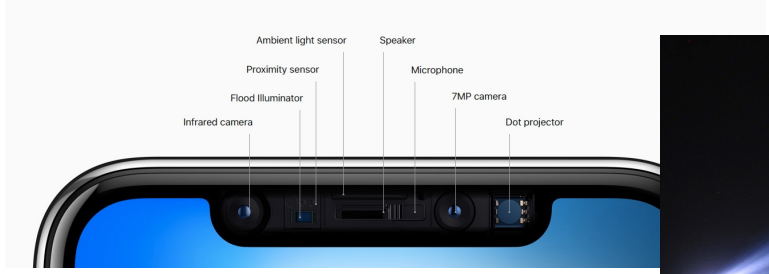
- Compare pixels: sliding window + SSD or normalized cross-correlation
- Global optimization (Y. Boykov, O. Veksler, and R. Zabih, Fast Approximate Energy Minimization via Graph Cuts, PAMI 2001)
- ***Deep Learning***: match learned representations, end-to-end estimation
- "Active stereo": project lasers or structured patterns (e.g., in IR)





L. Zhang, et al. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. *3DPVT* 2002

http://bbzippo.wordpress.com/2010/11/28/kinect-in-infrared/



https://www.cnet.com/news/apple-face-id-truedepth-how-it-works/

Bonus slides – SfM ambiguities

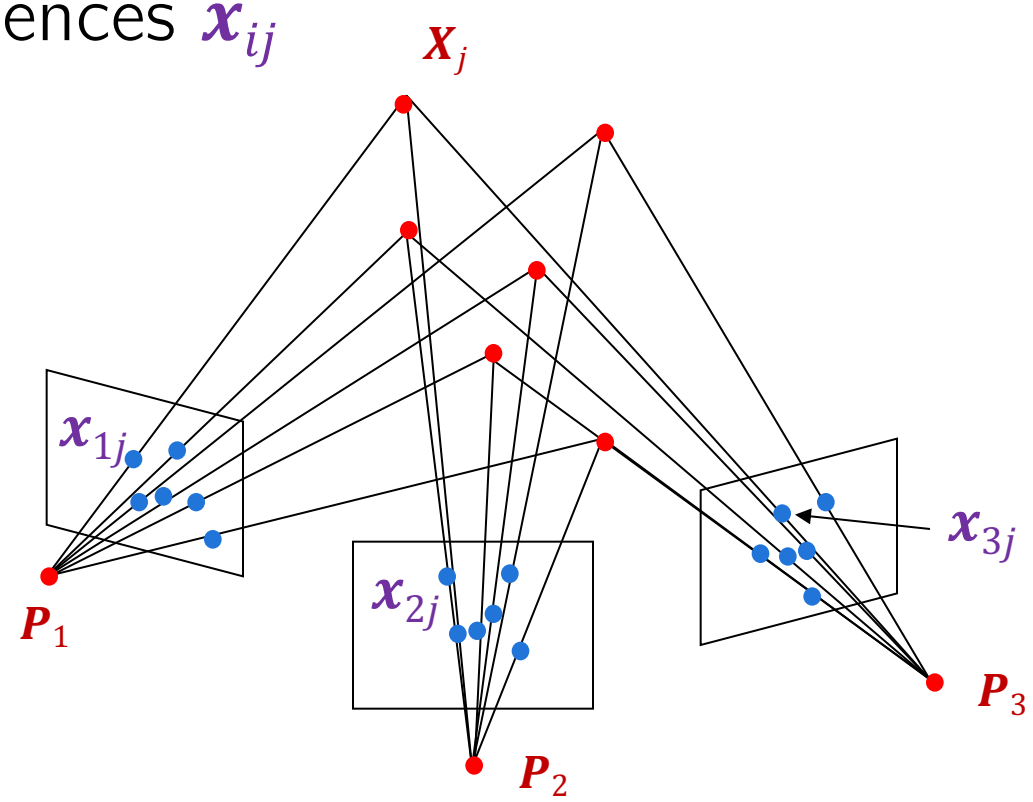
Slides credit: S. Lazebnik

Projective structure from motion

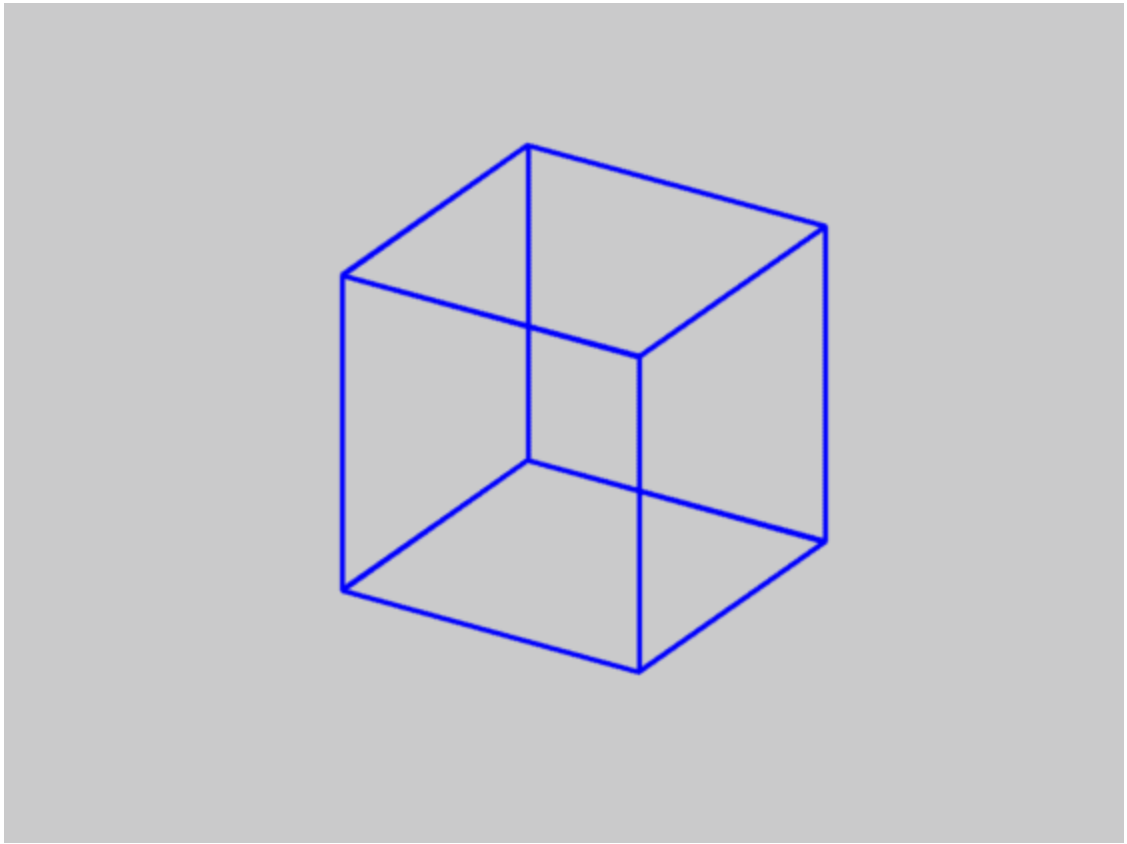
- *Given*: m images of n fixed 3D points such that (ignoring visibility):

$$\boldsymbol{x}_{ij} \sim \boldsymbol{P}_i \boldsymbol{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

- *Problem*: estimate m projection matrices $\boldsymbol{P}_i$ and n 3D points $\boldsymbol{X}_j$ from the mn correspondences $\boldsymbol{x}_{ij}$



Is SFM always uniquely solvable?



Necker cube

Source: N. Snavely

Structure from motion ambiguity

If we scale the entire scene by some factor k and, at the same time, scale the camera matrices by the factor of $1/k$, the projections of the scene points remain exactly the same:

$$\boldsymbol{x} \sim \boldsymbol{P}\boldsymbol{X} = \left(\frac{1}{k}\boldsymbol{P}\right)(k\boldsymbol{X})$$

Without a reference measurement, it is impossible to recover the absolute scale of the scene!

In general, if we transform the scene using a transformation $\boldsymbol{Q}$ and apply the inverse transformation to the camera matrices, then the image observations do not change:

$$\boldsymbol{x} \sim \boldsymbol{P}\boldsymbol{X} = (\boldsymbol{P}\boldsymbol{Q}^{-1})(\boldsymbol{Q}\boldsymbol{X})$$

Projective structure from motion

- *Given*: m images of n fixed 3D points such that (ignoring visibility):

$$\boldsymbol{x}_{ij} \sim \boldsymbol{P}_i \boldsymbol{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

- *Problem*: estimate m projection matrices $\boldsymbol{P}_i$ and n 3D points $\boldsymbol{X}_j$ from the mn correspondences $\boldsymbol{x}_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4×4 projective transformation $\boldsymbol{Q}$:

$$\boldsymbol{X} \rightarrow \boldsymbol{Q}\boldsymbol{X}, \boldsymbol{P} \rightarrow \boldsymbol{P}\boldsymbol{Q}^{-1} \Rightarrow \boldsymbol{x} \sim \boldsymbol{P}\boldsymbol{X} = \boldsymbol{P}\boldsymbol{Q}^{-1}\boldsymbol{Q}\boldsymbol{X}$$

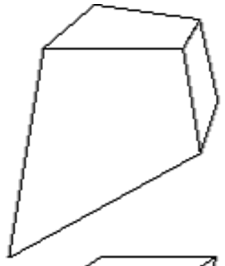
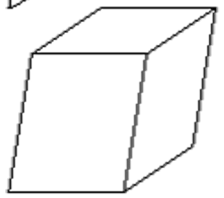
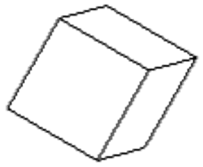
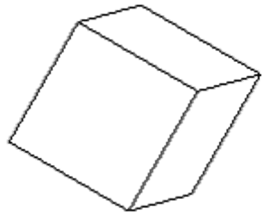
- We can solve for structure and motion when $2mn \geq 11m + 3n - 15$
- Inequality above --> for $m = 2$ cameras, need at least $n = 7$ points
- Remember: $\boldsymbol{F}$ has 7 d.o.f., not a coincidence!

Projective SFM: Two-camera case

1. Estimate fundamental matrix $\boldsymbol{F}$ between the two views
2. Set first camera matrix to $[\boldsymbol{I} \mid \boldsymbol{0}]$
3. Then the second camera matrix is given by $[\boldsymbol{A} \mid \boldsymbol{t}]$ where $\boldsymbol{t}$ is the epipole ($\boldsymbol{F}^T\boldsymbol{t} = \boldsymbol{0}$) and $\boldsymbol{A} = [\boldsymbol{t}]_\times \boldsymbol{F}$

For derivation: cf. CS231A or H&Z Chap.9 result 9.9

Types of ambiguity

Projective 15dof	$\begin{bmatrix} A & t \\ v^T & v \end{bmatrix}$		Preserves intersection and tangency
Affine 12dof	$\begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}$		Preserves parallellism, volume ratios
Similarity 7dof	$\begin{bmatrix} sR & t \\ 0^T & 1 \end{bmatrix}$		Preserves angles, ratios of length
Euclidean 6dof	$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$		Preserves angles, lengths

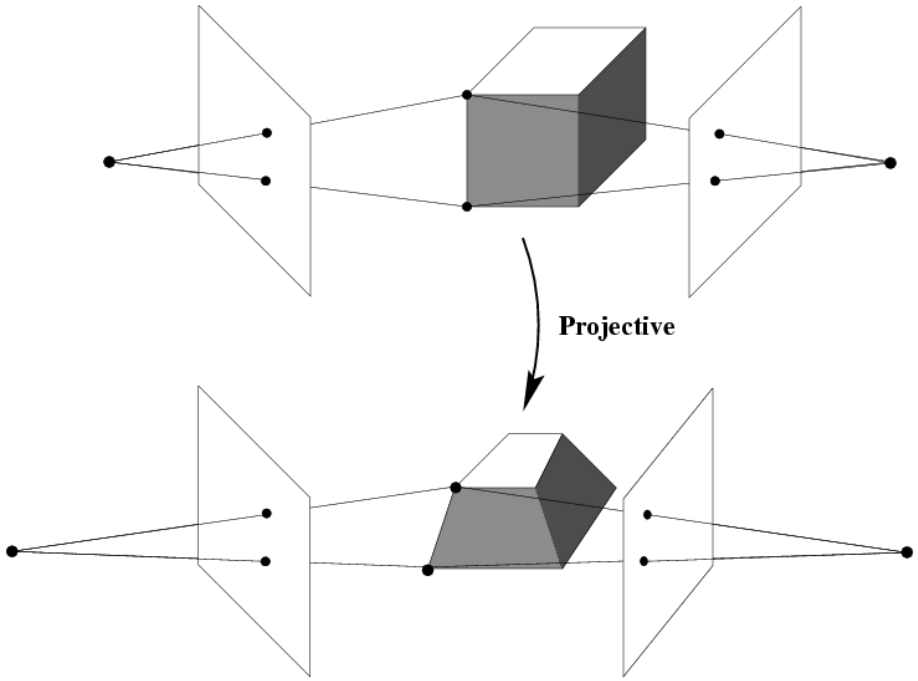
- With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction
- Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean

Projective ambiguity

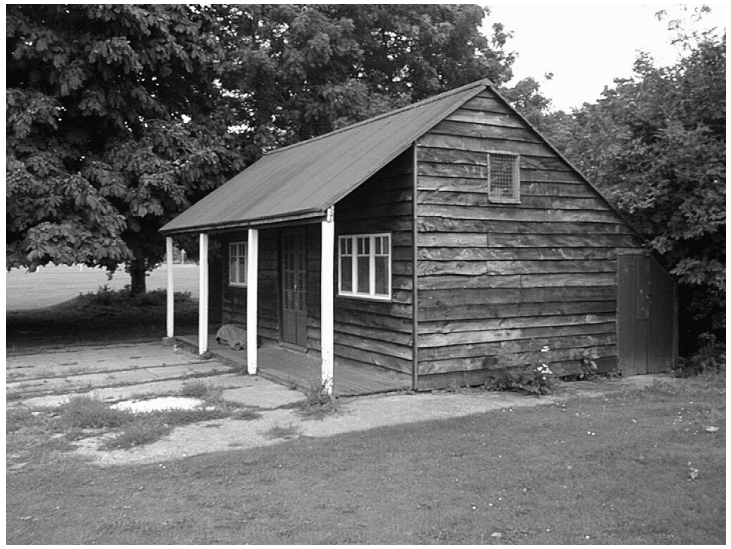
With no constraints on the camera calibration matrices or on the scene, we can reconstruct up to a *projective* ambiguity:

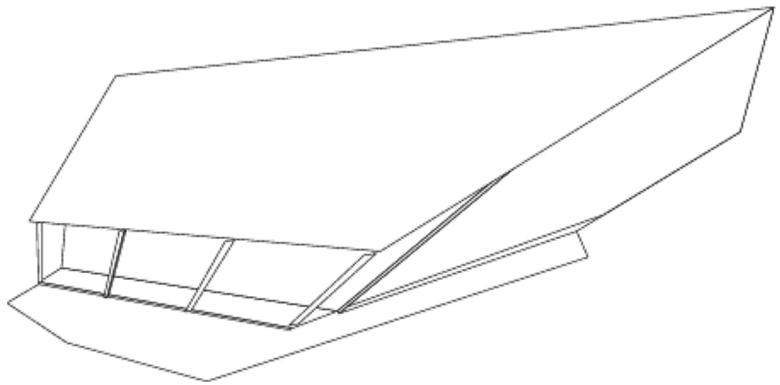
$$\boldsymbol{x \sim PX = (PQ^{-1})(QX)}$$

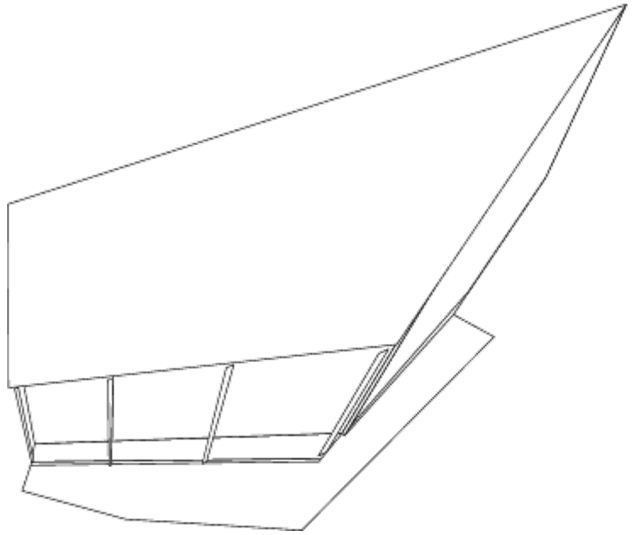
$\boldsymbol{Q}$ is a general full-rank 4×4 matrix



Projective ambiguity







Affine ambiguity

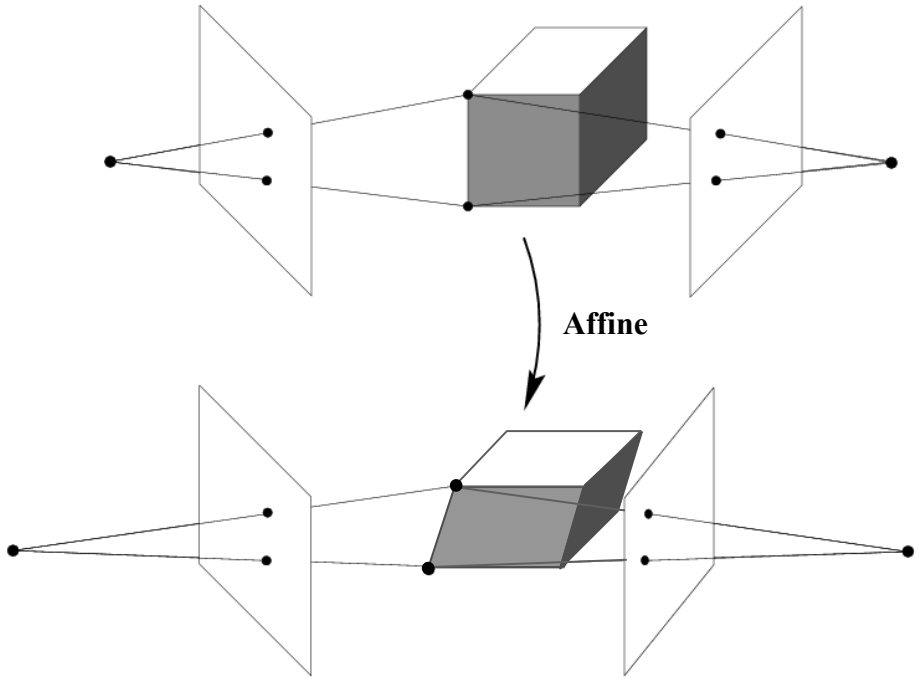
If we impose parallelism constraints, we can get a reconstruction up to an *affine* ambiguity:

$$\boldsymbol{x \sim PX = \left(PQ_A^{-1}\right)\left(Q_A X\right)}$$

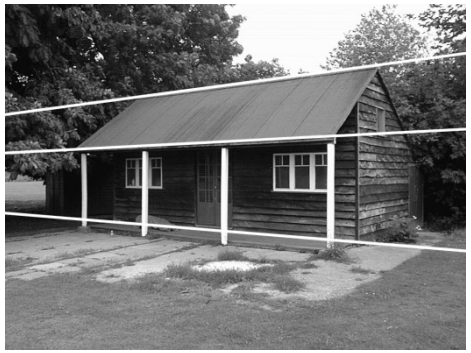
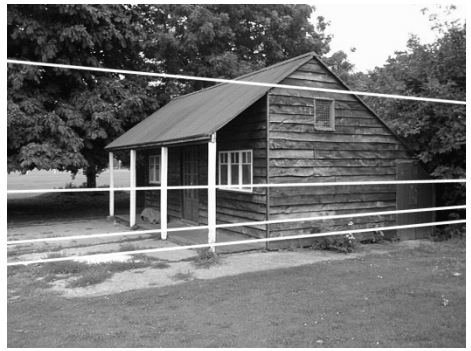
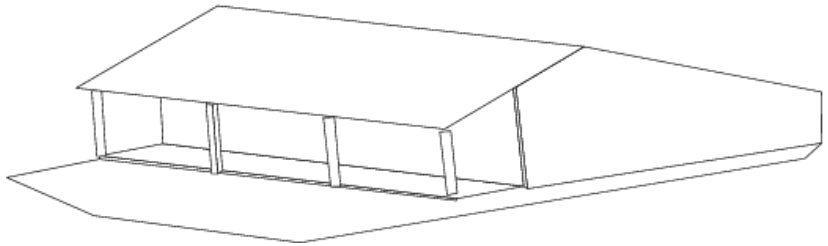
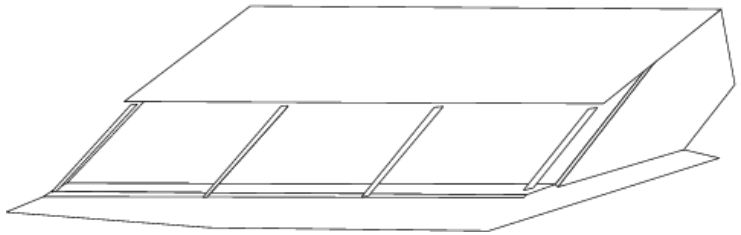
3×3 full-rank matrix

3×1 translation vector

$$\boldsymbol{Q_A = \begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}}$$



Affine ambiguity

Similarity ambiguity

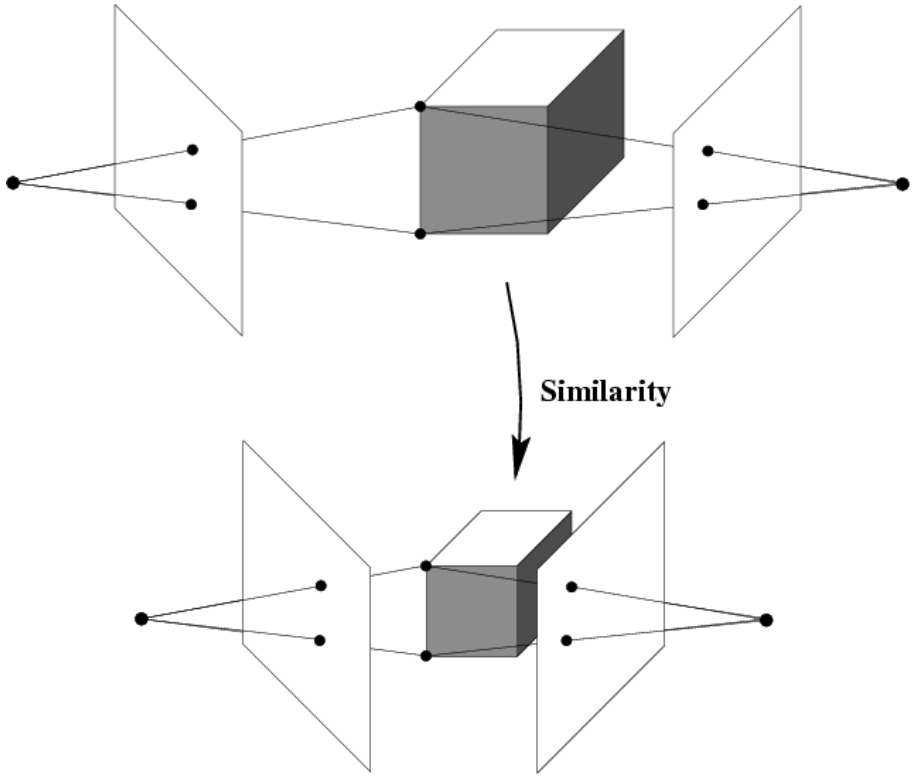
A reconstruction that obeys orthogonality constraints on camera parameters and/or scene

$$x \sim PX = \left(PQ_S^{-1}\right)\left(Q_S X\right)$$

3×3 rotation matrix

3×1 translation vector

$$Q_S = \begin{bmatrix} sR & t \\ 0^T & 1 \end{bmatrix}$$



Similarity ambiguity