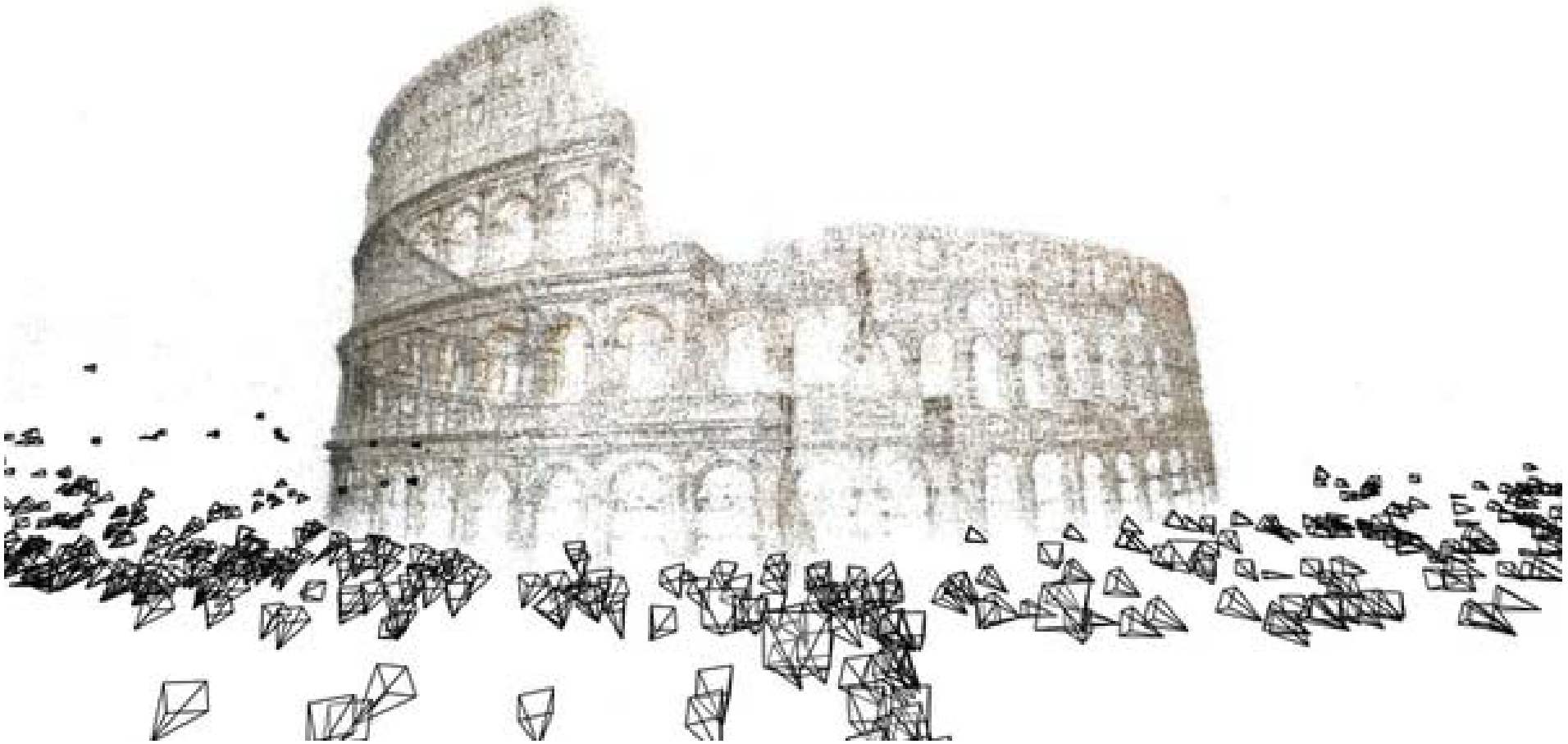
Structure from motion



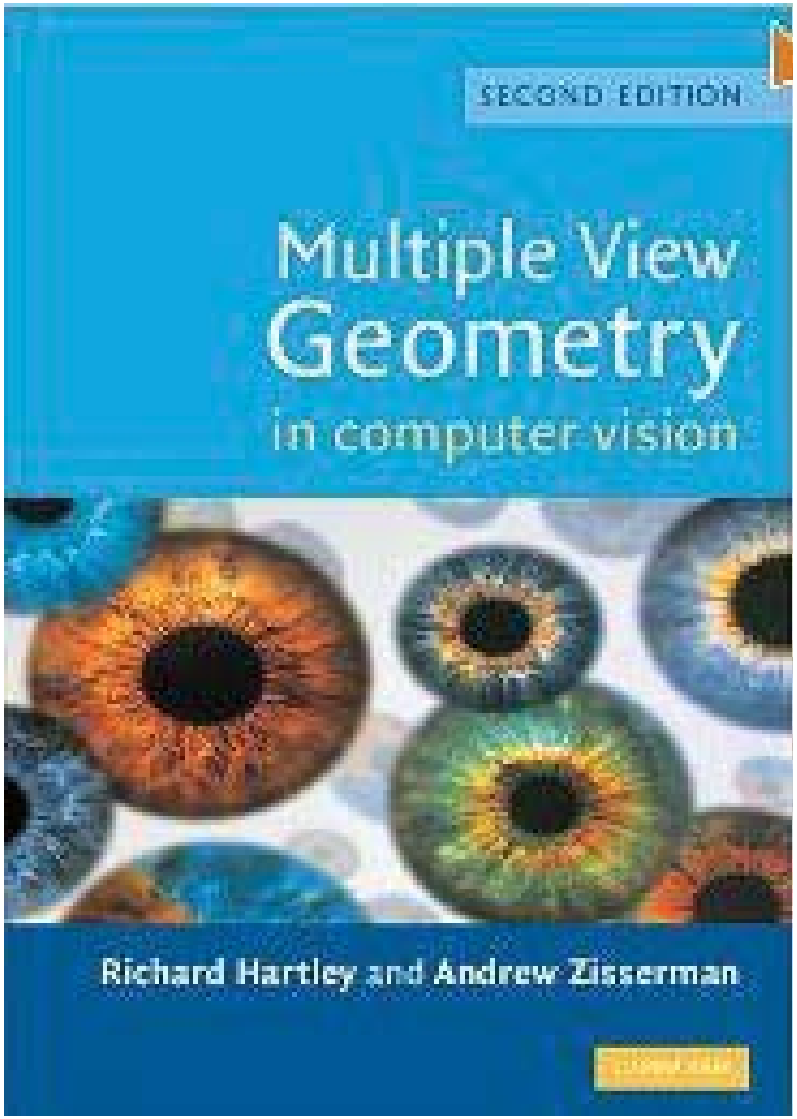
Slide credits: Ioannis Gkioulekas, Kris Kitani, Noah Snavely, Rob Fergus

Exam

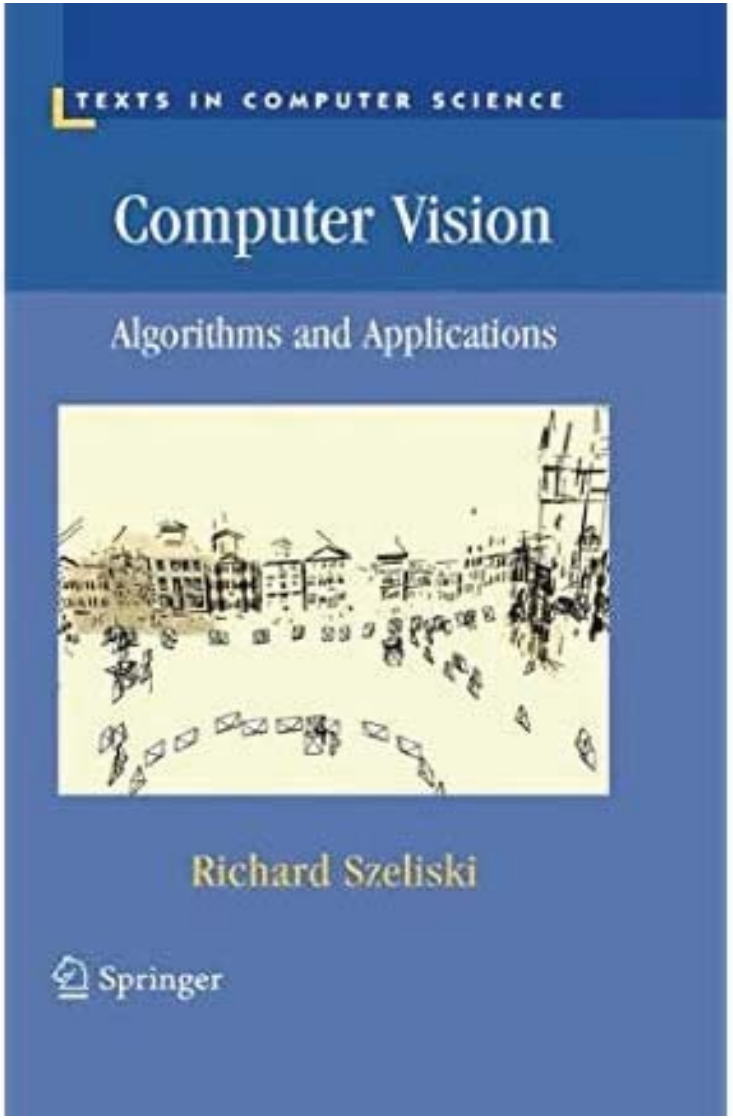
Open (printed only) material

Please refer to exams from previous years for preparation!
(I am going to review them for inspiration)

Recommended Books



Only about geometry, very detailed



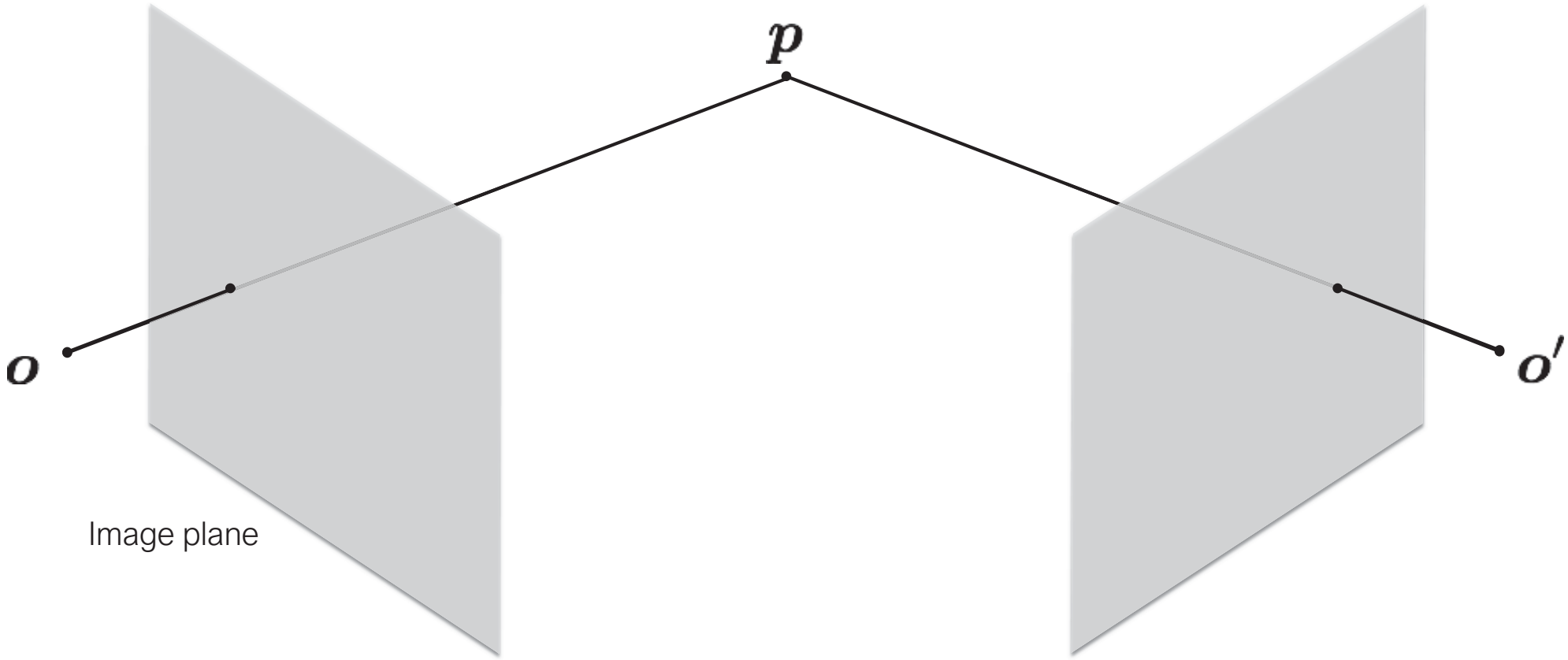
General computer vision, short geometry chapter

Overview of today's lecture

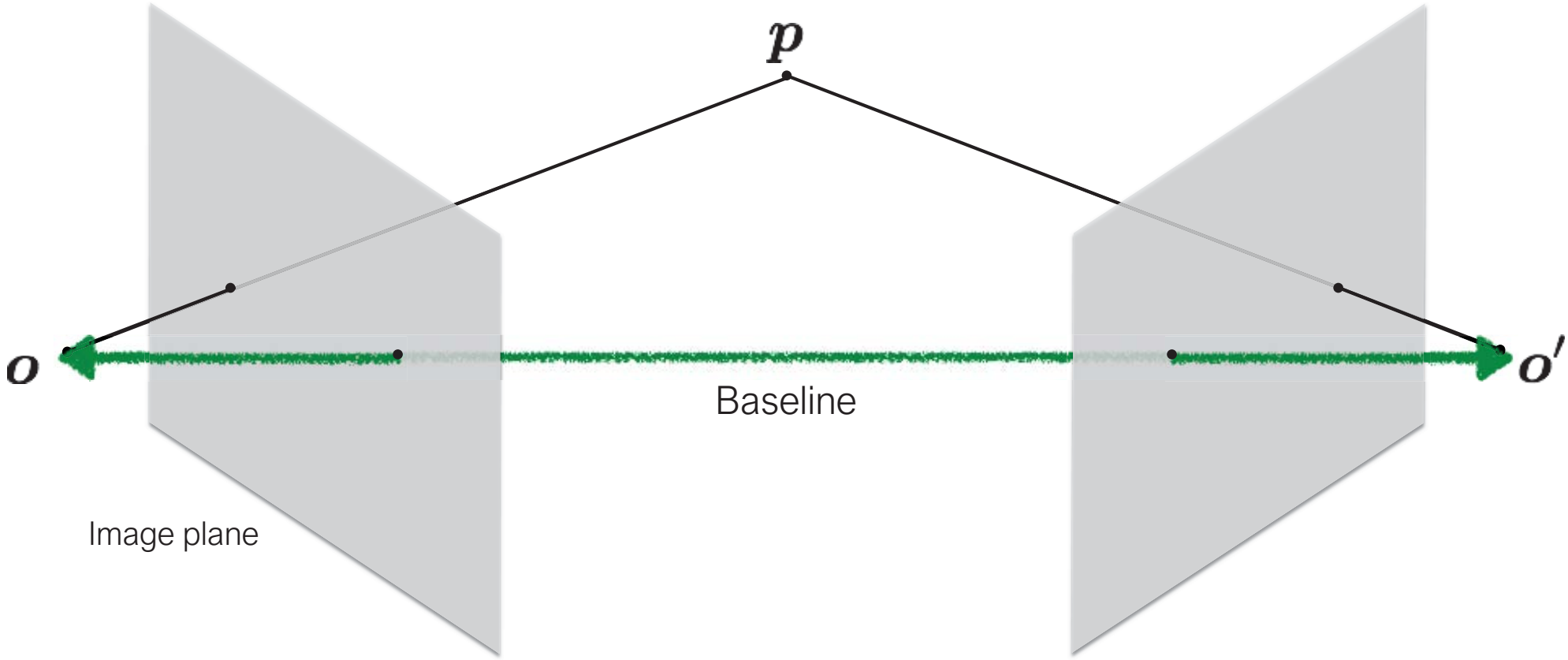
- Epipolar geometry review
- Estimating the F matrix
- A note on normalization.
- Two-view structure from motion.
- Ambiguities in structure from motion.
- Multi-view structure from motion.
- Large-scale structure from motion.
- Stereo

Epipolar geometry

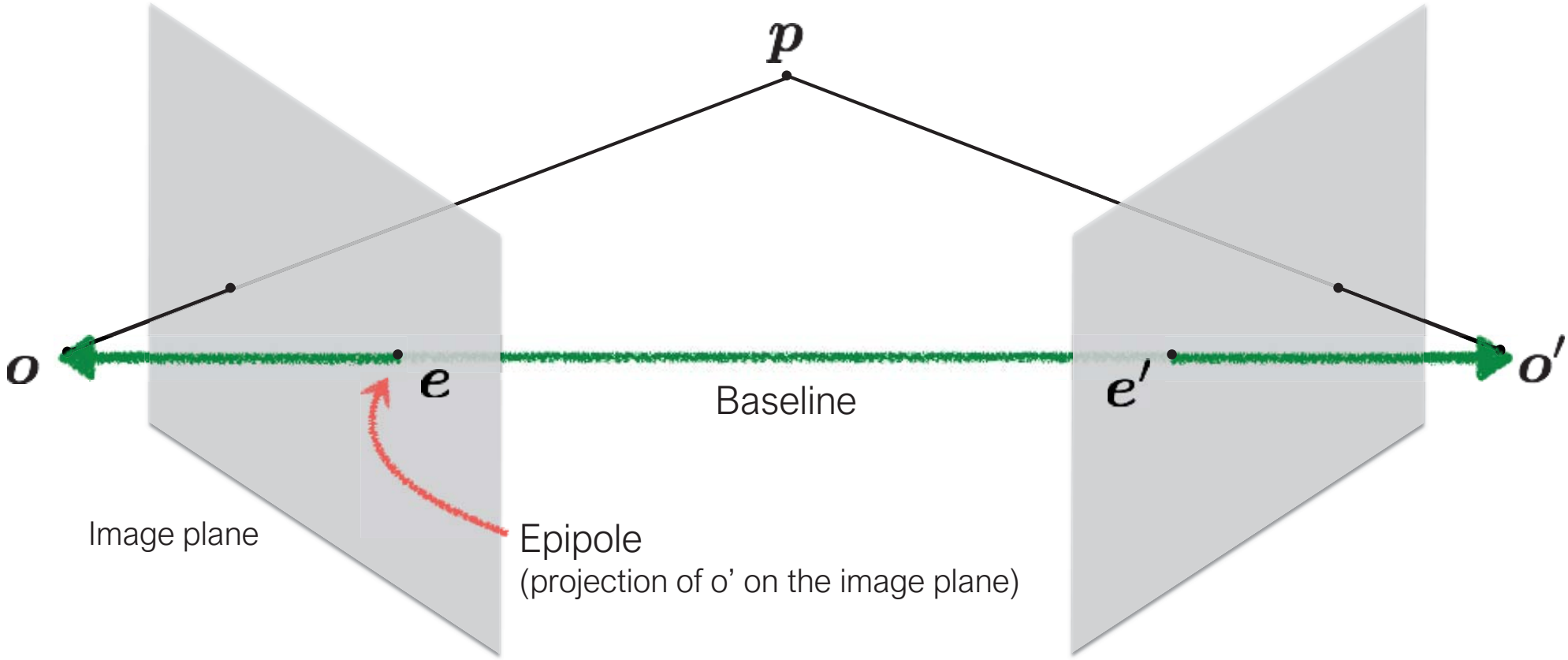
Epipolar geometry



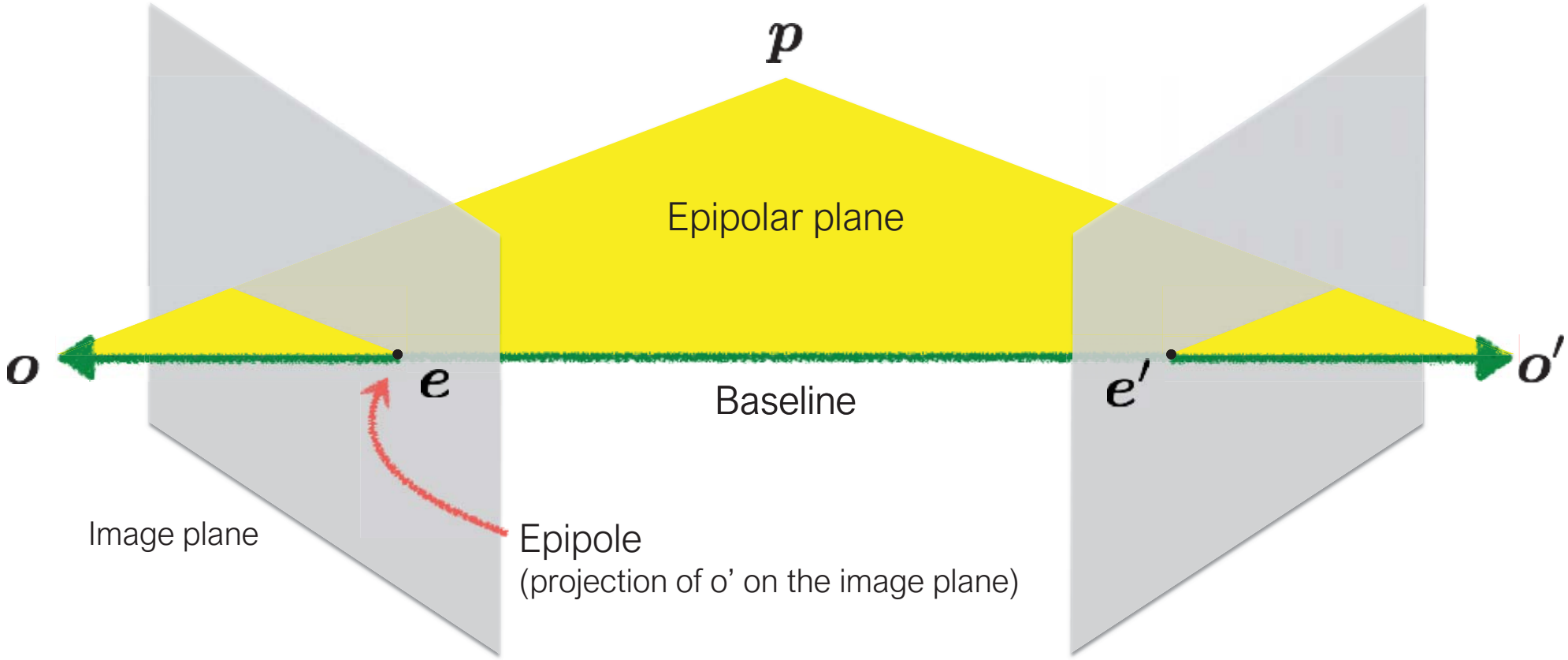
Epipolar geometry



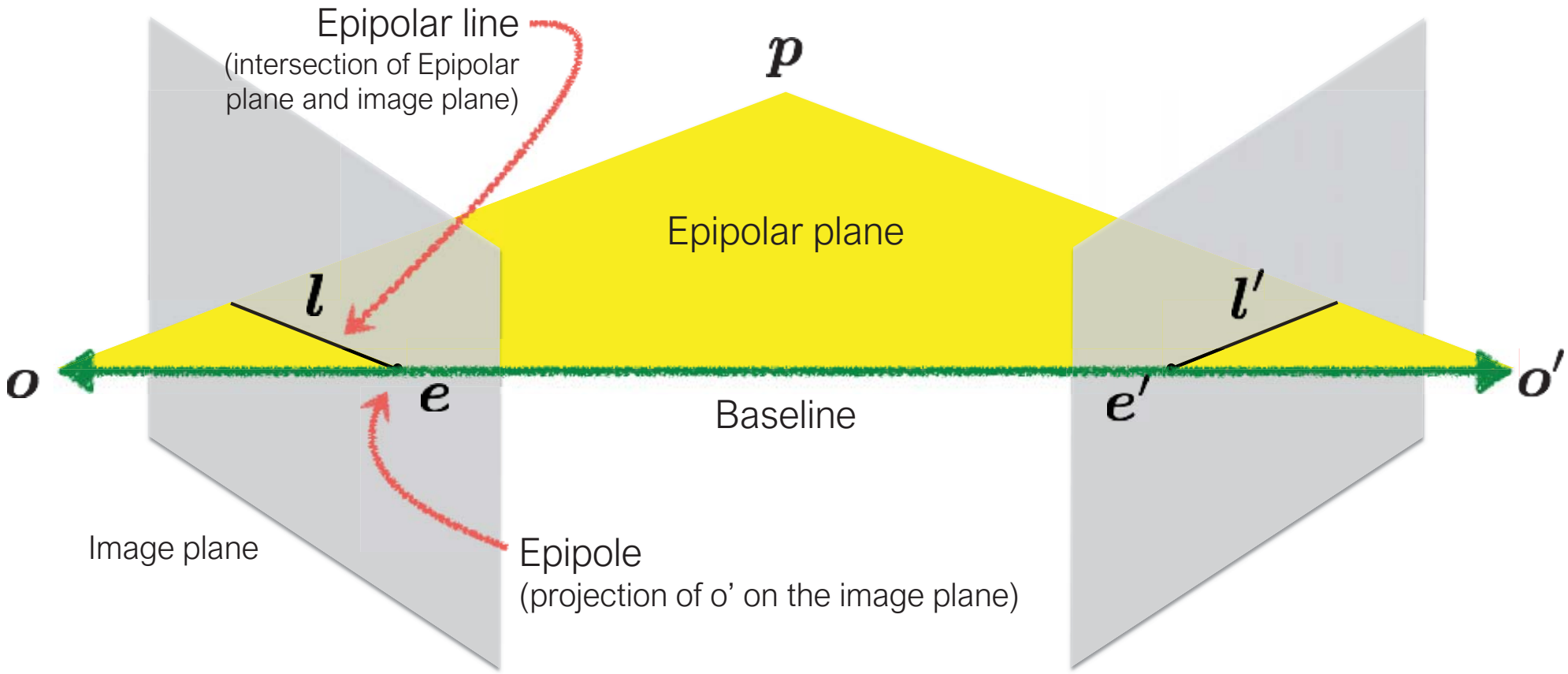
Epipolar geometry



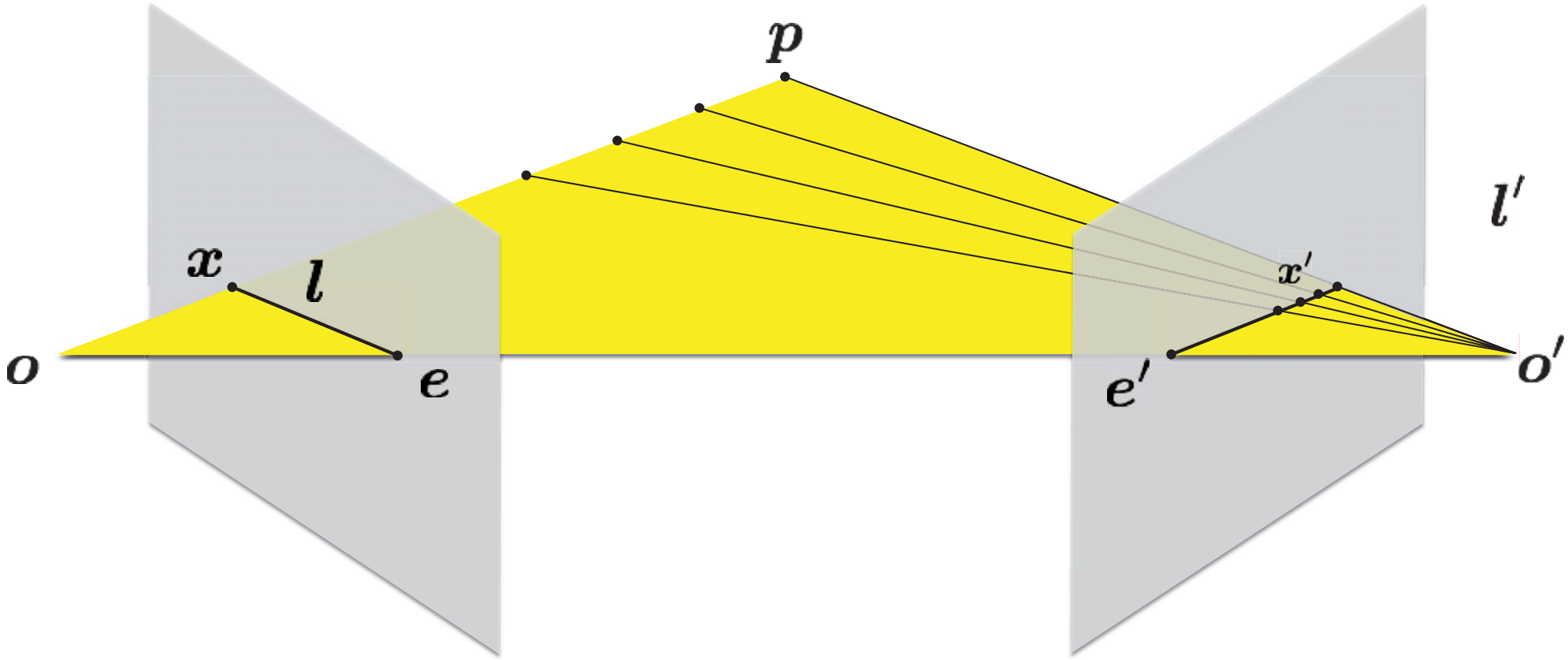
Epipolar geometry



Epipolar geometry

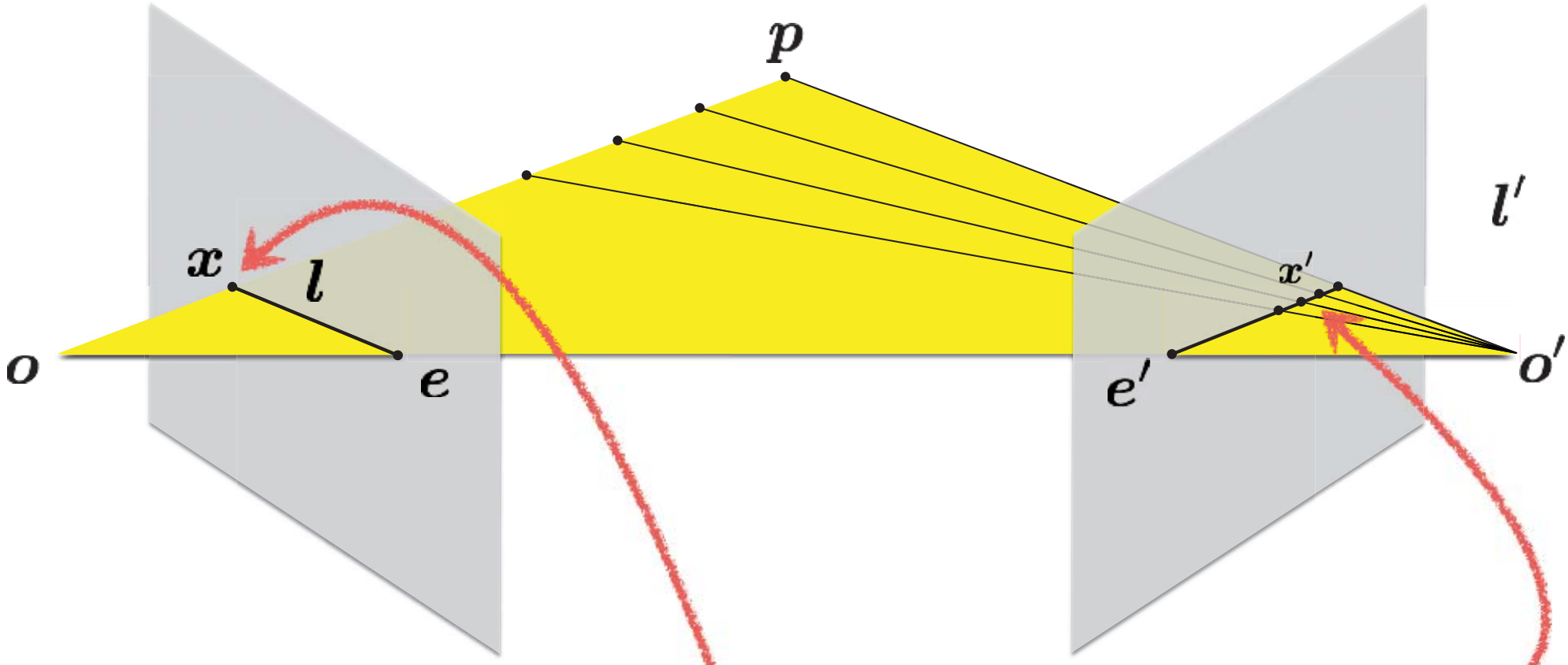


Epipolar constraint

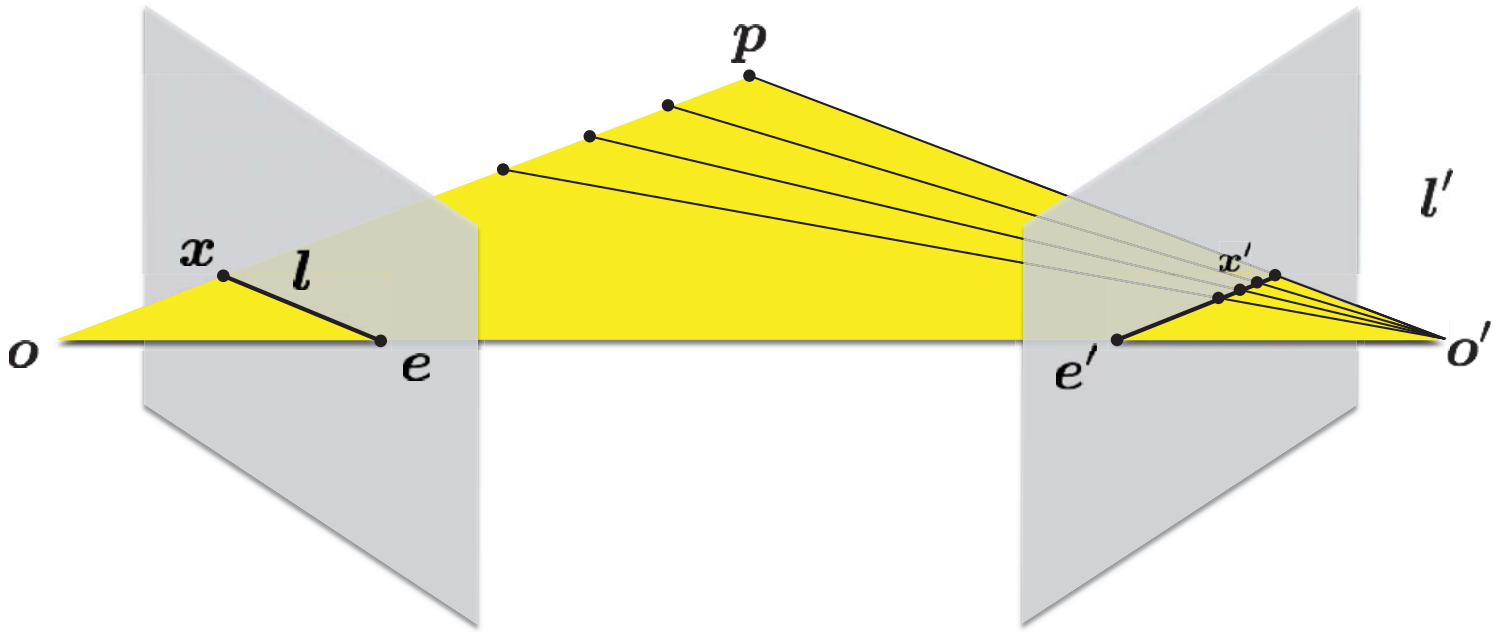


Potential matches for x lie on the epipolar line l'

Epipolar constraint



Potential matches for x lie on the epipolar line l'



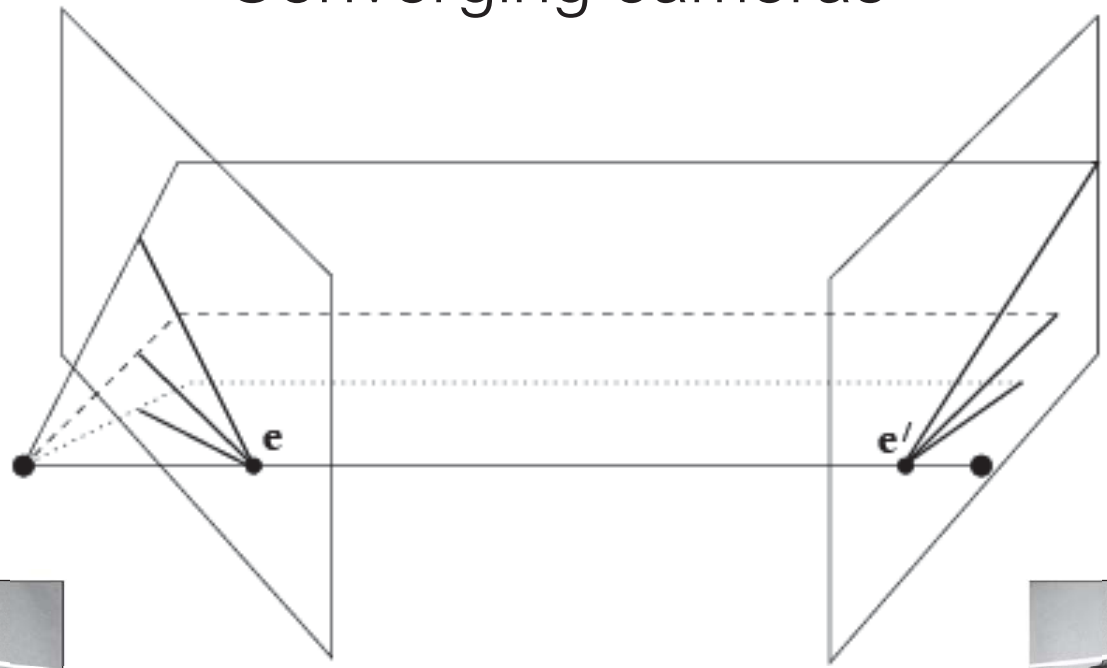
The point **x** (left image) maps to a Epipolar line in the right image

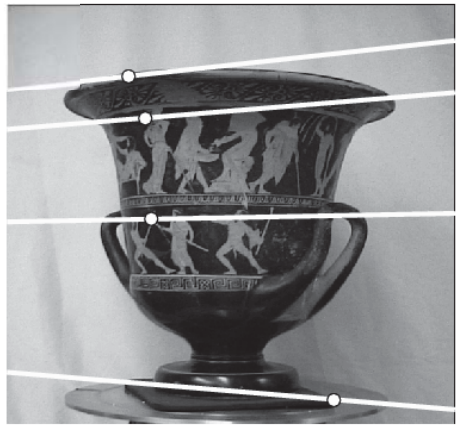
The baseline connects the Camera centers and epipols

An epipole **e** is a projection of the Camera center O' on the image plane

All epipolar lines in an image intersect at the epipole

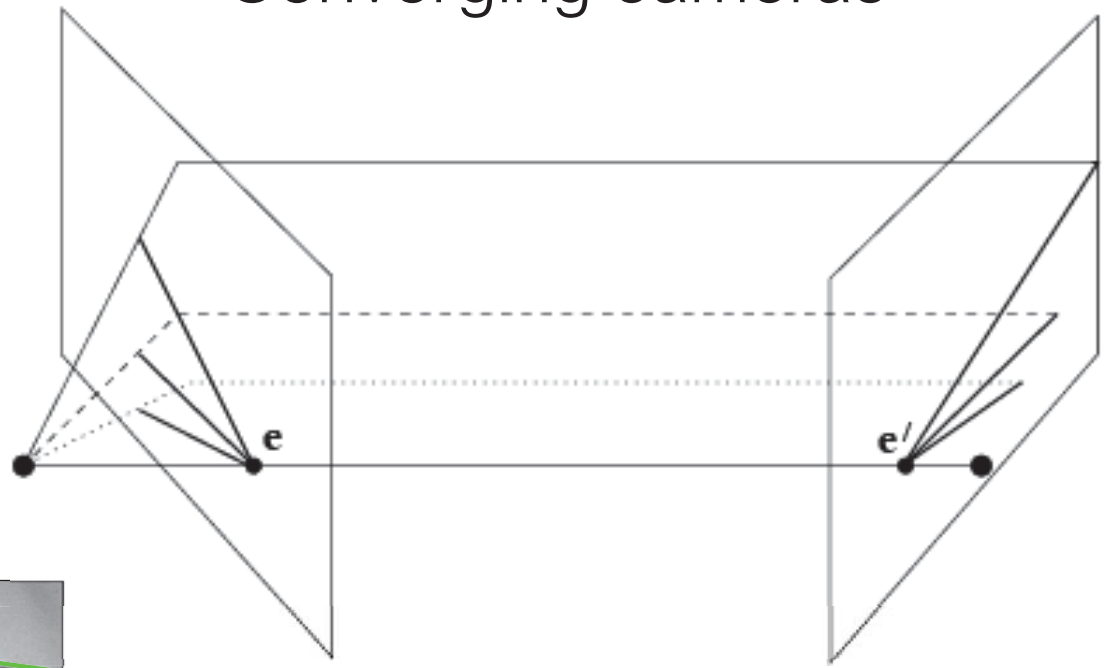
Converging cameras

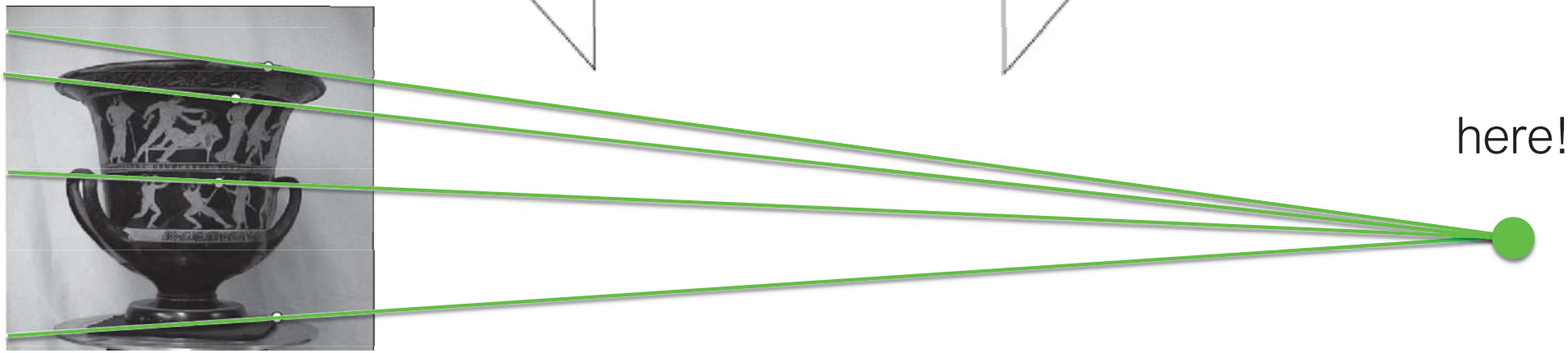




Where is the epipole in this image?

Converging cameras





here!

Where is the epipole in this image?

It's not always in the image

Plug: epipolar imaging.

A couple of interesting videos

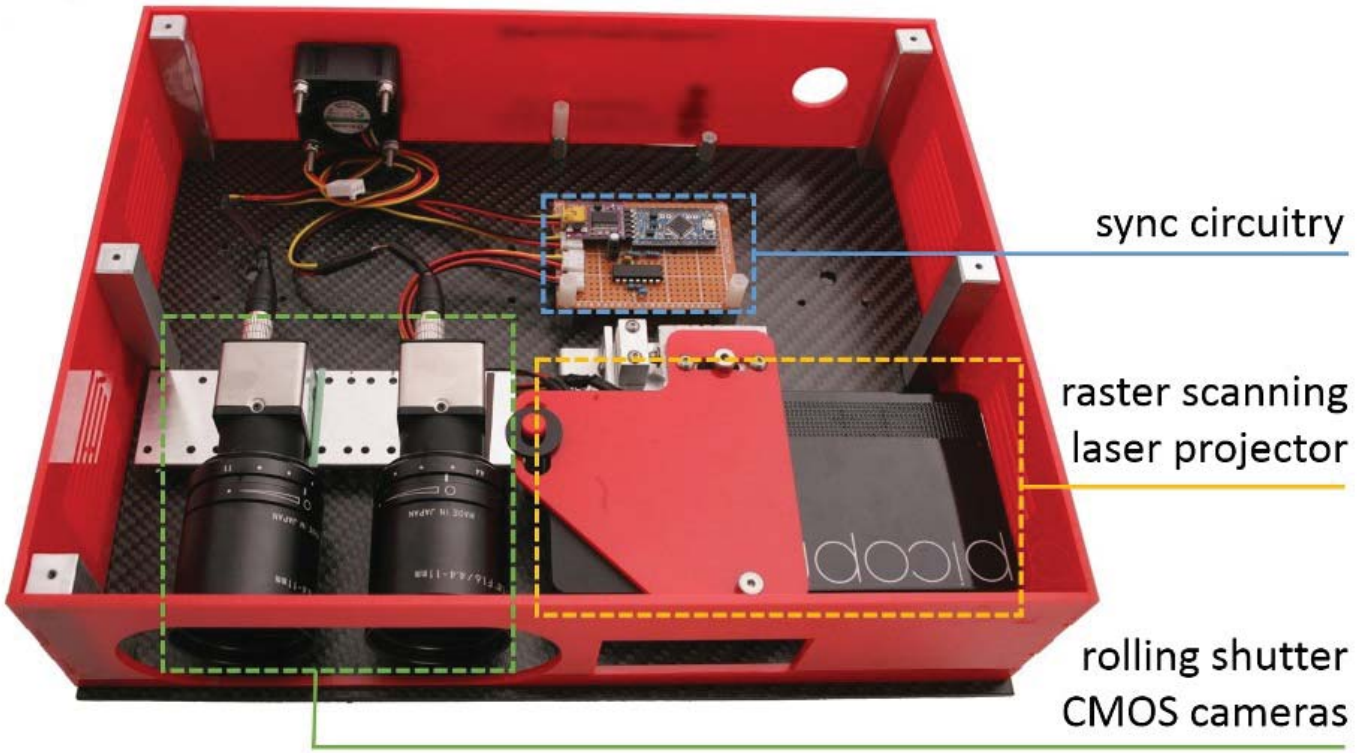
Notice anything unusual about them?

Video stream 1

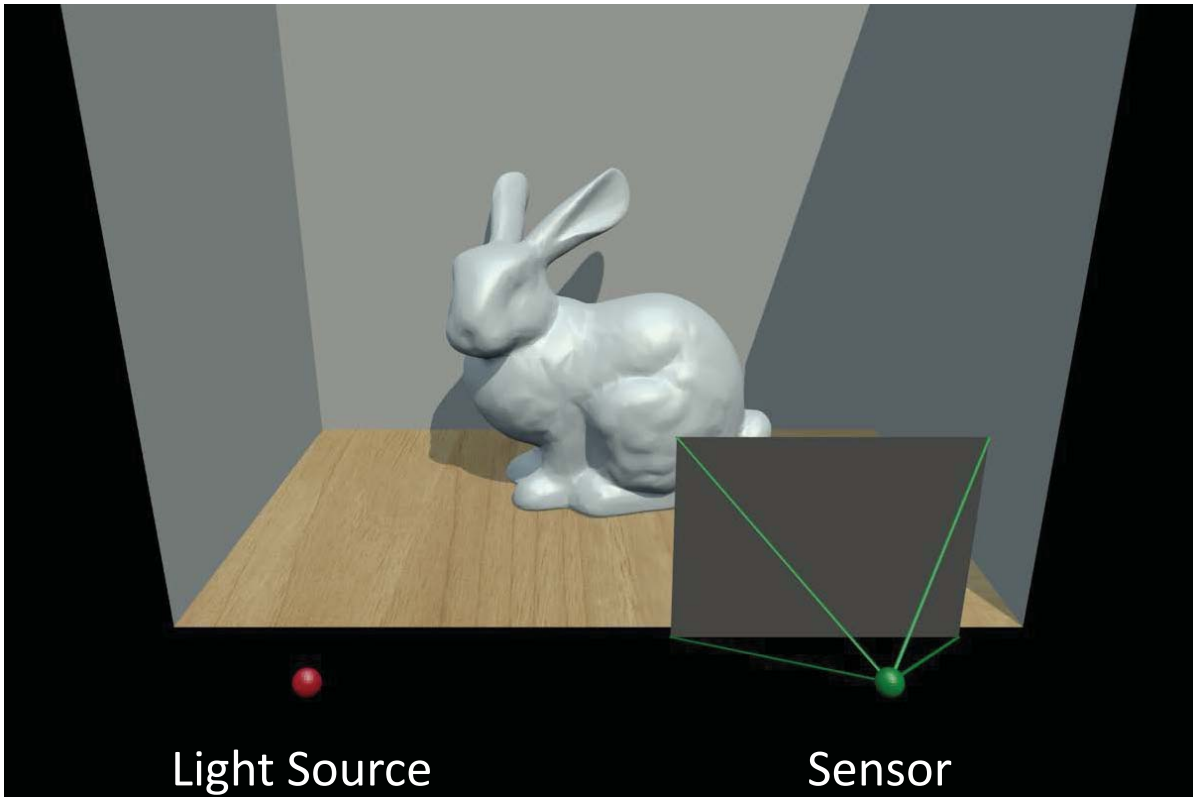
Video stream 2

Epipolar imaging camera

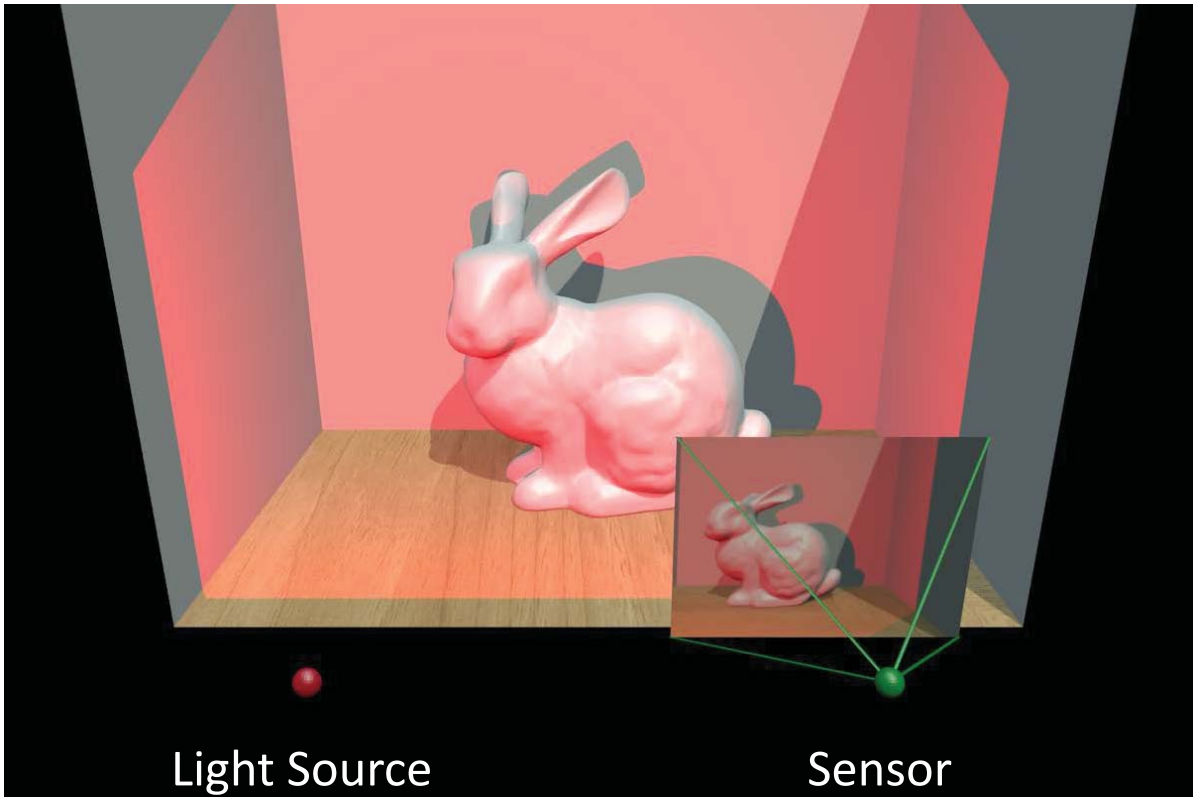
Built at CMU.



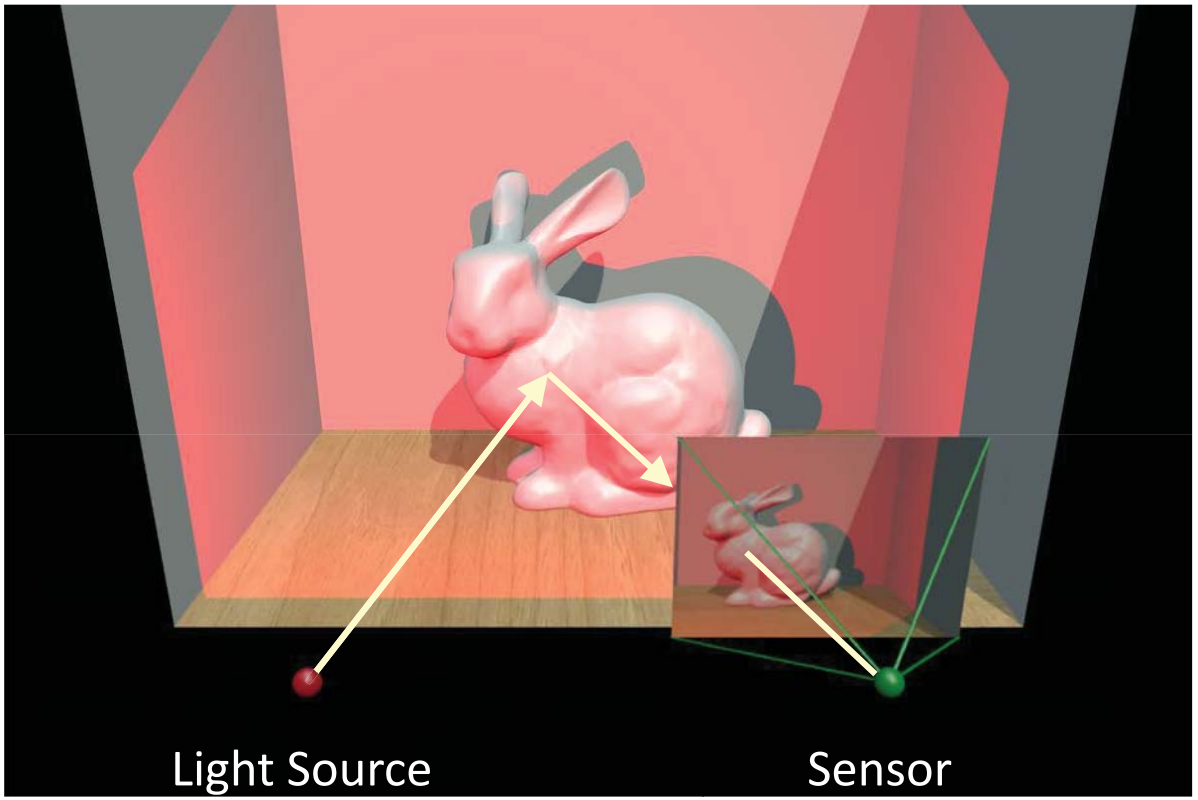
Regular Imaging



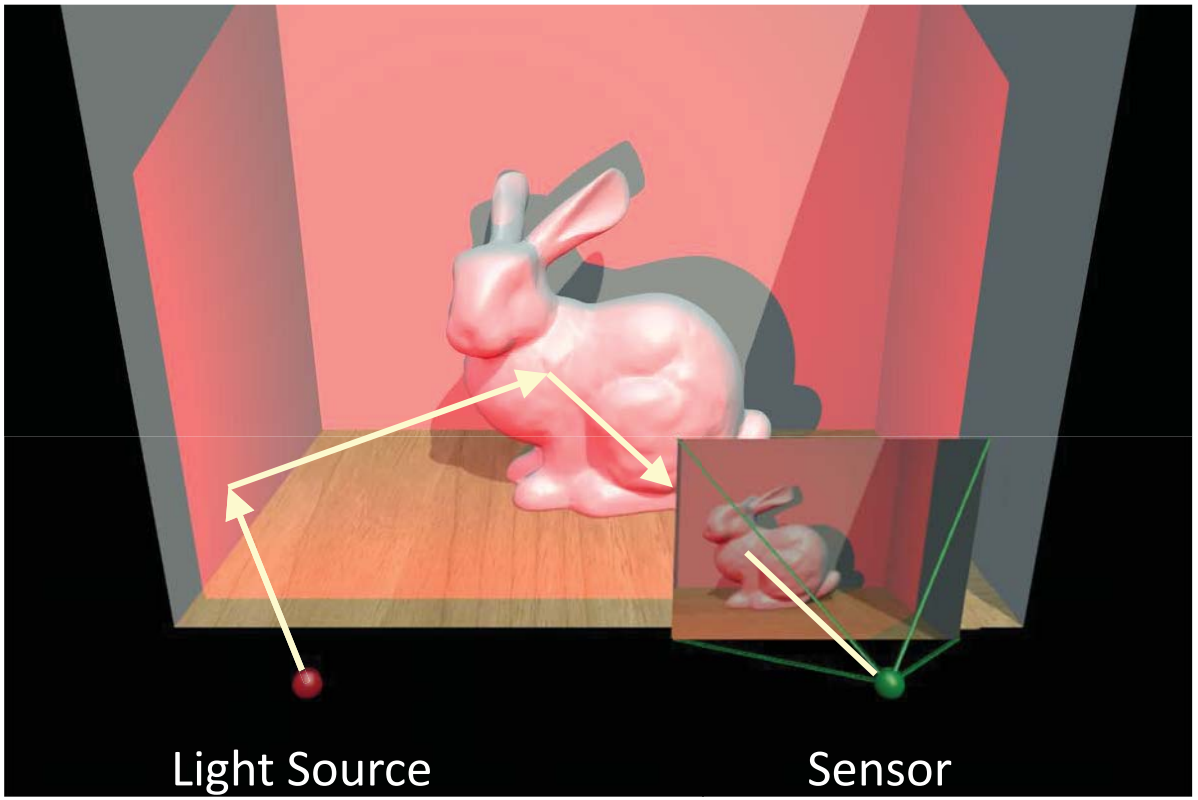
Regular Imaging



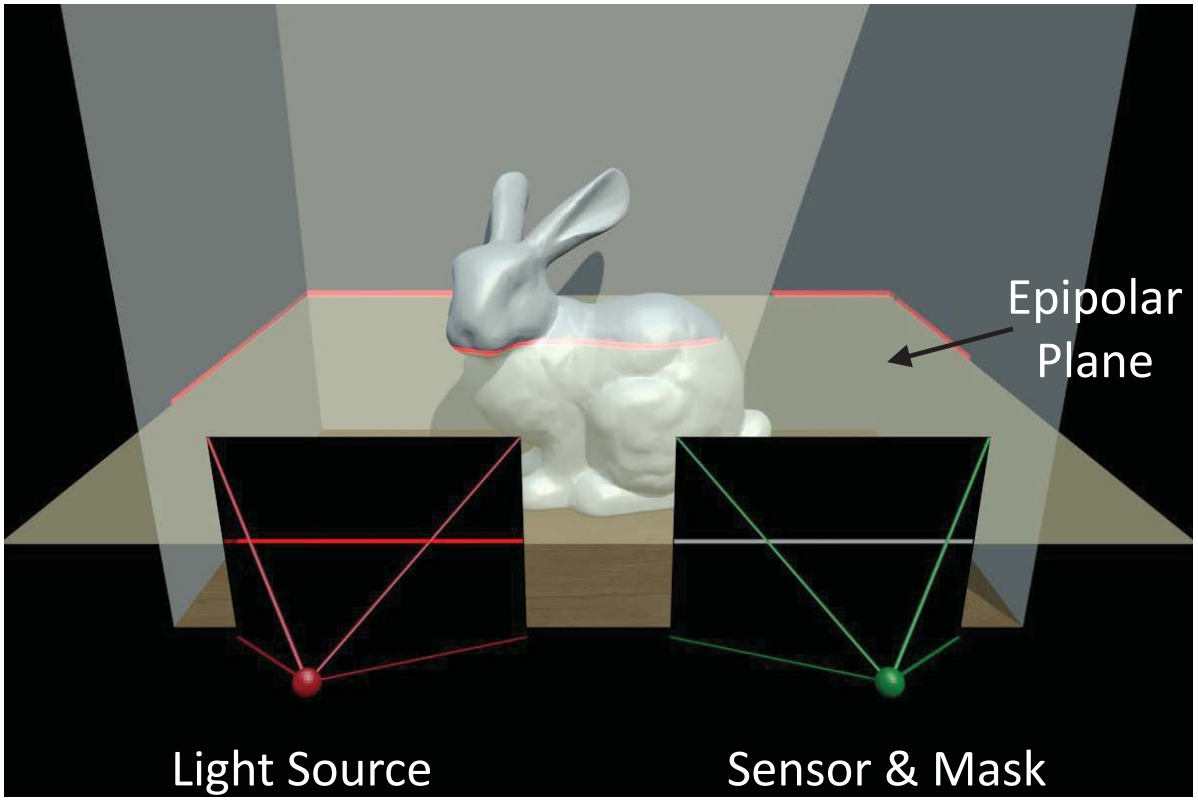
Regular Imaging



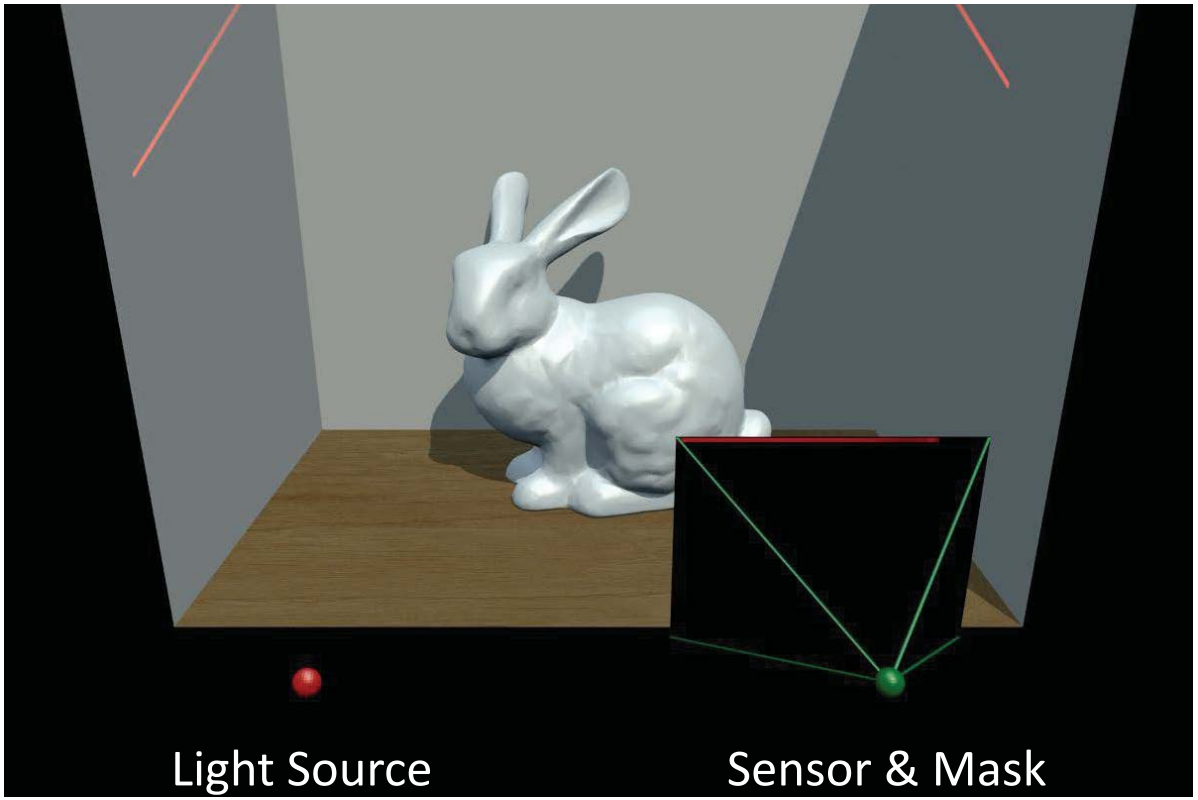
Regular Imaging



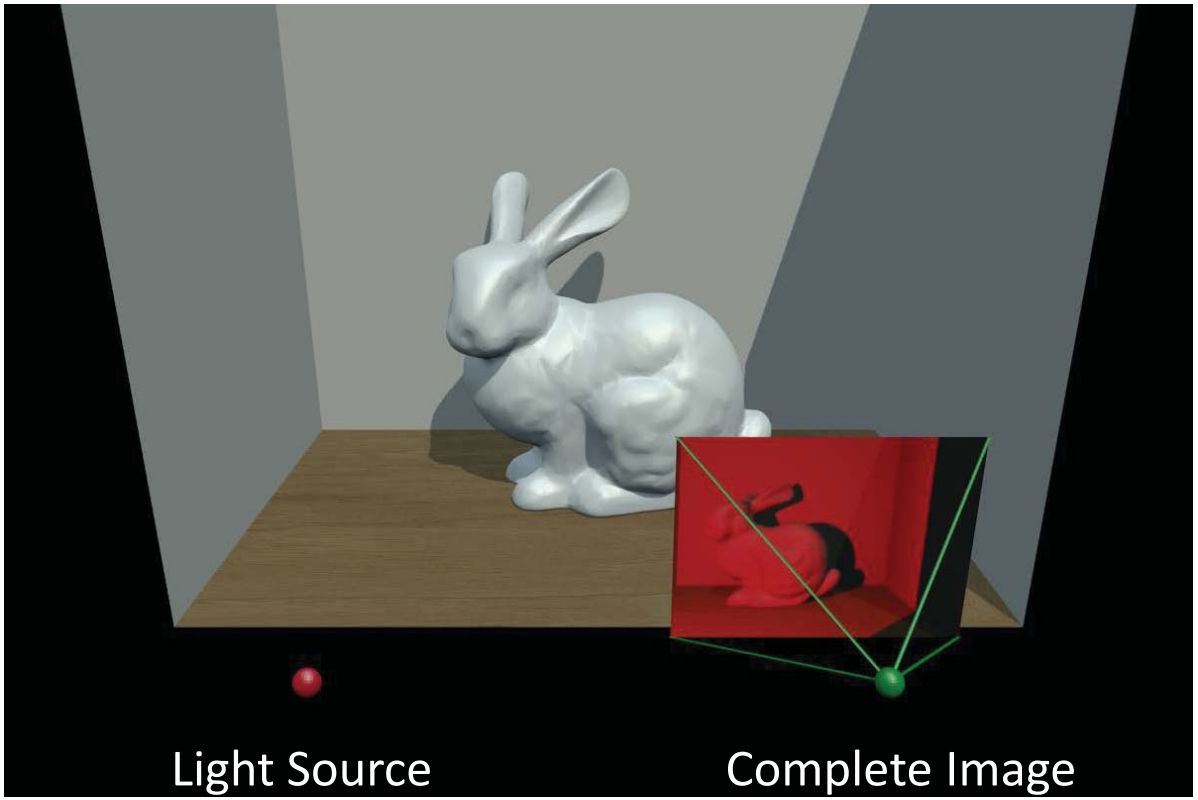
Epipolar Imaging



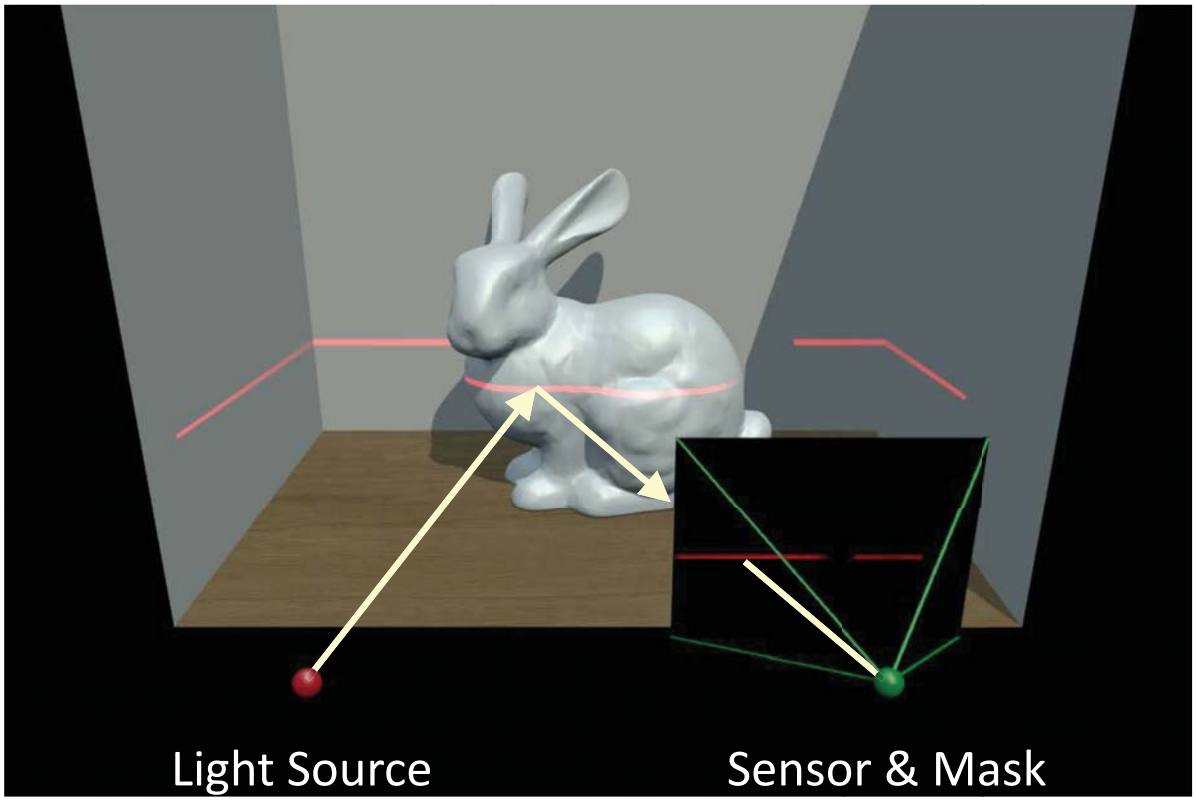
Epipolar Imaging



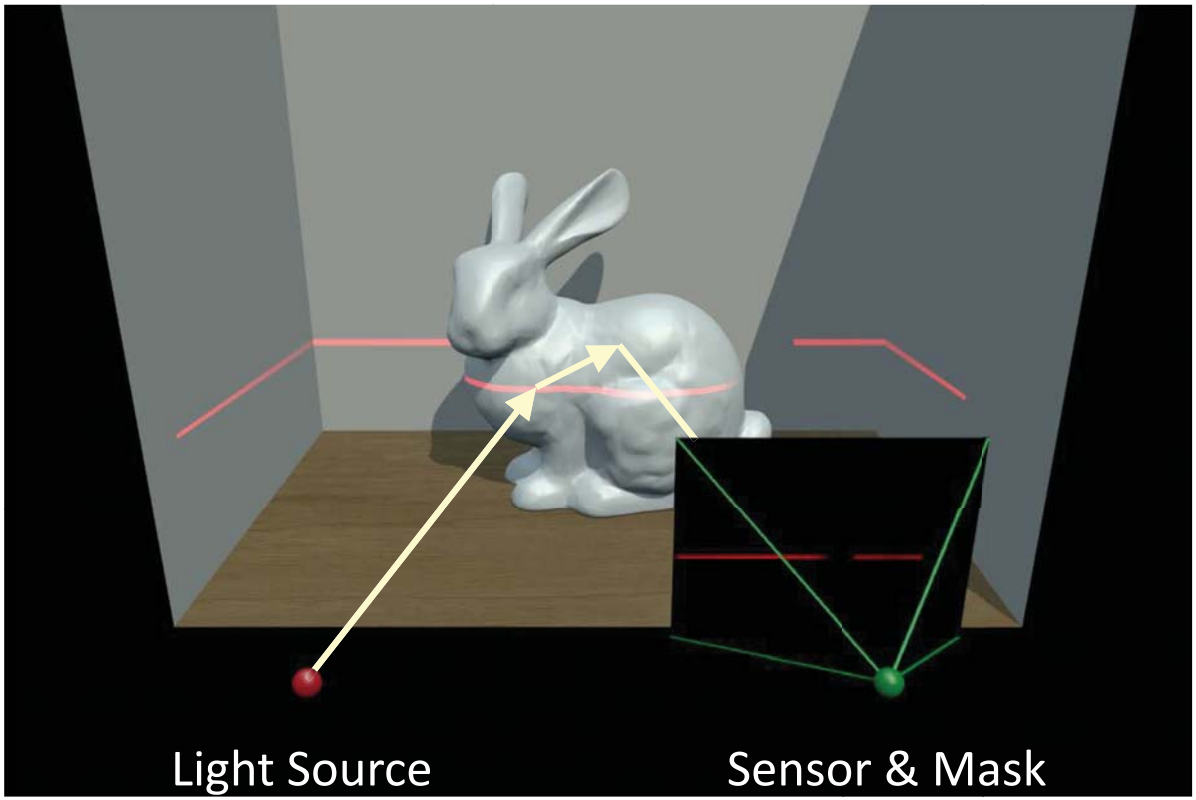
Epipolar Imaging



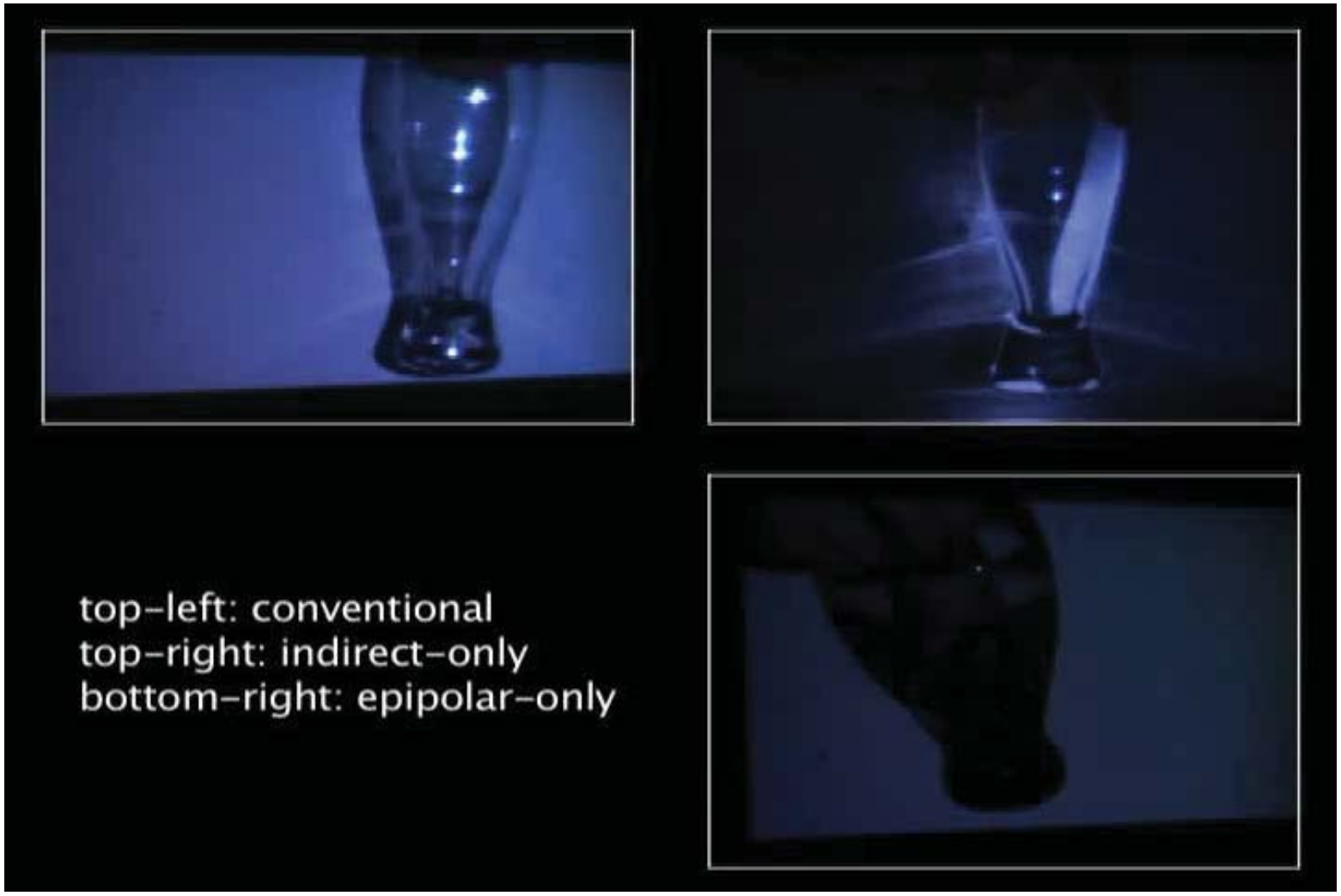
Epipolar Imaging



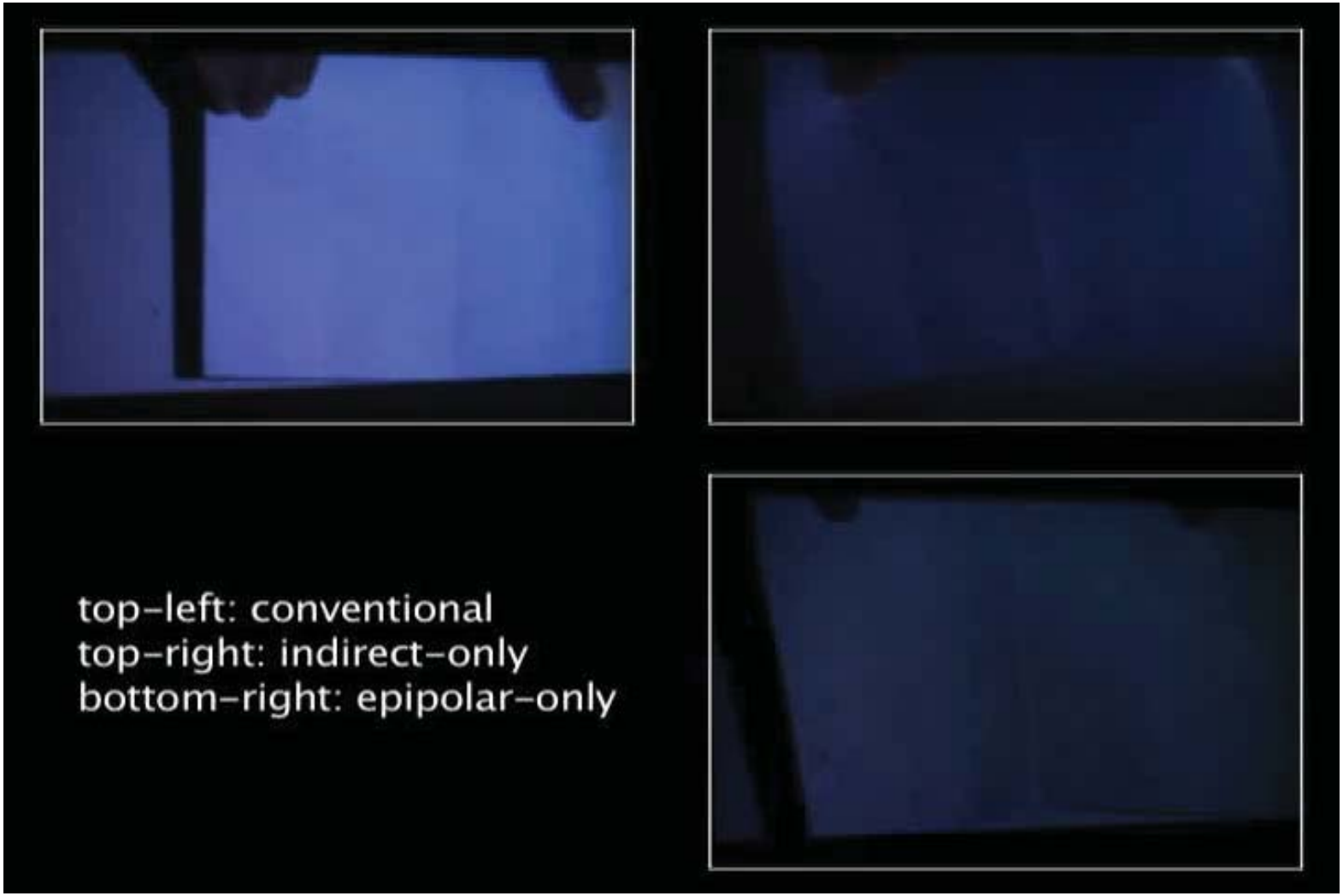
Epipolar Imaging




top-left: conventional
top-right: indirect-only
bottom-right: epipolar-only


top-left: conventional
top-right: indirect-only
bottom-right: epipolar-only


top-left: conventional
top-right: indirect-only
bottom-right: epipolar-only


top-left: conventional
top-right: indirect-only
bottom-right: epipolar-only

Computational Photography

Learn about this and other unconventional cameras
(Come to my class next year: fall 2020)

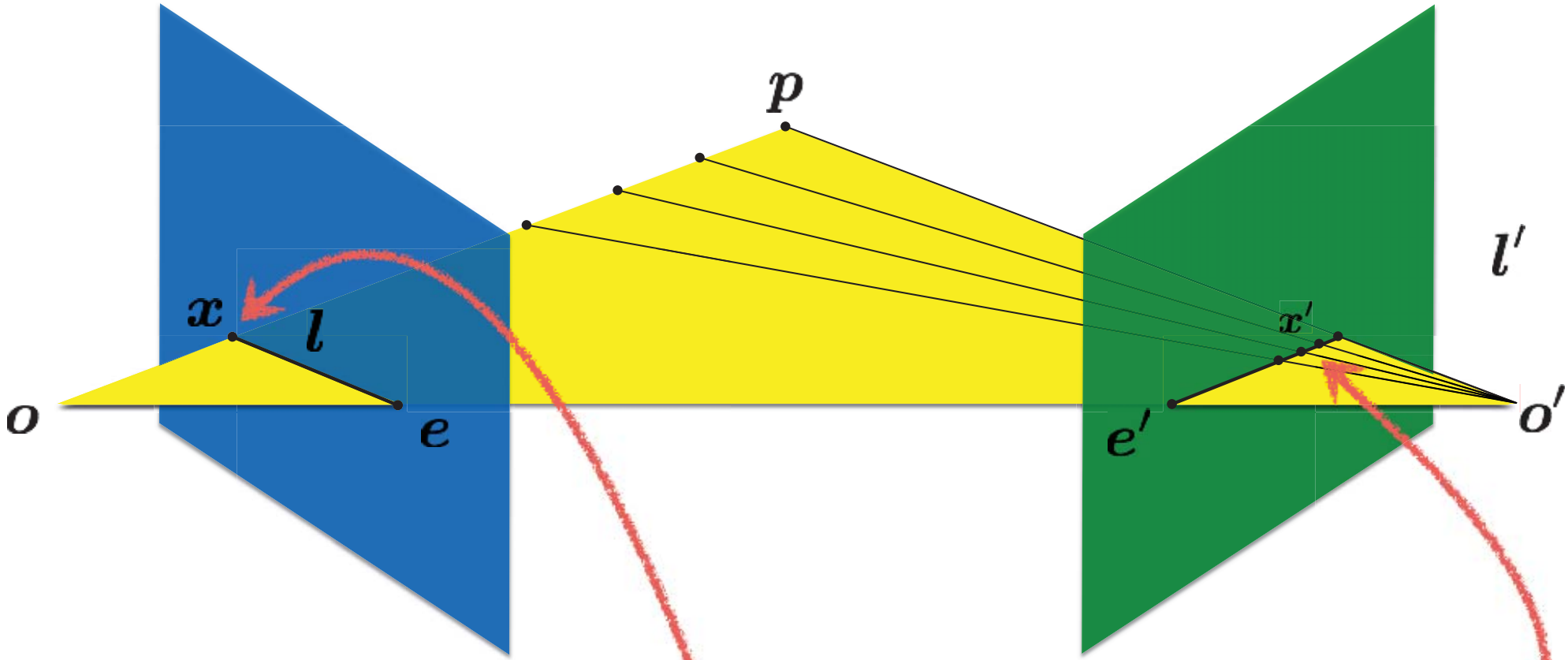
cameras that measure depth in real time

cameras that take video at the speed of light

cameras that see around corners

cameras that capture entire focal stacks

Recall:Epipolar constraint

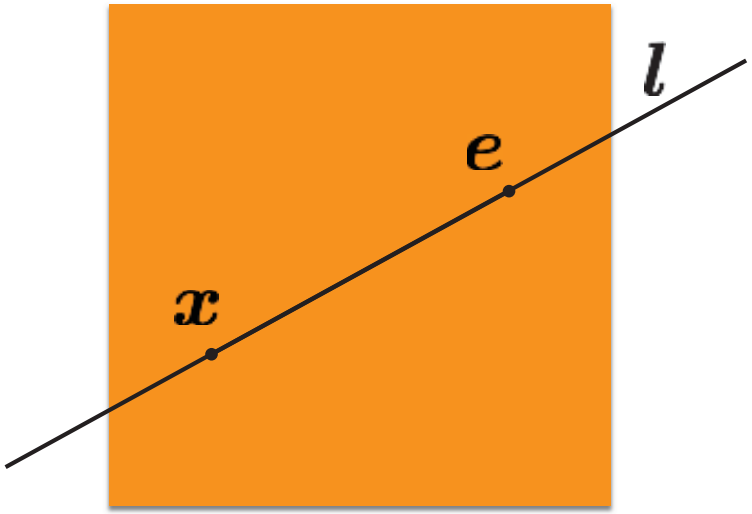


Potential matches for x lie on the epipolar line l'

Representing the …

Epipolar Line

$$ax + by + c = 0$$ in vector form $$\boldsymbol{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

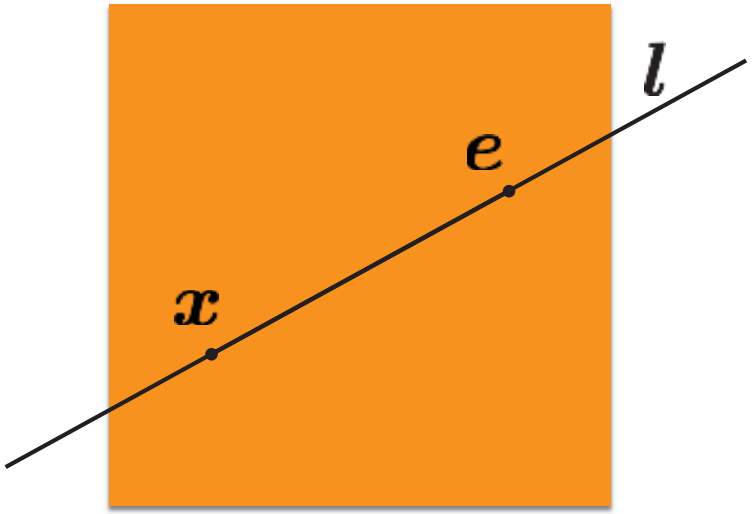


If the point $\boldsymbol{x}$ is on the epipolar line $\boldsymbol{l}$ then

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

Epipolar Line

$$ax + by + c = 0$$ in vector form $$\boldsymbol{l} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

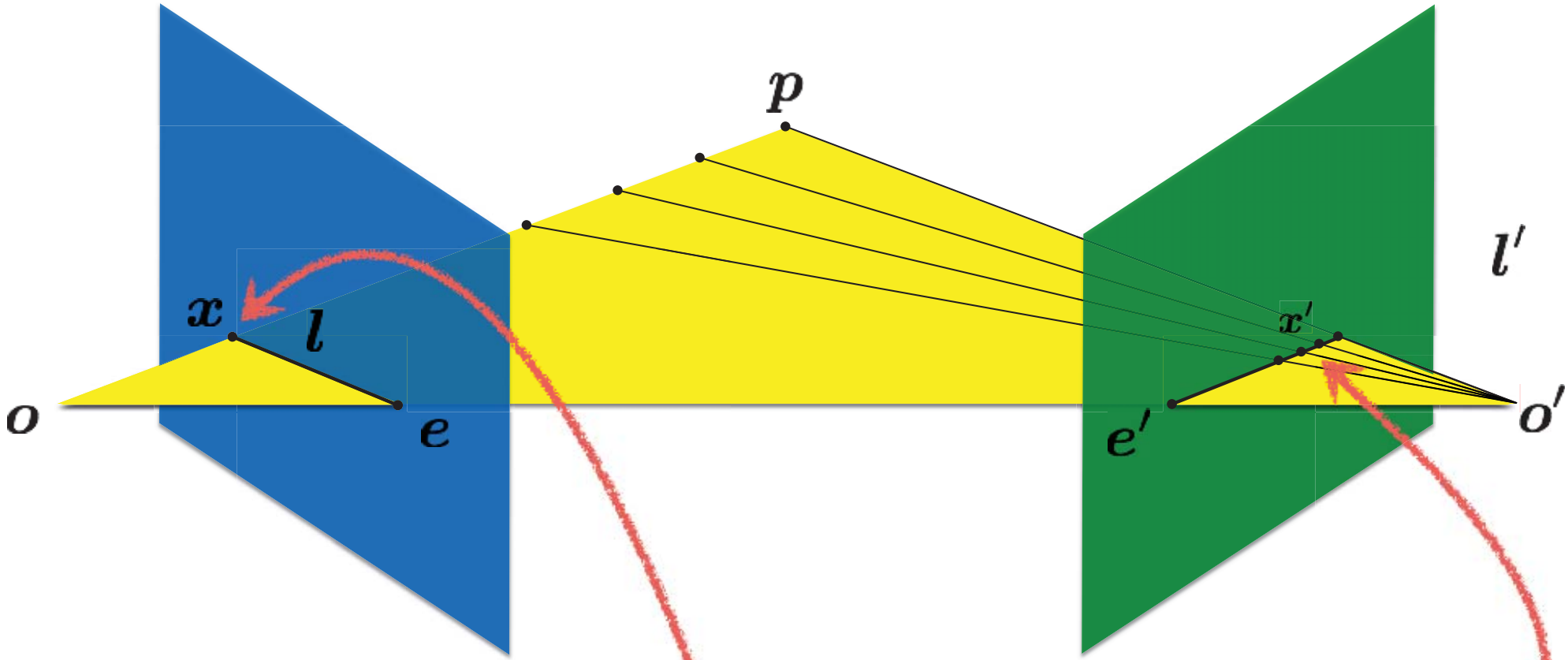


If the point $\boldsymbol{x}$ is on the epipolar line $\boldsymbol{l}$ then

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

The essential matrix

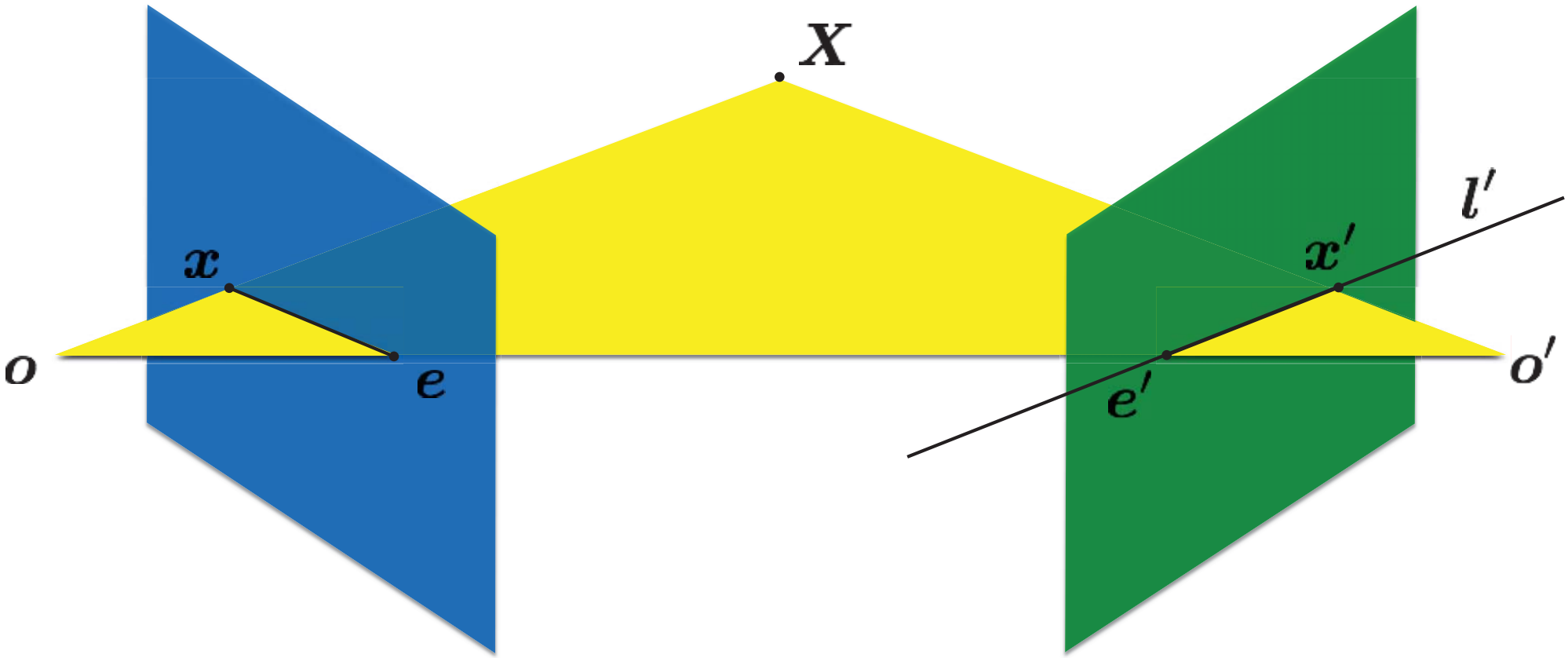
Recall:Epipolar constraint



Potential matches for x lie on the epipolar line l'

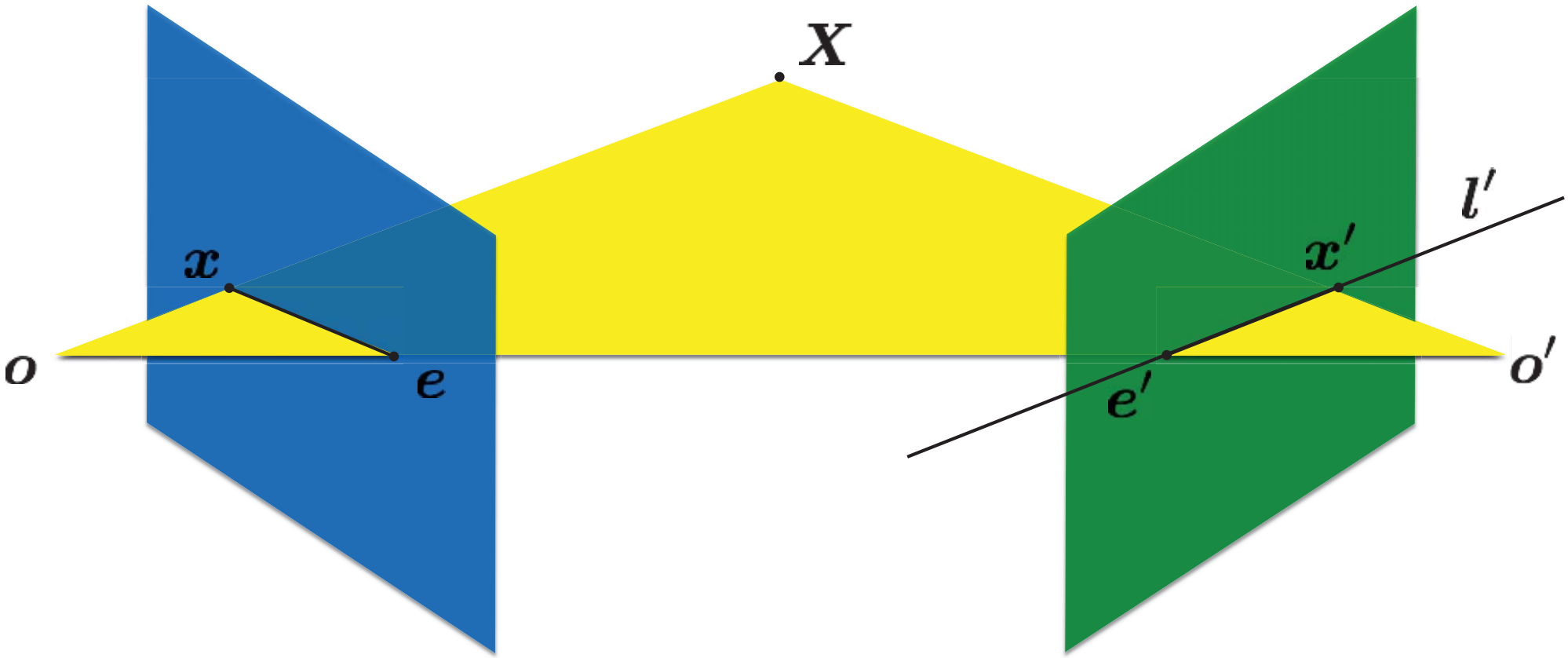
Given a point in one image,
multiplying by the **essential matrix** will tell us
the **epipolar line** in the second view.

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



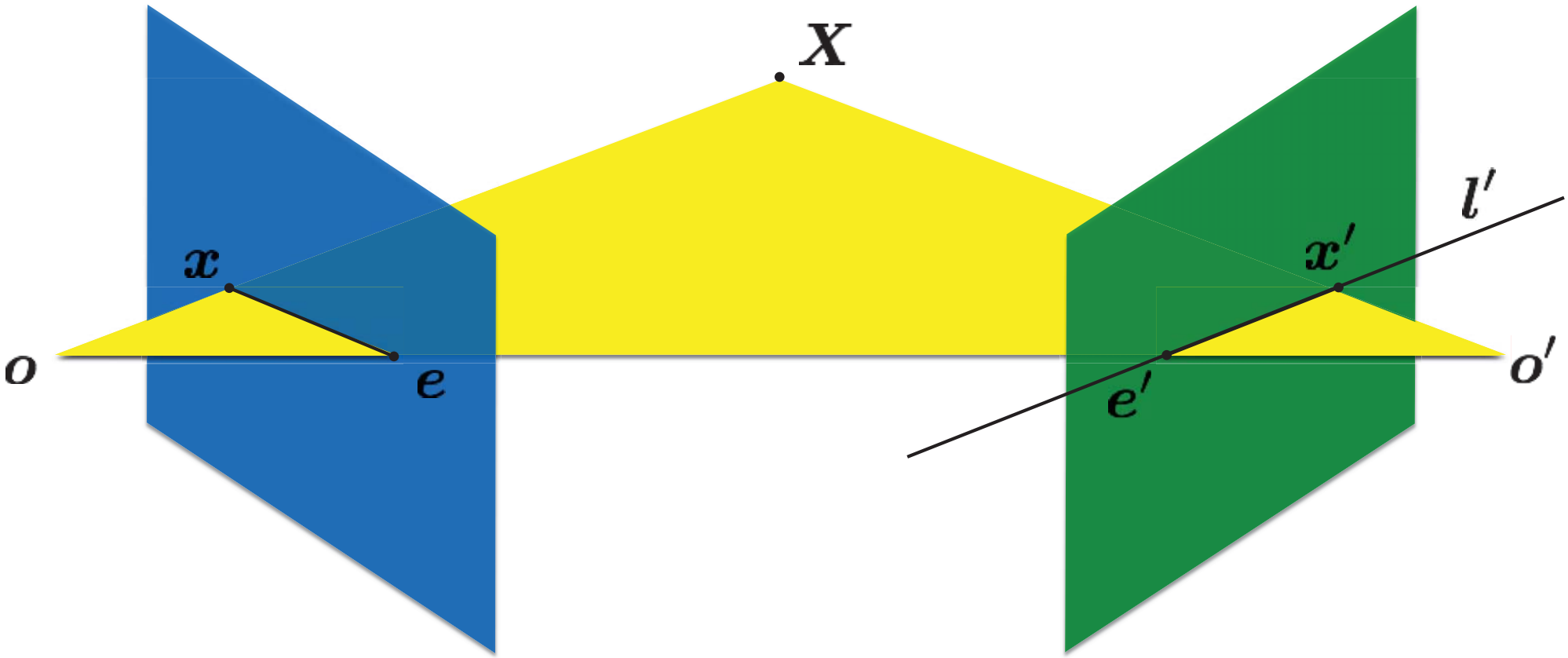
So if $\boldsymbol{x}^\top \boldsymbol{l} = 0$ and $\mathbf{E}\boldsymbol{x} = \boldsymbol{l}'$ then

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



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

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



properties of the E matrix

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

properties of the E matrix

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

Epipolar lines

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

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

properties of the E matrix

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

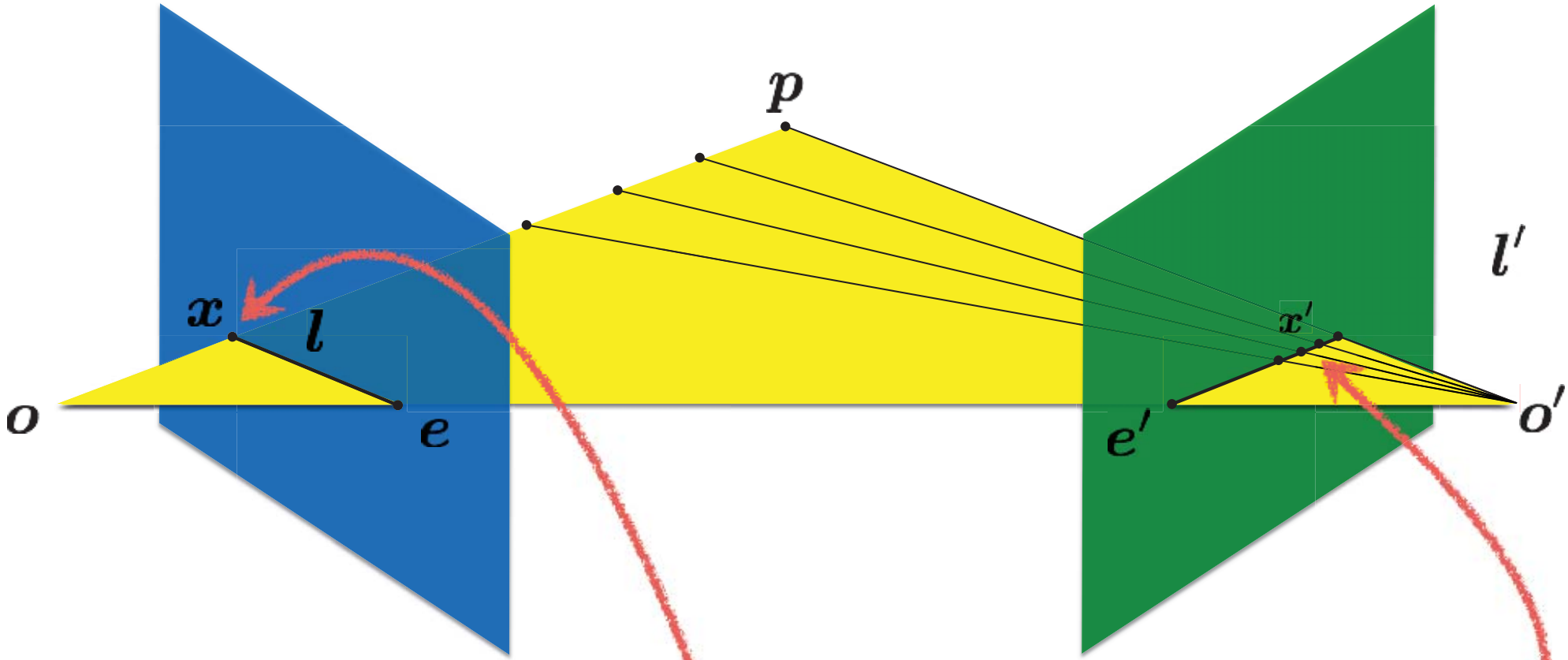
Epipolar lines $\boldsymbol{x}^{\top}\boldsymbol{l} = 0$ $\quad \boldsymbol{x'}^{\top}\boldsymbol{l'} = 0$

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

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

Hint: $e'l' = ?$

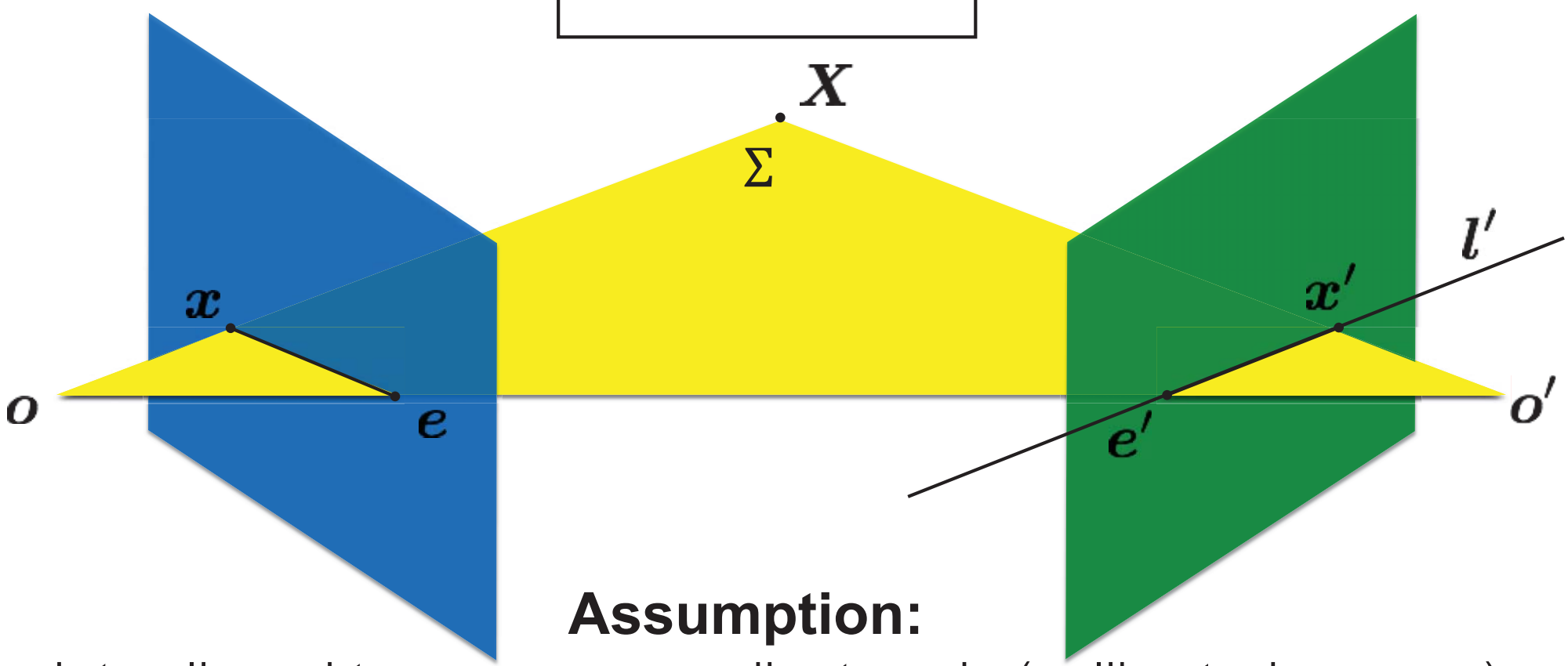
Recall:Epipolar constraint



Potential matches for x lie on the epipolar line l'

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

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



Assumption:
points aligned to camera coordinate axis (calibrated camera)
(internal matrix K was pre-applied)

Putting it all together

We can write everything into a single projection:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_\mathbf{w}$$

The camera matrix now looks like:

$$\mathbf{P} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & -\mathbf{RC} \end{bmatrix}$$

intrinsic parameters (3 x 3):
correspond to camera internals
(sensor not at f = 1 and origin shift)

extrinsic parameters (3 x 4):
correspond to camera externals
(world-to-image transformation)

More general camera matrices

Finite projective camera: sensor be skewed.

$$\mathbf{P} = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \left[\mathbf{R} \;\vdots\; -\mathbf{RC} \right]$$

How many degrees of freedom?

How do you generalize to uncalibrated cameras?

The fundamental matrix

The
Fundamental matrix
is a
generalization
of the
Essential matrix,
where the assumption of
calibrated cameras
is removed

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

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

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

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

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

camera point

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

image point

Writing out the epipolar constraint in terms of image coordinates

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

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

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

Same equation works in image coordinates!

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

it maps pixels to epipolar lines

properties of the E matrix

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

Epipolar lines

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

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

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

(points in **image** coordinates)

Breaking down the fundamental matrix

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

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

Depends on both intrinsic and extrinsic parameters

Breaking down the fundamental matrix

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

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

Depends on both intrinsic and extrinsic parameters

How would you solve for F?

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

The 8-point algorithm

Assume you have *M* matched *image* points

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

Each correspondence should satisfy

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

*How would you solve for the 3 x 3 **F** matrix?*

Assume you have *M* matched *image* points

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

Each correspondence should satisfy

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

*How would you solve for the 3 x 3 **F** matrix?*

S V D

Assume you have *M* matched *image* points

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

Each correspondence should satisfy

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

How would you solve for the 3 x 3 ***F*** *matrix?*

Set up a homogeneous linear system with 9 unknowns

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

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

How many equation do you get from one correspondence?

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

Set up a homogeneous linear system with 9 unknowns

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

How many equations do you need?

Each point pair (according to epipolar constraint) contributes only one scalar equation

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

Note: This is different from the Homography estimation where each point pair contributes 2 equations.

We need at least 8 points

Hence, the 8 point algorithm!

How do you solve a homogeneous linear system?

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

Total Least Squares

minimize $\|\mathbf{A}\boldsymbol{x}\|^2$

subject to $\|\boldsymbol{x}\|^2 = 1$

S V D !

Eight-Point Algorithm

0. (Normalize points)
1. Construct the M x 9 matrix **A**
2. Find the SVD of **A**
3. Entries of **F** are the elements of column of **V** corresponding to the least singular value
4. (Enforce rank 2 constraint on F)
5. (Un-normalize F)

Eight-Point Algorithm

0. (Normalize points)
1. Construct the M x 9 matrix **A**
2. Find the SVD of **A**
3. Entries of **F** are the elements of column of **V** corresponding to the least singular value
4. (Enforce rank 2 constraint on F)
5. (Un-normalize F)

How do we do this?

Eight-Point Algorithm

0. (Normalize points)
1. Construct the M x 9 matrix **A**
2. Find the SVD of **A**
3. Entries of **F** are the elements of column of **V** corresponding to the least singular value
4. (Enforce rank 2 constraint on F)
5. (Un-normalize F)

How do we do this?

S V D !

Enforcing rank constraints

Problem: Given a matrix **F**, find the matrix **F'** of rank k that is closest to **F**,

$$\min_{\substack{F' \\ \text{rank}(F')=k}} \|F - F'\|^2$$

Solution: Compute the singular value decomposition of **F**,

$$F = U\Sigma V^T$$

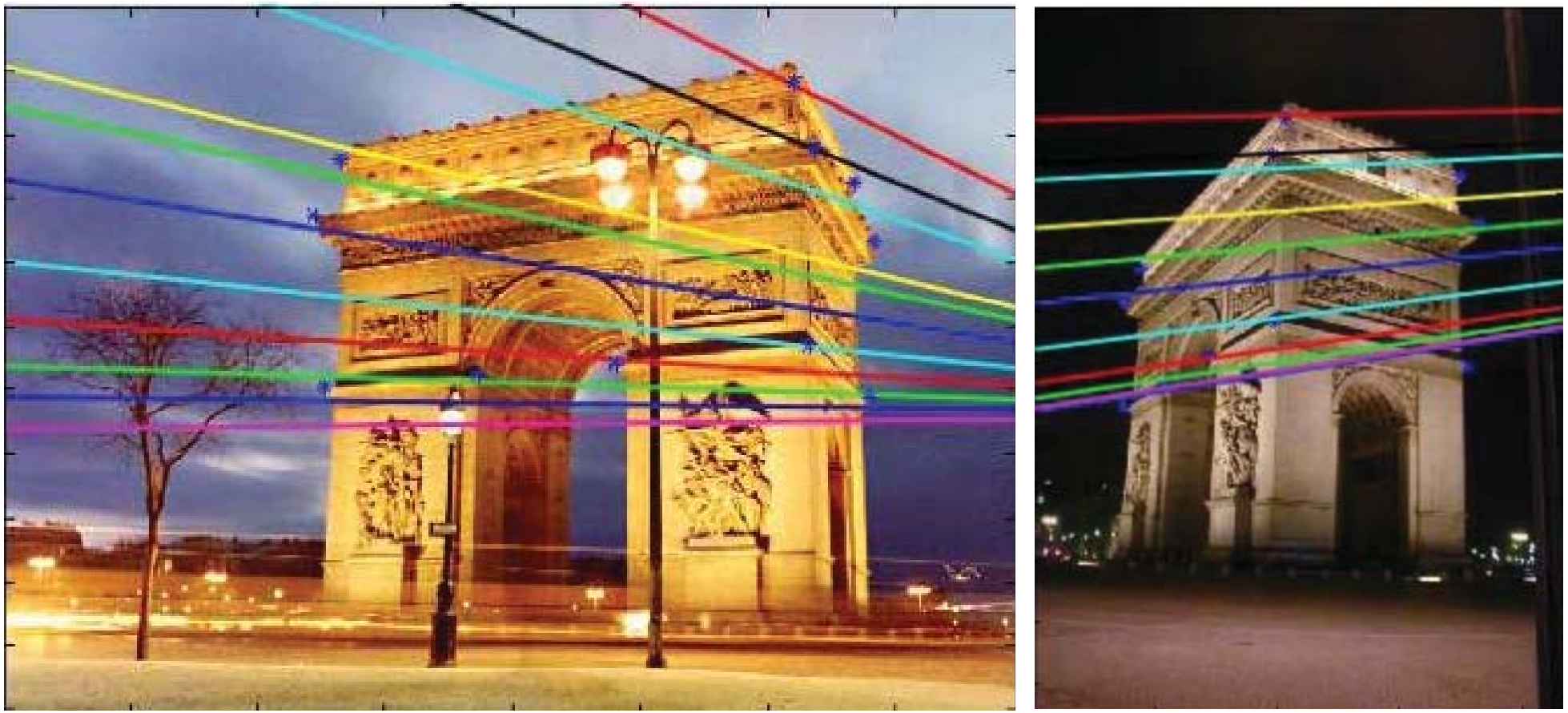
Form a matrix **Σ'** by replacing all but the k largest singular values in **Σ** with 0.

Then the problem solution is the matrix **F'** formed as,

$$F' = U\Sigma' V^T$$

Example

epipolar lines



$$\mathbf{F} = \begin{bmatrix} -0.00310695 & -0.0025646 & 2.96584 \\ -0.028094 & -0.00771621 & 56.3813 \\ 13.1905 & -29.2007 & -9999.79 \end{bmatrix}$$

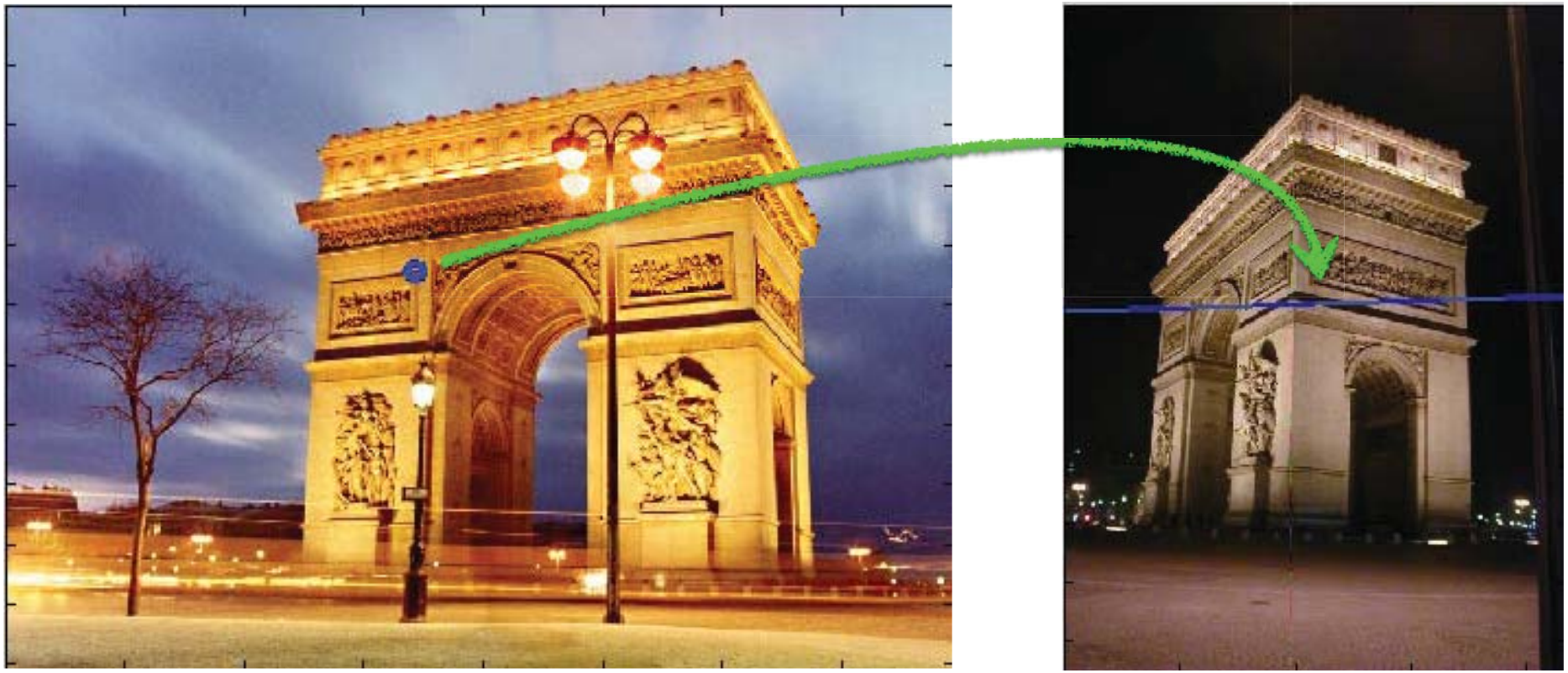
$$\boldsymbol{x} = \begin{bmatrix} 343.53 \\ 221.70 \\ 1.0 \end{bmatrix}$$

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

$$= \begin{bmatrix} 0.0295 \\ 0.9996 \\ -265.1531 \end{bmatrix}$$

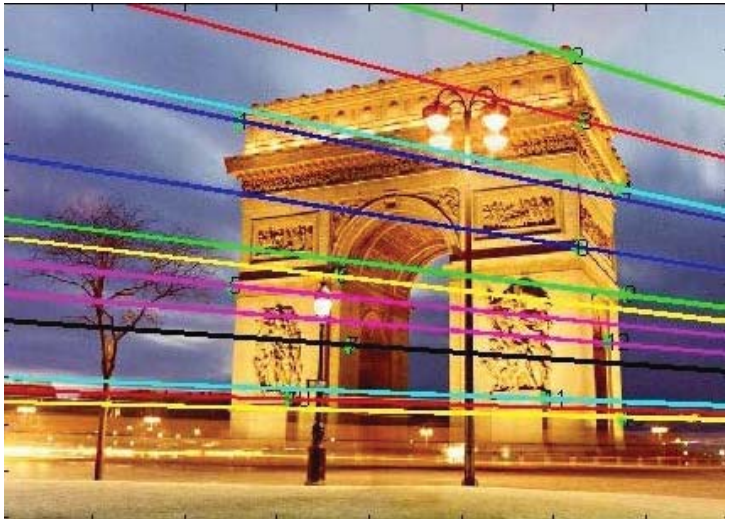
$$l' = \mathbf{F}x$$

$$= \begin{bmatrix} 0.0295 \\ 0.9996 \\ -265.1531 \end{bmatrix}$$



Where is the epipole?

How would you compute it?

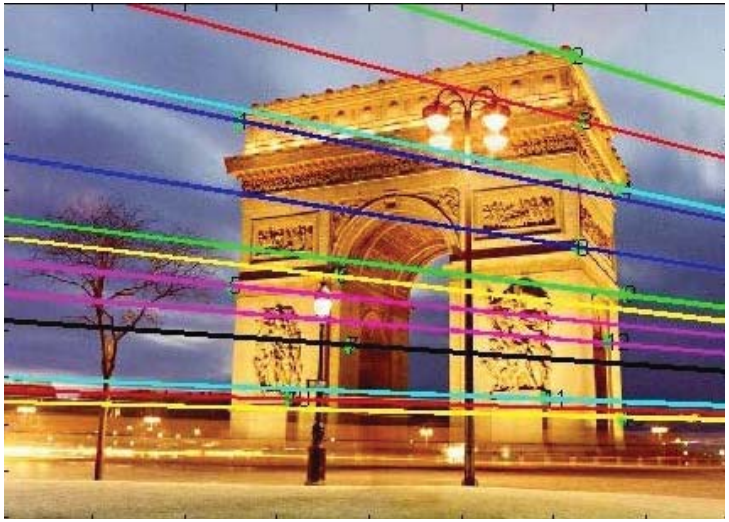


$$\mathbf{F}\boldsymbol{e} = \mathbf{0}$$

The epipole is in the right null space of **F**

How would you solve for the epipole?

(hint: this is a homogeneous linear system)



$$\mathbf{F}\boldsymbol{e} = \mathbf{0}$$

The epipole is in the right null space of **F**

How would you solve for the epipole?

(hint: this is a homogeneous linear system)

S V D !

```
>> [u,d] = eigs(F' * F)
```

eigenvectors

```
u =
    -0.0013    0.2586   -0.9660
     0.0029   -0.9660   -0.2586
     1.0000    0.0032   -0.0005
```

eigenvalue

```
d = 1.0e8*
   -1.0000         0         0
         0   -0.0000         0
         0         0   -0.0000
```

```
>> [u,d] = eigs(F' * F)
```

eigenvectors

```
u =
    -0.0013    0.2586   -0.9660
     0.0029   -0.9660   -0.2586
     1.0000    0.0032   -0.0005
```

eigenvalue

```
d = 1.0e8*
   -1.0000         0         0
         0   -0.0000         0
         0         0   -0.0000
```

```
>> [u,d] = eigs(F' * F)
```

eigenvectors

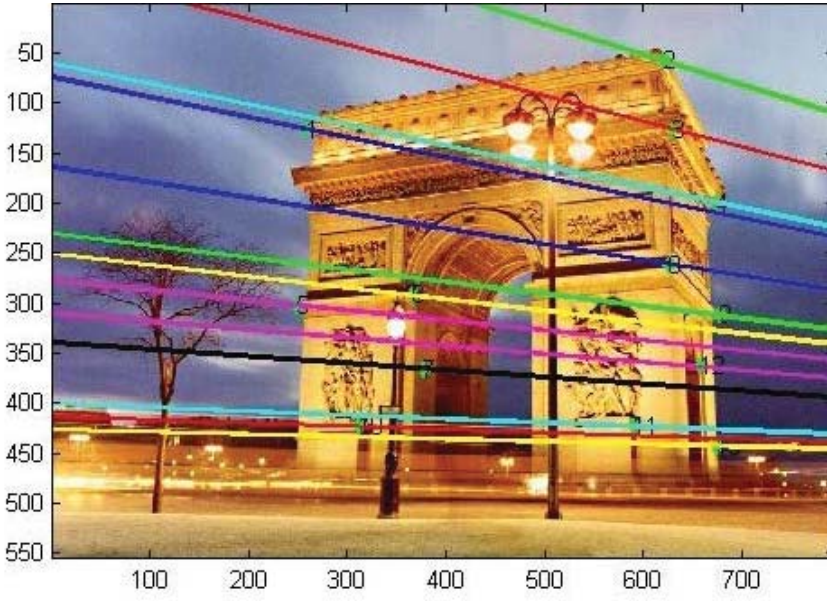
```
u =
    -0.0013    0.2586   -0.9660
     0.0029   -0.9660   -0.2586
     1.0000    0.0032   -0.0005
```

eigenvalue

```
d = 1.0e8*
   -1.0000         0         0
         0   -0.0000         0
         0         0   -0.0000
```

Eigenvector associated with smallest eigenvalue

```
>> uu = u(:,3)
(  -0.9660    -0.2586    -0.0005)
```




```
>> [u,d] = eigs(F' * F)
```

eigenvectors

```
u =
    -0.0013    0.2586   -0.9660
     0.0029   -0.9660   -0.2586
     1.0000    0.0032   -0.0005
```

eigenvalue

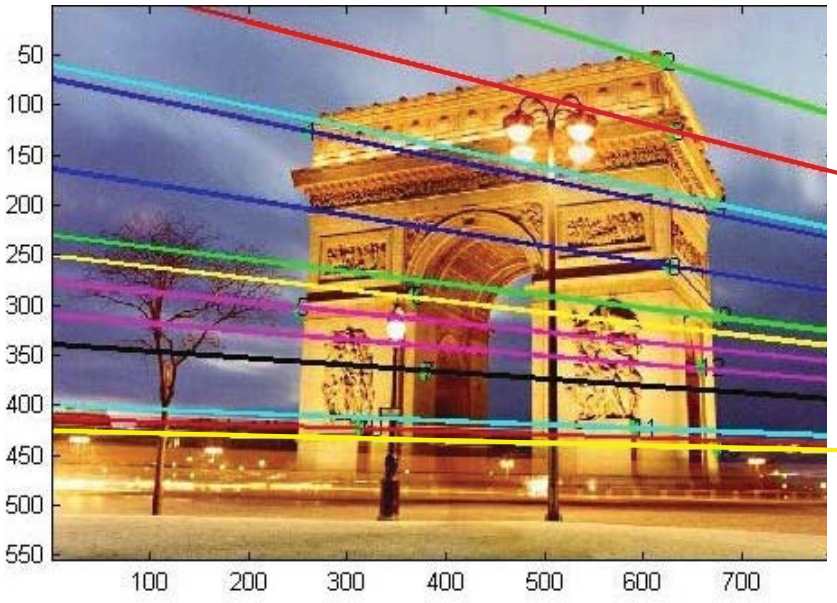
```
d = 1.0e8*
   -1.0000         0         0
         0   -0.0000         0
         0         0   -0.0000
```

Eigenvector associated with smallest eigenvalue

```
>> uu = u(:,3)
(  -0.9660    -0.2586    -0.0005)
```

Epipole projected to image coordinates

```
>> uu / uu(3)
(1861.02     498.21    1.0)
```



epipole

Epipole projected to image coordinates

```
>> uu / uu(3)
(1861.02      498.21      1.0)
```

A note on normalization

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

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

Problem with 8-point algorithm

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

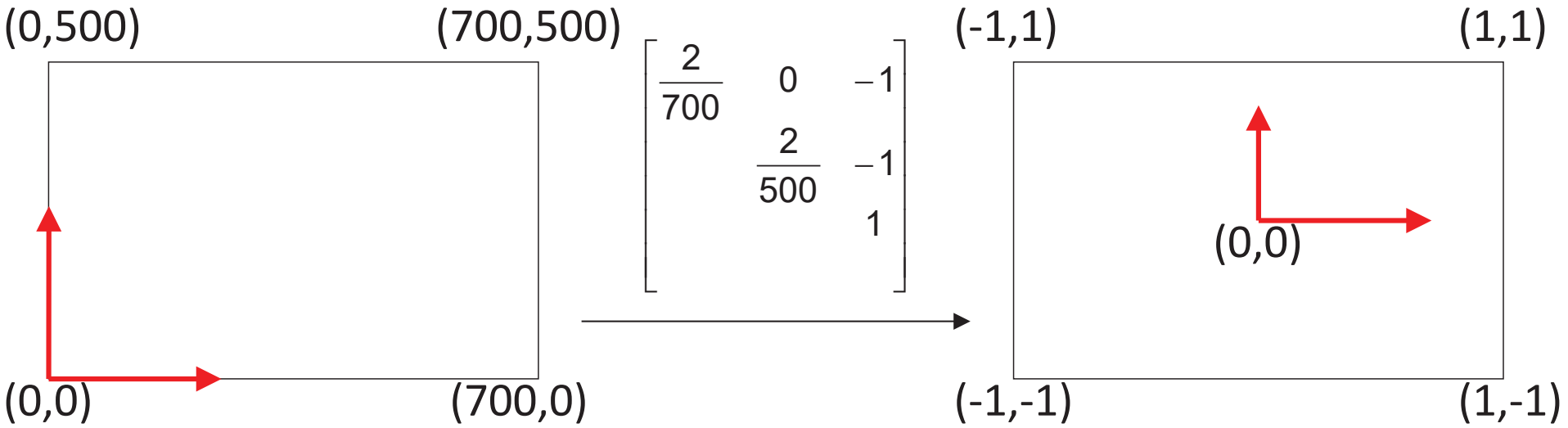
~10000 ~10000 ~100 ~10000 ~10000 ~100 ~100 ~100 1

Orders of magnitude difference
between column of data matrix
→ least-squares yields poor results

Normalized 8-point algorithm

normalized least squares yields good results

Transform image to ~[-1,1]x[-1,1]



Normalized 8-point algorithm

1. Transform input by $\hat{\mathbf{x}}_i = \mathbf{T}\mathbf{x}_i$, $\hat{\mathbf{x}}'_i = \mathbf{T}\mathbf{x}'_i$
2. Call 8-point on $\hat{\mathbf{x}}_i, \hat{\mathbf{x}}'_i$ to obtain $\hat{\mathbf{F}}$
3. $\mathbf{F} = \mathbf{T}'^{\mathrm{T}}\hat{\mathbf{F}}\mathbf{T}$

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

$$\hat{\mathbf{x}}'^{\mathrm{T}}\mathbf{T}'^{-\mathrm{T}}\,\mathbf{F}\,\mathbf{T}^{-1}\hat{\mathbf{x}} = 0$$

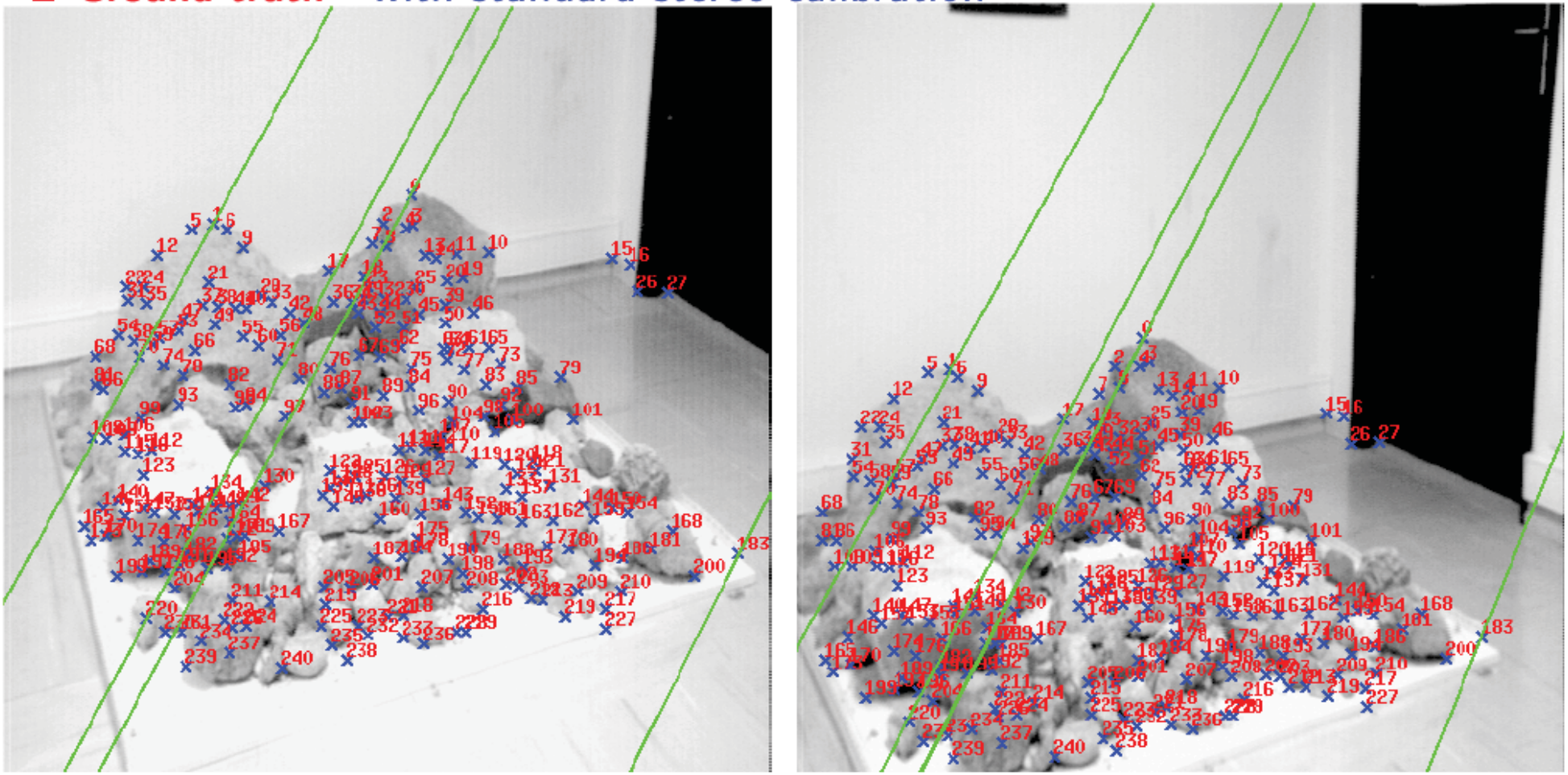
$$\hat{\mathbf{F}}$$

Normalized 8-point algorithm

```
[x1, T1] = normalise2dpts(x1);
[x2, T2] = normalise2dpts(x2);

A = [x2(1,:)'.*x1(1,:)'   x2(1,:)'.*x1(2,:)'  x2(1,:)' ...
     x2(2,:)'.*x1(1,:)'   x2(2,:)'.*x1(2,:)'  x2(2,:)' ...
     x1(1,:)'             x1(2,:)'            ones(npts,1) ];

[U,D,V] = svd(A);

F = reshape(V(:,9),3,3)';

[U,D,V] = svd(F);
F = U*diag([D(1,1) D(2,2) 0])*V';


% Denormalise
F = T2'*F*T1;
```

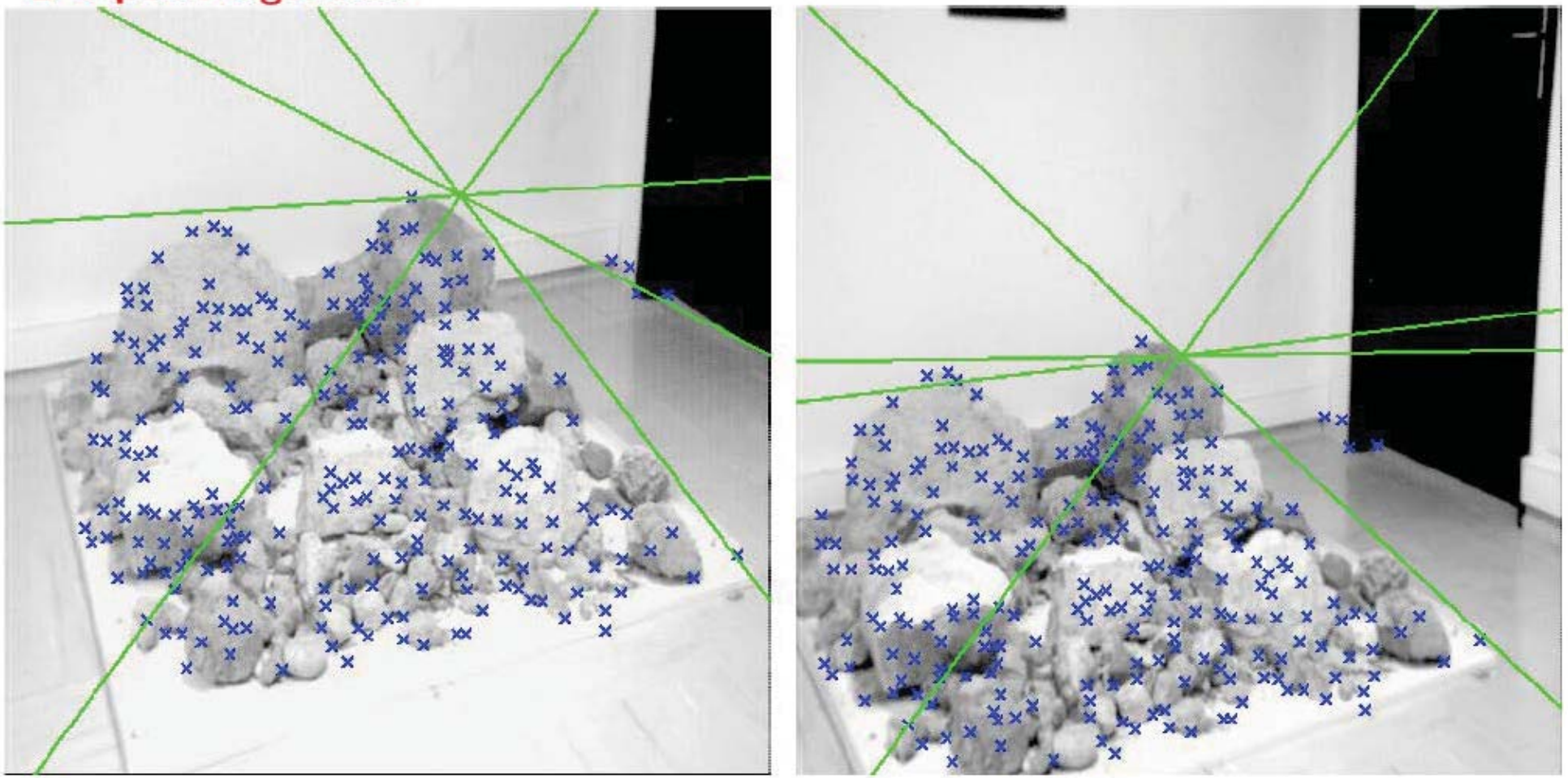
Results (ground truth)

- **Ground truth** with standard stereo calibration



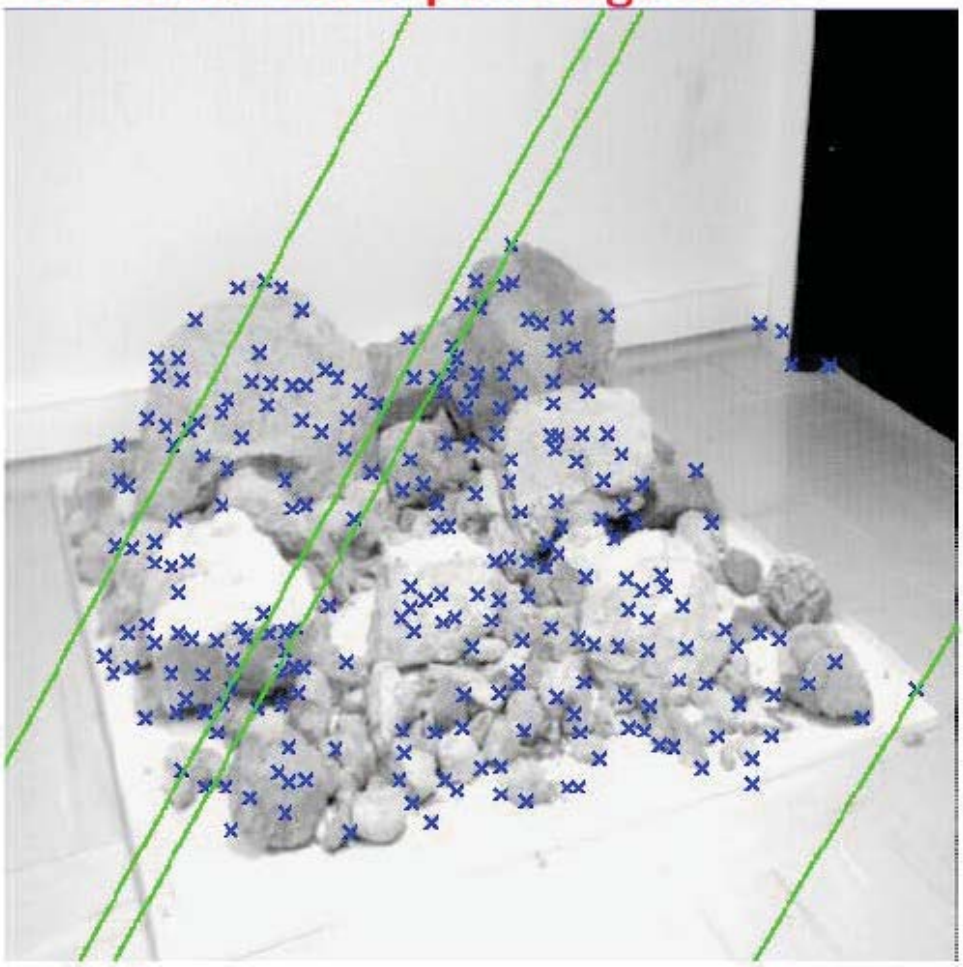
Results (8-point algorithm)

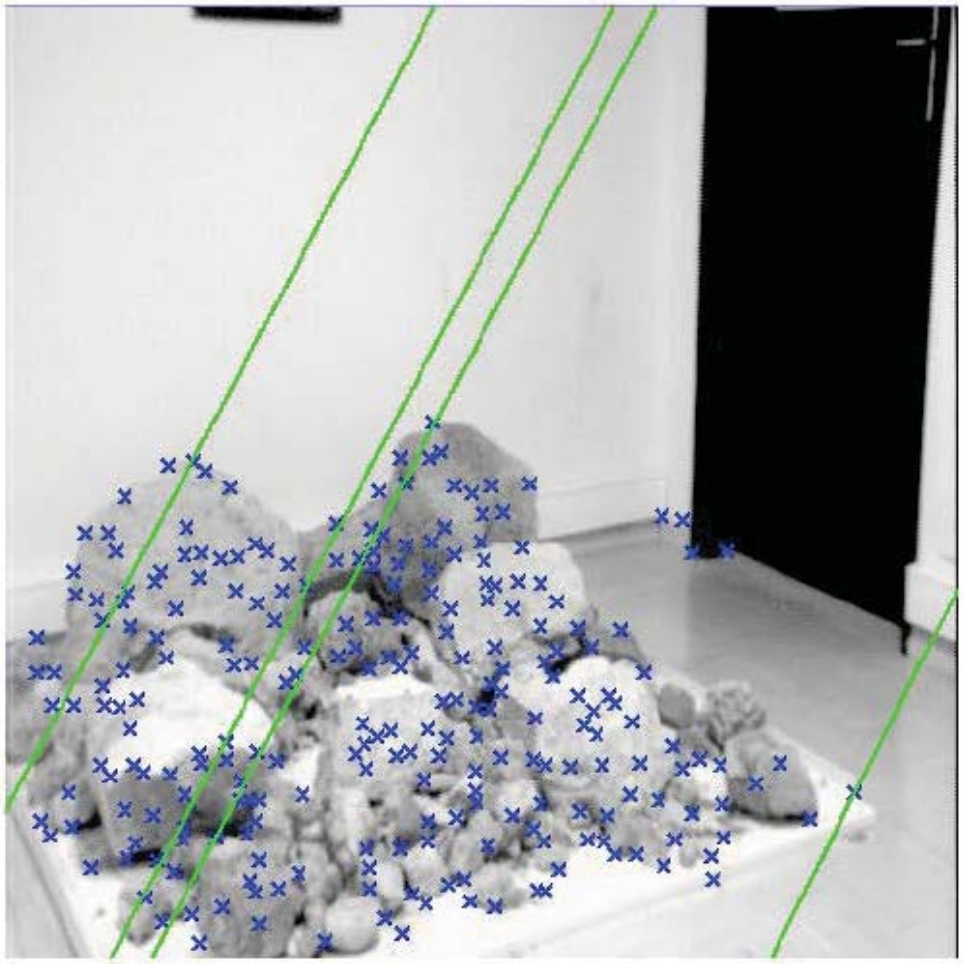
■ 8-point algorithm



Results (normalized 8-point algorithm)

- Normalized 8-point algorithm





Two-view structure from motion

	Structure (scene geometry)	Motion (camera geometry)	Measurements
Pose Estimation	known	estimate	3D to 2D correspondences
Triangulation	estimate	known	2D to 2D coorespondences
Reconstruction	estimate	estimate	2D to 2D coorespondences

Structure from motion

Драконъ, видимый подъ различными углами зрѣнія
По гравюрѣ на мѣди изъ „Oculus artificialis teledioptricus“ Цана. 1702 года.

Camera calibration & triangulation

- Suppose we know 3D points
  - And have matches between these points and an image
  - How can we compute the camera parameters?

- Suppose we have know camera parameters, each of which observes a point
  - How can we compute the 3D location of that point?

Structure from motion

- SfM solves both of these problems *at once*
- A kind of chicken-and-egg problem
  - (but solvable)

Reconstruction

(2 view structure from motion)

Given a set of matched points

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

Estimate the camera matrices

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

Estimate the 3D point

$$\mathbf{X}$$

Reconstruction

(2 view structure from motion)

Given a set of matched points

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

Estimate the camera matrices

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

'motion'

(of the cameras)

Estimate the 3D point

$$\mathbf{X}$$

'structure'

Two-view SfM

1. Compute the Fundamental Matrix **F** from points correspondences

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

Two-view SfM

1. Compute the Fundamental Matrix **F** from points correspondences
   8-point algorithm

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

Two-view SfM

1. Compute the Fundamental Matrix **F** from points correspondences
   8-point algorithm

2. Compute the camera matrices **P** from the Fundamental matrix
   $\mathbf{P} = [\, \mathbf{I} \mid \mathbf{0} \,]$ and $\mathbf{P'} = [\, [\mathbf{e}_x]\mathbf{F} \mid \mathbf{e'} \,]$

Camera matrices from fundamental matrix

$$\mathbf{P} = [\mathbf{I}|\mathbf{0}] \qquad \mathbf{P}' = [[\boldsymbol{e}_\times]\mathbf{F}|\boldsymbol{e}']$$

(See Hartley and Zisserman C.9 for proof)

Camera matrices from essential matrix

SVD: $\mathbf{E} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ Let $\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

We get FOUR solutions:

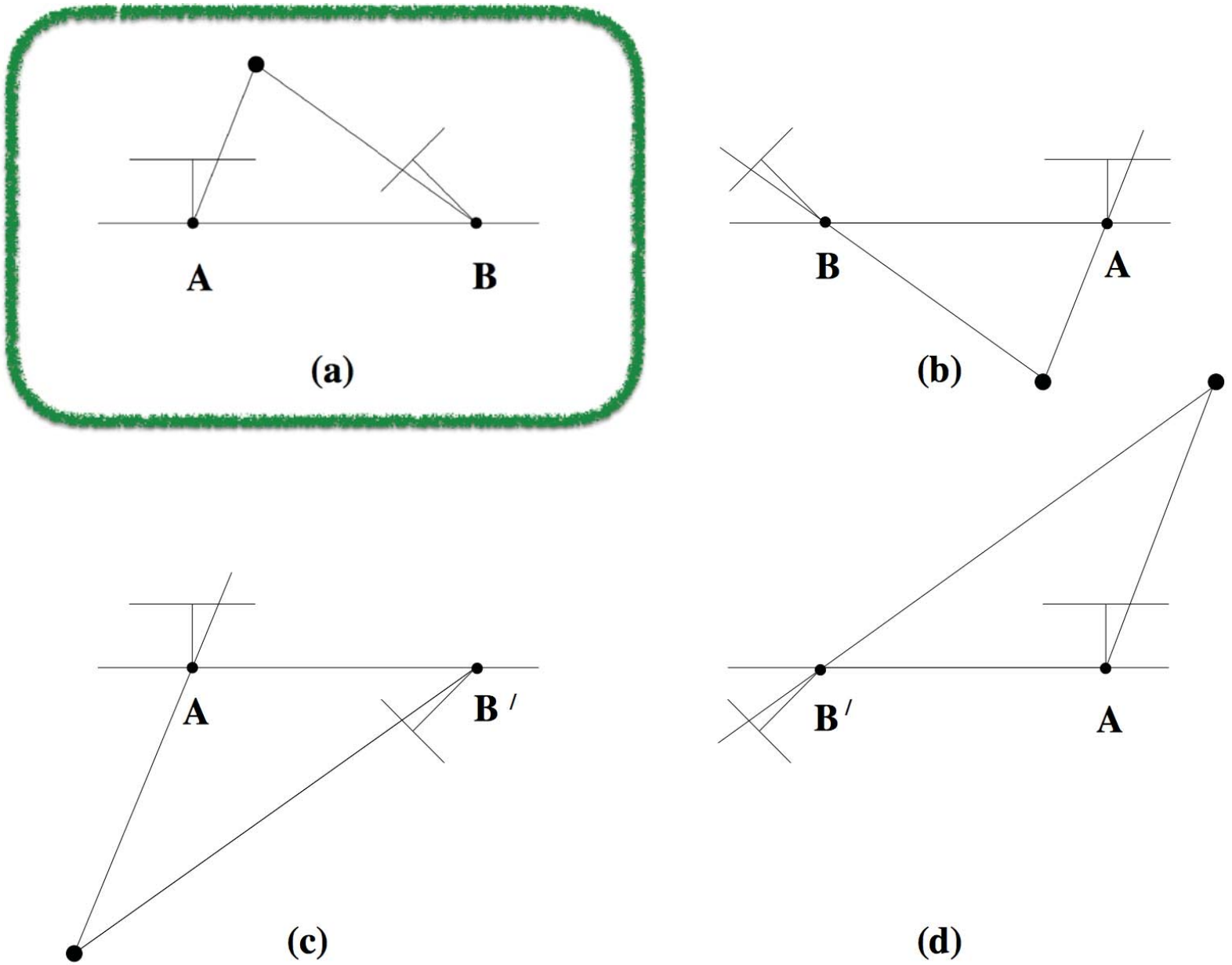
$$P' = [R|T]$$

$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top$ $\mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top$ two possible rotations

$\mathbf{T}_1 = U_3$ $\mathbf{T}_2 = -U_3$ two possible translations

(See Hartley and Zisserman C.9 for proof)

Find the configuration where the points is in front of both cameras



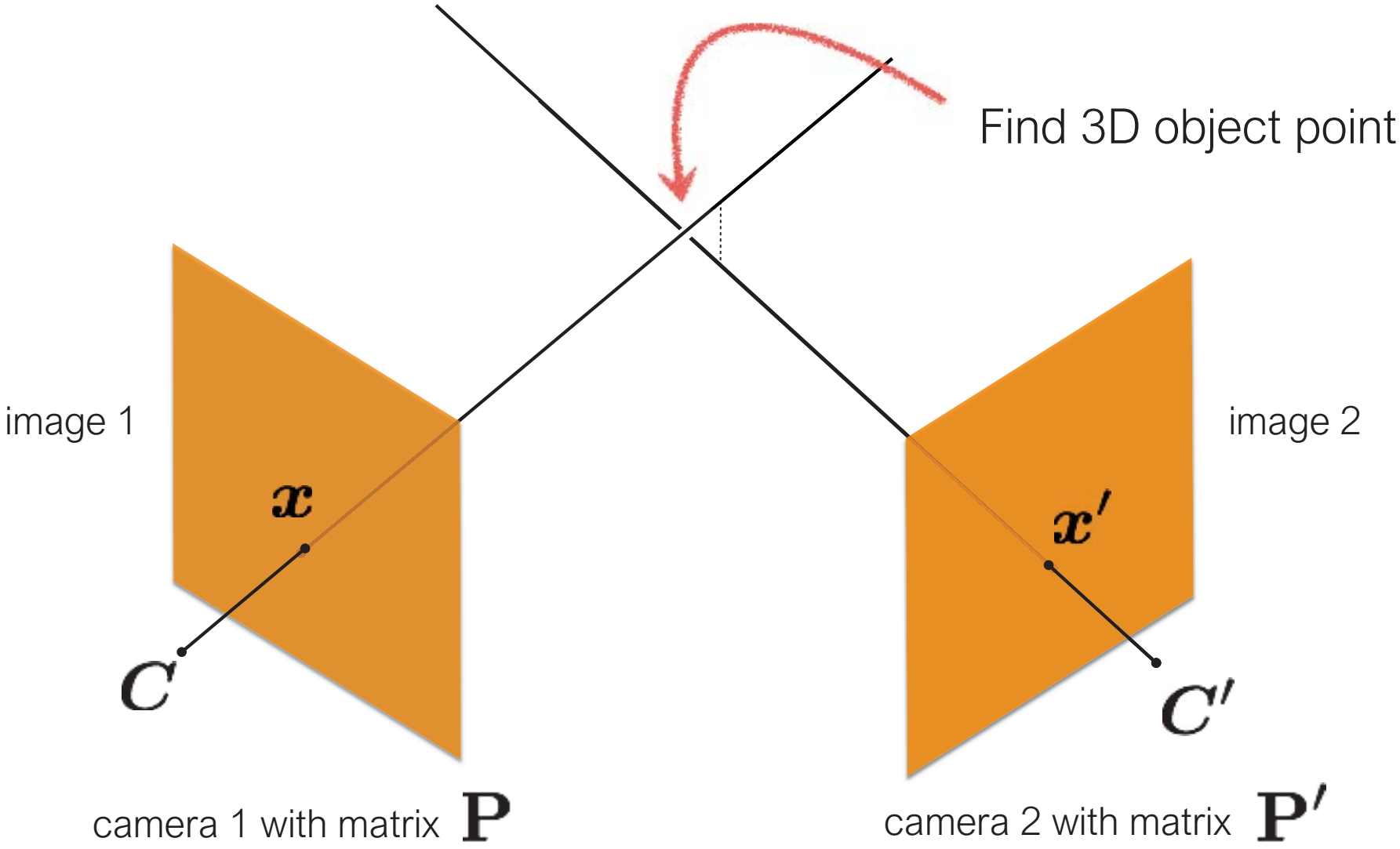
Two-view SfM

1. Compute the Fundamental Matrix **F** from points correspondences
   8-point algorithm

2. Compute the camera matrices **P** from the Fundamental matrix
   $\mathbf{P} = [\, \mathbf{I} \mid \mathbf{0} \,]$ and $\mathbf{P'} = [\, [\mathbf{e'}_x]\mathbf{F} \mid \mathbf{e'} \,]$

3. For each point correspondence, compute the point **X** in 3D space (triangularization)
   DLT with $x = \mathbf{P}\,X$ and $x' = \mathbf{P'}\,X$

Triangulation



Two-view SfM

1. Compute the Fundamental Matrix **F** from points correspondences
   8-point algorithm

2. Compute the camera matrices **P** from the Fundamental matrix
   $\mathbf{P} = [\, \mathbf{I} \mid \mathbf{0} \,]$ and $\mathbf{P'} = [\, [\mathbf{e'}_x]\mathbf{F} \mid \mathbf{e'} \,]$

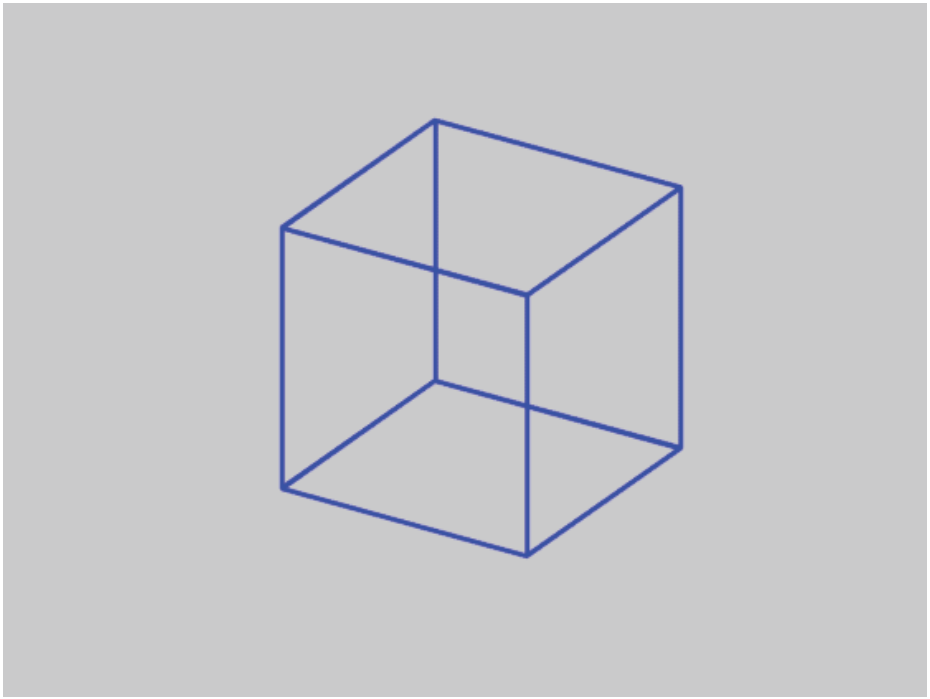
3. For each point correspondence, compute the point **X** in 3D space (triangularization)
   DLT with $x = \mathbf{P}\,X$ and $x' = \mathbf{P'}\,X$

Is SfM always uniquely solvable?

Ambiguities in structure from motion

Is SfM always uniquely solvable?

- No…



Projective Ambiguity

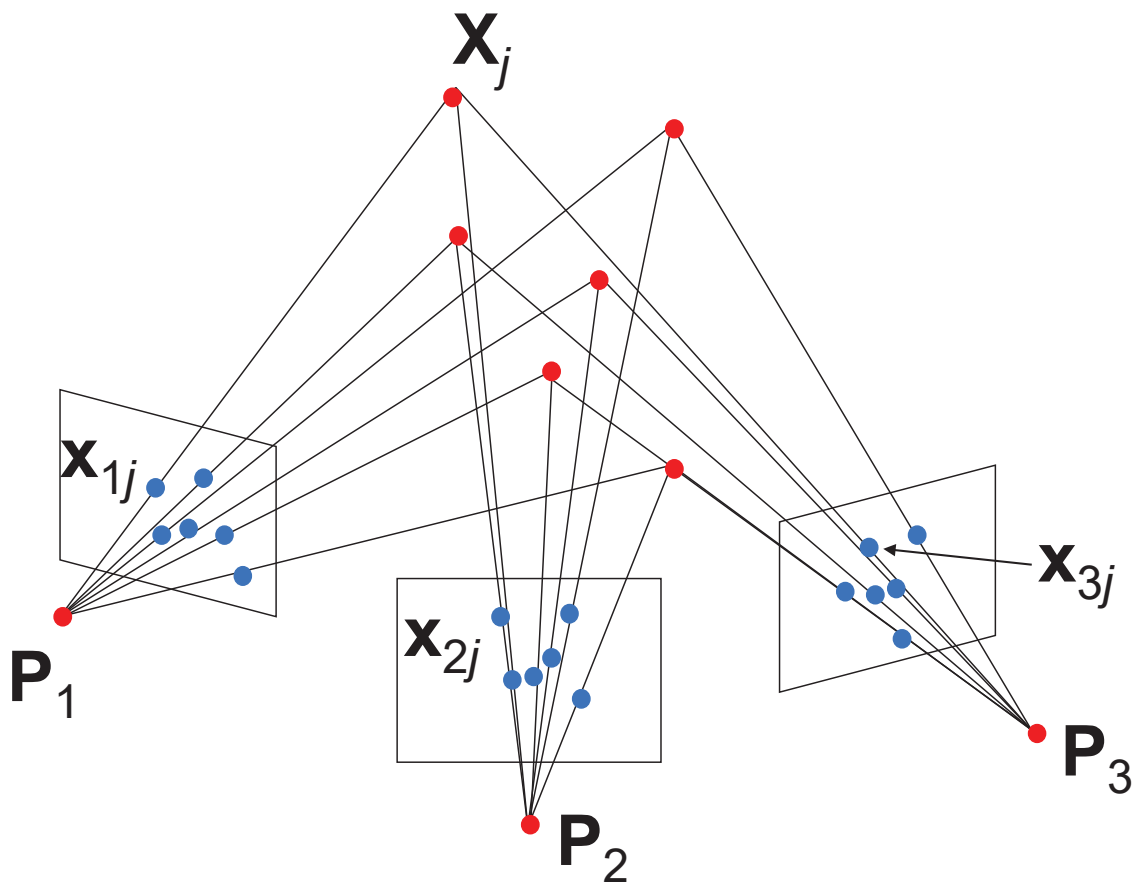
- Reconstruction is ambiguous by an arbitrary 3D projective transformation without prior knowledge of camera parameters

Structure from motion

- Given: *m* images of *n* fixed 3D points

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

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



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 in the image remain exactly the same:

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

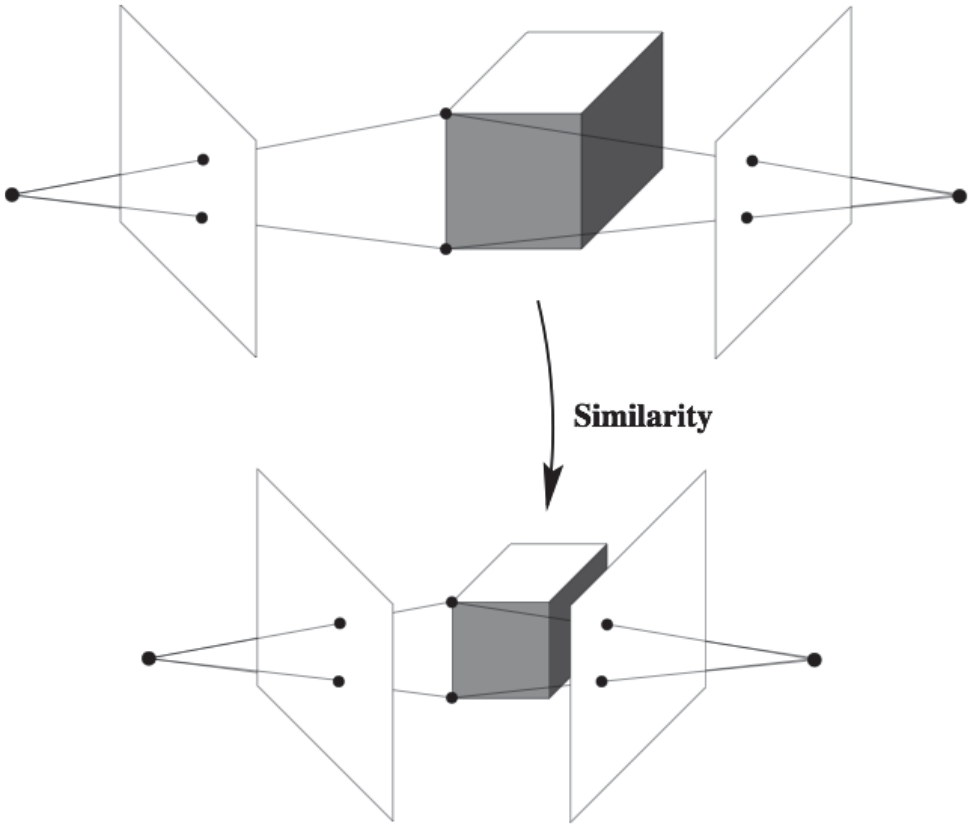
It is impossible to recover the absolute scale of the scene!

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 in the image remain exactly the same

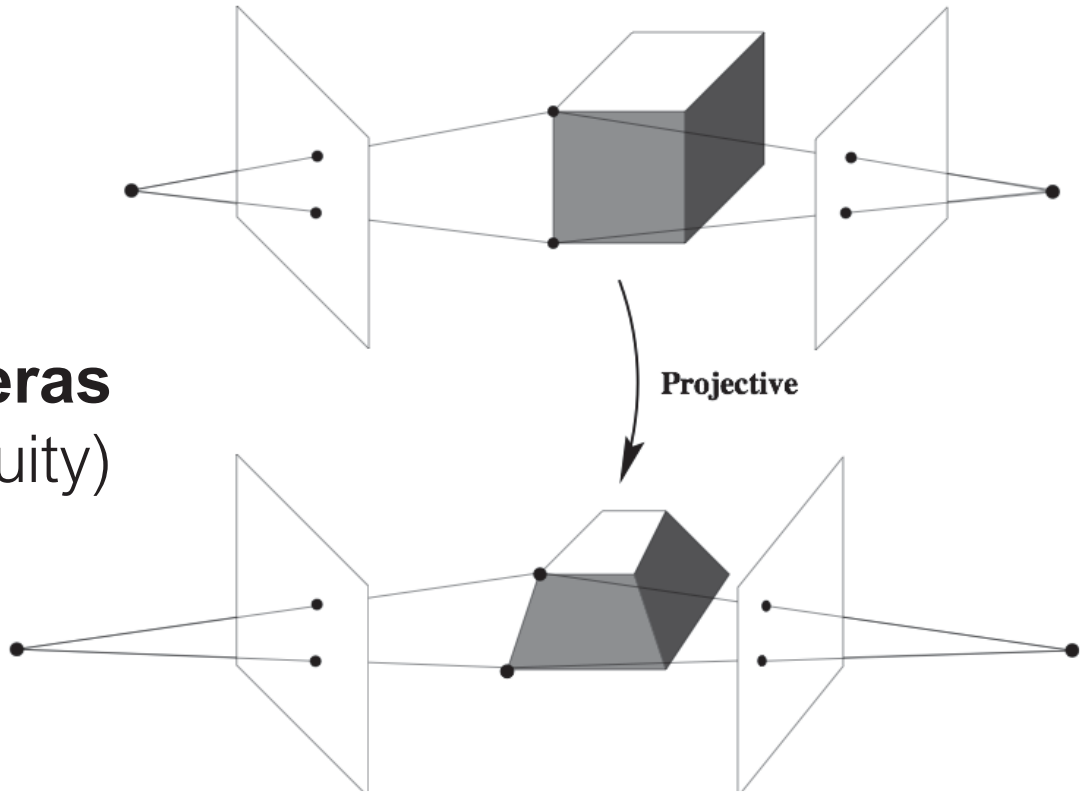
- More generally: if we transform the scene using a transformation **Q** and apply the inverse transformation to the camera matrices, then the images do not change

$$\mathbf{x} = \mathbf{PX} = \left(\mathbf{PQ}^{-1}\right)\left(\mathbf{QX}\right)$$

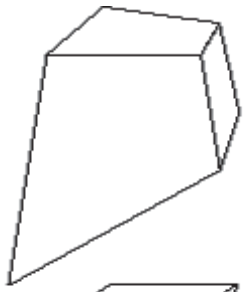
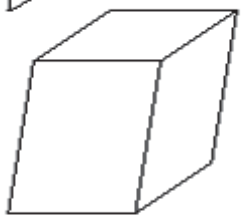
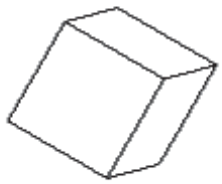
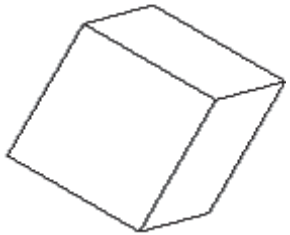


Calibrated cameras (= known K)
(similarity projection ambiguity)

Uncalibrated cameras
(projective projection ambiguity)



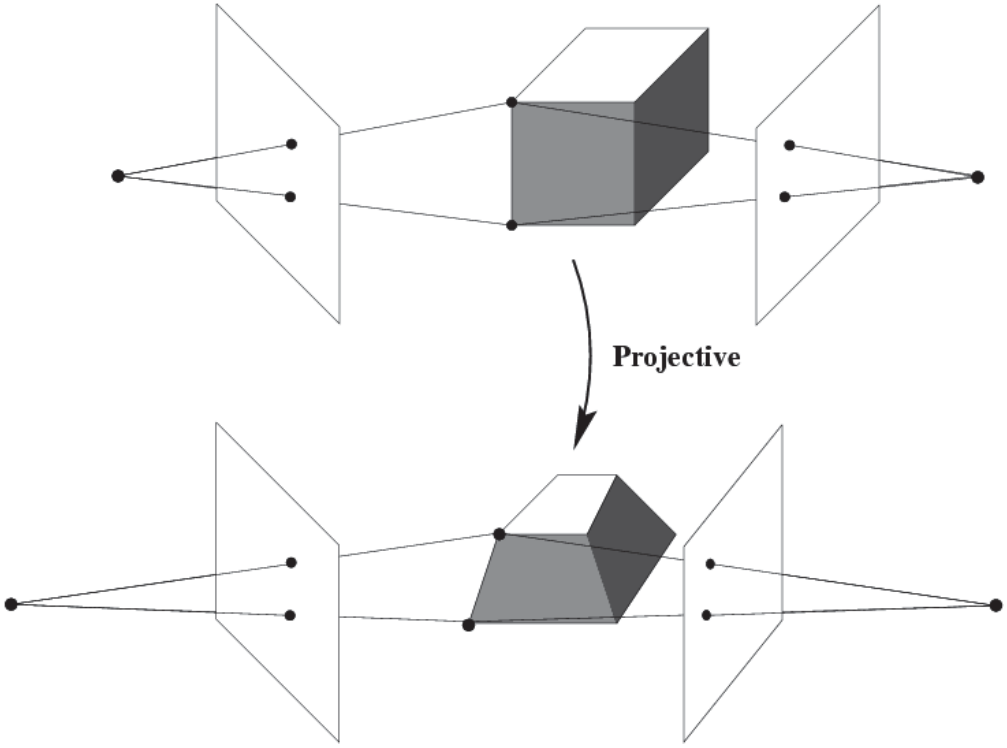
Types of ambiguity

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

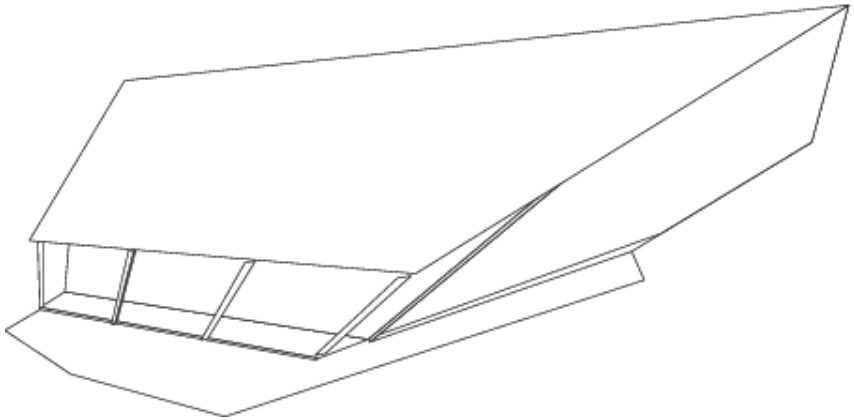
Slide: S. Lazebnik

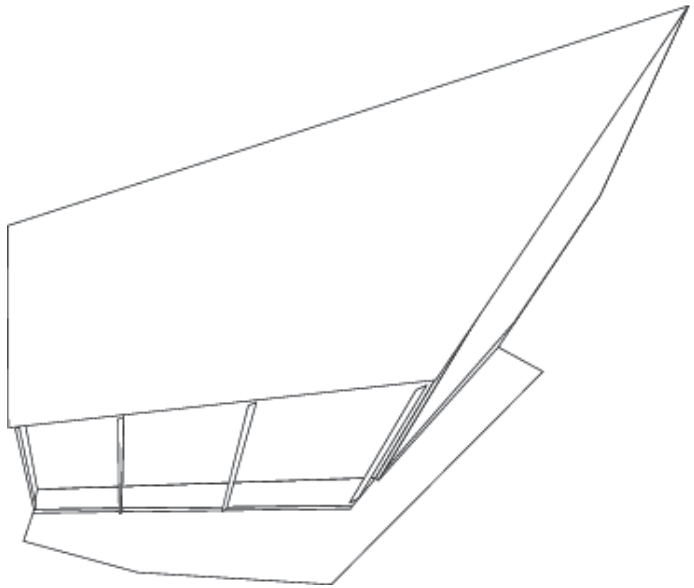
Projective ambiguity



$$\mathbf{x} = \mathbf{PX} = \left(\mathbf{PQ}_{\mathbf{P}}^{-1}\right)\left(\mathbf{Q}_{\mathbf{P}}\,\mathbf{X}\right)$$

Projective ambiguity





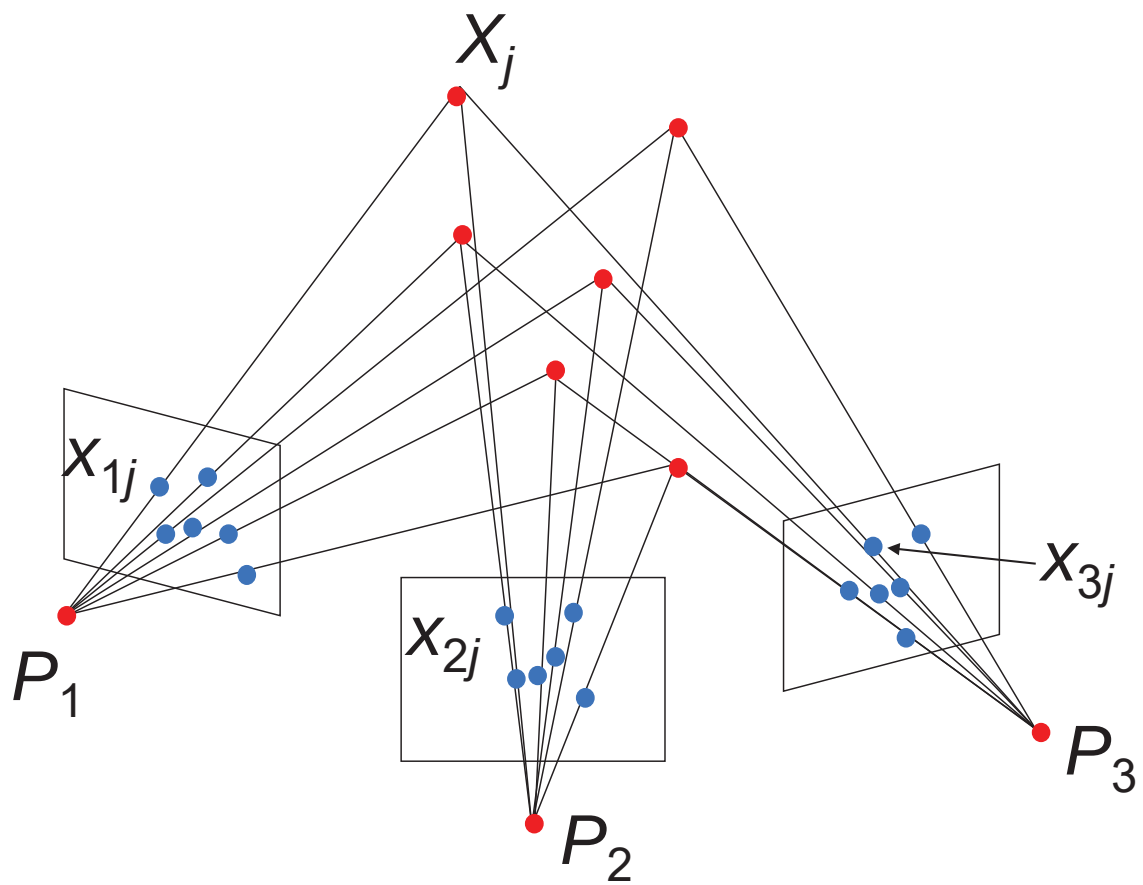
Multi-view projective structure from motion

Projective structure from motion

- Given: *m* images of *n* fixed 3D points

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

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



Projective structure from motion

- Given: *m* images of *n* fixed 3D points

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

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation **Q**:

$$\mathbf{X} \rightarrow \mathbf{QX},\ \mathbf{P} \rightarrow \mathbf{PQ}^{-1}$$

- We can solve for structure and motion when

$$2mn >= 11m + 3n - 15$$

- For two cameras, at least 7 points are needed

Projective SFM: Two-camera case

- Compute fundamental matrix $\mathbf{F}$ between the two views
- First camera matrix: $[\mathbf{I}|\mathbf{0}]$
- Second camera matrix: $[\mathbf{A}|\mathbf{b}]$
- Then $\mathbf{b}$ is the epipole ($\mathbf{F}^{\mathrm{T}}\mathbf{b} = 0$), $\mathbf{A} = -[\mathbf{b}_{\times}]\mathbf{F}$

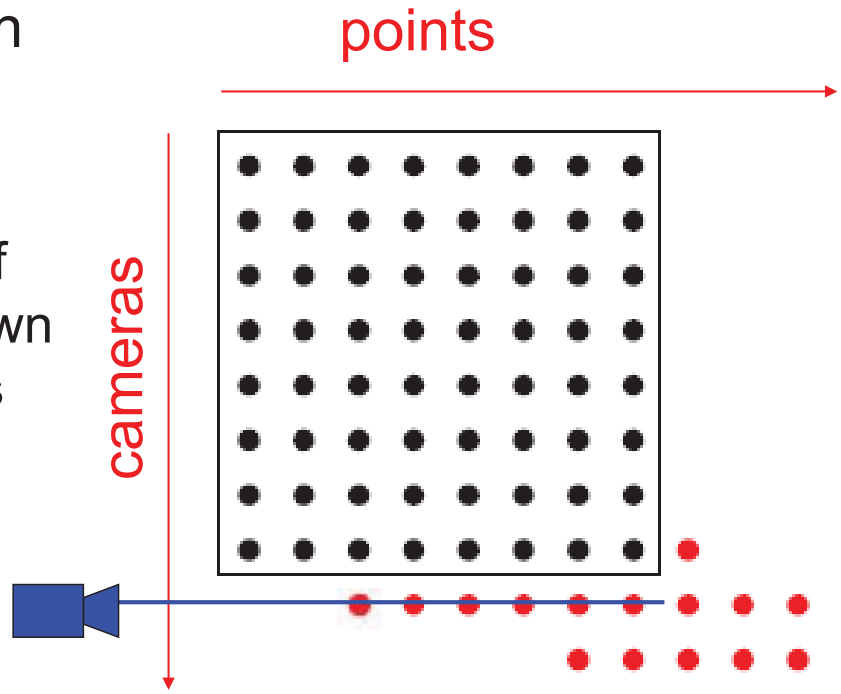
Sequential structure from motion

•Initialize motion from two images using 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*



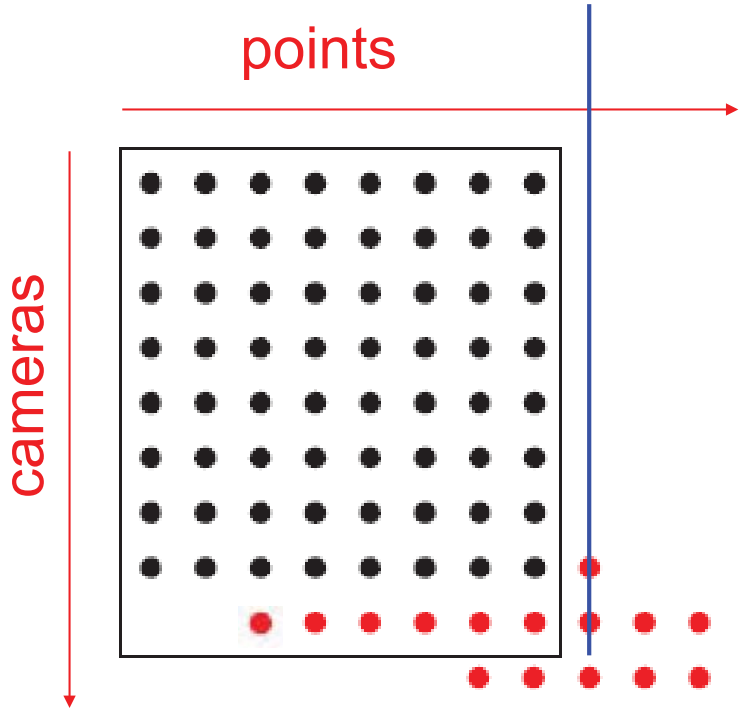
Sequential structure from motion

•Initialize motion from two images using 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 new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*



Sequential structure from motion

•Initialize motion from two images using fundamental matrix

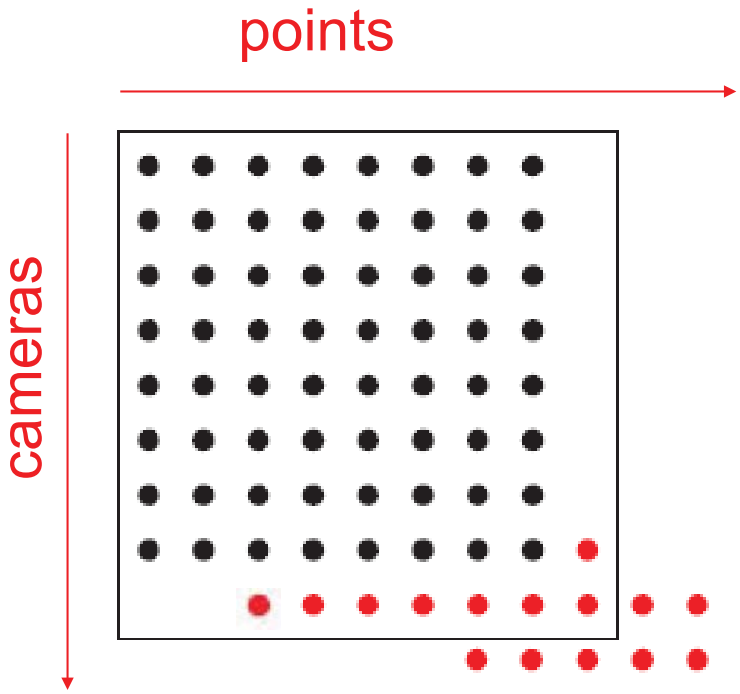
•Initialize structure by triangulation

•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 new 3D points, re-optimize existing points that are also seen by this camera – *triangulation*

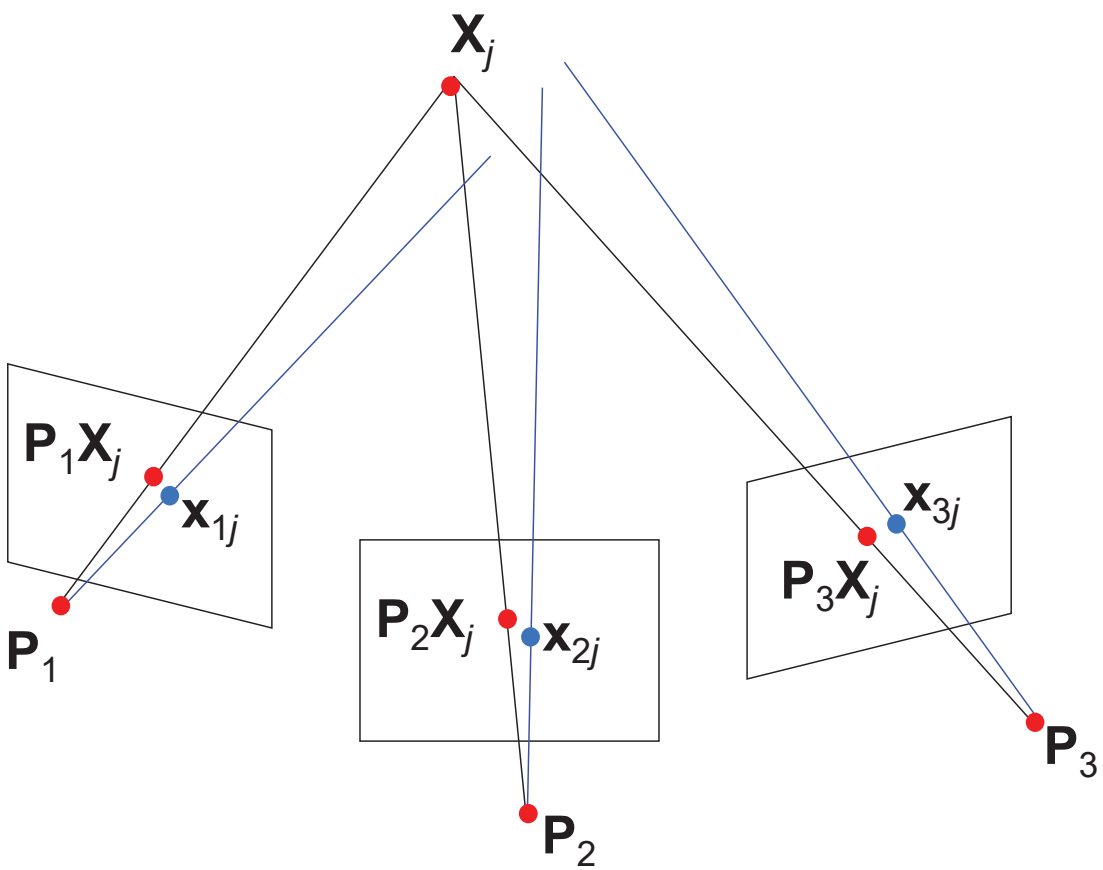
•Refine structure and motion: bundle adjustment



Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizing reprojection error

$$E(\mathbf{P},\mathbf{X}) = \sum_{i=1}^{m}\sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i\mathbf{X}_j\right)^2$$



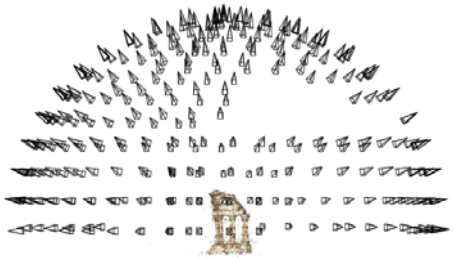
Review: Structure from motion

- Ambiguity
- Dealing with missing data
  - Incremental structure from motion
- Projective structure from motion
  - Bundle adjustment

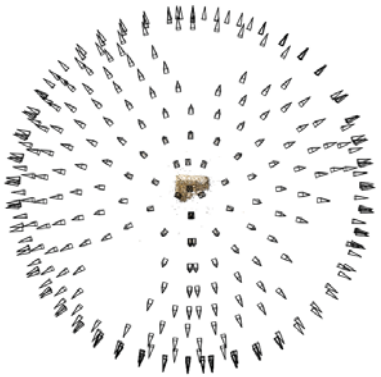
	Structure (scene geometry)	Motion (camera geometry)	Measurements
Pose Estimation	known	estimate	3D to 2D correspondences
Triangulation	estimate	known	2D to 2D coorespondences
Reconstruction	estimate	estimate	2D to 2D coorespondences

Large-scale structure from motion

Structure from motion



Reconstruction (side)



(top)

- Input: images with points in correspondence
  $p_{i,j} = (u_{i,j}, v_{i,j})$
- Output
  - structure: 3D location $\mathbf{x}_i$ for each point p_i
  - motion: camera parameters $\mathbf{R}_j$, $\mathbf{t}_j$ possibly $\mathbf{K}_j$
- Objective function: minimize *reprojection error*

15,464

37,383

76,389

Standard way to view photos

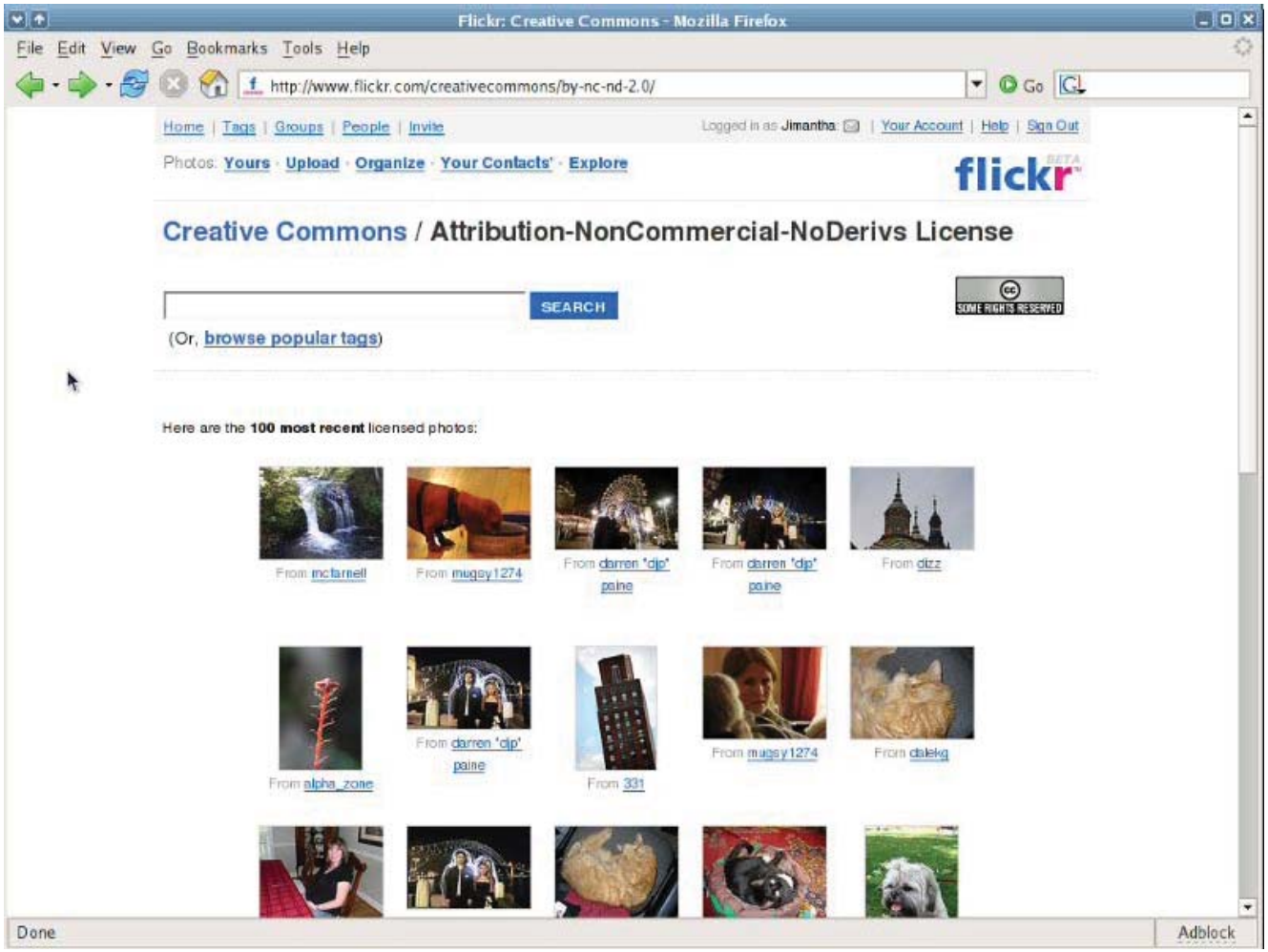
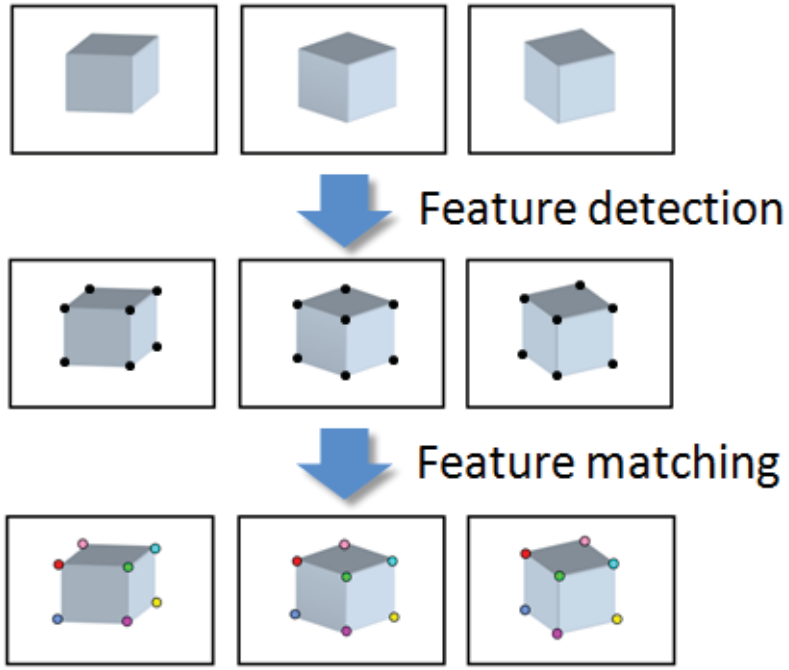


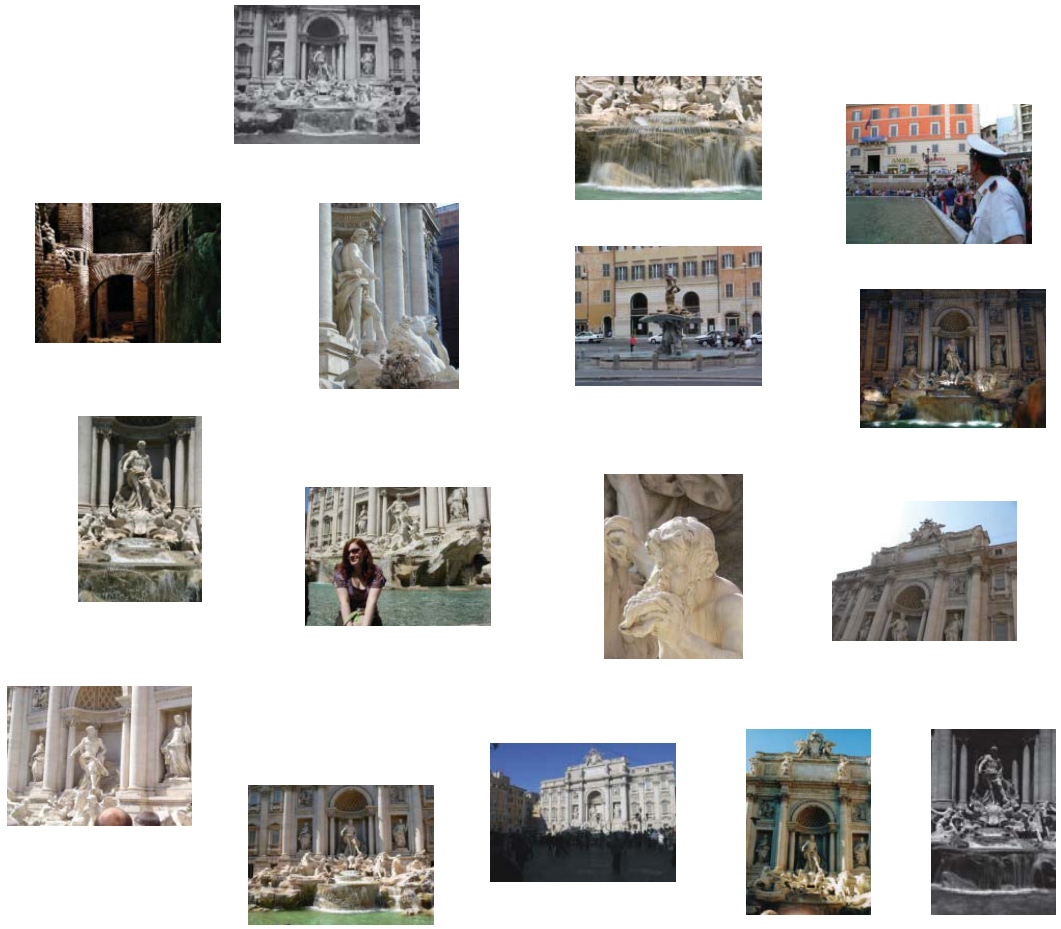
Photo Tourism

Input: Point correspondences



Feature detection

Detect features using SIFT [Lowe, IJCV 2004]

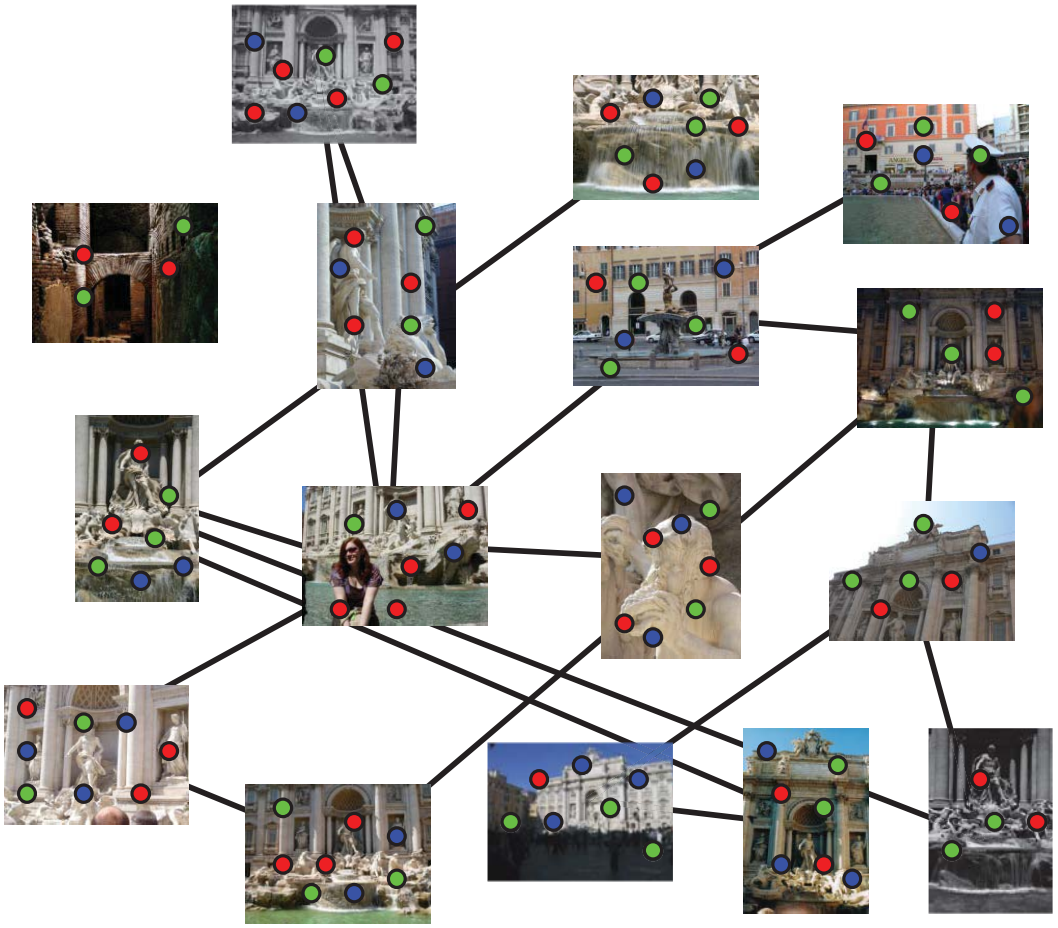


Feature description

Describe features using SIFT [Lowe, IJCV 2004]

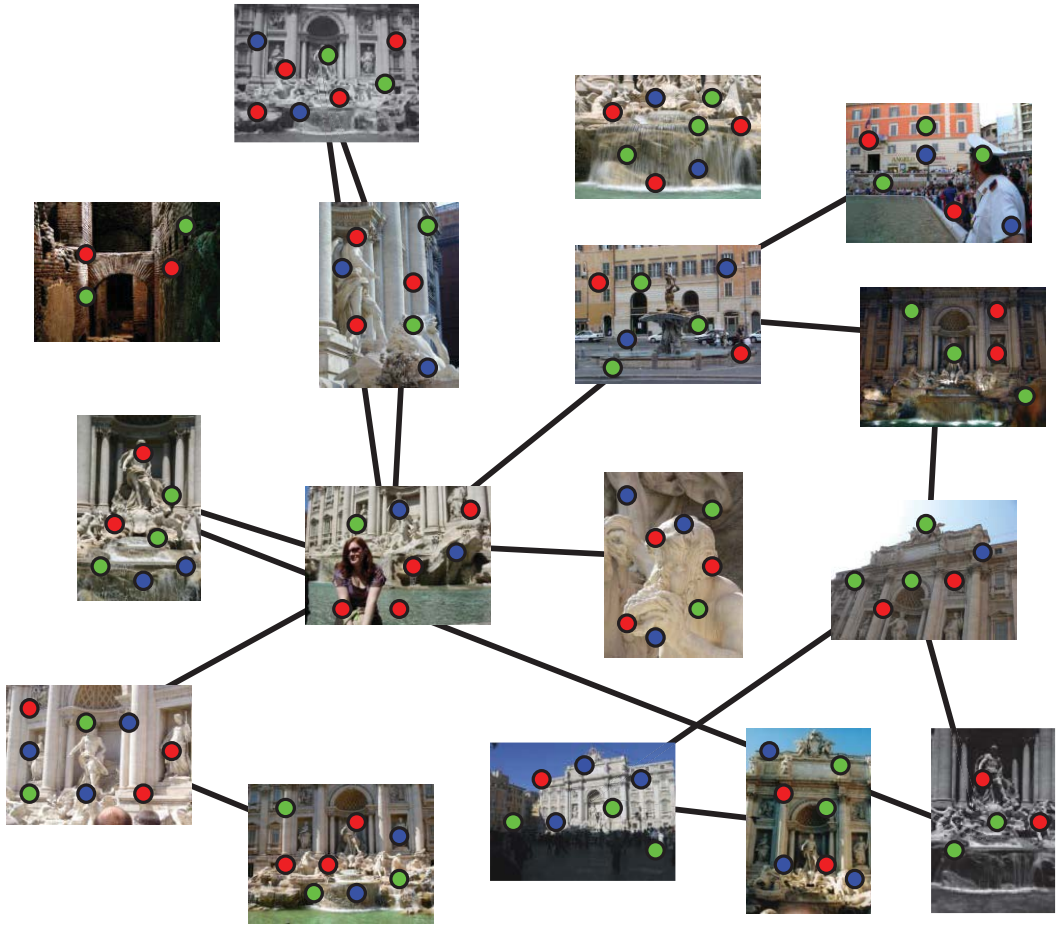
Feature matching

Match features between each pair of images



Feature matching

Refine matching using RANSAC to estimate fundamental matrix between each pair



Correspondence estimation

- Link up pairwise matches to form connected components of matches across several images

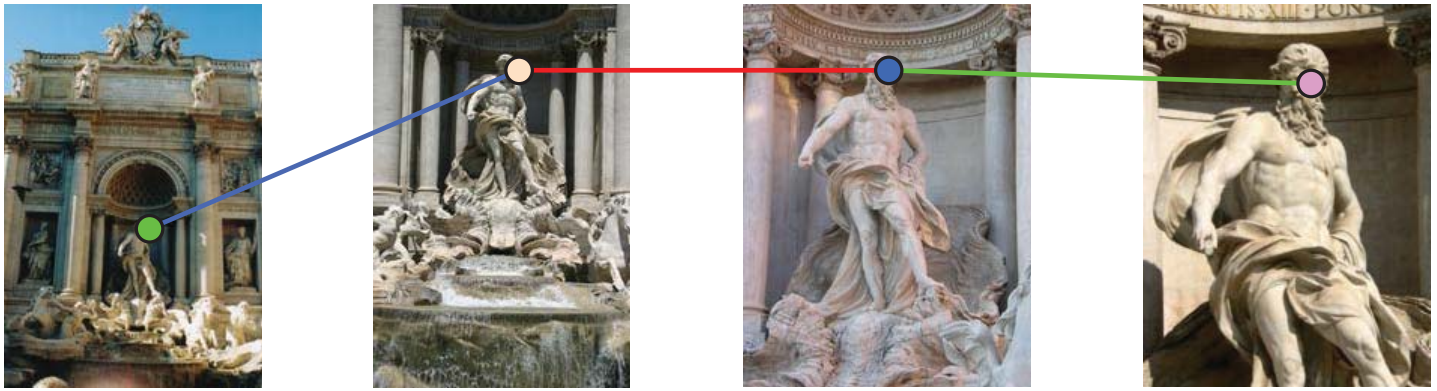
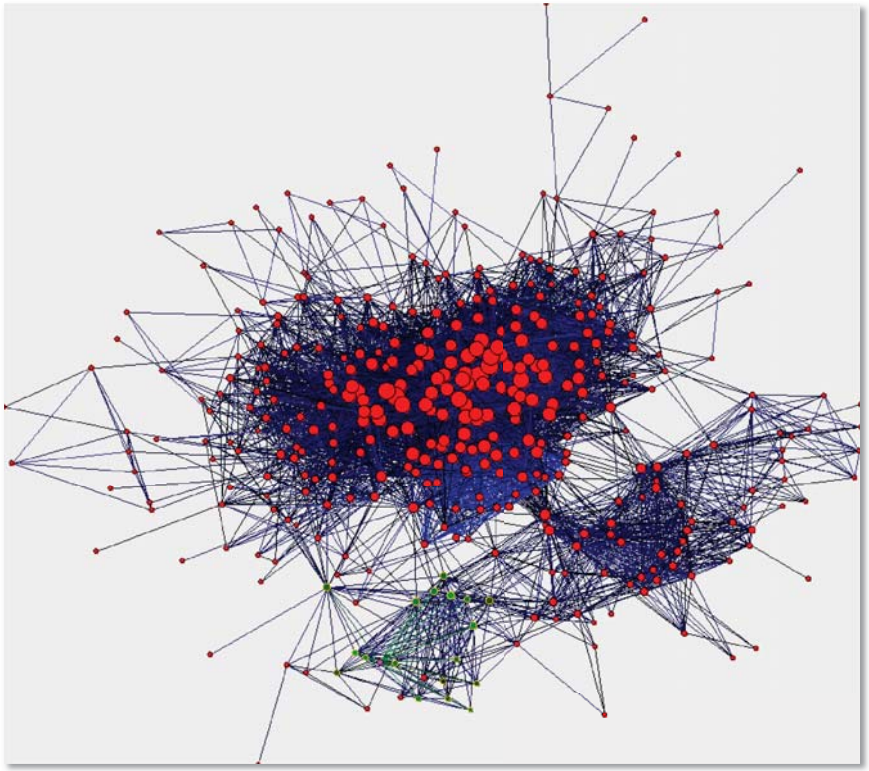


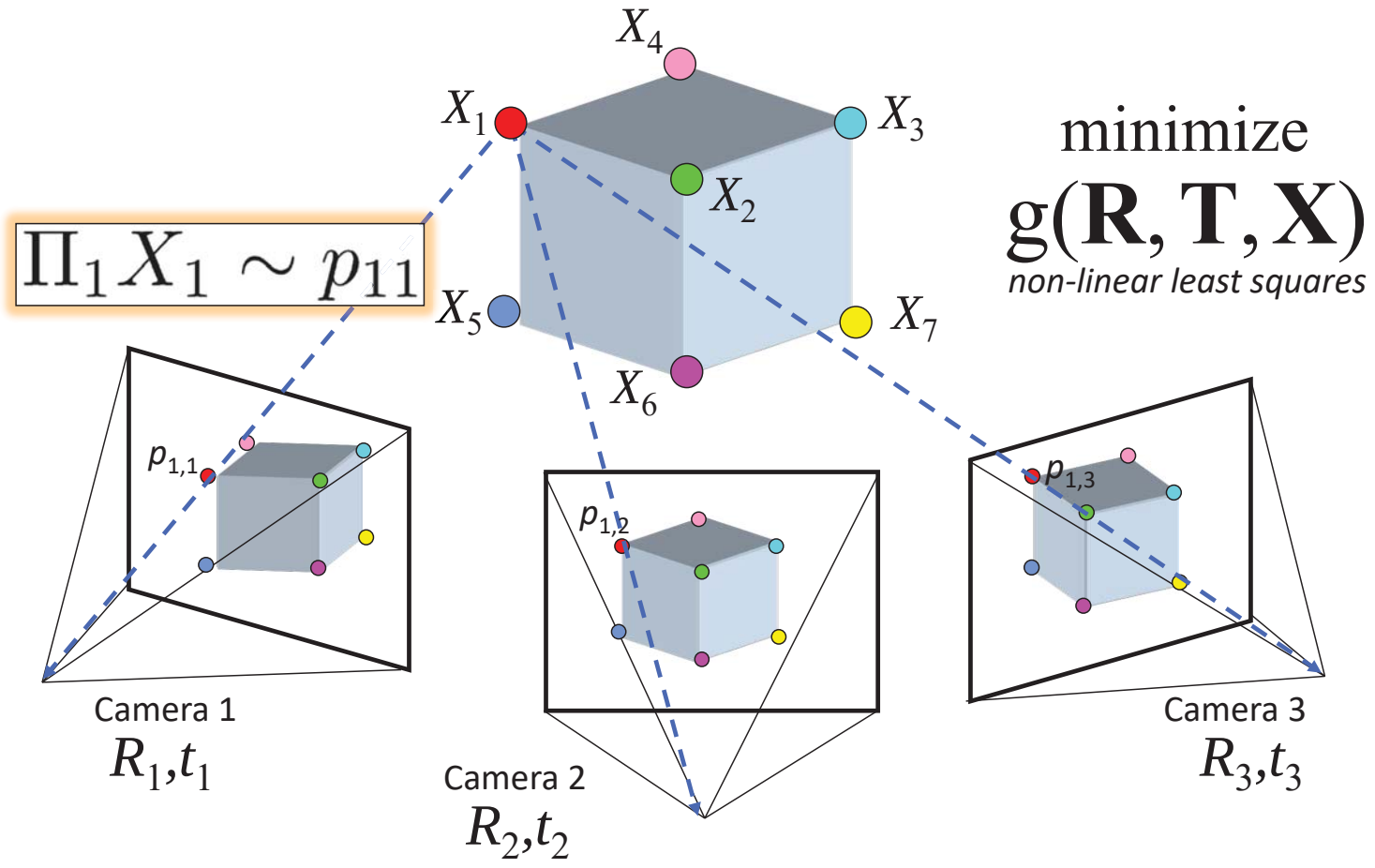
Image 1 Image 2 Image 3 Image 4

Image connectivity graph



(graph layout produced using the Graphviz toolkit: http://www.graphviz.org/)

Structure from motion



Global structure from motion

- Minimize sum of squared reprojection errors:

$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)}_{\text{predicted image location}} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}}_{\text{observed image location}} \right\|^2$$

w_{ij}: *indicator variable*: is point *i* visible in image *j* ?

- Minimizing this function is called *bundle adjustment*
  - Optimized using non-linear least squares, e.g. Levenberg-Marquardt

Problem size

- What are the variables?
- How many variables per camera?
- How many variables per point?

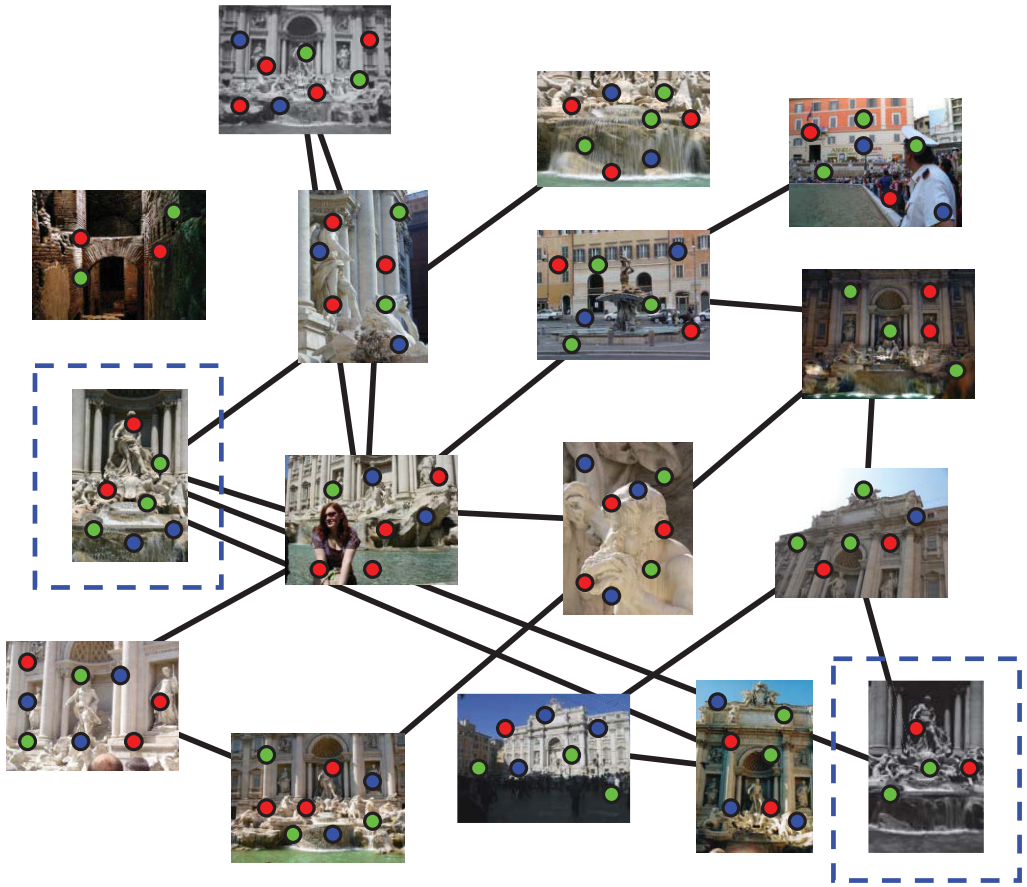
- Trevi Fountain collection

  466 input photos

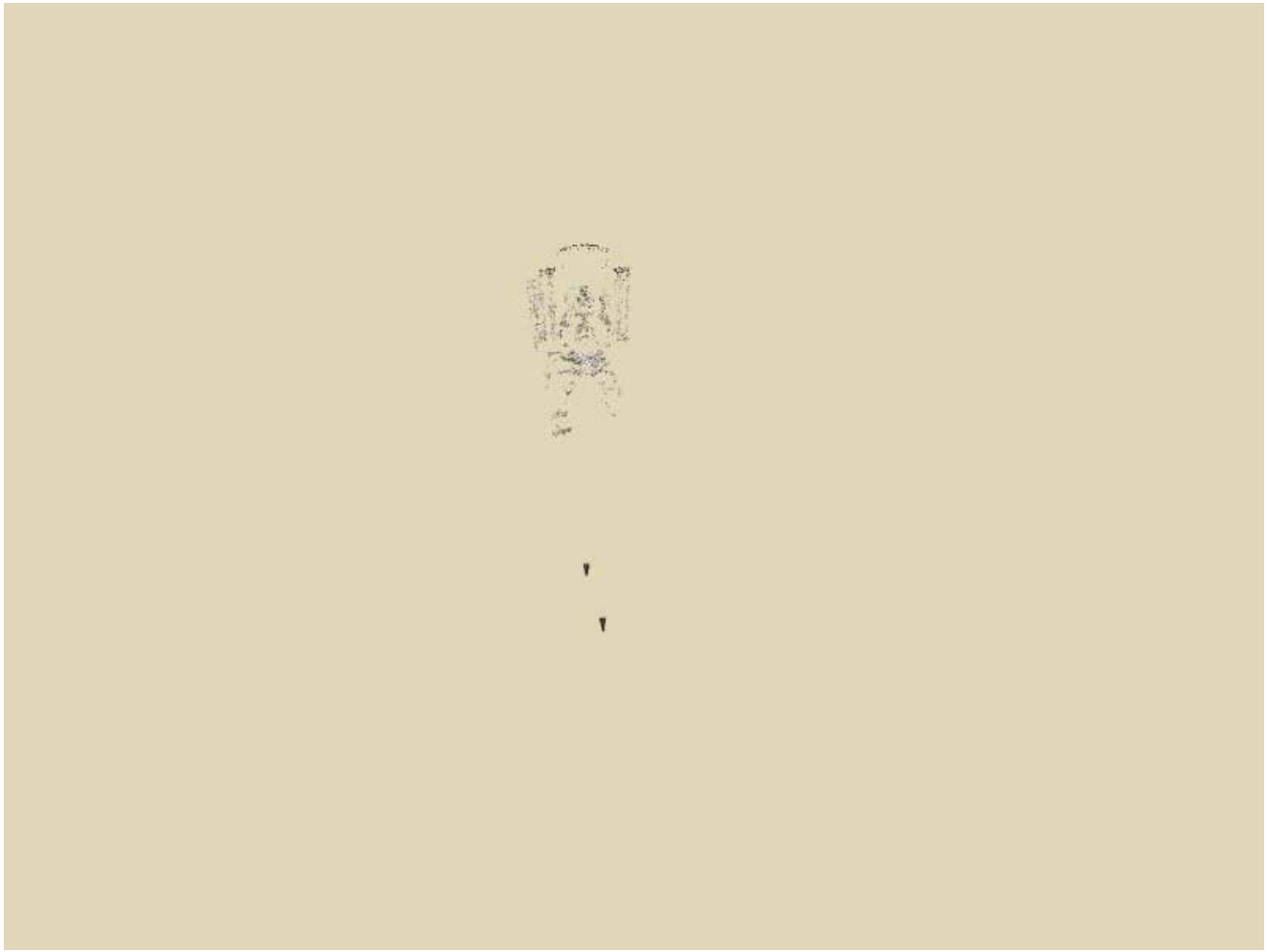
  \+ > 100,000 3D points

  = very large optimization problem

Initialization: Incremental structure from motion



Incremental structure from motion



Final reconstruction

More examples

More examples

Even larger scale SfM

City-scale structure from motion

- “Building Rome in a day”

http://grail.cs.washington.edu/projects/rome/

SfM applications

- 3D modeling
- Surveying
- Robot navigation and mapmaking
- Visual effects (“Match moving”)
  - https://www.youtube.com/watch?v=RdYWp70P_kY

Applications – Photosynth

Applications – Hyperlapse

https://www.youtube.com/watch?v=SOpwHaQnRSY

Stereo

Revisiting triangulation

How would you reconstruct 3D points?

Left image

Right image

How would you reconstruct 3D points?

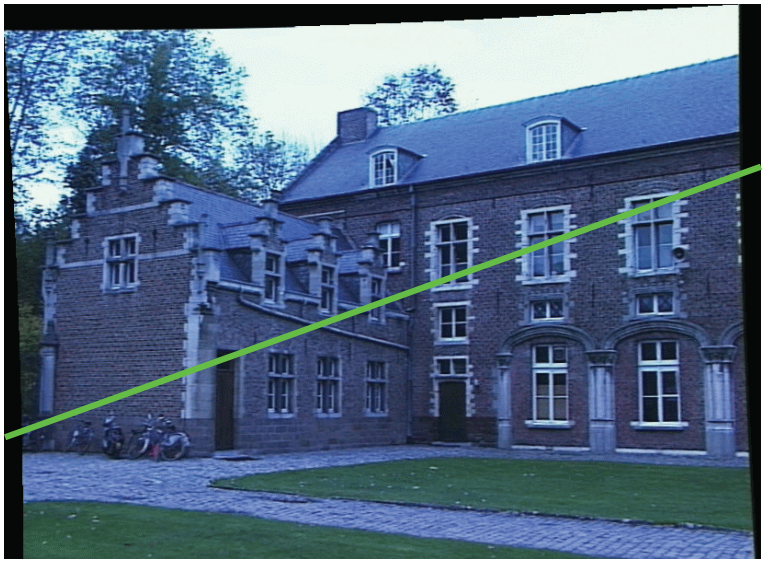
Left image

Right image

1. Select point in one image (how?)

How would you reconstruct 3D points?

Left image

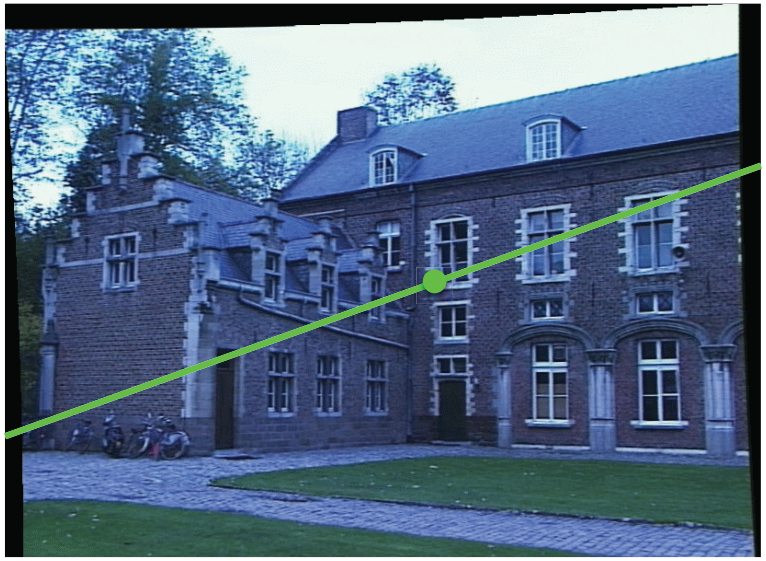


Right image

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

How would you reconstruct 3D points?

Left image

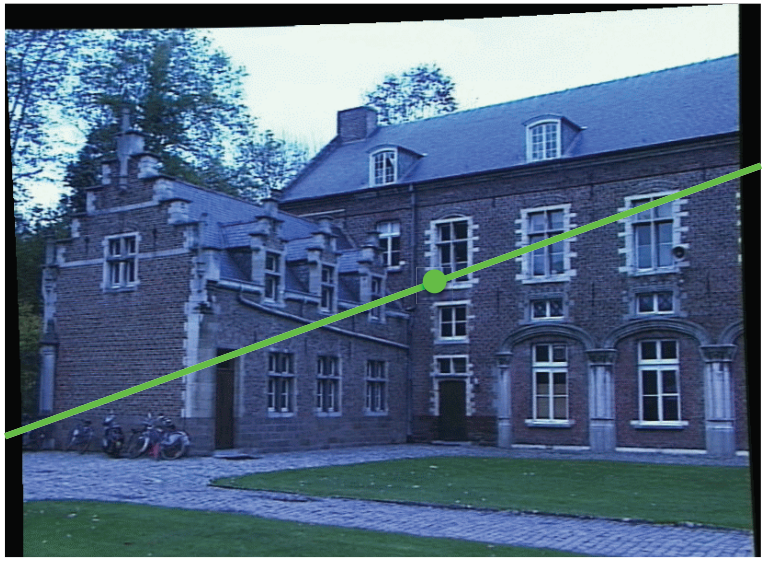


Right image

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

How would you reconstruct 3D points?

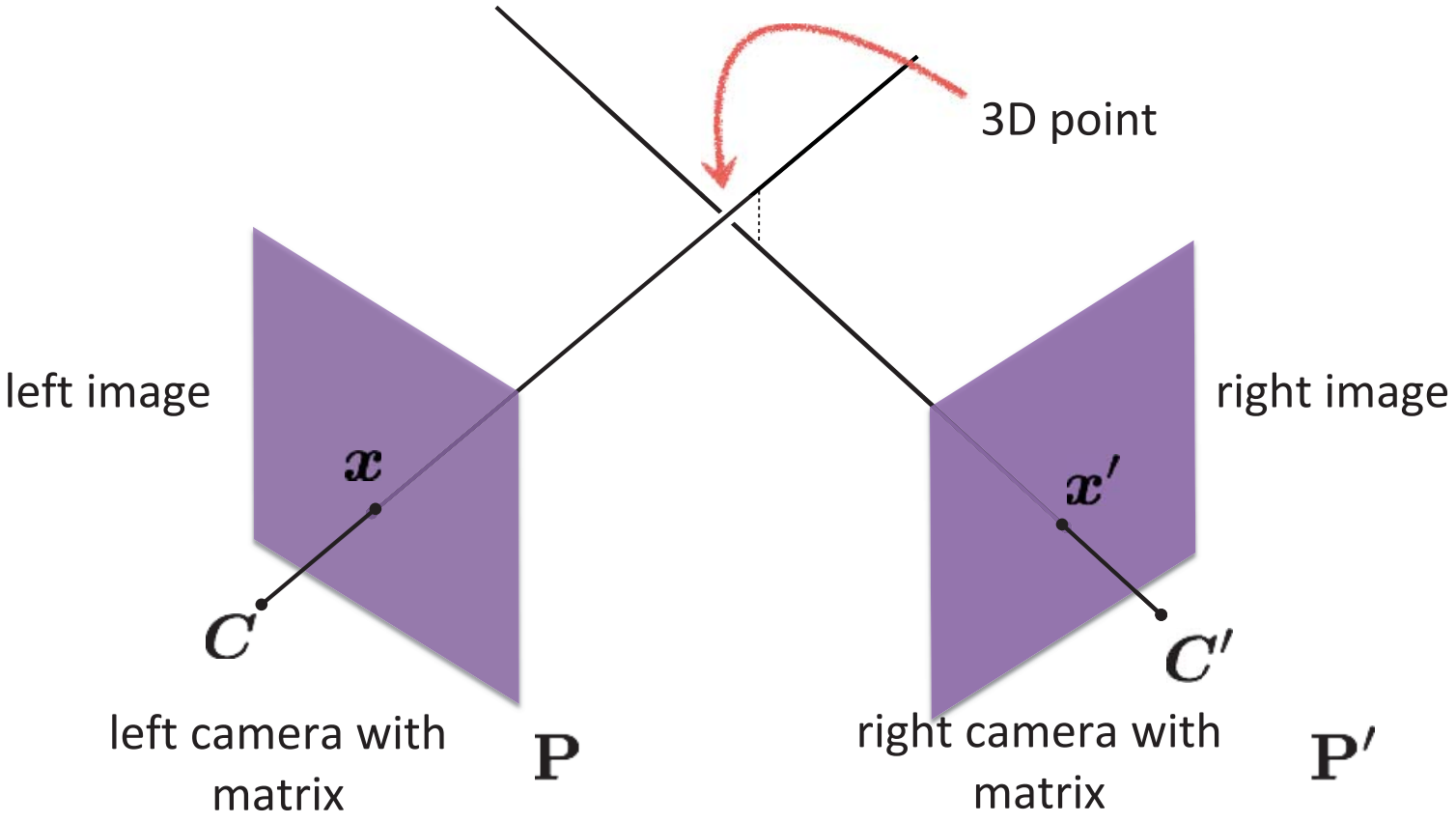
Left image



Right image

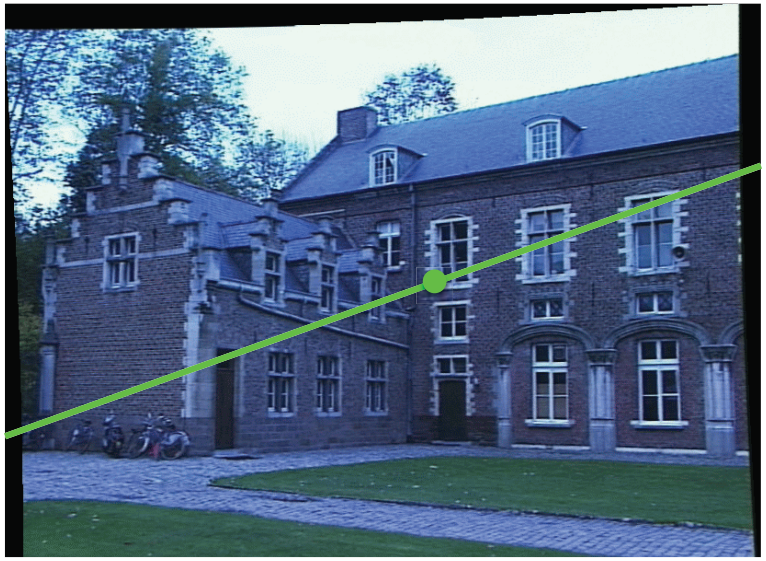
1. Select point in one image (how?)
2. Form epipolar line for that point in second image (how?)
3. Find matching point along line (how?)
4. Perform triangulation (how?)

Triangulation



How would you reconstruct 3D points?

Left image

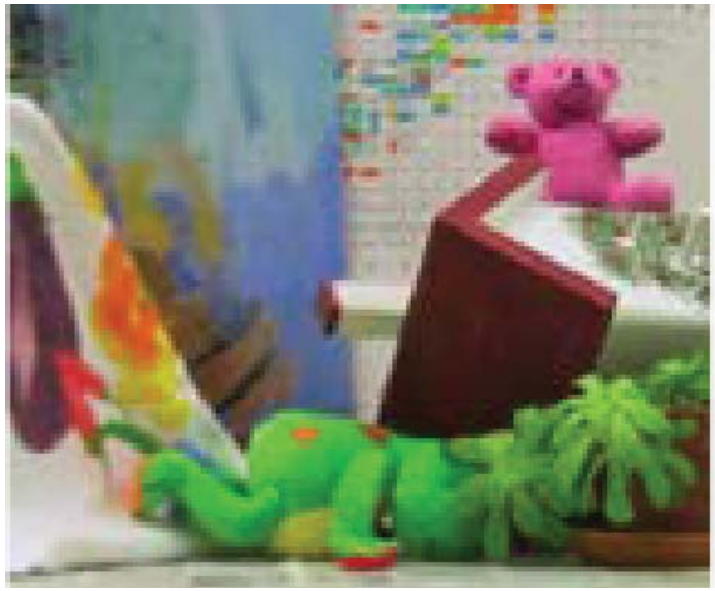


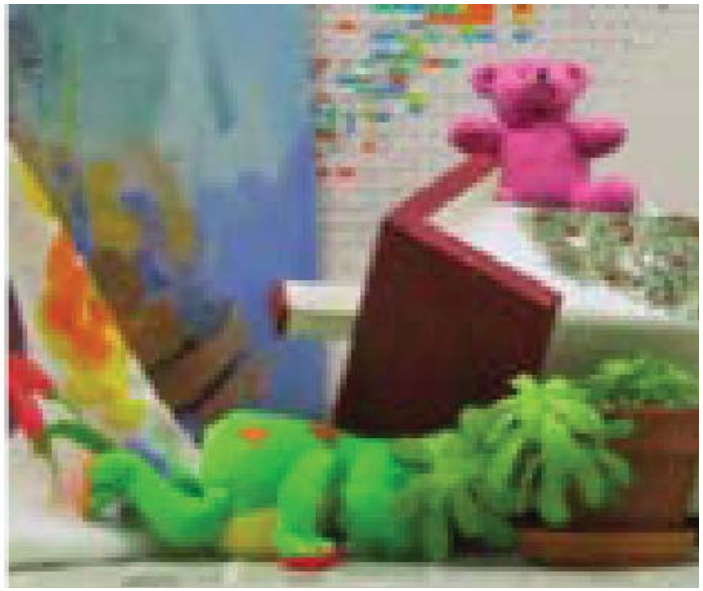
Right image

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

What are the disadvantages of this procedure?

Stereo rectification





What's different between these two images?

Objects that are close move more or less?

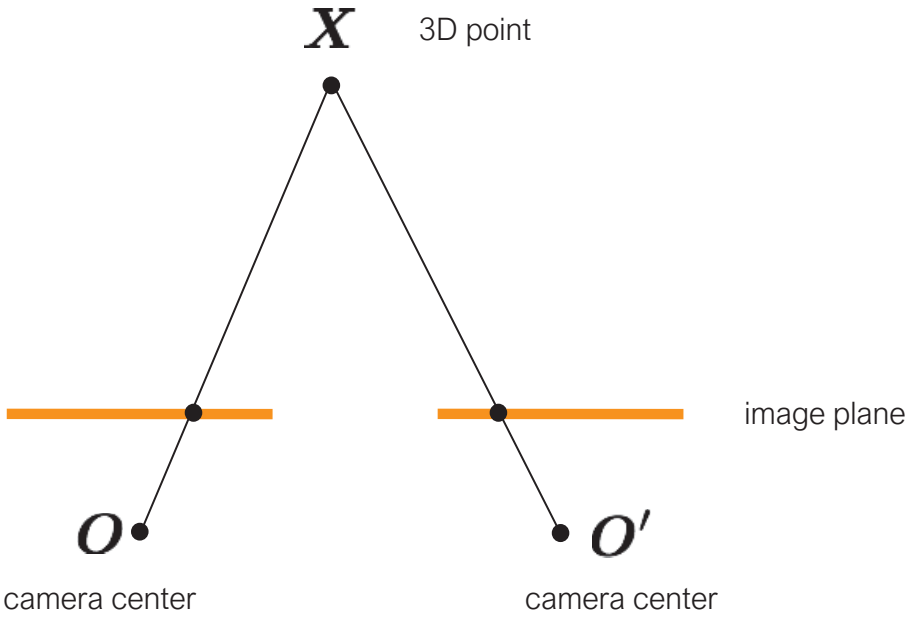
The amount of horizontal movement is
inversely proportional to …

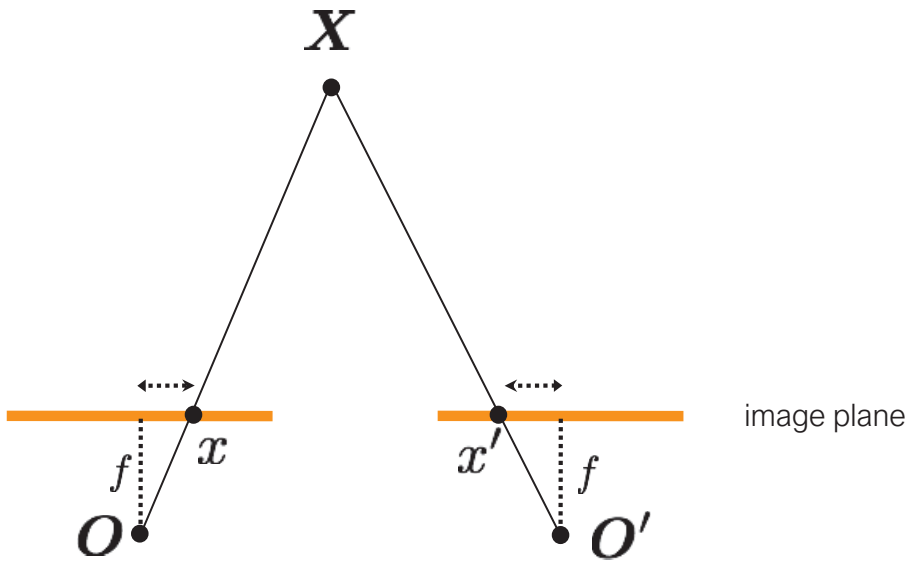


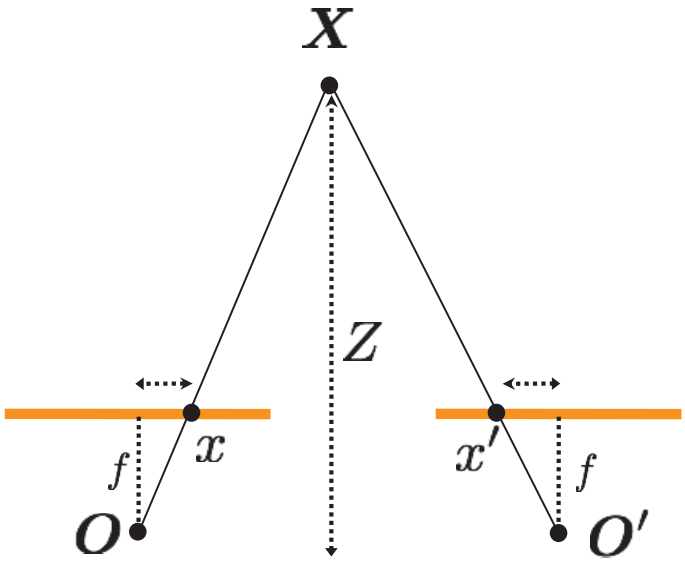
The amount of horizontal movement is
inversely proportional to …

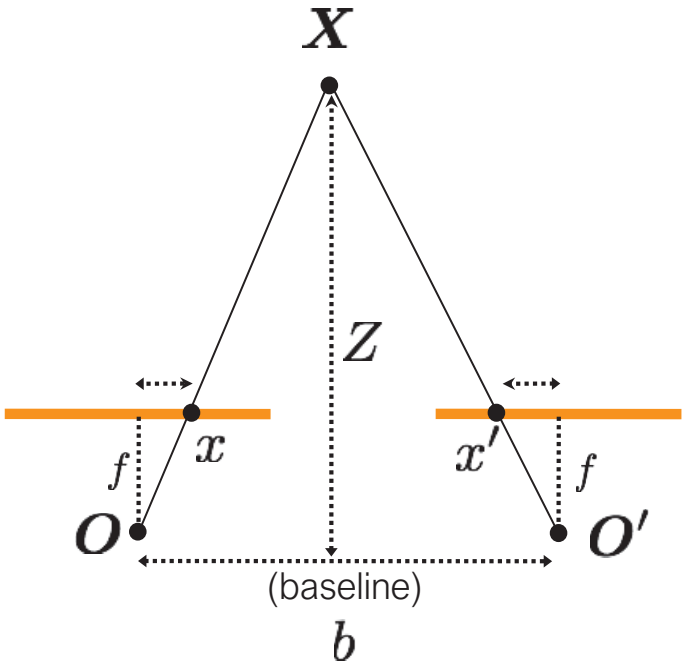
… the distance from the camera.

More formally…

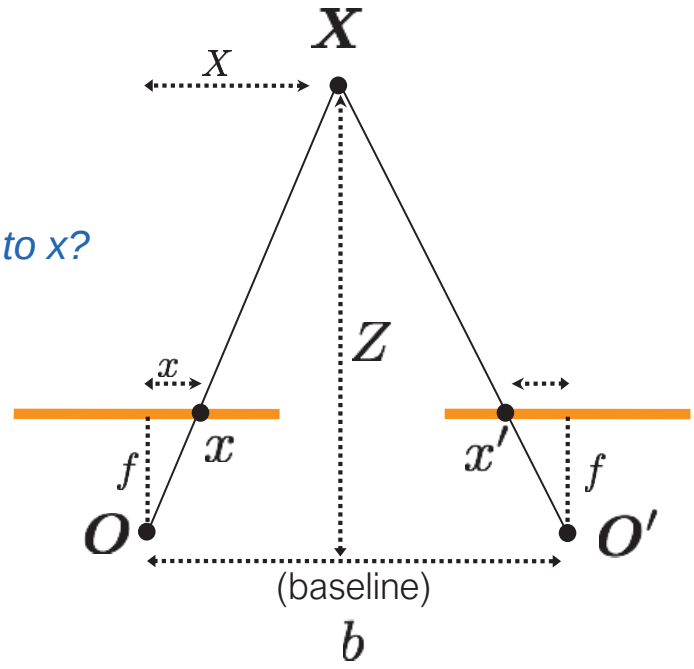

X
3D point
image plane
O
O′
camera center
camera center


X
x
x′
f
f
O
O′
image plane

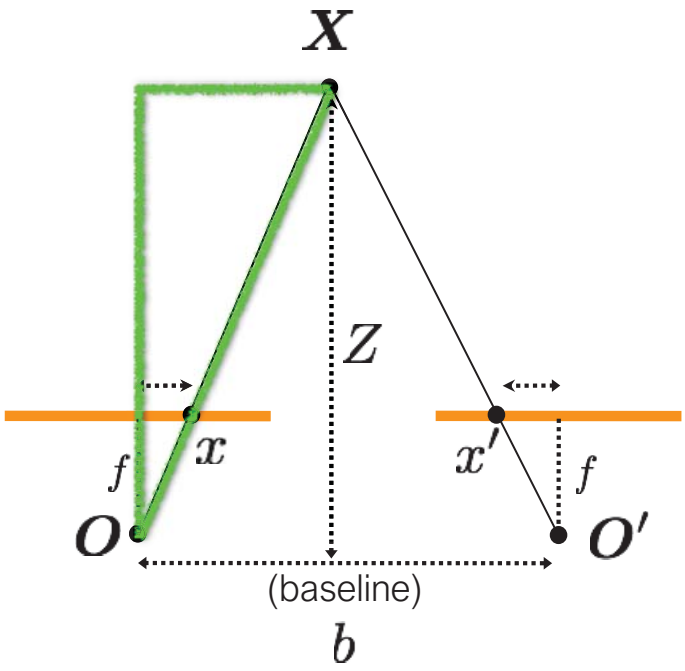

X
Z
x
x′
f
f
O
O′


X
Z
x
x′
f
f
O
O′
(baseline)
b

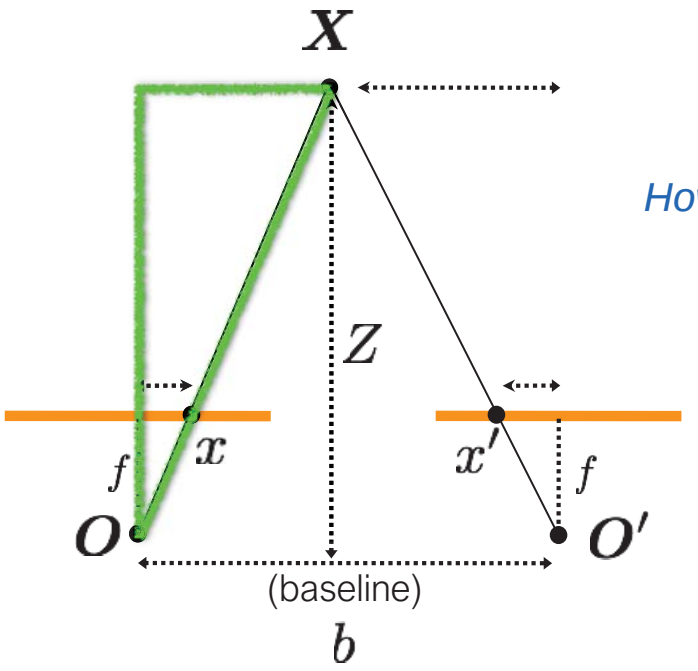
How is X related to x?



$$\frac{X}{Z} = \frac{x}{f}$$

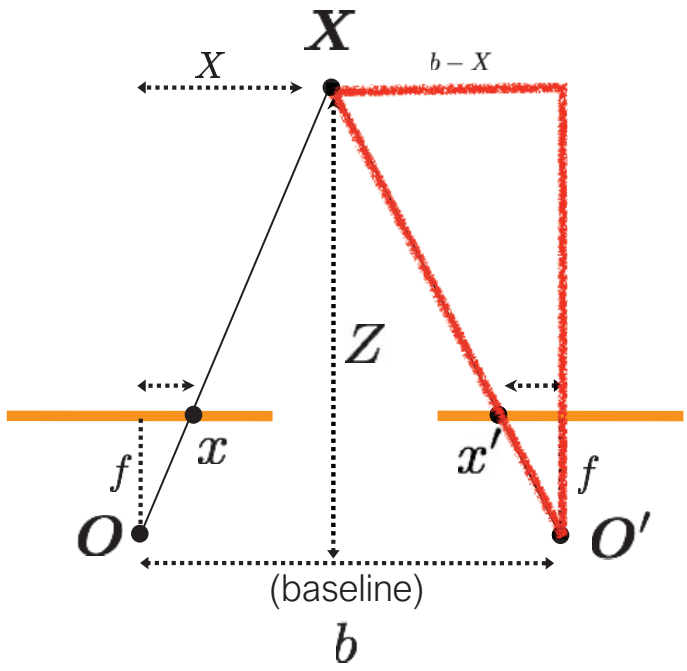


$$\frac{X}{Z} = \frac{x}{f}$$



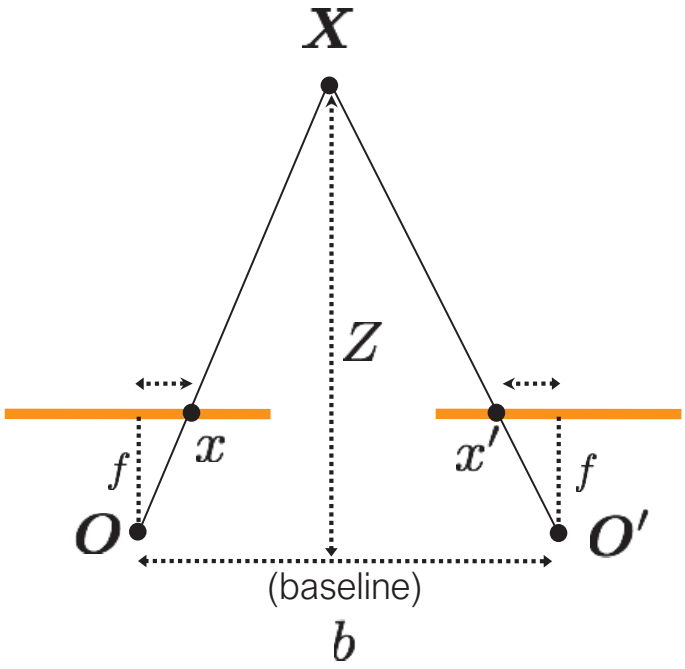
How is X related to x'?

$$\frac{X}{Z} = \frac{x}{f}$$



$$\frac{b - X}{Z} = \frac{x'}{f}$$

$$\frac{X}{Z} = \frac{x}{f}$$



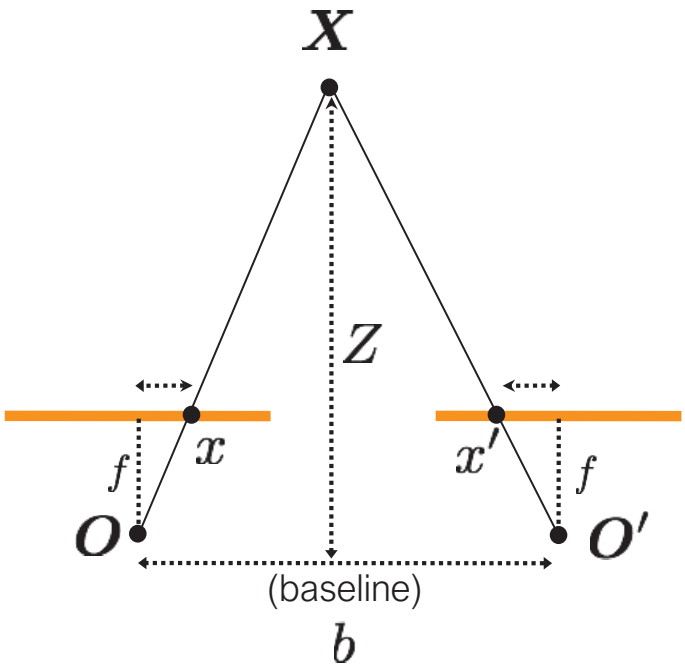
$$\frac{b - X}{Z} = \frac{x'}{f}$$

Disparity

$$d = x - x'$$ (wrt to camera origin of image plane)

$$= \frac{bf}{Z}$$

$$\frac{X}{Z} = \frac{x}{f}$$



$$\frac{b - X}{Z} = \frac{x'}{f}$$

Disparity

$$d = x - x'$$

$$= \frac{bf}{Z}$$

inversely proportional to depth

Real-time stereo sensing

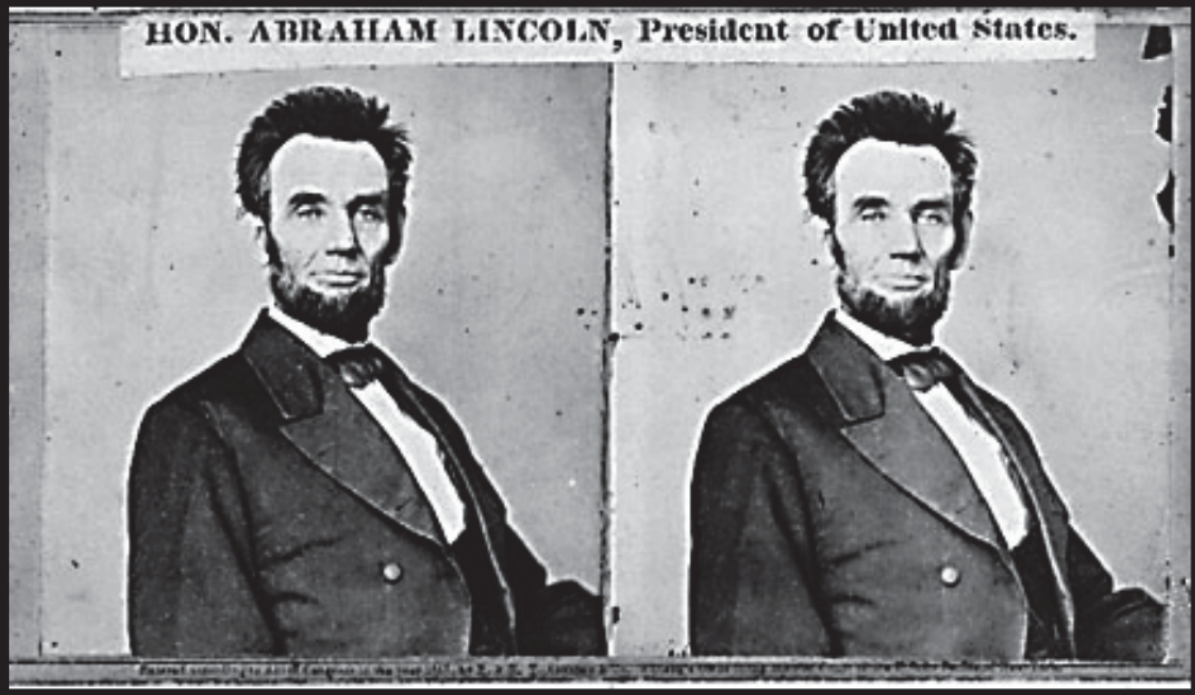
Nomad robot searches for meteorites in Antartica

http://www.frc.ri.cmu.edu/projects/meteorobot/index.html

Pre-collision
braking

Subaru
Eyesight system

Stereoscopes: A 19th Century Pastime

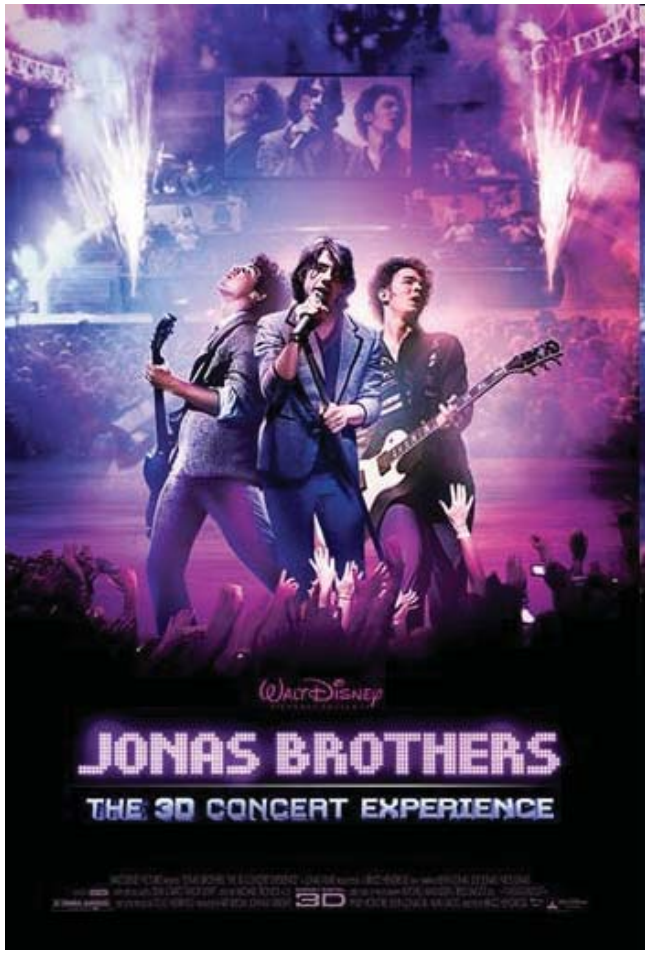

HON. ABRAHAM LINCOLN, President of United States.

Public Library, Stereoscopic Looking Room, Chicago, by Phillips, 1923

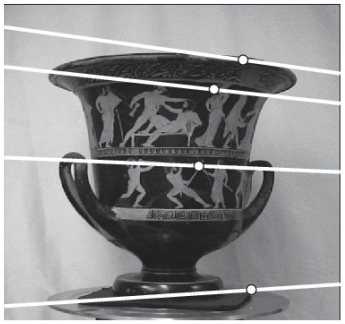
Teesta suspension bridge-Darjeeling, India

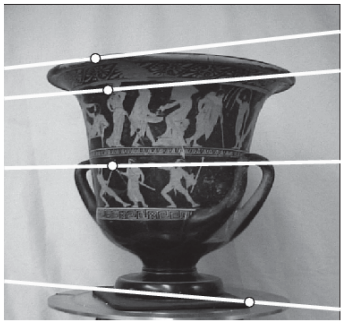
Mark Twain at Pool Table", no date, UCR Museum of Photography

This is how 3D movies work

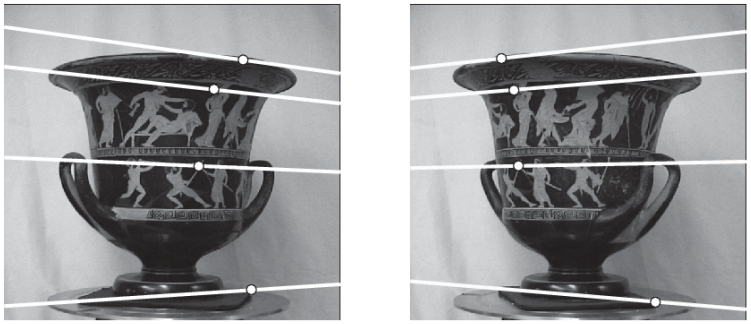


So can I compute depth from any two images of the same object?



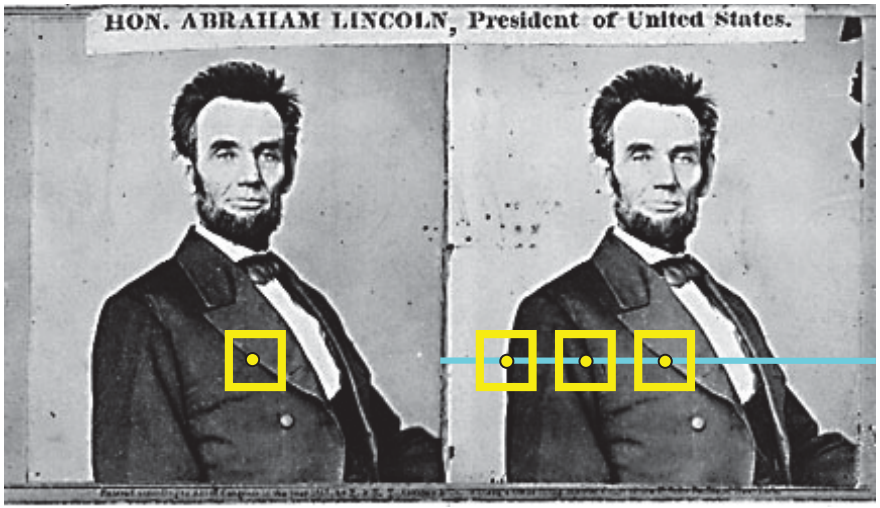


So can I compute depth from any two images of the same object?



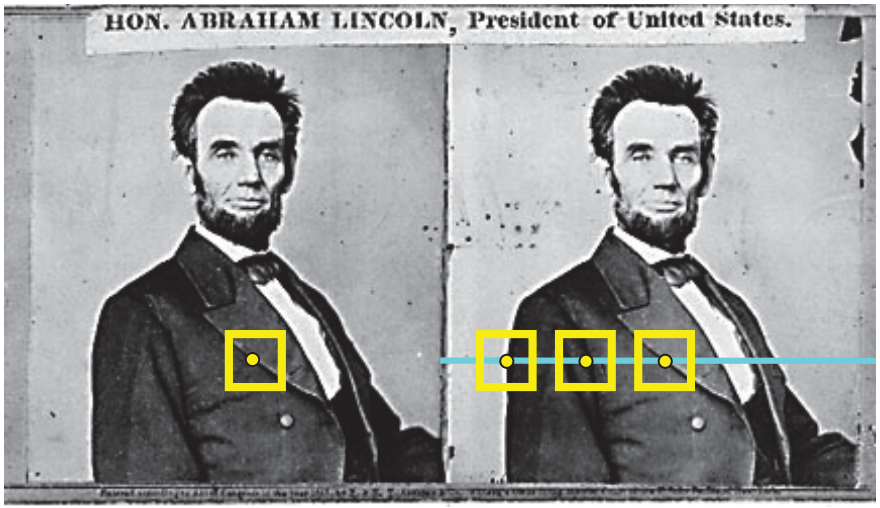
1. Need sufficient baseline

2. Images need to be ‘rectified’ first (make epipolar lines horizontal)

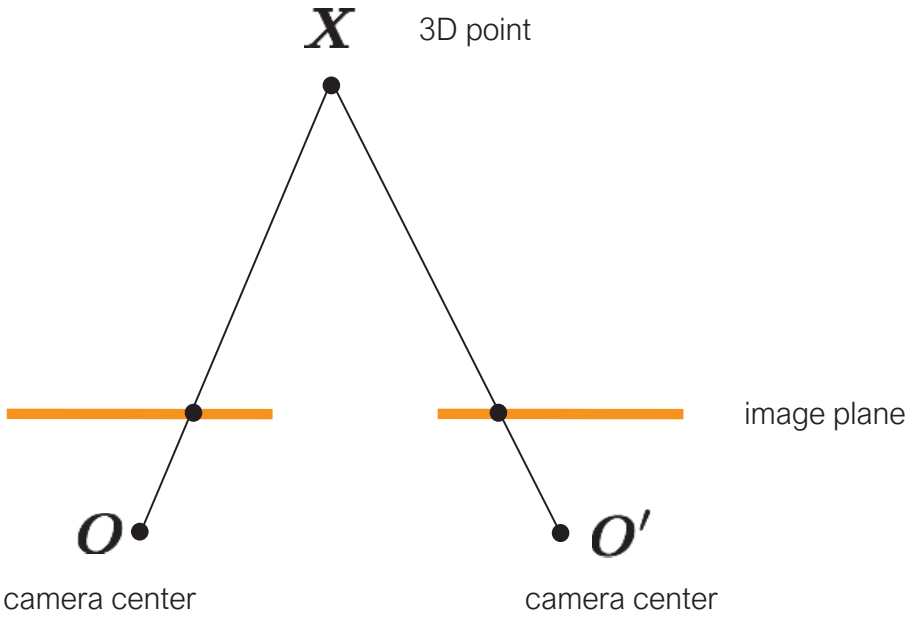


1. Rectify images
(make epipolar lines horizontal)
2. For each pixel
   a. Find epipolar line
   b. Scan line for best match
   c. Compute depth from disparity

$$Z = \frac{bf}{d}$$

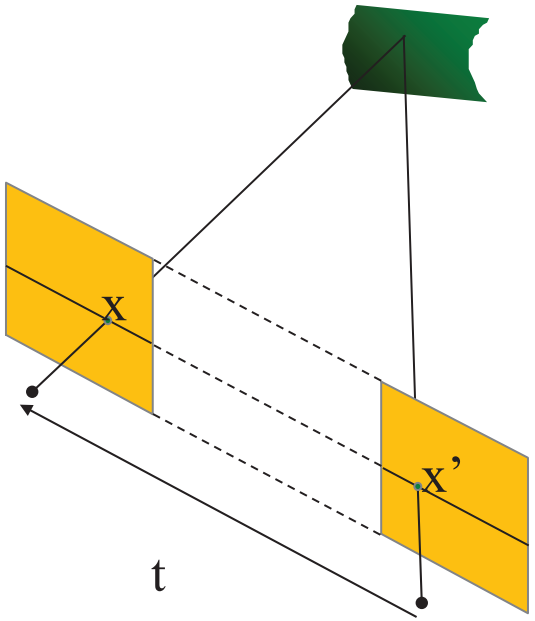


How can you make the epipolar lines horizontal?



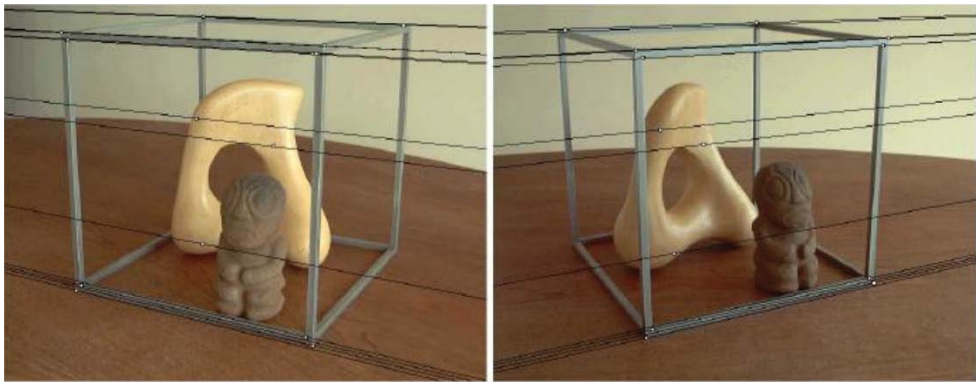
What's special about these two cameras?

When are epipolar lines horizontal?

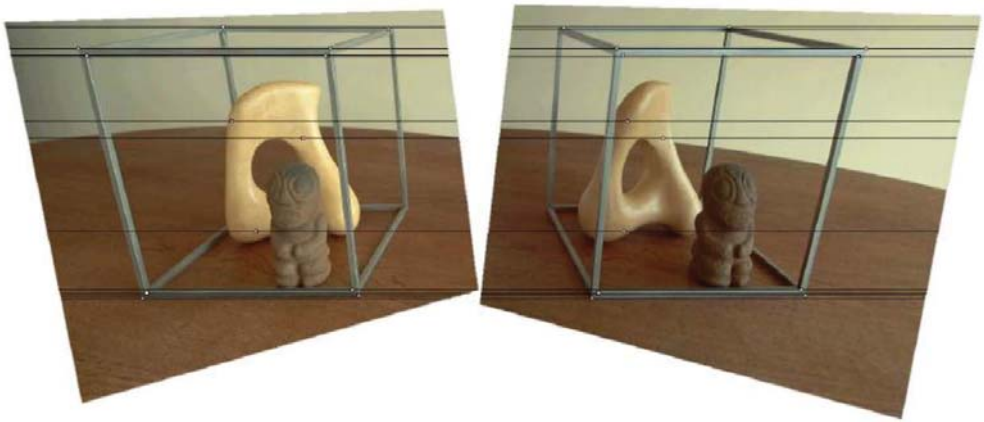


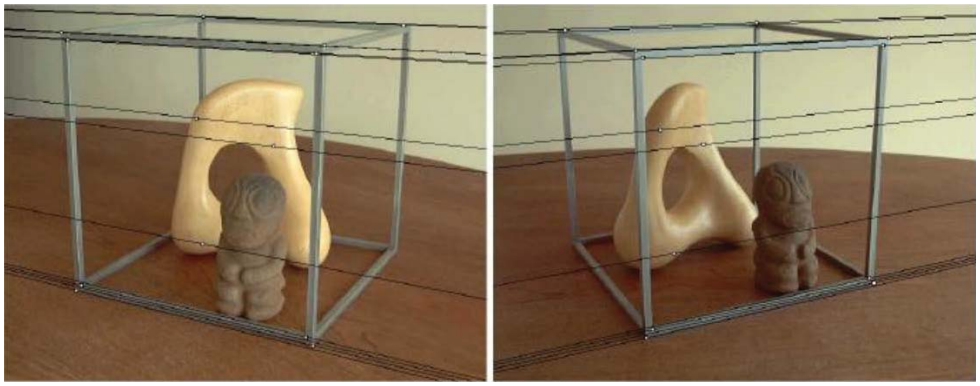
When this relationship holds:

$$R = I \qquad t = (T, 0, 0)$$

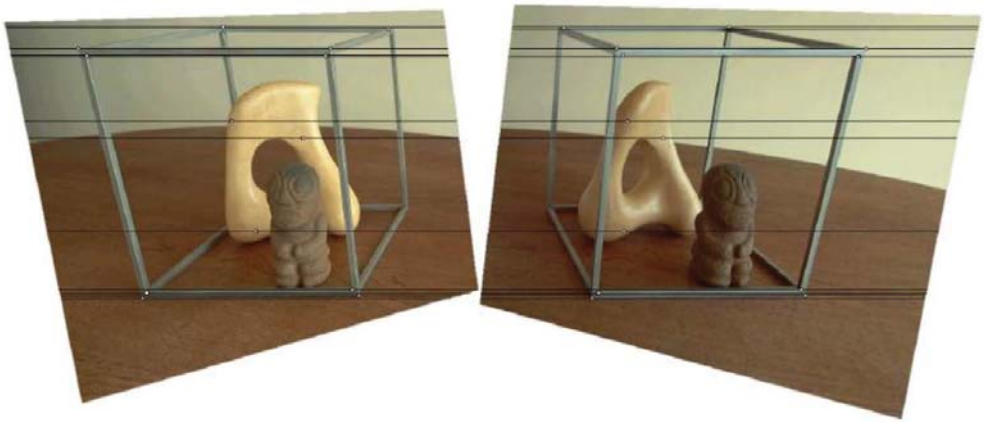


How can you make the epipolar lines horizontal?

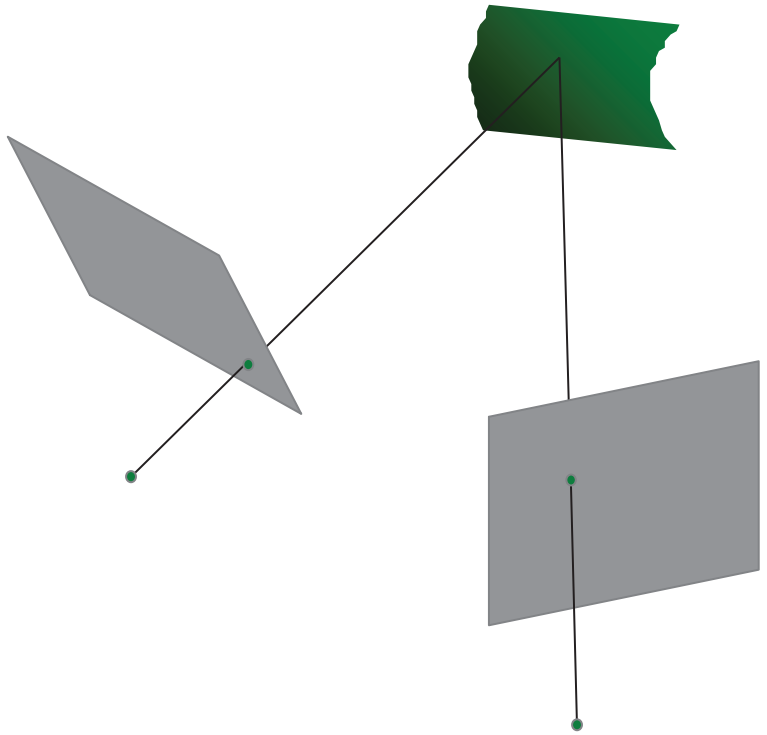




Use stereo rectification?

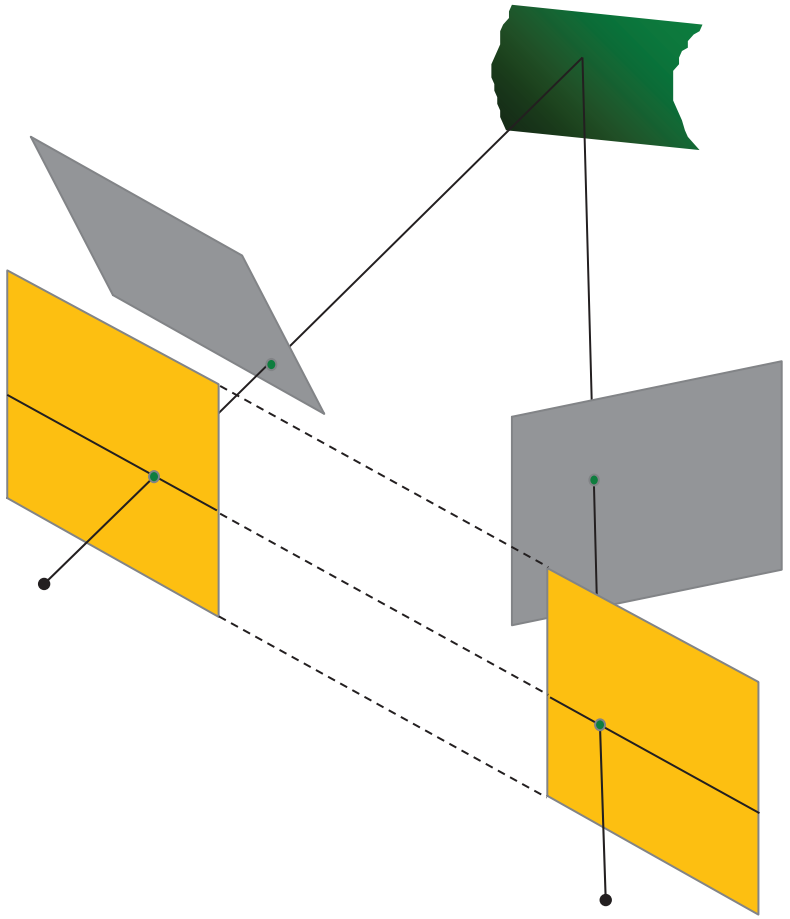


What is stereo rectification?



What is stereo rectification?

Reproject image planes onto a common plane parallel to the line between camera centers

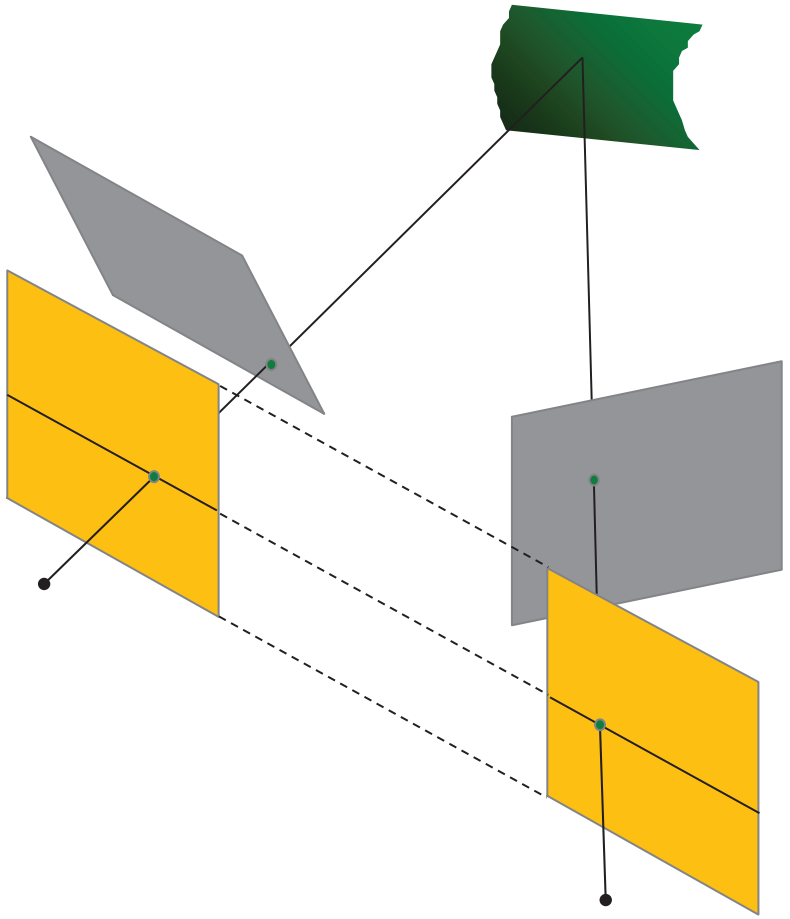


How can you do this?

What is stereo rectification?

Reproject image planes onto a common plane parallel to the line between camera centers

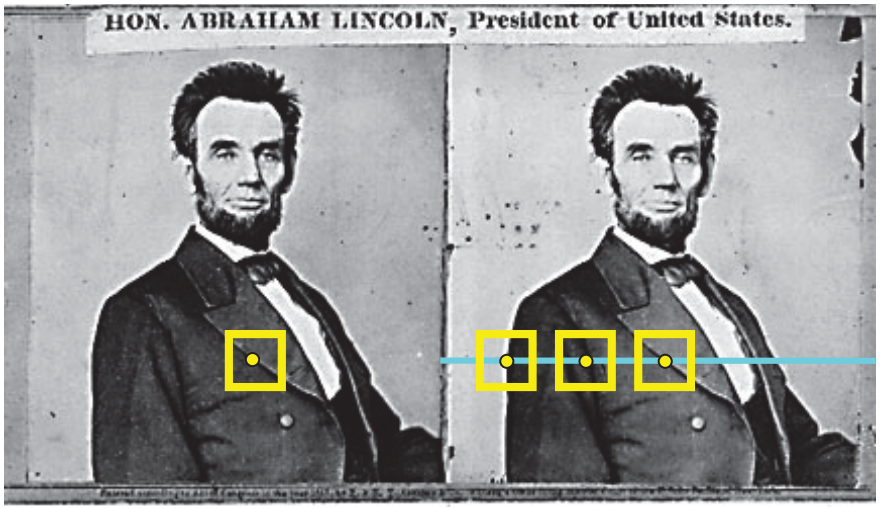
Need two homographies (3x3 transform), one for each input image reprojection



C. Loop and Z. Zhang. Computing Rectifying Homographies for Stereo Vision.Computer Vision and Pattern Recognition, 1999.

Stereo matching

Depth Estimation via Stereo Matching

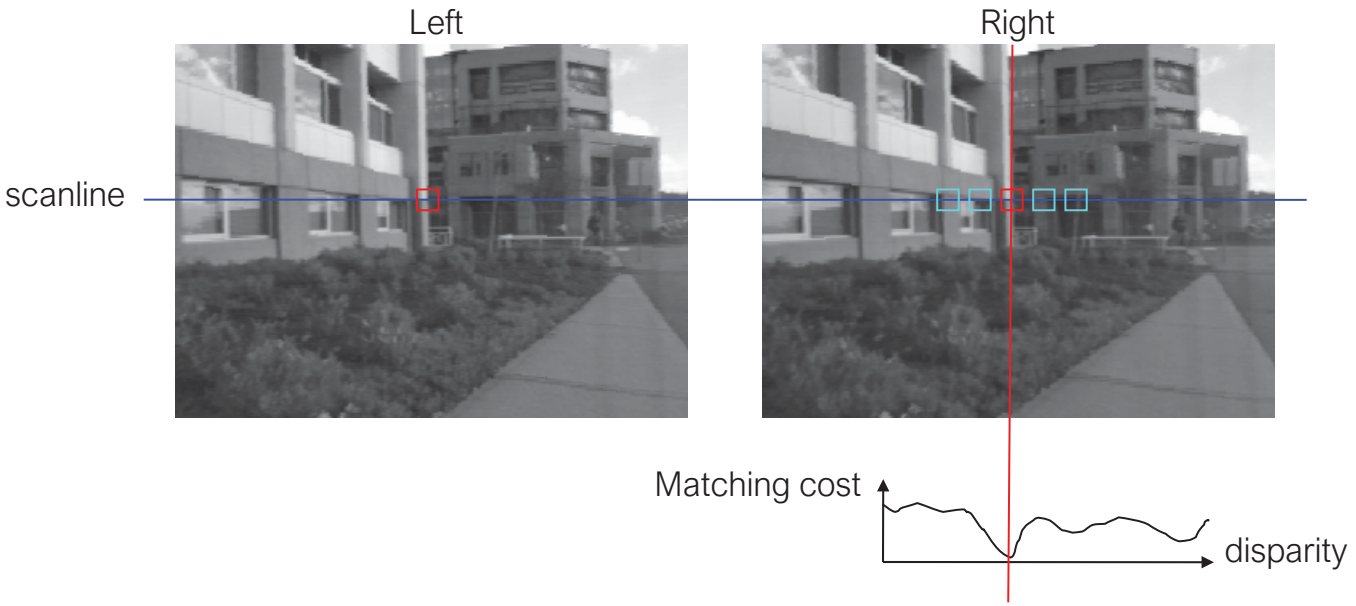


1. Rectify images
(make epipolar lines horizontal)
2. For each pixel
   a. Find epipolar line
   b. Scan line for best match
   c. Compute depth from disparity

How would you do this?

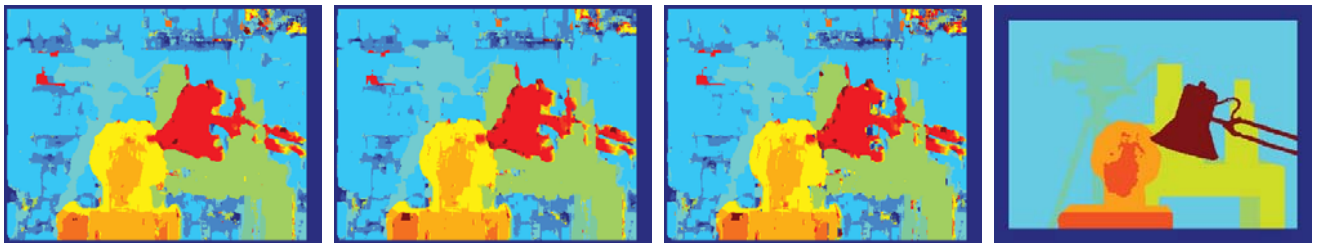
$$Z = \frac{bf}{d}$$

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

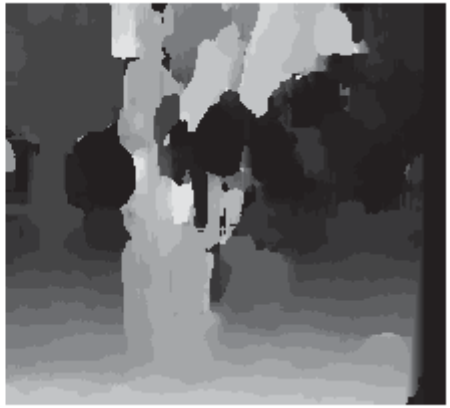
Similarity Measure	Formula
Sum of Absolute Differences (SAD)	$\sum_{(i,j)\in W} \lvert I_1(i,j) - I_2(x+i,y+j)\rvert$
Sum of Squared Differences (SSD)	$\sum_{(i,j)\in W} \left(I_1(i,j) - I_2(x+i,y+j)\right)^2$
Zero-mean SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \bar{I}_1(i,j) - I_2(x+i,y+j) + \bar{I}_2(x+i,y+j)\rvert$
Locally scaled SAD	$\sum_{(i,j)\in W} \lvert I_1(i,j) - \frac{\bar{I}_1(i,j)}{\bar{I}_2(x+i,y+j)} I_2(x+i,y+j)\rvert$
Normalized Cross Correlation (NCC)	$\frac{\sum_{(i,j)\in W} I_1(i,j).I_2(x+i,y+j)}{\sqrt{\sum_{(i,j)\in W} I_1^2(i,j).\sum_{(i,j)\in W} I_2^2(x+i,y+j)}}$



SAD SSD NCC Ground truth

Effect of window size

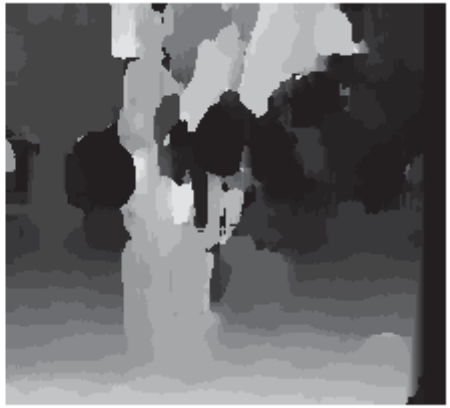
W = 3



W = 20

Effect of window size

W = 3



W = 20

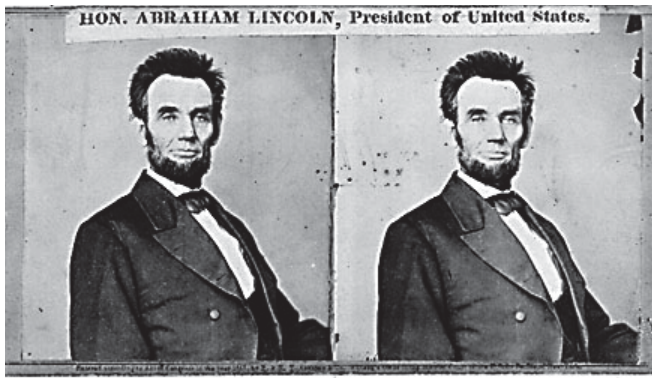
Smaller window

+ More detail
- More noise

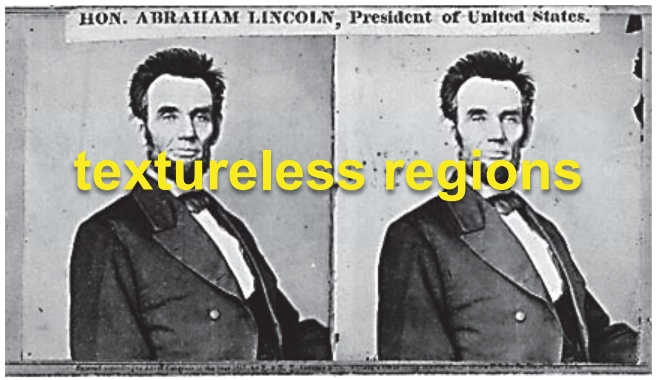
Larger window

+ Smoother disparity maps
- Less detail
- Fails near boundaries

When will stereo block matching fail?

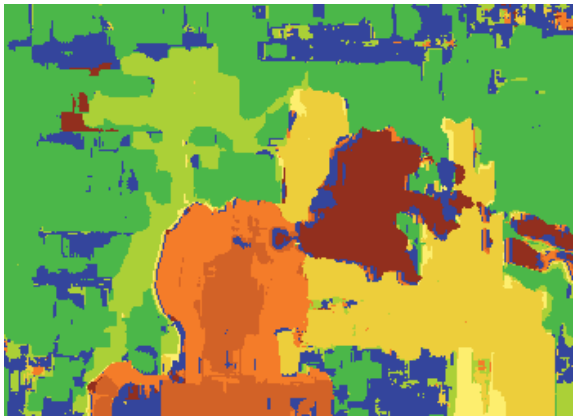


When will stereo block matching fail?



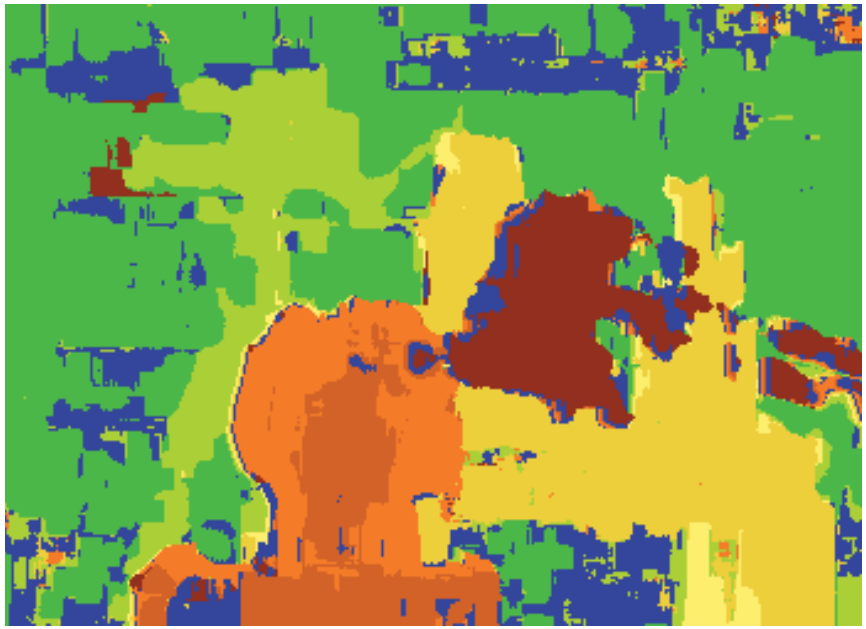
Improving stereo matching

Block matching

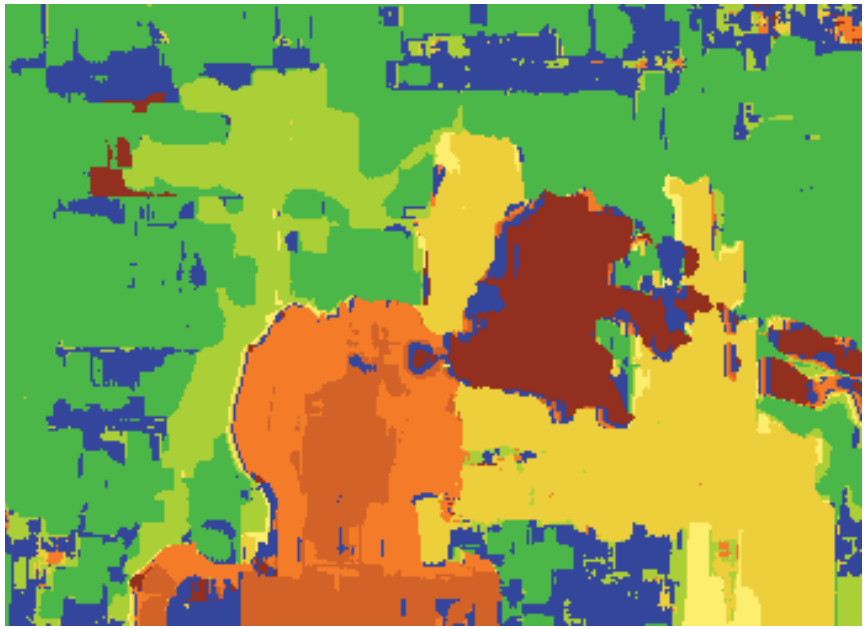


Ground truth

What are some problems with the result?



How can we improve depth estimation?



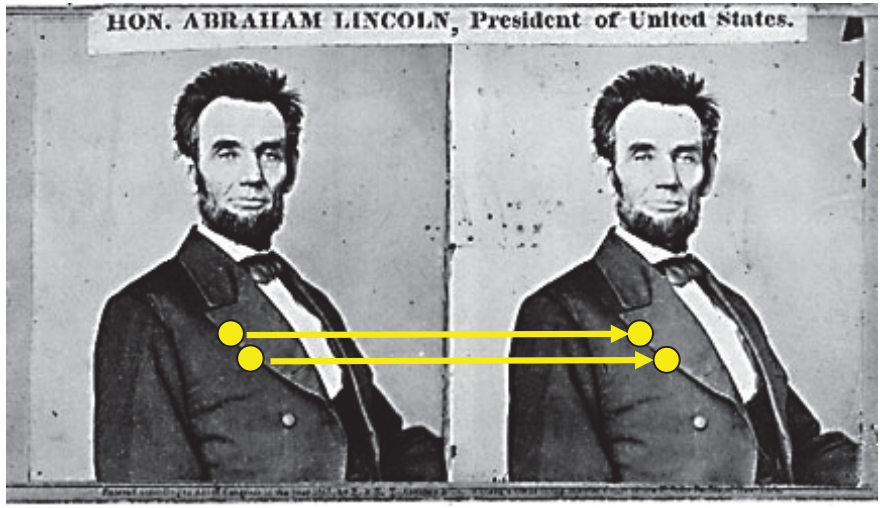
How can we improve depth estimation?

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

Let's make an assumption:
depth should change smoothly

Stereo matching as …

Energy Minimization



What defines a good stereo correspondence?

1. **Match quality**
   - Want each pixel to find a good match in the other image
2. **Smoothness**
   - If two pixels are adjacent, they should (usually) move about the same amount

energy function
(for one pixel)

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

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

(block matching result)

Adjacent pixels should (usually) move about the same amount

(smoothness function)

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

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

data term

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

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

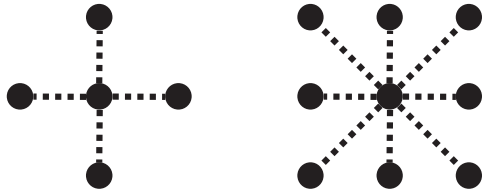
$$E_d(d) = \sum_{(x,y)\in I} C(x, y, d(x, y))$$

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

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

smoothness term

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



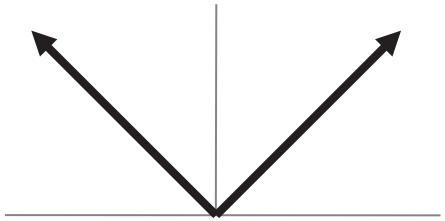
4-connected neighborhood

8-connected neighborhood

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

smoothness term

$$V(d_p, d_q) = aff(I_p, I_q)|d_p - d_q|$$



L_1 distance

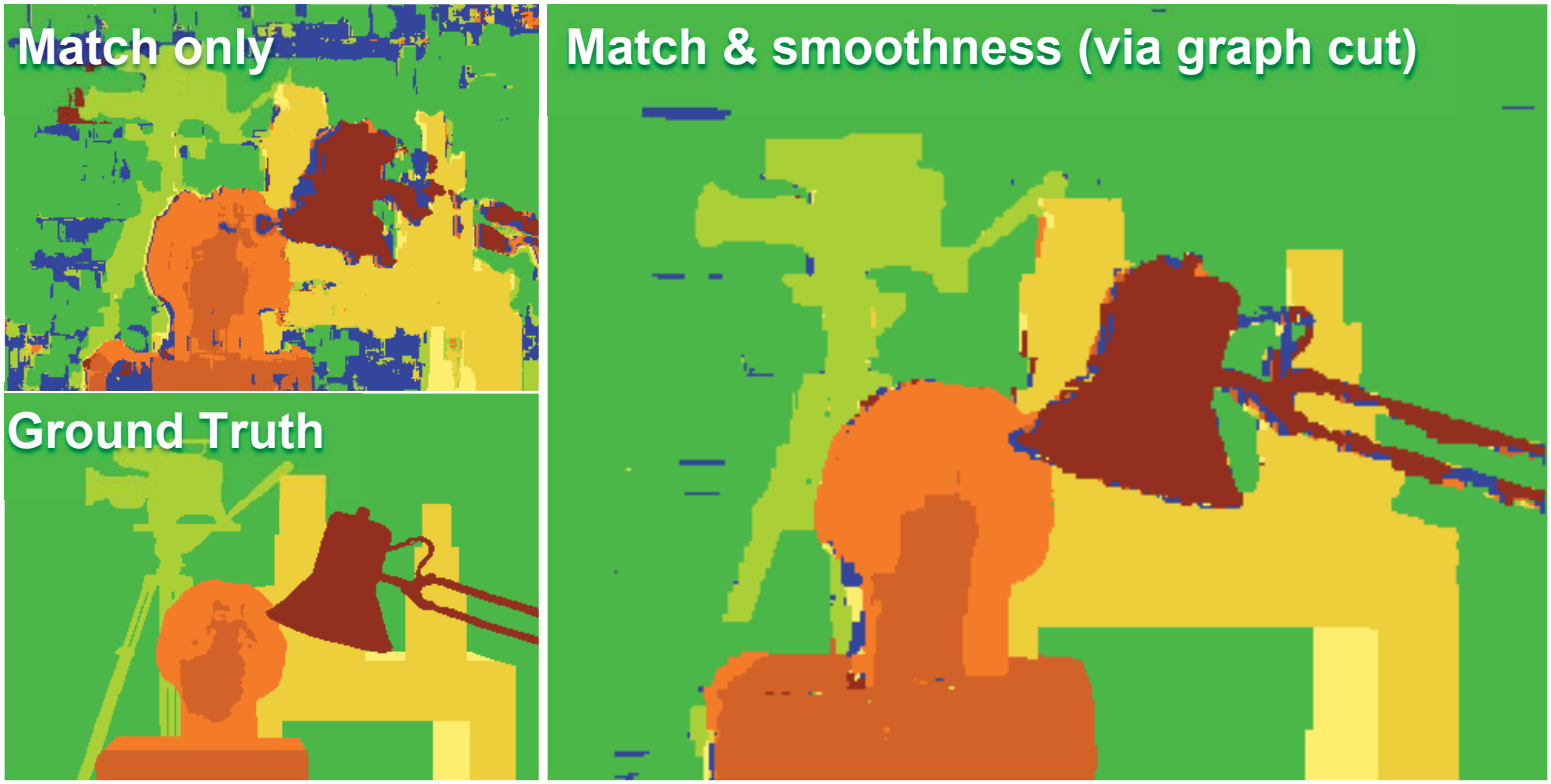
$$aff(I_p, I_q) = e^{\frac{-|I_p - I_q|^2}{\sigma}}$$

Penalty for discontinuities is lower if edge in the image exists

Energy minimization

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

Can minimize this using the graph cut framework presented for segmentation



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

Summary: 3D geometric vision

- Single-view geometry
  - The pinhole camera model
    - Variation: orthographic projection
  - The perspective projection matrix
  - Intrinsic parameters
  - Extrinsic parameters
  - Calibration
- Multiple-view geometry
  - Triangulation
  - The epipolar constraint
    - Essential matrix and fundamental matrix
  - Stereo
    - Binocular, multi-view
  - Structure from motion
    - Reconstruction ambiguity
    - Affine SFM
    - Projective SFM

References

Basic reading:

- Szeliski textbook, Chapter 7.
- Hartley and Zisserman, Chapter 9,18.