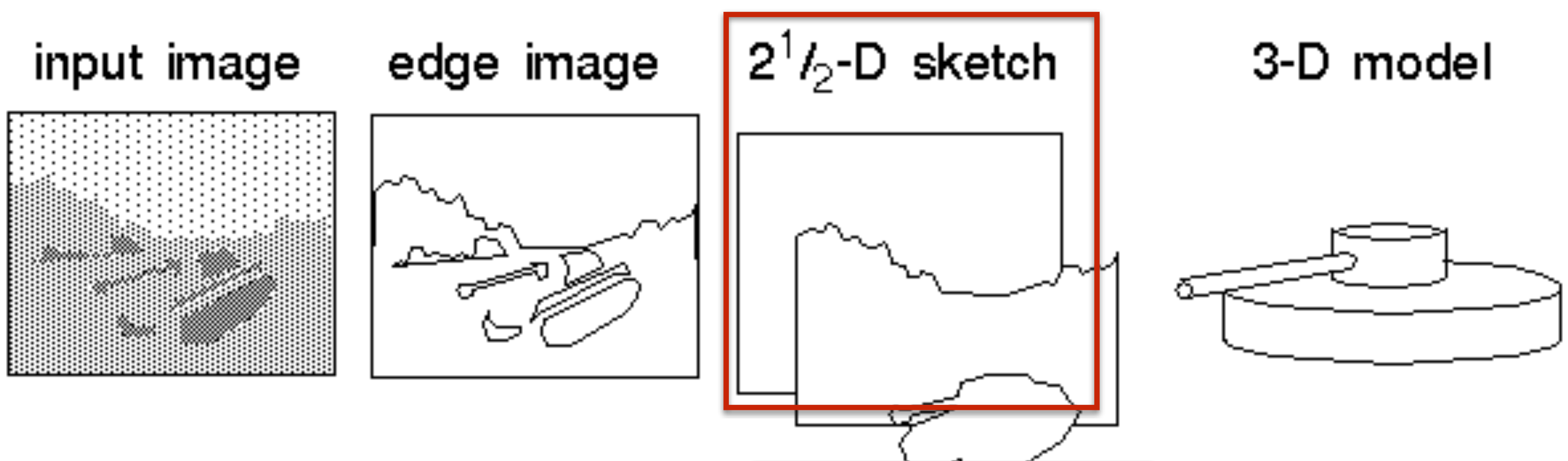
Two-view geometry

David Marr’s Taxonomy of Vision

	Image Processing + Correspondence	Motion + Geometry	Recognition
	Viewer centred		*Object centred*
Input Image	**Primal Sketch**	**2 1/2-D Sketch**	**3-D Model Representation**
Perceived intensities	Zero crossings, blobs,edges, bars, ends, virtual lines, groups, curves boundaries.	Local surface orientation and discontinuities in depth and in surface orientation	3-D models hierarchically organised in terms of surface and volumetric primitives



Low-level Mid-level High-level

Brief Review from Last Time

Estimating Optical Flow

Brightness constancy equation: $\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \frac{\partial I}{\partial t} = 0$

image gradient / derivative

pixel velocity ("flow")

change in pixel brightness over time keeping (x,y) fixed

Aperture problem

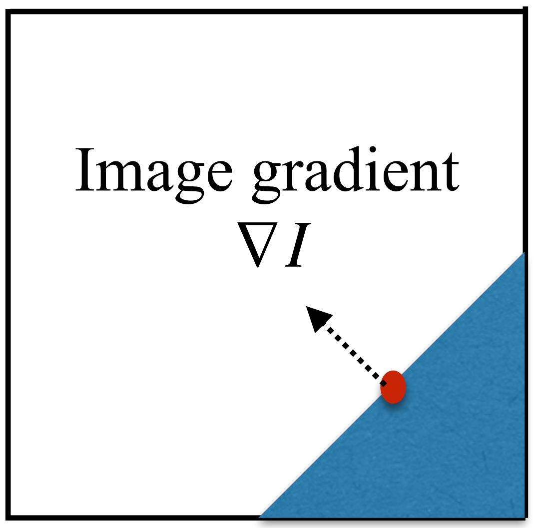
$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u_\perp \\ v_\perp \end{bmatrix} + \begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix}$$

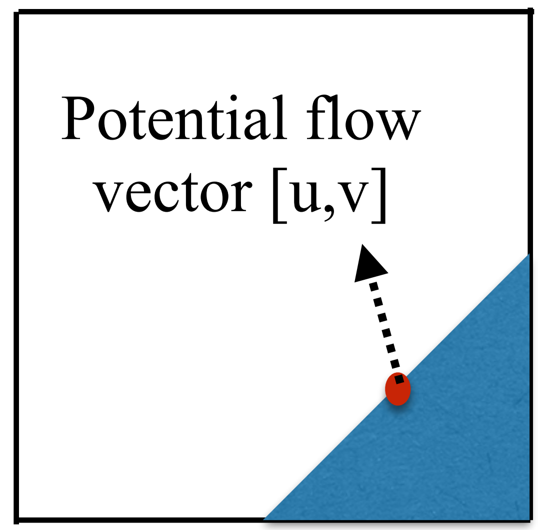
Decompose flow vector into parallel and perpendicular components:

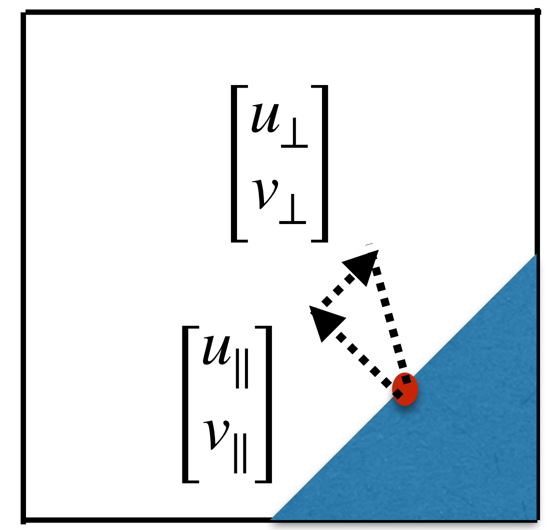
pixel velocity ("flow")

flow component perpendicular to image gradient

flow component parallel to image gradient







What happens when we compute the brightness constancy equation?

$$\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} + \frac{\partial I}{\partial t} = 0$$

$$\nabla I \cdot \begin{bmatrix} u \\ v \end{bmatrix} = \cancel{\nabla I \cdot \begin{bmatrix} u_\perp \\ v_\perp \end{bmatrix}} + \nabla I \cdot \begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix} \qquad = \nabla I \cdot \begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix}$$

We can only estimate the component of the flow that is parallel to the image gradient! $\begin{bmatrix} u_\parallel \\ v_\parallel \end{bmatrix}$

We <u>cannot</u> estimate the component of the flow that is <u>perpendicular</u> to the image gradient

Today: Multi-view geometry

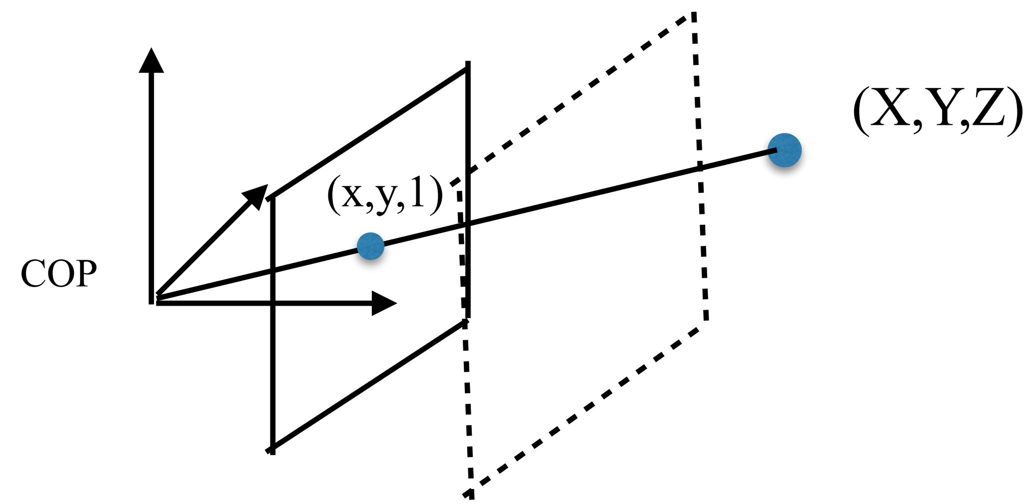
Three questions:

(i) **Correspondence geometry:** Given two camera views and an image point x in one image, what points x' can this correspond to in the second image (geometrically)?

(ii) **Camera geometry (motion):** Given a set of corresponding image points $\{x_i \leftrightarrow x'_i\}$, i=1,…,n across two camera views, what are the camera poses P and P' for the two views?

(iii) **Scene geometry (structure):** Given corresponding image points $x_i \leftrightarrow x'_i$ and camera poses P, P', what is the position of the 3D location X_i of these points?

Recall perspective projection



Assume *calibrated* camera with known intrinsics

We can rescale the image to be the image that would have been taken if f = 1

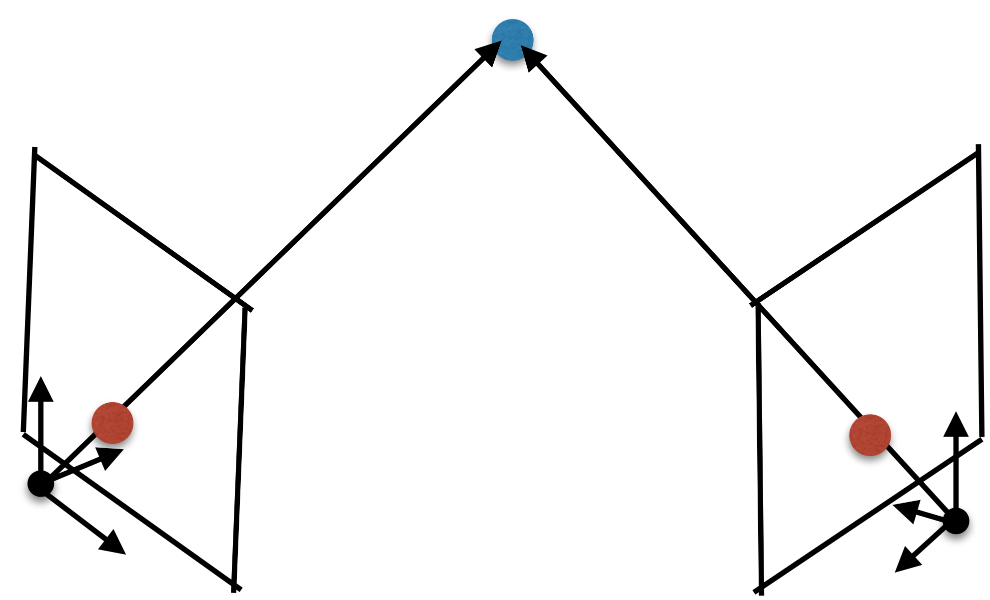
Then, let's assume that f = 1

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

(Note: λ = Z)

Two-view (stereo)

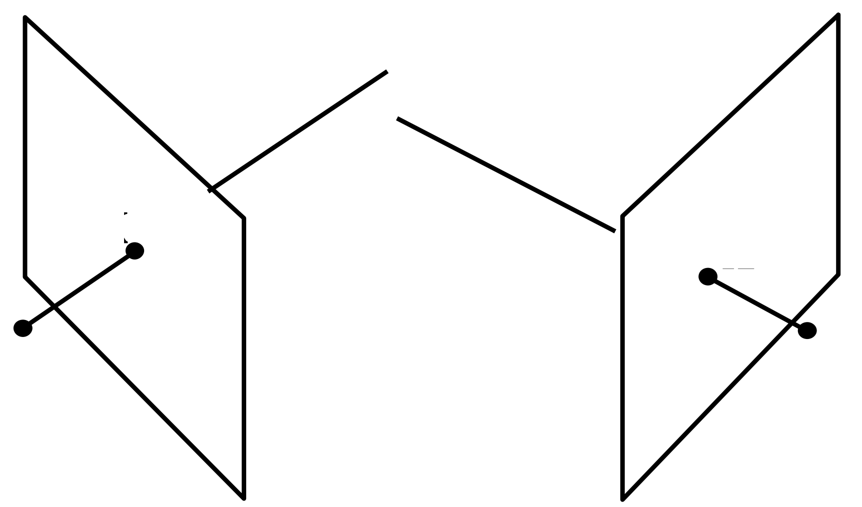
Given 2 corresponding points in 2 cameras, how do we get the 3D location of this point?



See where the cast rays meet

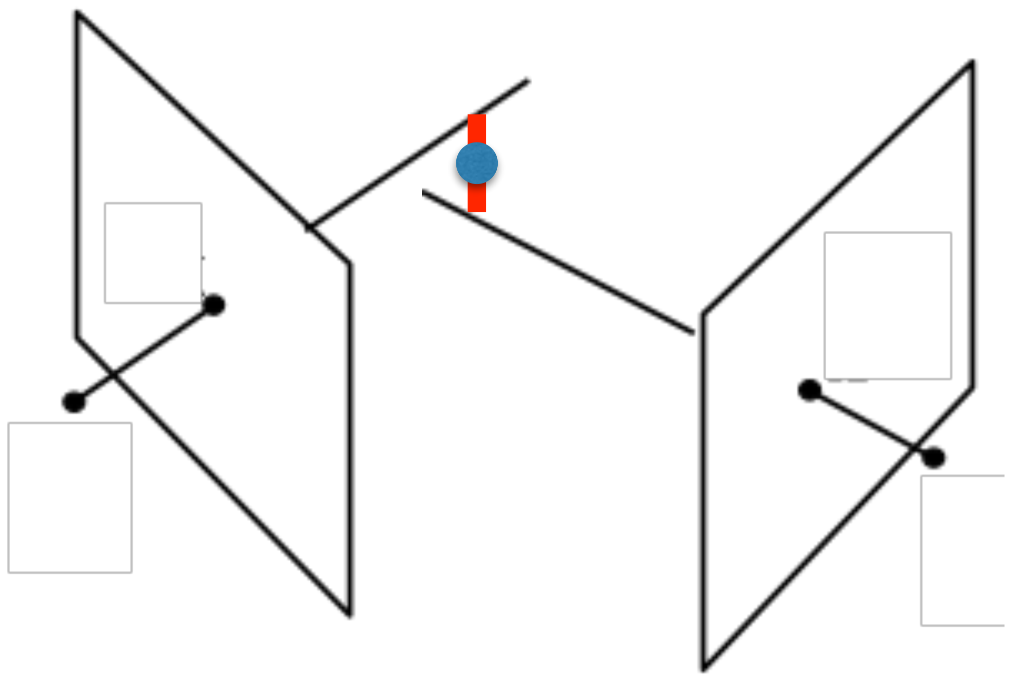
Much of the basics of stereo geometry can be derived from this picture

An annoying "detail"

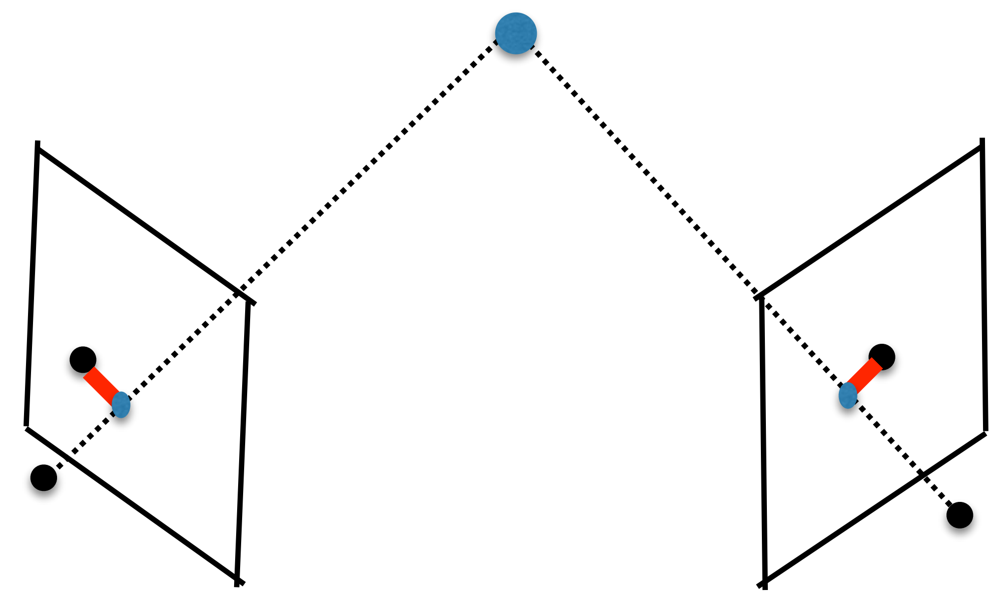


What to do when ray's don't intersect?

Possible solutions



Find 3D point closest to 2 rays

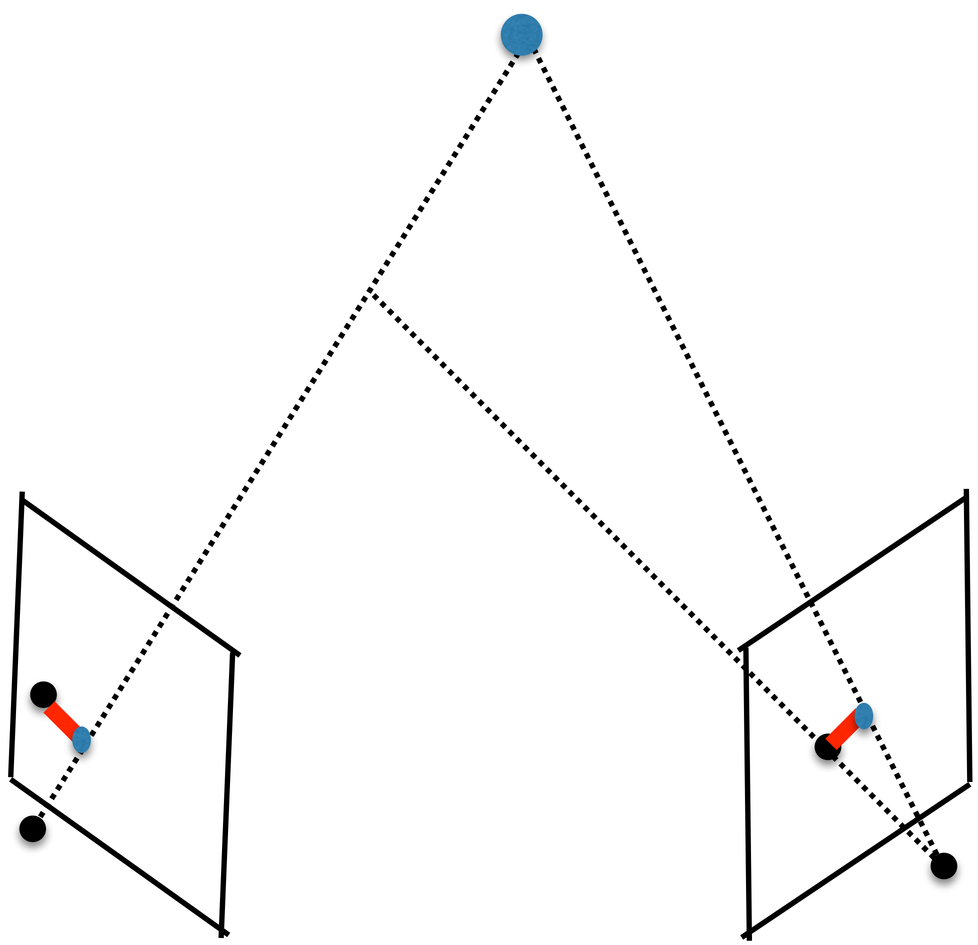


Find 3D point with low reprojection error

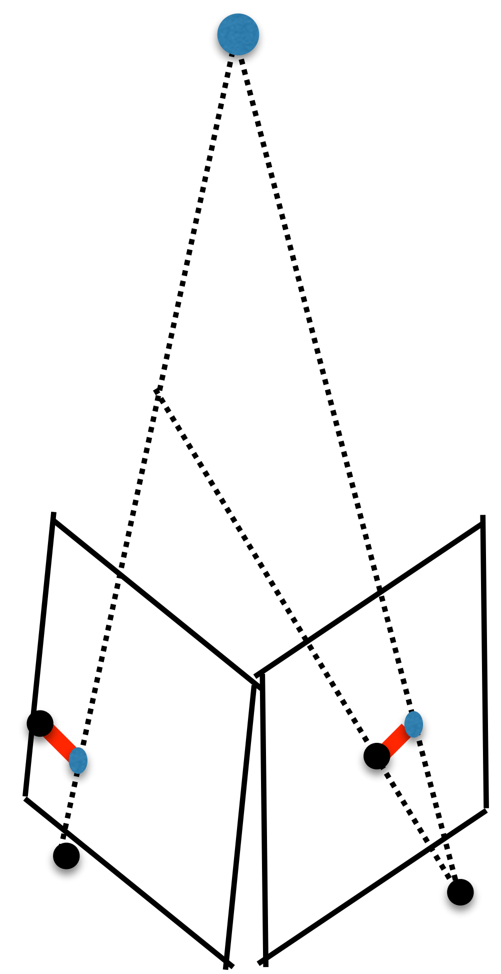
Which makes more sense?

(Maximum likelihood estimation, under a model of Gaussian noise on the projected points)

Numerical stability



Cameras are far apart



Cameras are close together

Which setup produces more accurate estimates of depth (as a function of image noise)?

large baselines (left)

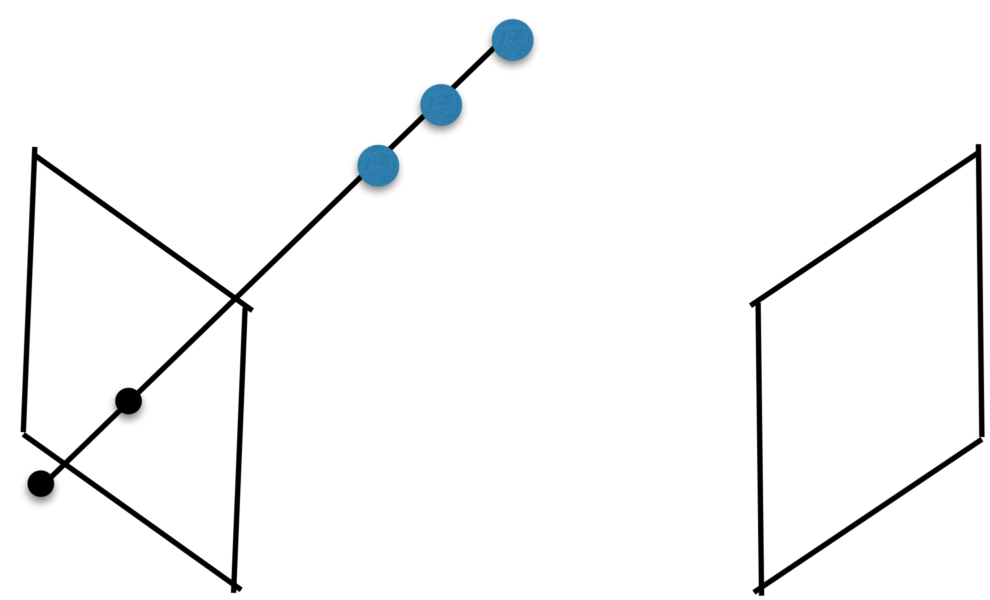
Which setup will produce points that are easier to match?

small baselines (right)

Multi-view geometry

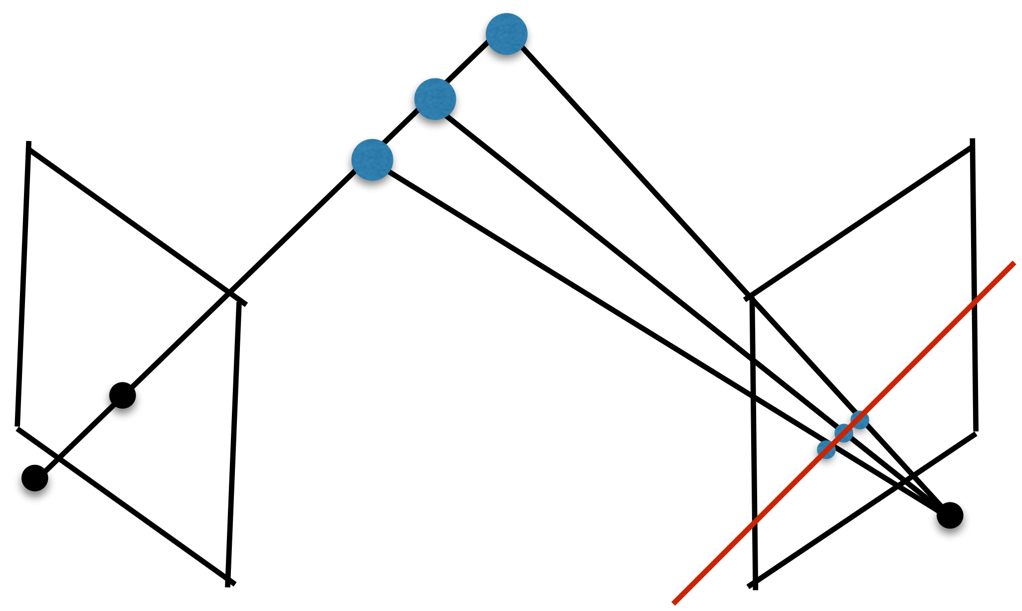
Questions

Given a point in the left view image, what is the set of points it could project to in the right view?



Questions

Given a point in the left view image, what is the set of points it could project to in the right view?



This is called the "epipolar line"

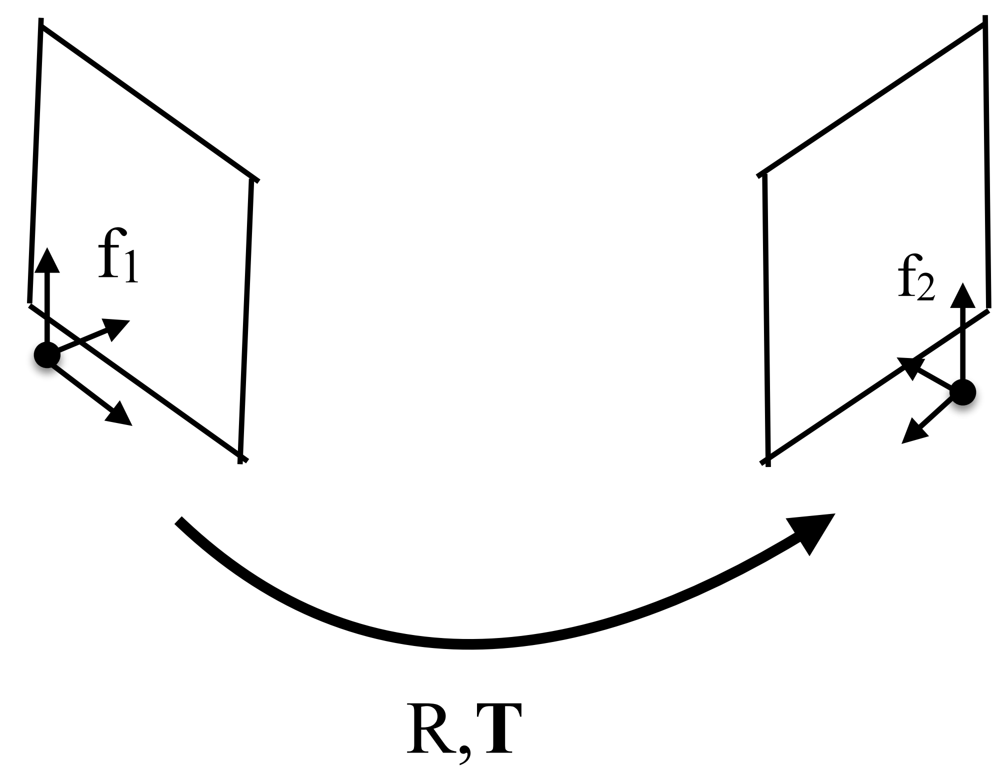
Implies that *for known camera geometry,* we need search for correspondence only over 1D line

Given known camera translation, "epipolar geometry" will help us find that line!

f(point) = line

Does the function f need to know the 3D scene geometry? **No!**

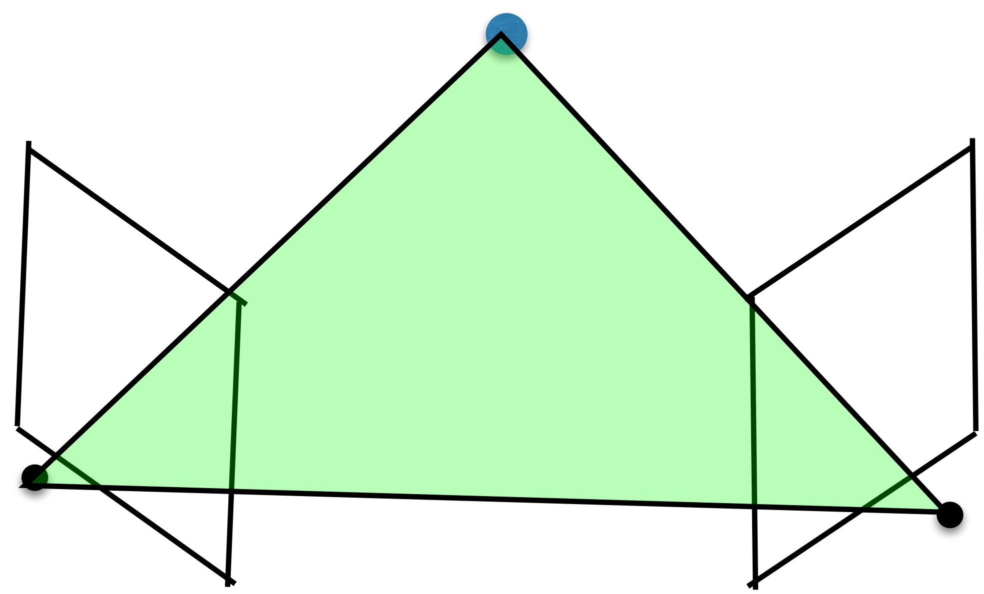
Epipolar geometry



Epipolar geometry describes the set of candidate correspondences across 2 views as a function of camera extrinsics (R,**T**) and intrinsics (f)

Epipolar geometry is *not* a function of the 3D scene

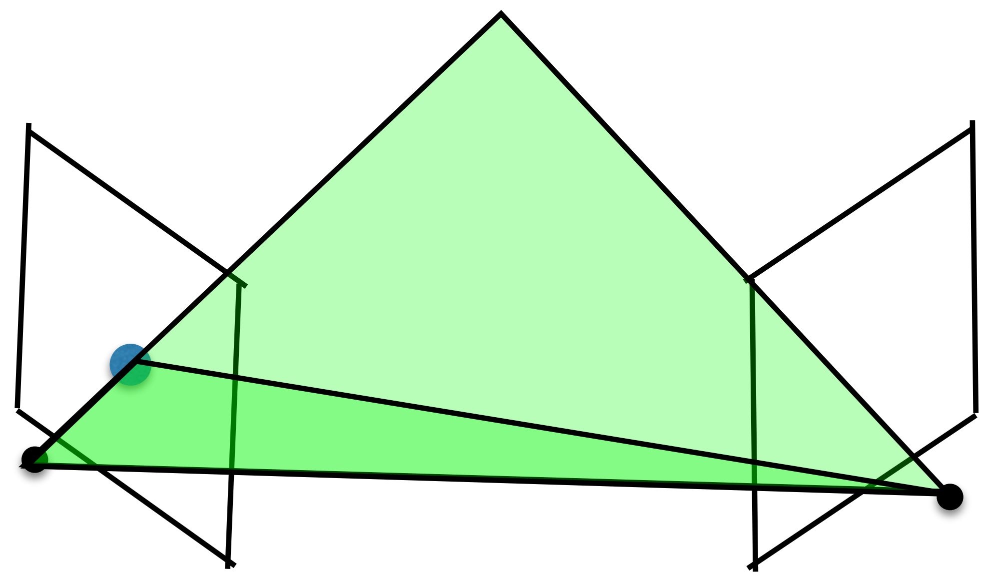
Definitions



Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

…but didn't we just state that epipolar geometry doesn't depend on the 3D scene?

Definitions



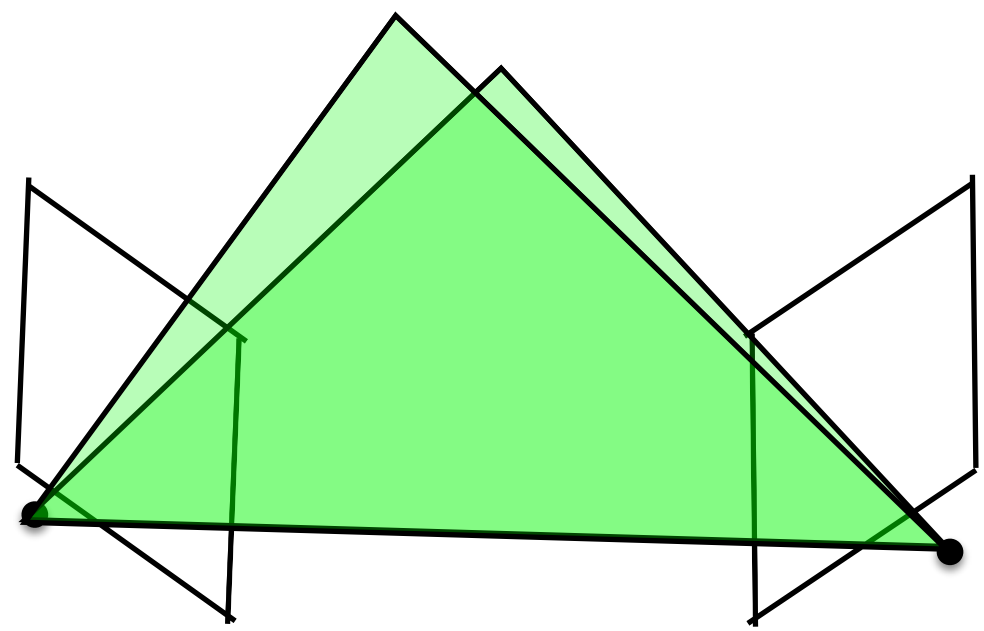
Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

(formally defined by 2 camera centers and any 1 point in either image plane;
for convenience, we'll draw the triangle connecting to a 3D point)

How does the epipolar plane change when we double the distance between the two cameras?

Doesn't change

Definitions

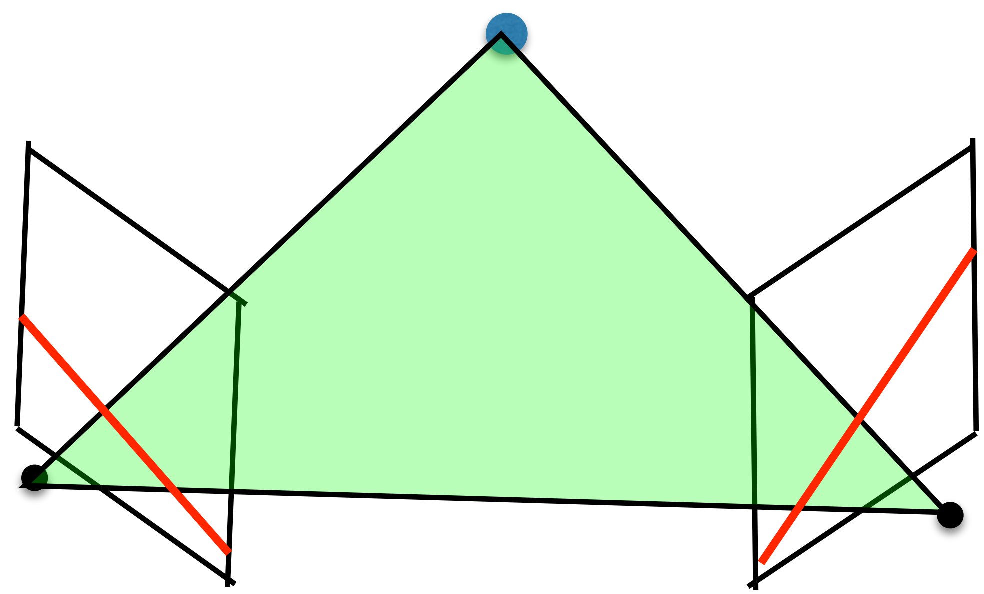


Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

How large is the *family* of epipolar planes? (given fixed camera centers). How many degrees of freedom are there?

1 DOF (epipolar planes *hinged* at 2 camera centers)

Definitions

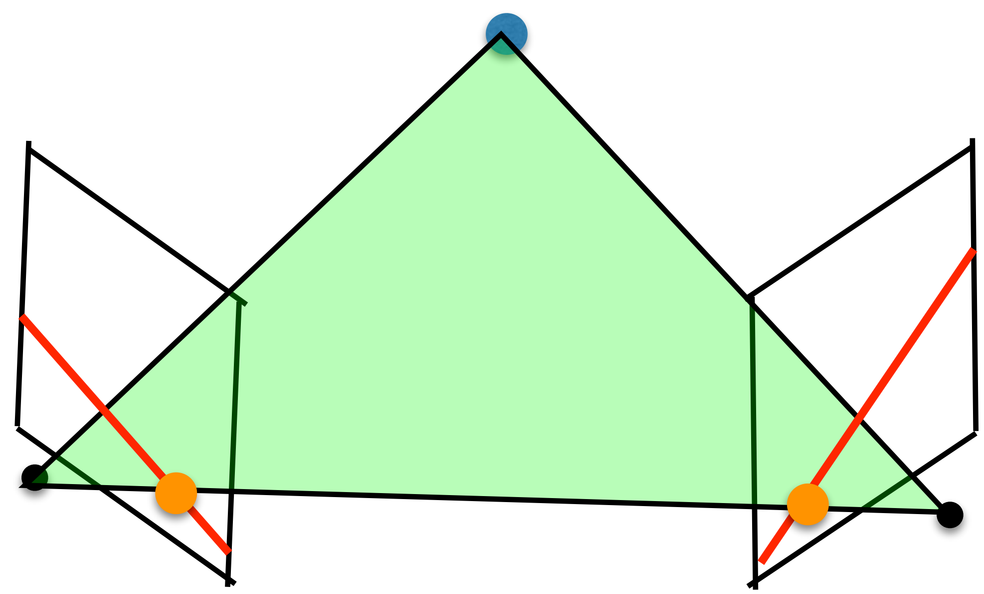


Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

(also defined by 2 camera centers any 1 points in either image plane)

Epipolar lines (red): intersection of epipolar plane and image planes

Definitions



Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

(also defined by 2 camera centers any 1 points in either image plane)

Epipolar lines (red): intersection of epipolar plane and image planes

Epipoles (orange): projection of camera center 1 in camera 2 (& vice versa)

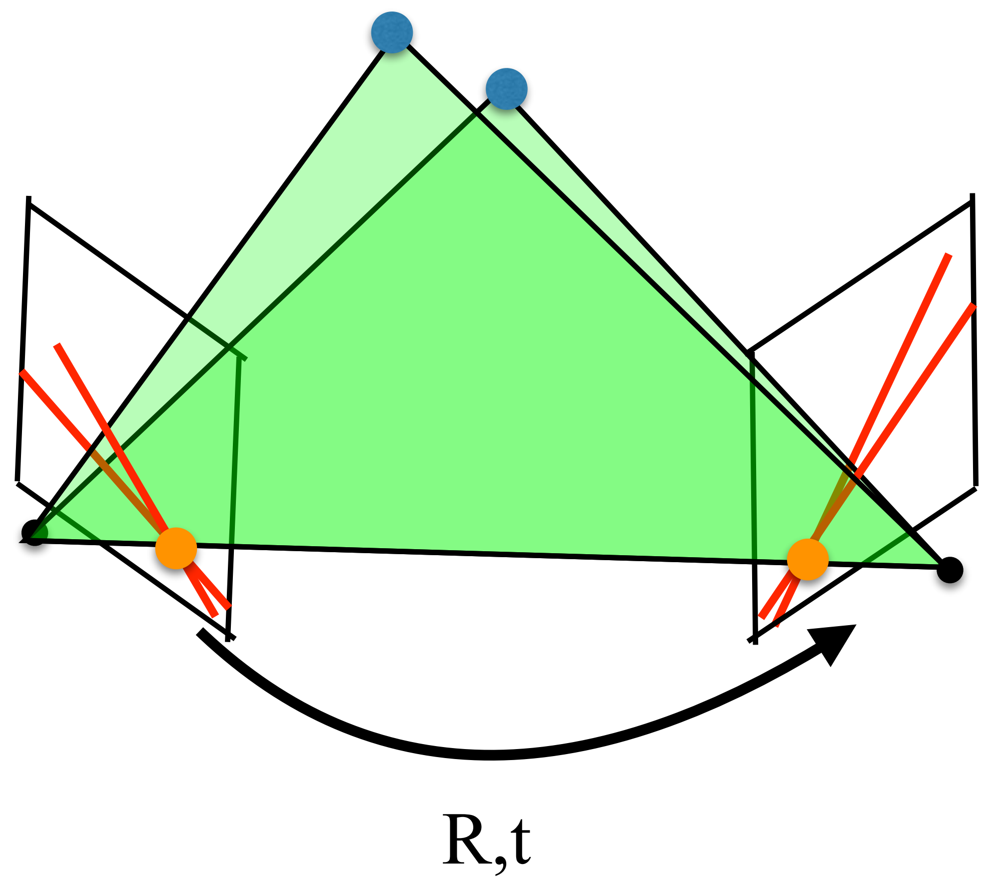
All epipolar lines include the epipole; **the set of all epipolar lines intersect at the epipoles**

What happens if we scale the translation vector?

epipolar geometry doesn’t depend on camera translation

Question: do the epipolar lines depend on scene structure, cameras, or both?

Epipolar geometry is purely determined by camera extrinsics and camera instrinics



Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

(also defined by 2 camera centers any 1 points in either image plane)

Epipolar lines (red): intersection of epipolar plane and image planes

Epipoles (orange): projection of camera center 1 in camera 2 (& vice versa)

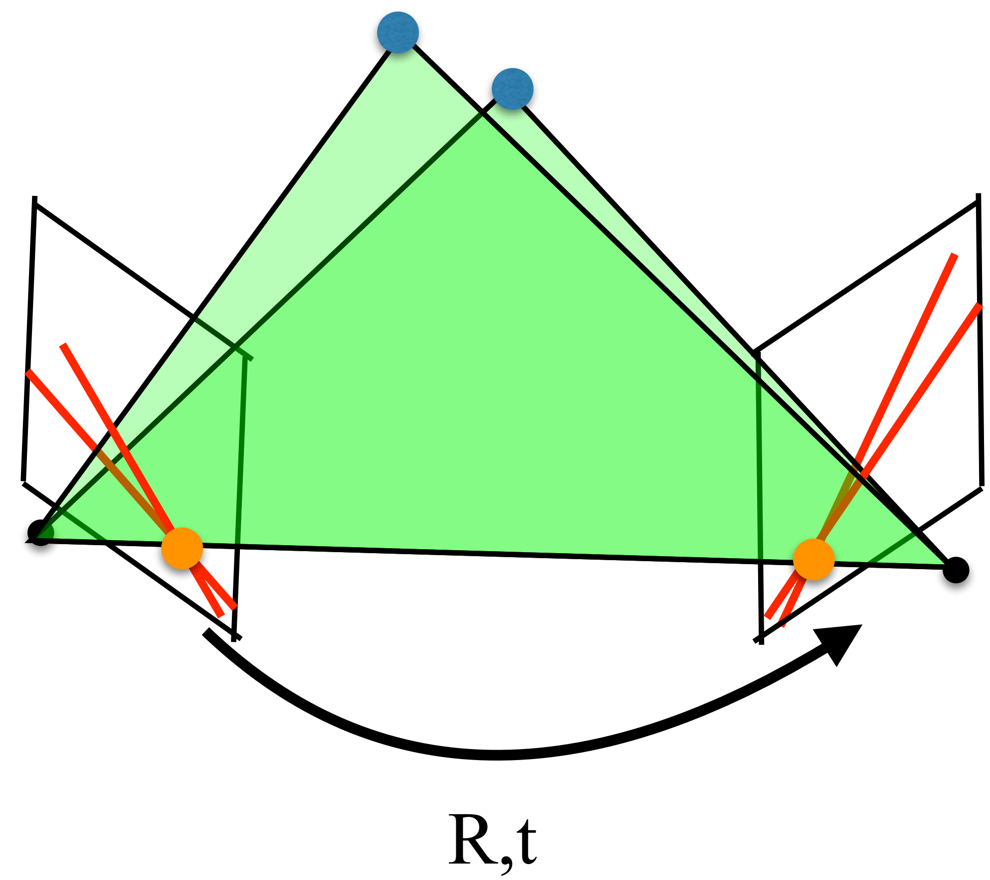
All epipolar lines include the epipole; **the set of all epipolar lines intersect at the epipoles**

Special case (I)

What happens if we don't rotate the cameras (R=I), only translate horizontally?

Where is the epipole? **At infinity (imagine slowly rotating one camera)**

What do the epipolar lines look like?



Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

(also defined by 2 camera centers any 1 points in either image plane)

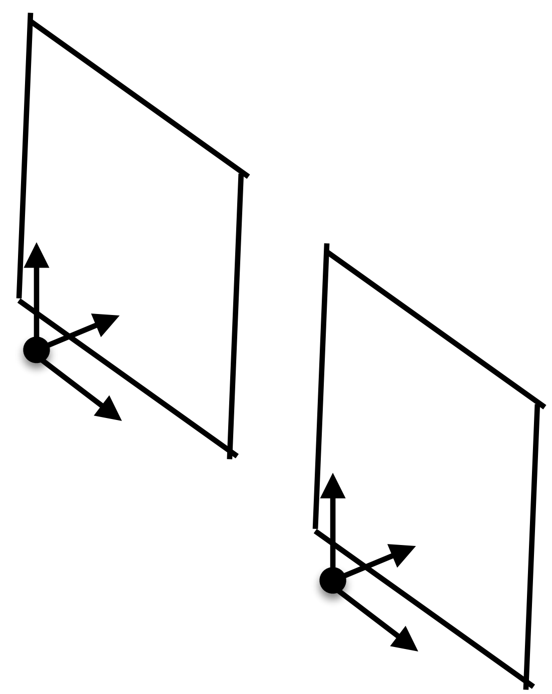
Epipolar lines (red): intersection of epipolar plane and image planes

Epipoles (orange): projection of camera center 1 in camera 2 (& vice versa)

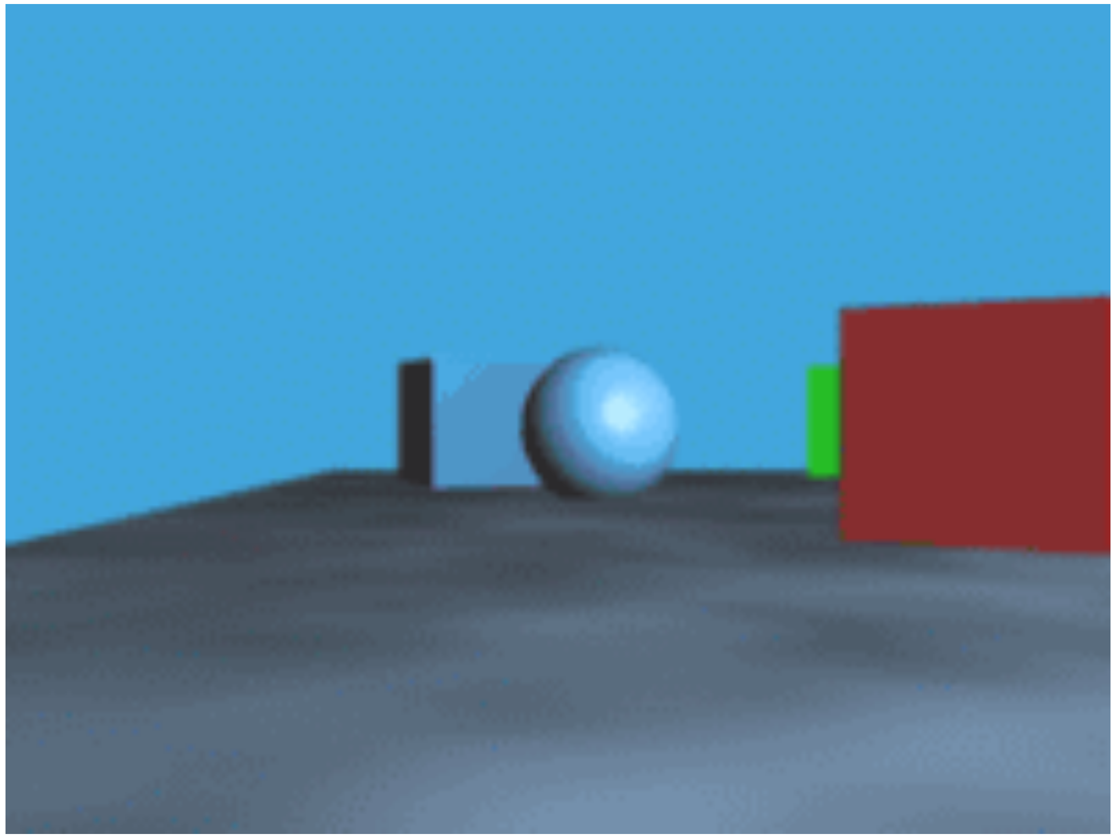
All epipolar lines include the epipole; **the set of all epipolar lines intersect at the epipoles**

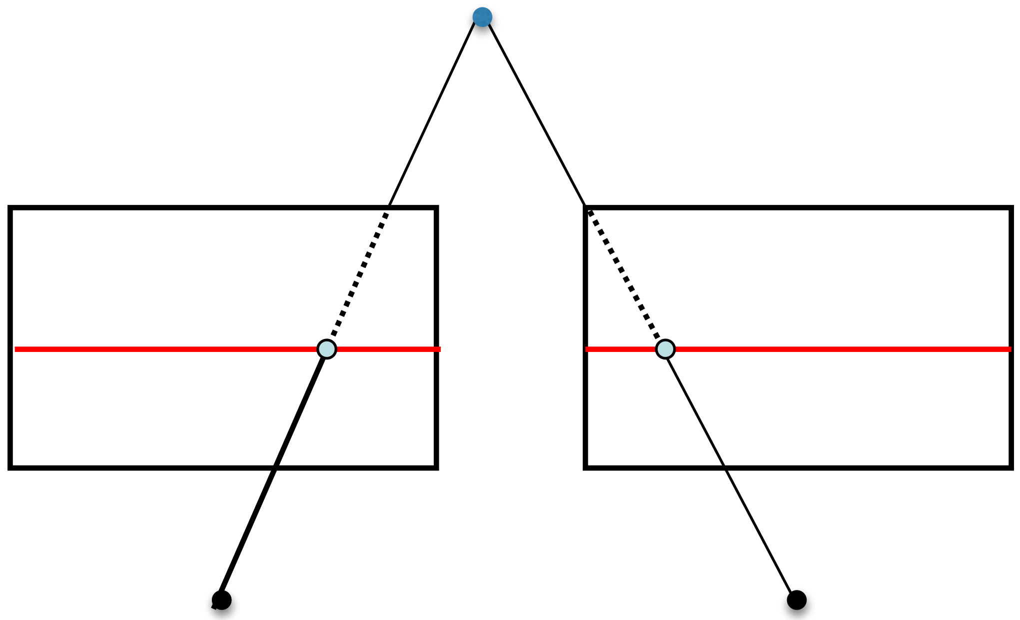
Special case (I)

Parallel, offset cameras (T except for T_x, R = I)



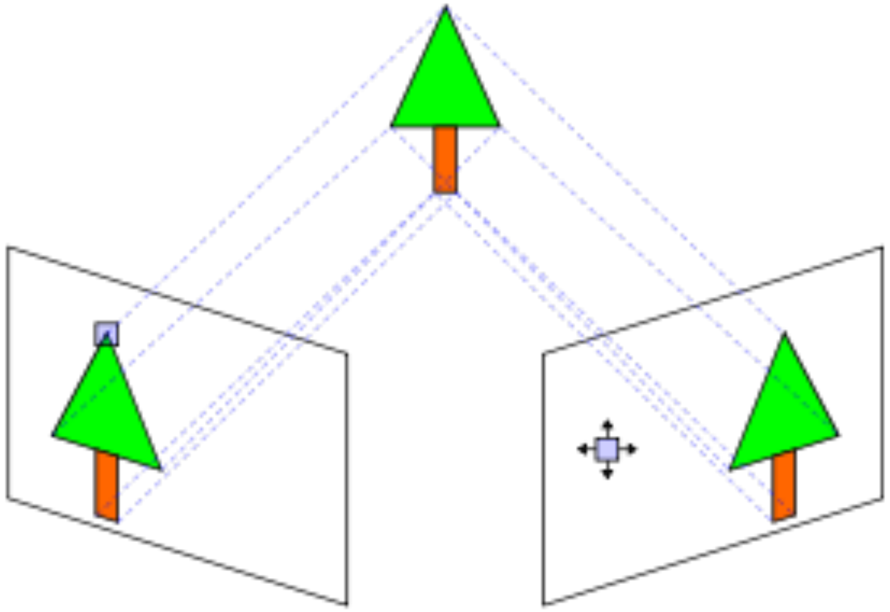
What do the epipolar lines look like?



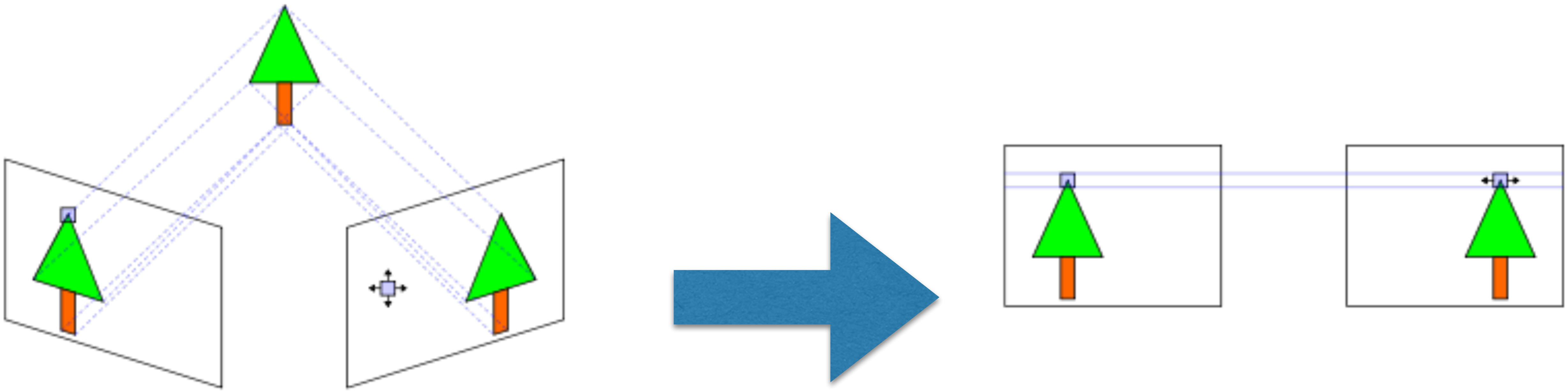


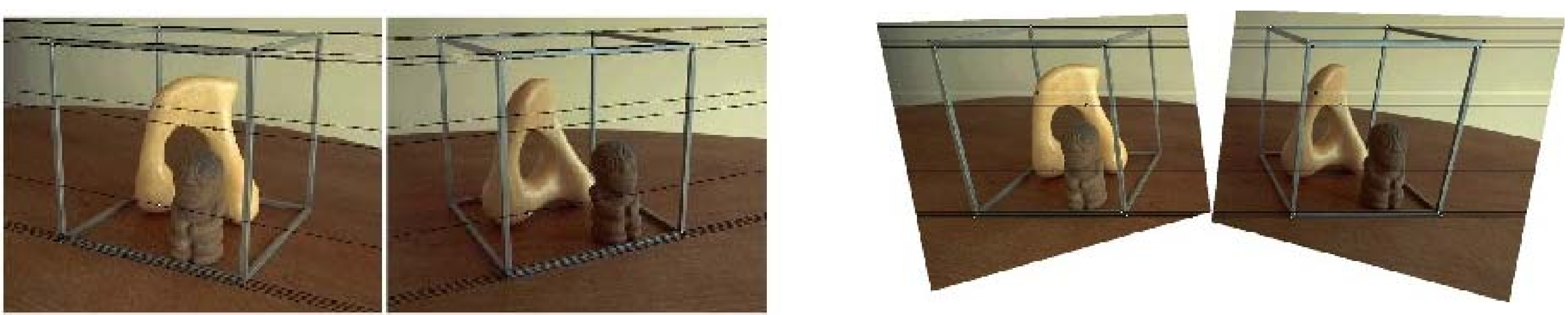
Epipolar lines are parallel to the lines between the two cameras
(horizontal if the cameras only translate in x)

Epipoles are at infinity (derive by slowly rotating image planes)



Stereo Pair
(Rotation and translation)





Stereo Pair

(Rotation and translation)

Rectified Stereo Pair

(Only translation, no rotation)

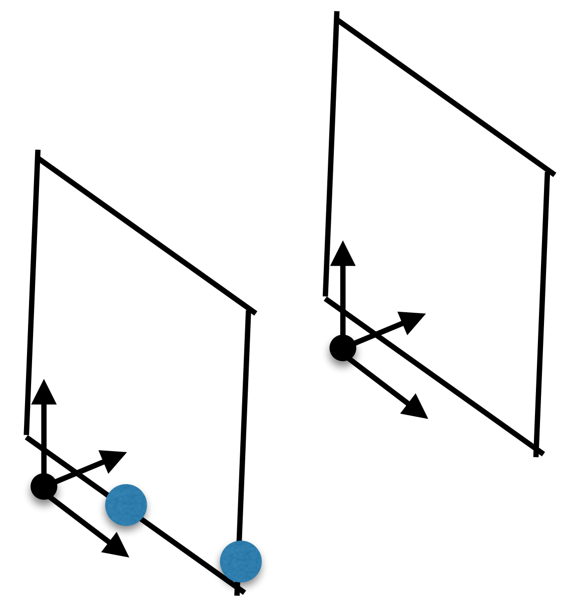
Rotate image plane about fixed camera centers
so that the cameras are facing the same direction (only translation, no rotation)

What kind of transformation is this?

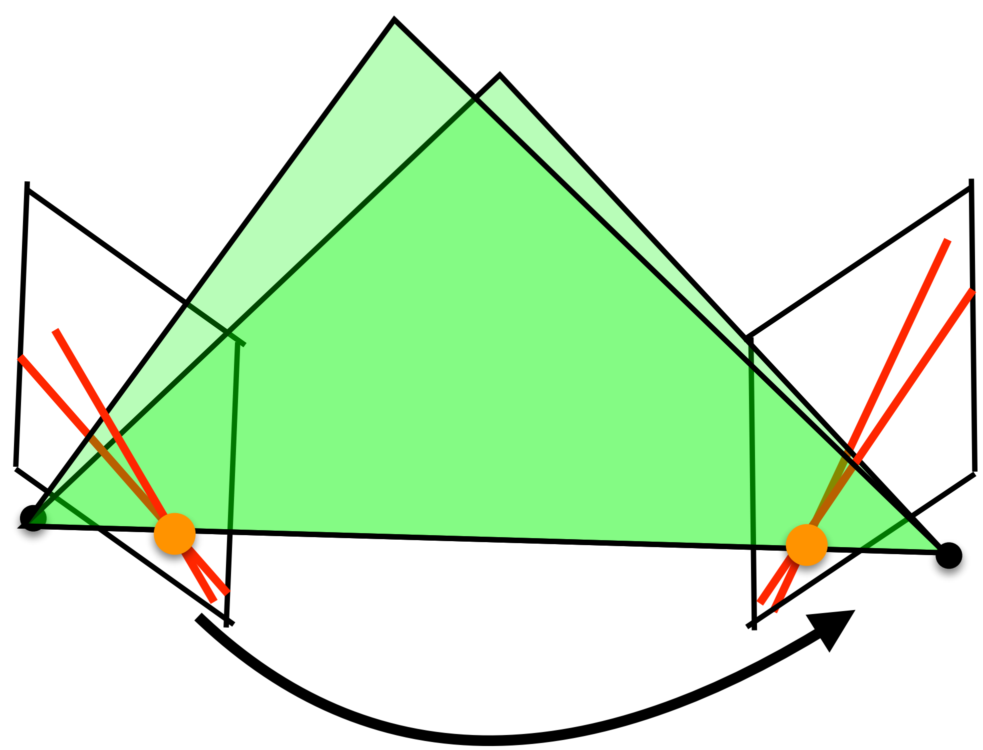
Homography!

Special case (II)

Forward camera motion ($T = 0$ except for T_z, $R = I$)



What would epipolar lines look like?

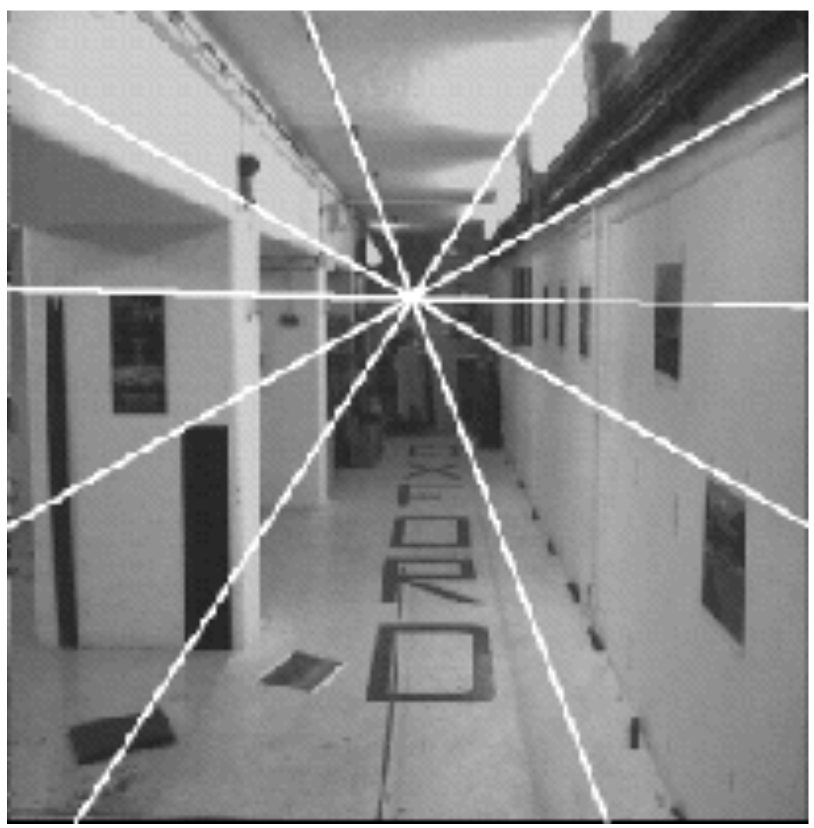


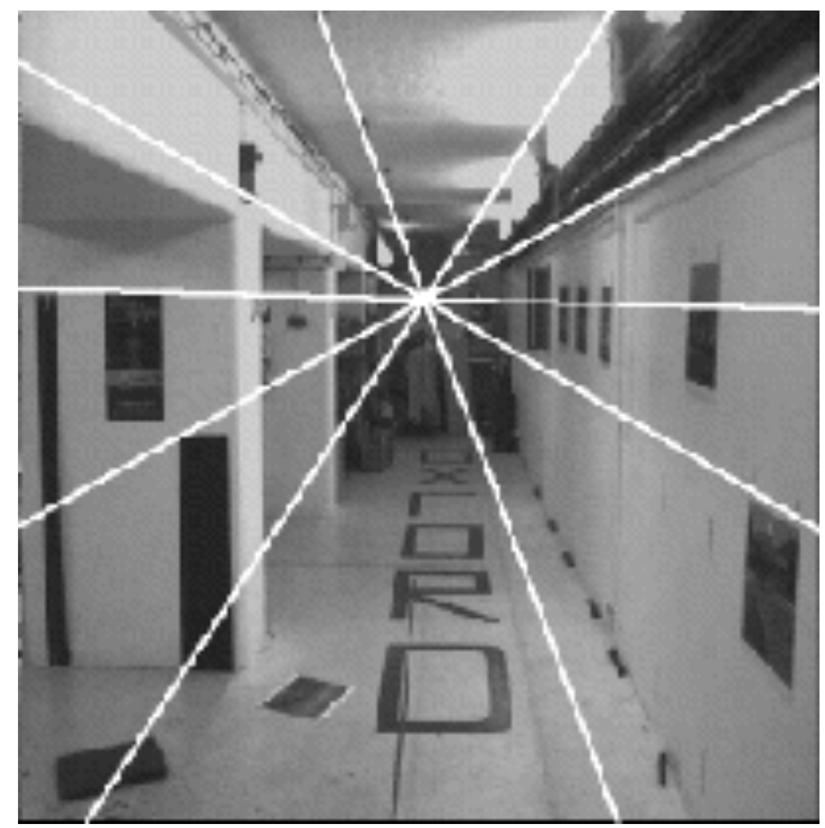
R,t

Epipolar lines (red): intersection of epipolar plane and image planes

Special case (II)

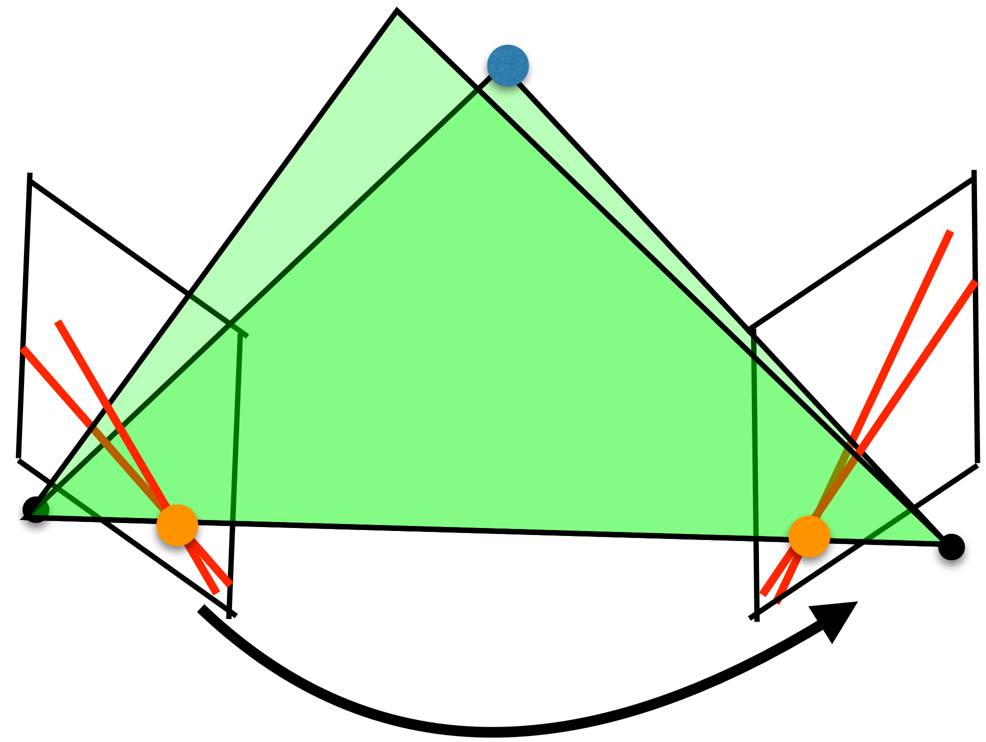
Forward camera motion





Similar to ego motion about flow when the camera moves forward

Key observations from geometric intuition



- Epipolar geometry is characterized by the family of **epipolar planes** that is determined by R,**T** (and intrinsics K_1,K_2).
- Epipolar geometry maps points-on-the-left to **epipolar lines**-on-the-right (and vice versa)
- All such lines intersect at a single point, the **epipole** (which is the projection of the other camera COP)

Questions:

- Do multiple points on-the-left map to the same epipolar line-on-the-right?

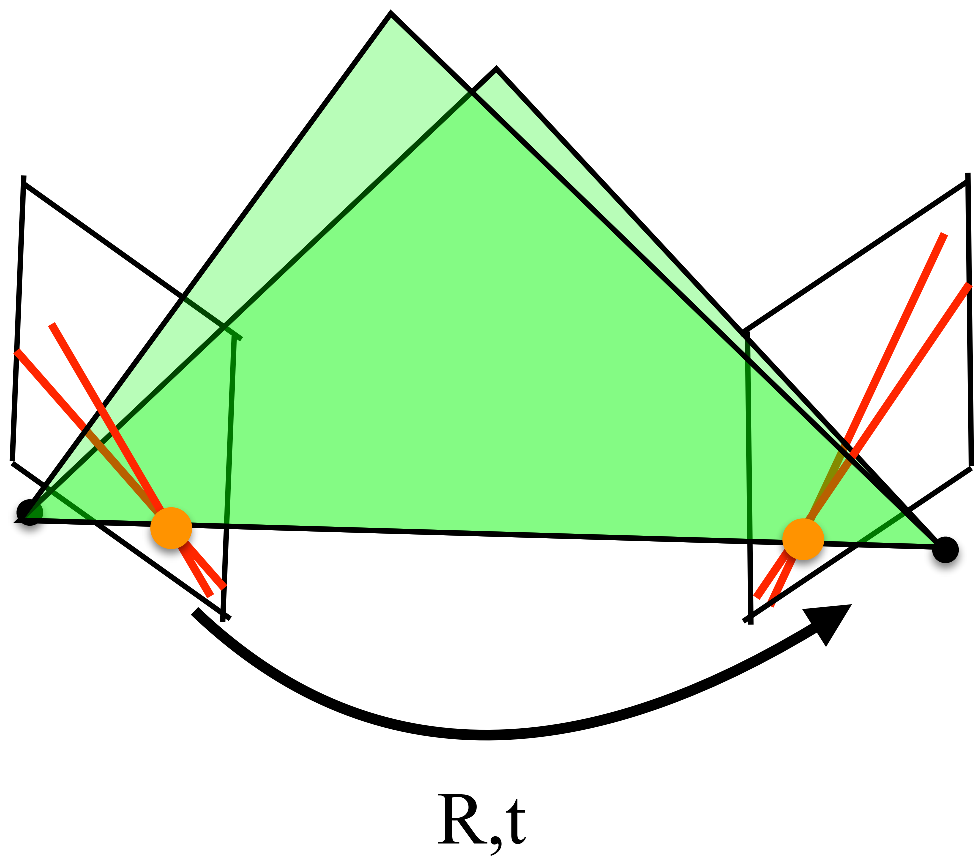
- If so, how can I characterize the *set* of points-on-the-left that map to the same line-on-the-right?

This set is given by the intersection of the epipolar plane with the left image
(*itself* an epipolar line!)

Roadmap

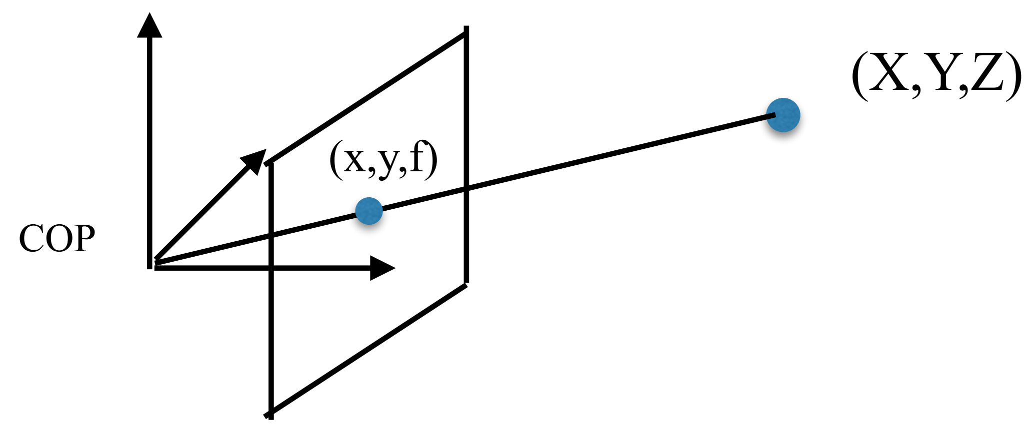
- logistics
- review of motion segmentation
- two-view
  - geometric intuition
  - **essential matrix**, fundamental matrix
  - properties
  - estimation

Mathematical formulation



Goal: given point-on-the-left, we want a mathematical operator that returns the line on-the-right (the epipolar line)

Projecting from camera coordinate system to image coordinates



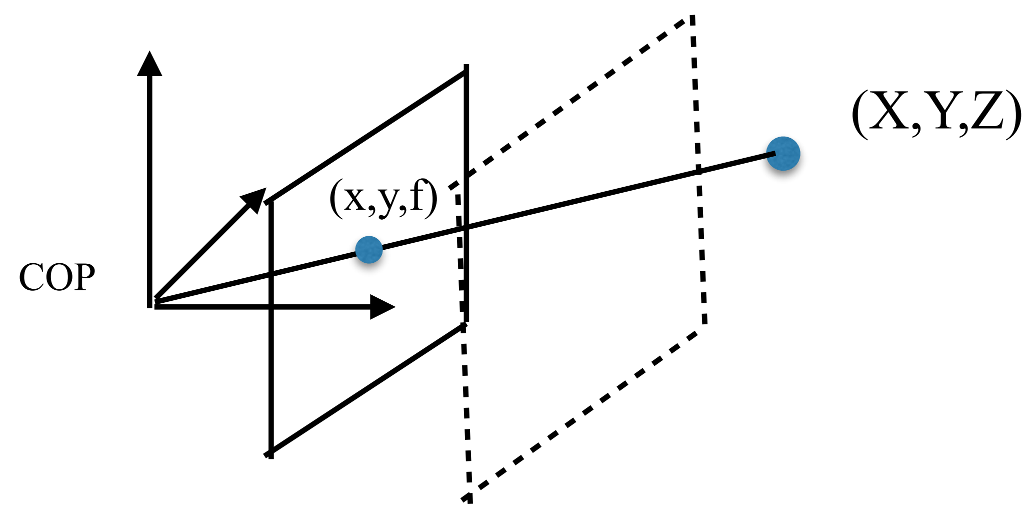
Assume we are projecting a point in camera coordinates

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f s_x & f s_\theta & o_x \\ 0 & f s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\lambda \mathbf{x} = K \mathbf{X}$$

$$\lambda \mathbf{x} = K\mathbf{X}$$

Projecting from camera coordinate system to *normalized* image coordinates (f=1)



If K is known, work with warped image

$$\lambda \underline{K^{-1}x} = X$$

Normalized image coordinates $\lambda x' = X$

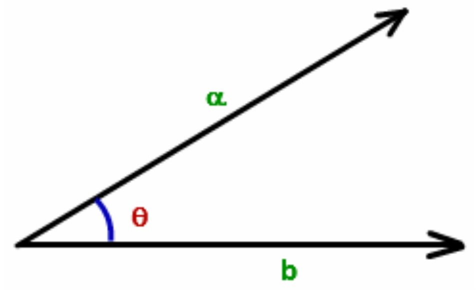
In the normalized image,
with the 3D point in camera coordinates,
the camera projection equation is just:

$$\lambda \mathbf{x}' = \mathbf{X}$$

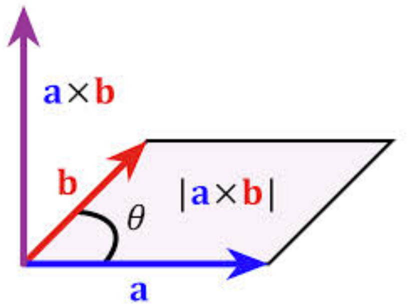
To simplify notation, we'll use **x** instead of **x**'

Recall

Dot product: $\mathbf{a} \cdot \mathbf{b} = ||\mathbf{a}|| \ \ ||\mathbf{b}||cos\theta$



Cross product: $\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \sin\theta \ \mathbf{n}$



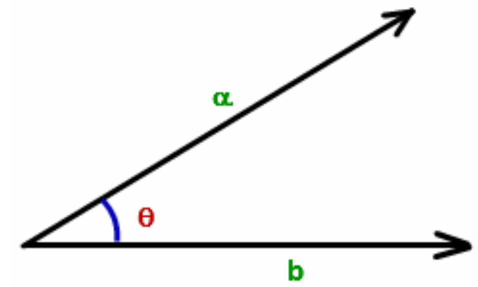
Cross product matrix: $\mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \equiv \hat{\mathbf{a}}\mathbf{b}$

Important property (skew symmetric): $\hat{\mathbf{a}}^T = -\hat{\mathbf{a}}$

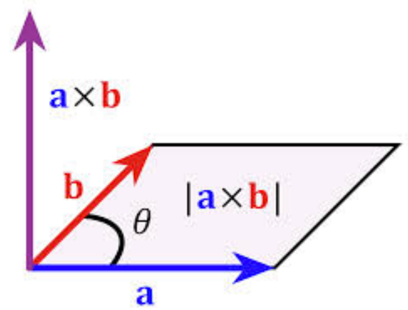
Note: every skew symmetric matrix must look like this!

Recall

Dot product: $\mathbf{a} \cdot \mathbf{b} = ||\mathbf{a}|| \ \ ||\mathbf{b}|| cos\theta$

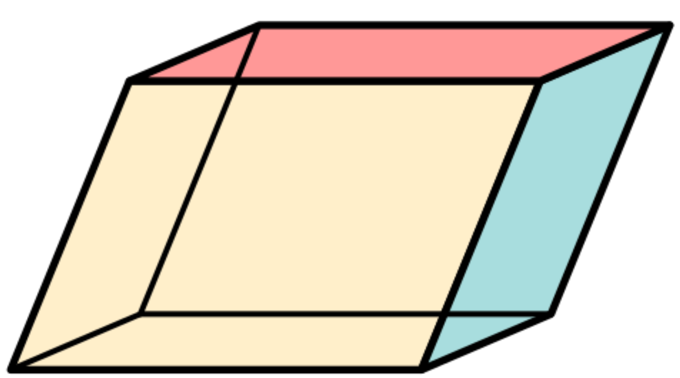


Cross product: $\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \sin\theta \ \mathbf{n}$



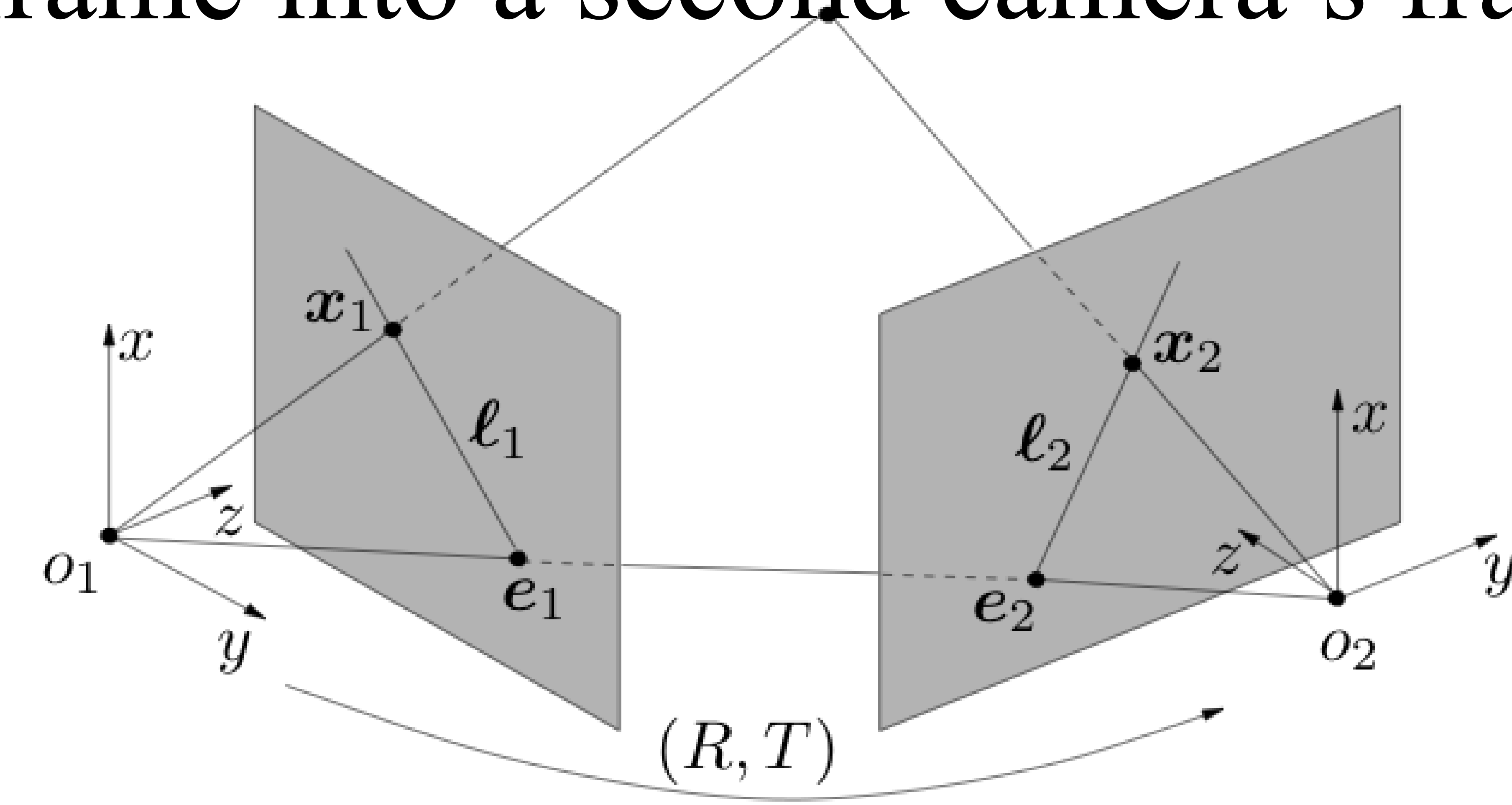
Cross product matrix: $\mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \equiv \hat{\mathbf{a}}\mathbf{b}$

$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})$ = volume of parallelpiped
= 0 for coplanar vectors



(This is a technique to see if the 3 vectors are coplanar)

Transform a point from one camera frame into a second camera's frame

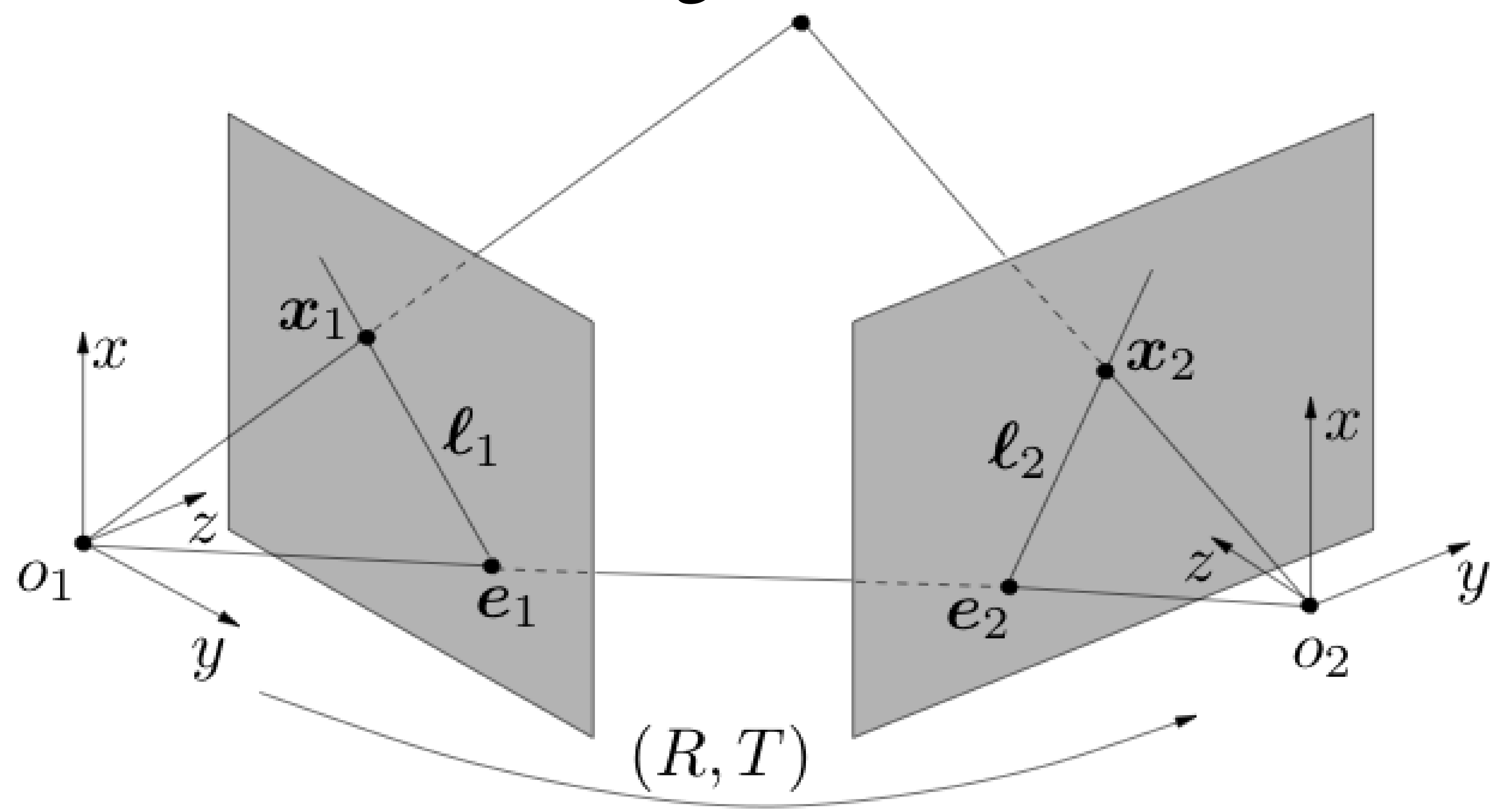


$$\boxed{\boldsymbol{X}_2 = R\boldsymbol{X}_1 + \boldsymbol{T}}$$

$\mathbf{X}_1$: position of 3D point in camera 1's coordinate system

$\mathbf{X}_2$: position of 3D point in camera 2's coordinate system

Projection



$$\boxed{\boldsymbol{X}_2 = R\boldsymbol{X}_1 + \boldsymbol{T}}$$

Projection with normalized intrinsics:

$$\boldsymbol{X}_1 = \lambda_1 \boldsymbol{x}_1, \quad \boldsymbol{X}_2 = \lambda_2 \boldsymbol{x}_2$$

Now I have a relation between a point in 1 image and a point in another image!

Epipolar geometry

$$\boxed{\boldsymbol{X}_2 = R\boldsymbol{X}_1 + \boldsymbol{T}}$$

$$\boldsymbol{X}_1 = \lambda_1 \boldsymbol{x}_1, \quad \boldsymbol{X}_2 = \lambda_2 \boldsymbol{x}_2$$

Plug in: $$\lambda_2 \boldsymbol{x}_2 = R\lambda_1 \boldsymbol{x}_1 + \boldsymbol{T}$$

Take cross product of both sides with the translation vector ***T*** *(on the left)*

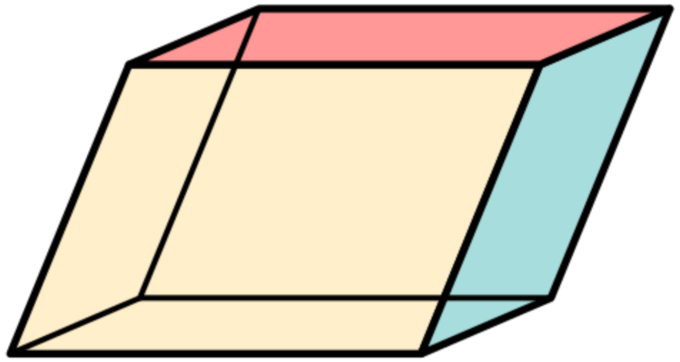
$$\lambda_2 \hat{\mathbf{T}}\mathbf{x}_2 = \lambda_1 \hat{\mathbf{T}} R \mathbf{x_1} + \underbrace{\hat{\mathbf{T}}\mathbf{T}}_{0}$$

Take dot product of both sides with $\boldsymbol{x_2}$ *(on the left)*

$$\lambda_2 \mathbf{x}_2^T \hat{\mathbf{T}}\mathbf{x}_2 = \lambda_1 \mathbf{x}_2^T \hat{\mathbf{T}} R \mathbf{x_1}$$

This is 3 vectors:
dot product and cross product
but 2 vectors are the same so volume is 0!

$$0 = \mathrm{x}_2^T \hat{\mathrm{T}} R \mathrm{x}_1$$



$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})$$

= volume of parallelpiped
= 0 for coplanar vectors

$$\mathbf{x}_2^T \hat{\mathbf{T}} R \mathbf{x}_1 = 0$$

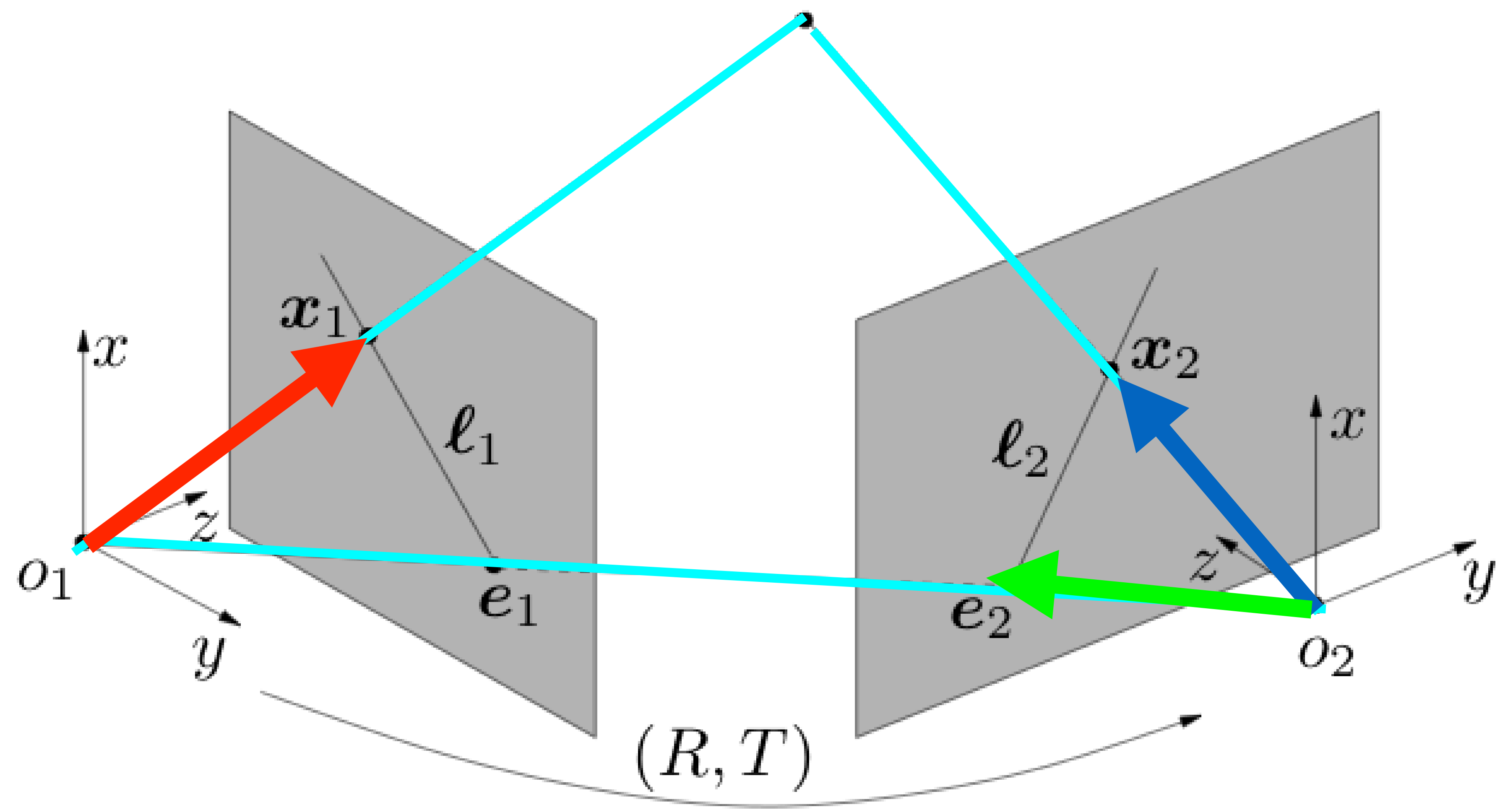
This expression relates points in 1 image x_1 to points in another image x_2

In between is a matrix: $\hat{T}R$

(which does not depend on scene geometry!)

But any points x_1 and x_2 which correspond must satisfy this expression!

Geometric derivation



Simply the coplanar constraint applied to 3 vectors **from camera 2's coordinate system**

$$\mathbf{x}_2 \cdot (\mathbf{T} \times R\mathbf{x}_1) = 0$$

$$\mathbf{x}_2^T \hat{\mathbf{T}} R \mathbf{x}_1 = 0$$

Epipolar geometry

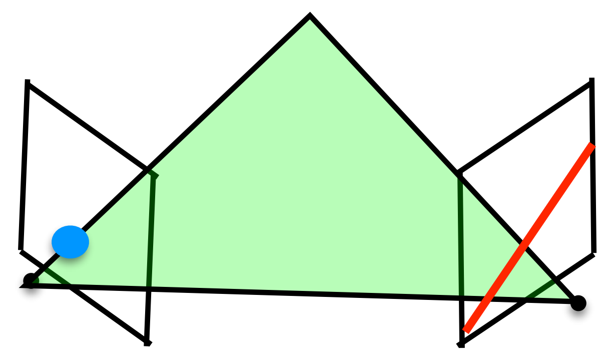
$$\mathbf{x}_2^T \hat{\mathbf{T}} R \mathbf{x}_1 = 0$$

$$\boxed{\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0}$$

E is known as the *essential* matrix

Essential matrix

$$\boxed{\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0}$$



$$\begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \overset{Ex_1}{\begin{bmatrix} a \\ b \\ c \end{bmatrix}} = 0$$

$$ax_2 + by_2 + c = 0 \qquad \text{(equation of a line!)}$$

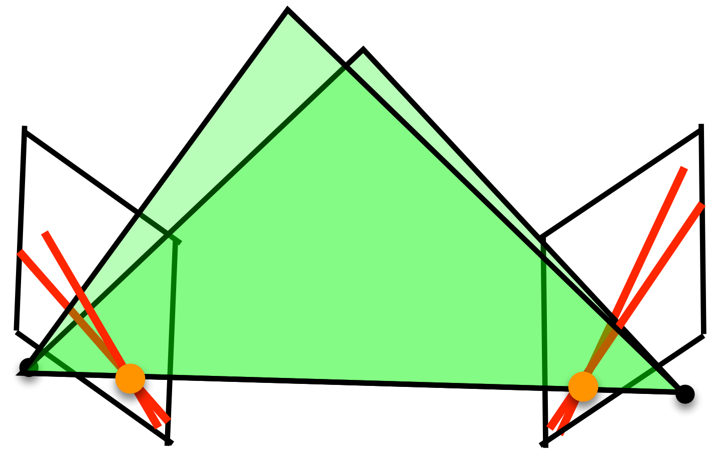
Maps point (x_1,y_1) from left image to line (a,b,c) in right image… and vice versa.

This is the line of points that might correspond to the point (x_1, y_1)

This is an "epipolar line"

Epipoles

$$\boxed{\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0}$$



Recall that all lines in the right image intersect at the epipole
Thus for any point on the left (x_1, y_1),
the point e_2 = right epipole will satisfy this equation

$$\overbrace{\begin{bmatrix} e_x & e_y & 1 \end{bmatrix}}^{\mathbf{e_2}} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = 0, \quad \forall x_1, y_1$$

$$\begin{bmatrix} e_x & e_y & 1 \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} = [? \quad 0 \quad 0]$$

Recall: SVD

Any matrix $A \in \mathbb{R}^{mxn}$ can be decomposed into a product of 3 matrices:

where U and V are orthonormal and Σ is diagonal

Then $Ax = U\Sigma V^\top x$

$$V^T = \begin{bmatrix} v_1^\top \\ v_2^\top \\ \vdots \\ v_N^\top \end{bmatrix} \in \mathbb{R}^{nxn}$$

Right singular vectors

$\Sigma \in \mathbb{R}^{mxn}$ is a rectangular-diagonal matrix: $\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots \\ 0 & \sigma_2 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$ $\quad \sigma_i \geq 0$

Singular values

$$U = [u_1, \dots, u_m] \in \mathbb{R}^{mxm}$$

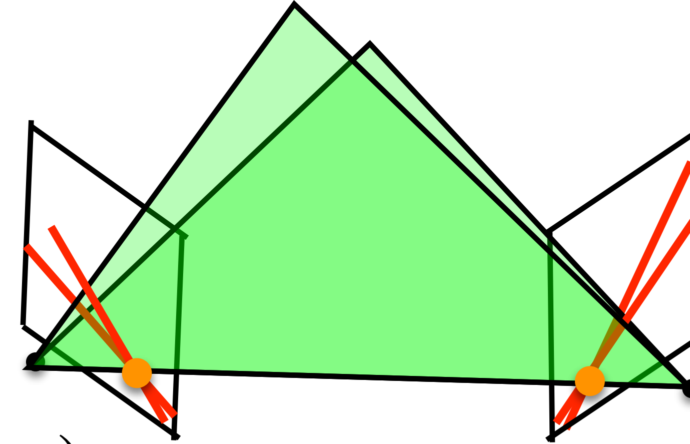
Left singular vectors

$$\boxed{A = U\Sigma V^T = \sum_i^{\min(m,n)} u_i \sigma_i v_i^T}$$

We will define the order:
σ_1 is the largest
σ_N is the smallest

Epipoles

$$\boxed{\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0}$$



Thus for any point on the left (x_1, y_1),
the point e_2 = right epipole will satisfy this equation

$$\begin{bmatrix} e_x & e_y & 1 \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

$$A = U\Sigma V^T = \sum_i^{\min(m,n)} u_i \sigma_i v_i^T$$

$$e_2^T E = \sum_i e_2^T u_i \sigma_i v_i^T = 0$$

What does this imply? Recall that all of the vectors $u_1, \ldots, u_N$ are orthonormal

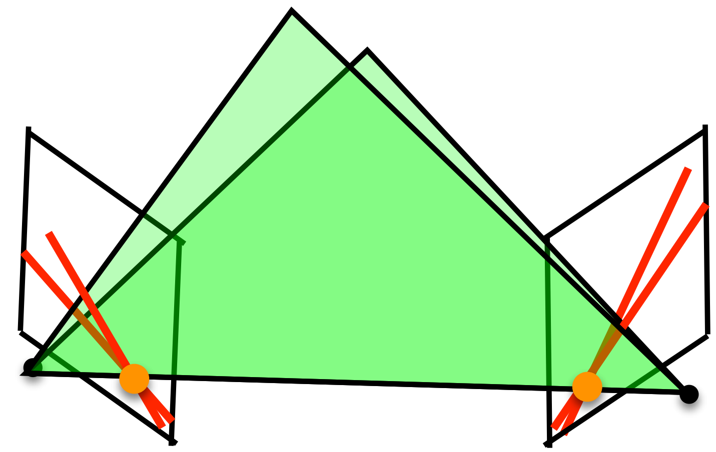
This implies that e_2 is equal to one of the singular vectors u_N
with an associated singular value $\sigma_N = 0$

Thus, the right epipole e_2 is equal to the smallest left singular vector u_N

What is the left epipole e_1?

Epipoles

$$\boxed{\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0}$$



Recall that all lines in the **left** image intersect at the epipole

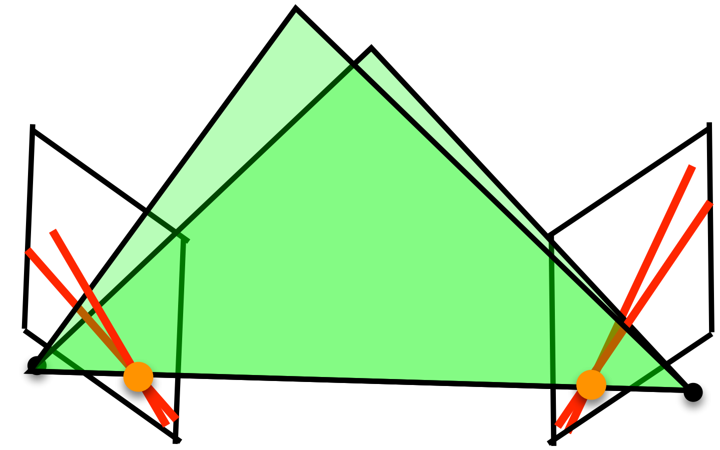
Thus for any point on the right (x_2, y_2),
the point e_1 = left epipole will satisfy this equation

$$\begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ 1 \end{bmatrix} = 0, \quad \forall x_2, y_2$$

$$\begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ 1 \end{bmatrix} = 0$$

Epipoles

$$\boxed{\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0}$$



Thus for any point on the right (x_2, y_2),
the point e_1 = left epipole will satisfy this equation

$$\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ 1 \end{bmatrix} = 0$$

$$A = U\Sigma V^T = \sum_i^{\min(m,n)} u_i \sigma_i v_i^T$$

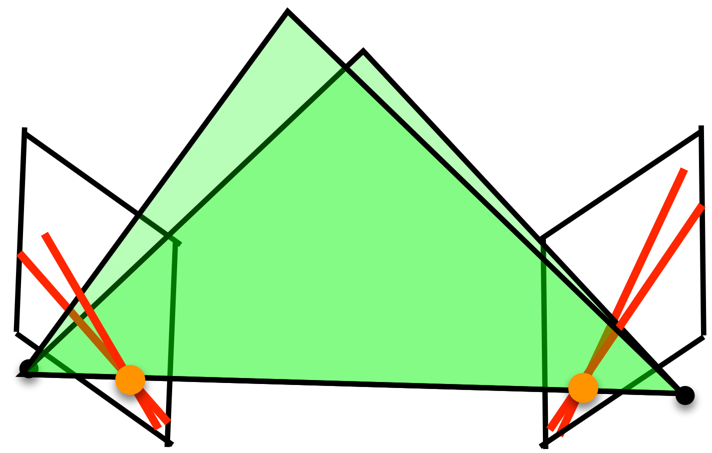
$$Ee_1 = \sum_i u_i \sigma_i v_i^T e_1 = 0$$

What does this imply? Recall that all of the vectors $v_1, \ldots, v_N$ are orthonormal

This implies that e_1 is equal to one of the right singular vectors v_N
with an associated singular value $\sigma_N = 0$

Thus, the **left** epipole e_1 is equal to the smallest right singular vector v_N

Unnormalized cameras (don’t assume f = 1, K = I)



$$\lambda_1 \mathbf{x_1} = K_1 \mathbf{X}_1$$

$$\lambda_2 \mathbf{x_2} = K_2 \mathbf{X}_2$$

$$\boxed{\boldsymbol{X}_2 = R\boldsymbol{X}_1 + \boldsymbol{T}}$$

Previous equation of correspondences

$$\mathbf{x}_2^T \hat{\mathbf{T}} R \mathbf{x}_1 = 0$$

New equation of correspondences

$$\mathbf{x_2}^T K_2{}^{-T} \hat{\mathbf{T}} R K_1^{-1} \mathbf{x_1} = 0$$

$$\mathbf{x_2}^T F \mathbf{x_1} = 0$$

We’ll call F the *fundamental matrix*

Equations for corresponding points x_1 and x_2

Essential matrix (normalized cameras: K = I):

$$\mathbf{x}_2^T \hat{\mathbf{T}} R \mathbf{x}_1 = 0$$

$$\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0$$

Fundamental matrix (unnormalized cameras: K ≠ I):

$$\mathbf{x_2}^T F \mathbf{x_1} = 0$$

$$\mathbf{x_2}^T K_2{}^{-T} \hat{\mathbf{T}} R K_1^{-1} \mathbf{x_1} = 0$$

Roadmap

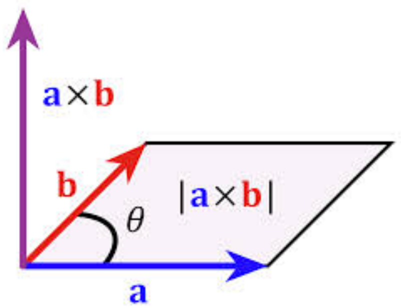
- logistics
- review of motion segmentation
- two-view
    - geometric intuition
    - essential matrix E, fundamental matrix F
    - **properties of E, F**
    - estimating E,F from correspondences
    - inferring R,T from E,F

Background: SVDs of skew symmetric matrices

Any skew-symmetric matrix ($A = -A^T$) can be thought of as a cross-product

Cross product: $\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \sin\theta \; \mathbf{n}$

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



SVD of a skew-symmetric matrix:

$$\hat{\mathbf{a}} = \begin{bmatrix} -\mathbf{e}_2 & \mathbf{e}_1 & \mathbf{e}_3 \end{bmatrix} \begin{bmatrix} ||\mathbf{a}|| & 0 & 0 \\ 0 & ||\mathbf{a}|| & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{e}_1^T \\ \mathbf{e}_2^T \\ \mathbf{e}_3^T \end{bmatrix}$$

where $\mathbf{e}_3 = \mathbf{a} \,/\, \|\mathbf{a}\|$

$$\hat{\mathbf{a}} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix} \begin{bmatrix} ||\mathbf{a}|| & 0 & 0 \\ 0 & ||\mathbf{a}|| & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{e}_1^T \\ \mathbf{e}_2^T \\ \mathbf{e}_3^T \end{bmatrix}$$

Crucial properties:

- smallest left and right singular vector of skew-symmetric matrix given by **a** (i.e., $\mathbf{a} \times \mathbf{a} = \mathbf{0}$)
- non-zero singular values are equal to $\|\mathbf{a}\|$ (i.e., $\mathbf{a} \times \mathbf{b} = ||\mathbf{a}|| R_{90^\circ}\mathbf{b} \quad \textbf{for} \quad \mathbf{a} \perp \mathbf{b}$)

Properties (essential matrix)

https://en.wikipedia.org/wiki/Essential_matrix#Properties_of_the_essential_matrix

Q. How many DOFs are needed to specify an essential matrix?

$$\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0$$

3 (rotation) + 2 (translation direction)

$$\mathbf{x}_2^T \hat{\mathbf{T}} R \mathbf{x}_1 = 0$$

Q. Can any 3x3 matrix be an essential matrix?

No…

E is the product of a rotation and skew-symmetric matrix $\hat{\mathbf{T}}$

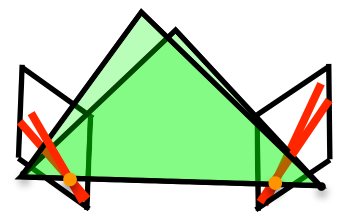
Singular values of E = singular values of $\hat{\mathbf{T}}$ = $(\sigma,\sigma,0)$

[rotations do not affect singular values]

*Q. Given E, can we uniquely recover R,**T**?*

Almost. It is unique up to symmetries

Properties (fundamental matrix)



$$\mathbf{x_2}^T K_2^{-T} \hat{\mathbf{T}} R K_1^{-1} \mathbf{x_1} = 0$$

$$\boxed{\mathbf{x_2}^T F \mathbf{x_1} = 0}$$

Q. Can any 3x3 matrix be a fundamental matrix?

Let $\mathbf{e}_2 = K_2\mathbf{T}$

$e_2^T F = ?$ $\mathbf{e}_2^T F = 0$

similar argument for $\mathbf{e}_1$

What does this imply about the answer to this question?

No! epipoles are in the null space, implying rank(F) = 2

Q. How many DOFs are needed to specify F?

7 = 9 - 1 (for scale) - 1 (for 0-determinant)

Formal characterizations

Ma et al, An Invitation to 3D Vision

Theorem 5.1 (Characterization of the essential matrix). *A non-zero matrix $E \in \mathbb{R}^{3\times3}$ is an essential matrix* if and only if *E has a singular value decomposition (SVD): $E = U\Sigma V^T$ with*

$$\Sigma = diag\{\sigma, \sigma, 0\}$$

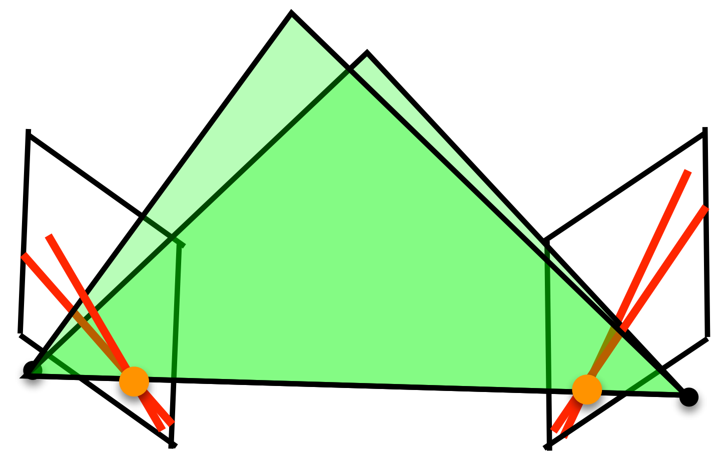
for some $\sigma \in \mathbb{R}_+$ and $U, V \in SO(3)$.

Remark 6.1. *Characterization of the fundamental matrix. A non-zero matrix $F \in \mathbb{R}^{3\times3}$ is a fundamental matrix if F has a singular value decomposition (SVD): $E = U\Sigma V^T$ with*

$$\Sigma = diag\{\sigma_1, \sigma_2, 0\}$$

for some $\sigma_1, \sigma_2 \in \mathbb{R}_+$.

Essential and Fundamental Matrices



$$E = \hat{\mathbf{T}}R$$

$$\mathbf{x}_2^T E \mathbf{x}_1 = 0$$

$$E = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \mathbf{e}_2 \end{bmatrix} \begin{bmatrix} \sigma & & \\ & \sigma & \\ & & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_0^T \\ \mathbf{v}_1^T \\ \mathbf{e}_1^T \end{bmatrix}$$

$$F = {K_2}^{-T} E K_1^{-1}$$

$$\mathbf{x}_2^T F \mathbf{x}_1 = 0$$

$$F = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \mathbf{e}_2 \end{bmatrix} \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_0^T \\ \mathbf{v}_1^T \\ \mathbf{e}_1^T \end{bmatrix}$$

*where **e_1, e_2** are epipoles in right and left images and singular values can be arbitrarily scaled*

SVD characterization allows us to snap *any* 3x3 matrix to the closest E, F matrix (in sense of Frebonius norm): How?

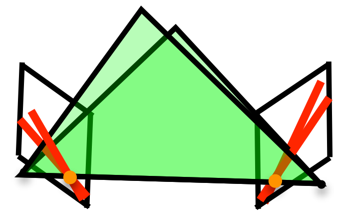
F: zero out smallest singular value

E: zero out smallest singular value and average the over 2

Roadmap

- logistics
- review of motion segmentation
- two-view
  - geometric intuition
  - essential matrix E, fundamental matrix F
  - properties of E,F
  - **estimating E,F from correspondences**
  - inferring R,T from E,F

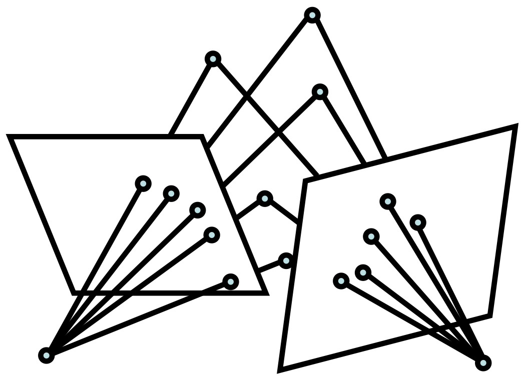
Estimation (fundamental matrix)



Assume we have a corresponding pair of points: in noise-free case….

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = 0 \iff \begin{bmatrix} xx' & xy' & x & yx' & yy' & y & x' & y' & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

Estimation (fundamental matrix)

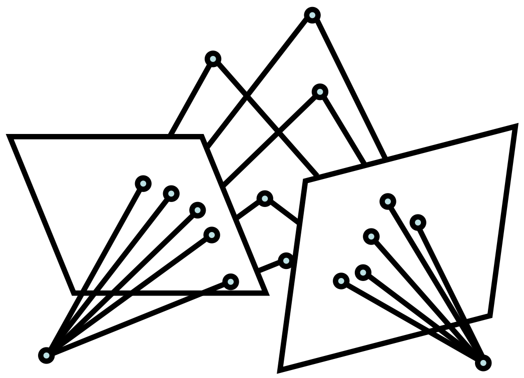


Given m point correspondences (x_i,y_i) and (x'_i,y'_i):

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_mx'_m & x_my'_m & x_m & y_mx'_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

AF(:) = 0

Estimation (fundamental matrix)



Given m point correspondences (x_i,y_i) and (x'_i,y'_i):

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_mx'_m & x_my'_m & x_m & y_mx'_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

AF(:) = 0 noisy case: $\min_{||F||=1} ||AF(:)||^2$

How do I solve this? **SVD!**

How many pairs of points are needed?

m = 8 point algorithm due to Longuet-Higgens

Issue:

$$\begin{array}{c}\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_mx'_m & x_my'_m & x_m & y_mx'_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \\ \begin{array}{ccccccccc} \sim 10000 & \sim 10000 & \sim 100 & \sim 10000 & \sim 10000 & \sim 100 & \sim 100 & \sim 100 & 1 \end{array}\end{array} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

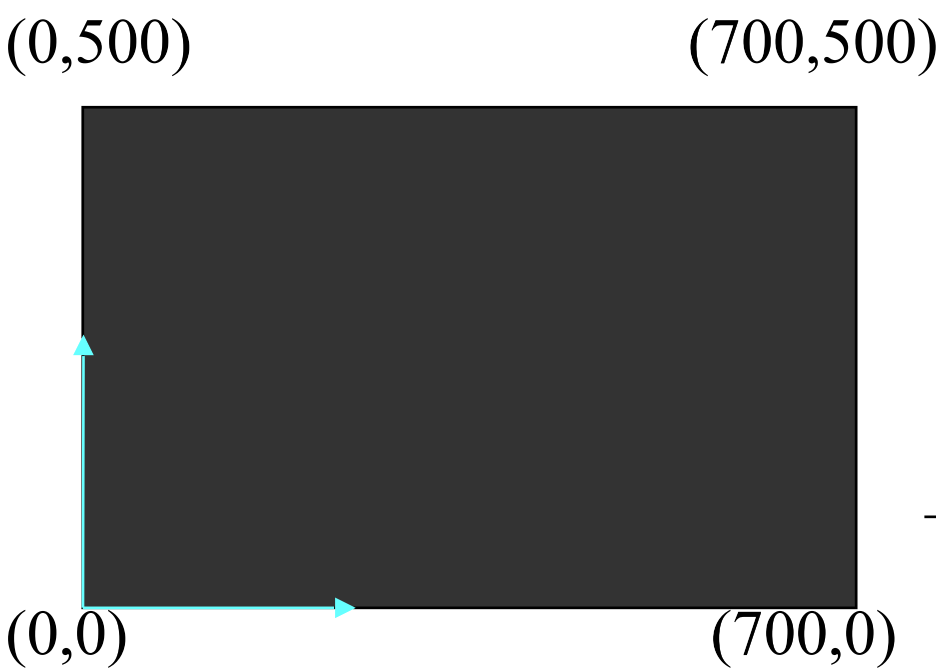
!

Orders of magnitude difference
Between column of data matrix
$\rightarrow$ least-squares yields poor results

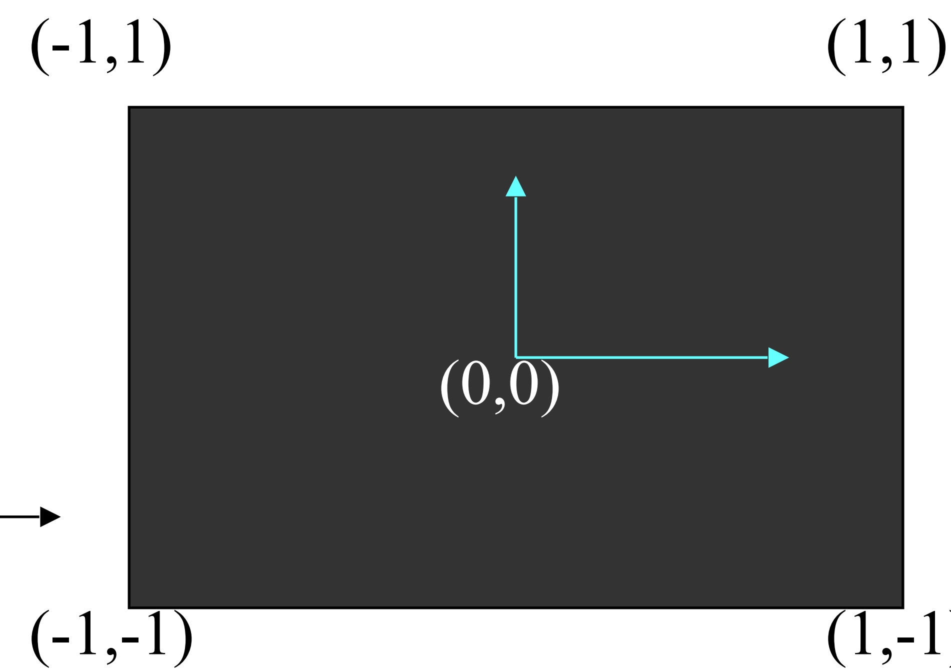
“In Defense of the 8-point Algorithm”

(Hartley, PAMI´97)

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



$$\begin{bmatrix} \frac{2}{700} & 0 & -1 \\ & \frac{2}{500} & -1 \\ & & 1 \end{bmatrix}$$



SVD now produces good results

Final “annoying” issue

Least squares solution won’t produce F that’s rank 2
(or rank-2 E with 2 identical singular values)

Solution: find the closest F/E (Frebonius norm) with SVD

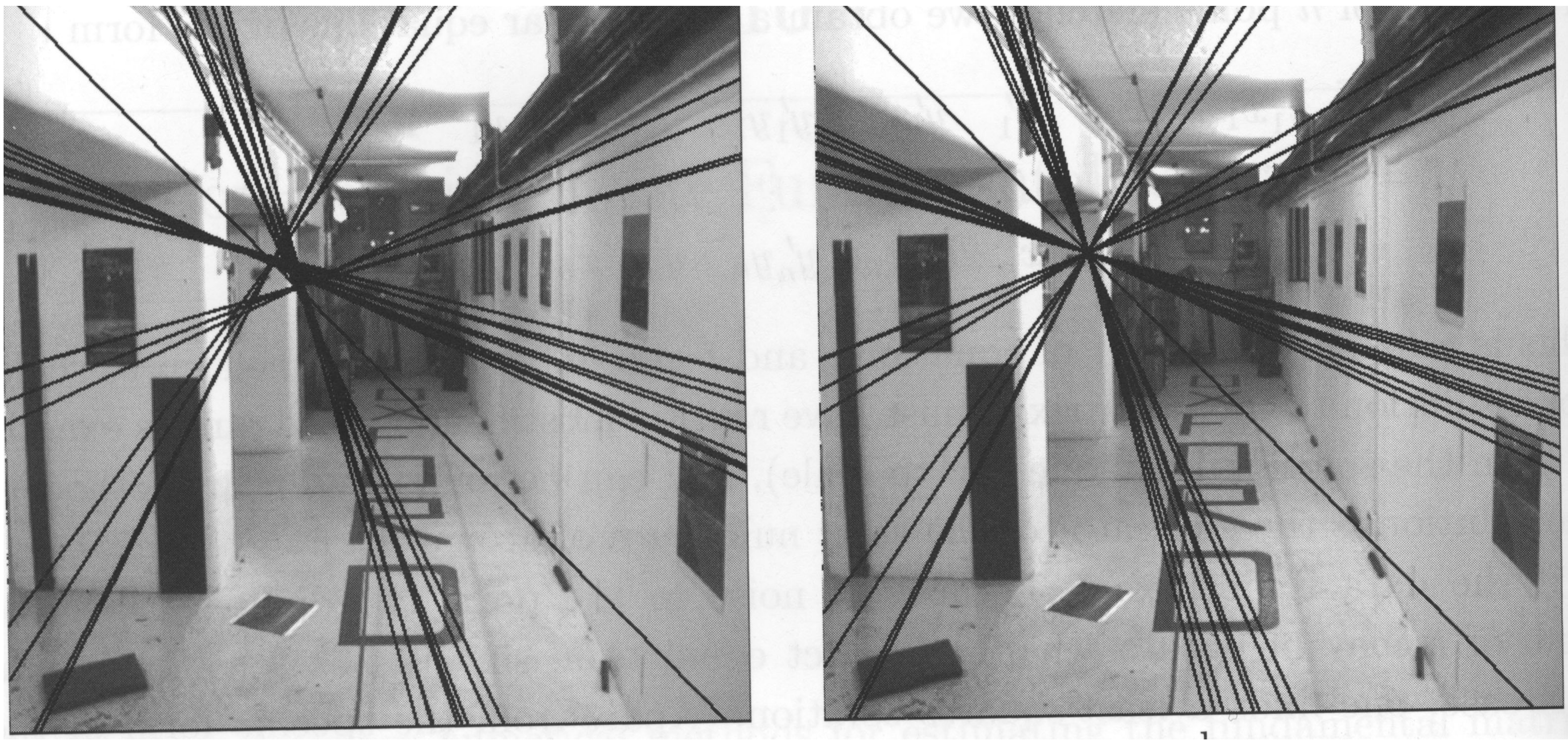
$$X = U \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T$$

How do we fix this?

Closest fundamental matrix: set $\sigma_3 = 0$

Closest essential matrix: set $\sigma_3 = 0$, $\sigma = .5*(\sigma_1+\sigma_2)$

Rank-2 Fundamental Matrix



Closest fundamental matrix: set $\sigma_3 = 0$

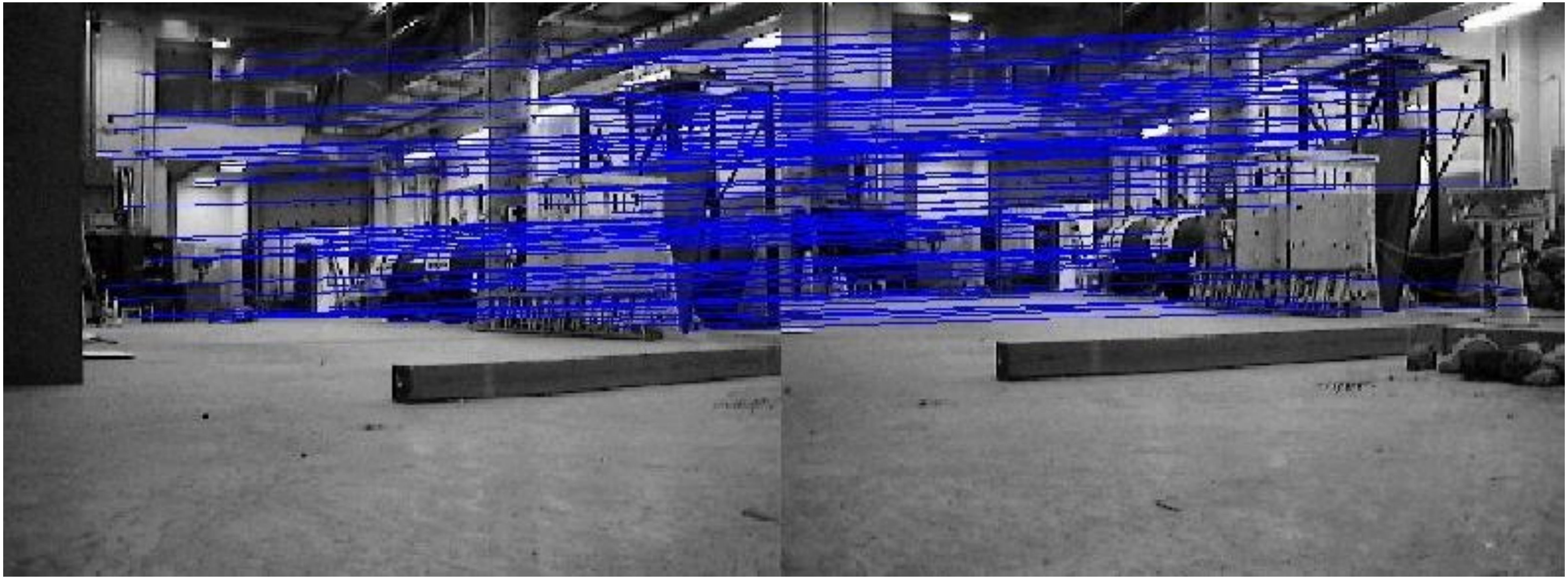
Recall: RANSAC

RANSAC loop:

1. Select feature pairs (at random)
2. Compute transformation T (exact)
3. Compute *inliers* (point matches where $d(\mathbf{x}'_i, F\mathbf{x}_i)^2 \leq \epsilon$)
   Keep largest set of inliers

5. Re-compute least-squares estimate of transformation **T** using all of the inliers

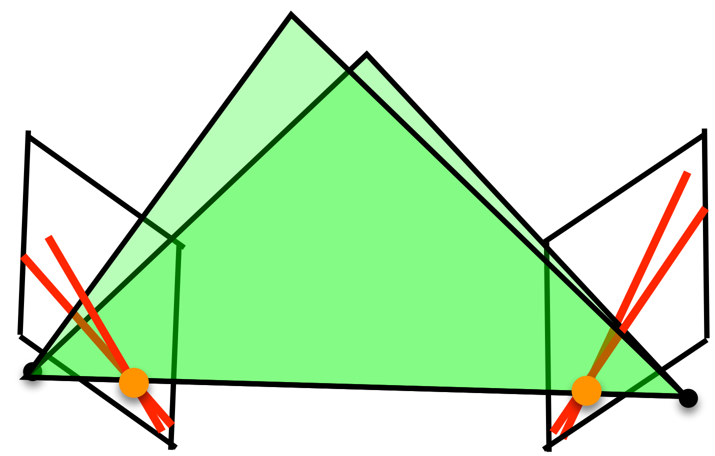
Fundamental matrix estimation with RANSAC



Roadmap

- logistics
- review of motion segmentation
- two-view
  - geometric intuition
  - essential matrix E, fundamental matrix F
  - properties of E,F
  - estimating E,F from correspondences
  - **inferring R,T from E,F**

Recovering T,R from E



1. Universal scale ambiguity

Doubling **T** results in same epipolar lines

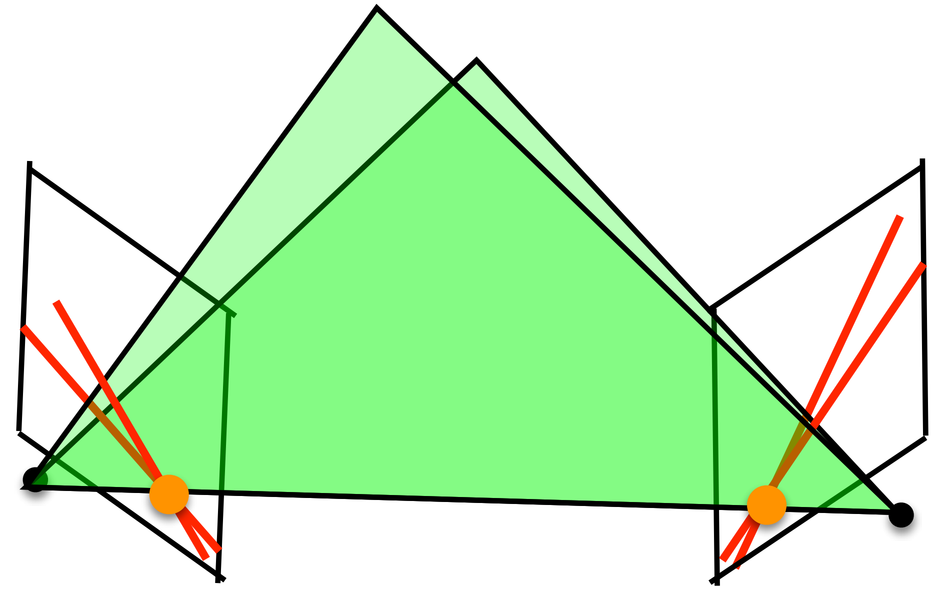
Let’s fix $||\mathbf{T}|| = 1$

Numerous methods for recovering **T**,R from E exist:
SVD, Lounget-Higgen’s alg, etc.

Recovering T from E

SVD-based approach for noise-free E (Szeliski Chap 7.2)

$$\boxed{\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0}$$



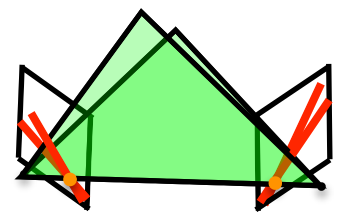
Take (left-handside) dot product of E = $\hat{\mathbf{T}}$R with **T**

$$\mathbf{T}^T E = -(\mathbf{T} \times \mathbf{T})R = 0$$

Implies that translation vector = left singular vector of E associated with smallest singular value

Recovering T from E

SVD-based approach for noise-free E (Szeliski Chap 7.2)



$$\begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = 0$$

(recall we scale E so that singular values are 1)

$$E = \hat{\mathbf{T}} R = U \Sigma V^T = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \mathbf{T} \end{bmatrix} \begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_0^T \\ \mathbf{v}_1^T \\ \mathbf{v}_2^T \end{bmatrix}$$

Set translation direction = smallest left singular vector of E

But we can't distinguish E from -E, so we only know direction up to a sign

Aside: $\mathbf{v}_2$ = epipole in left image

Roadmap

- logistics
- review of motion segmentation
- two-view
  - geometric intuition
  - essential matrix E, fundamental matrix F
  - properties of E,F
  - estimating E,F from correspondences
  - inferring R,T from E,F
- stereo