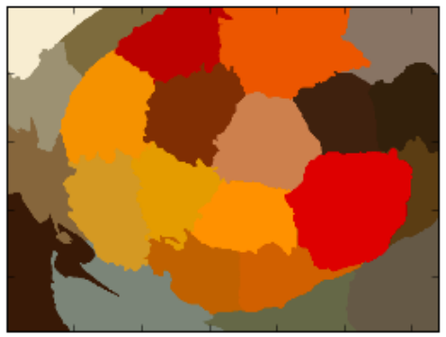
Two-view geometry

Finish up motion…

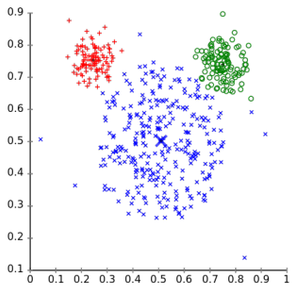
- Lucas Kanade
- Egomotion
    - Motivation
    - Time-to-Contact, Parallax, Focus-of-Expansion
- Optical Flow
    - Motivation, aperture problem
    - Sparse (KLT) vs Dense (variational)
    - Optimization tools: robust losses, varational coarse-to-fine, markov-random feilds
    - **Segmentation** (dominant motion estimation, background subtraction, layered models)

Motion segmentation

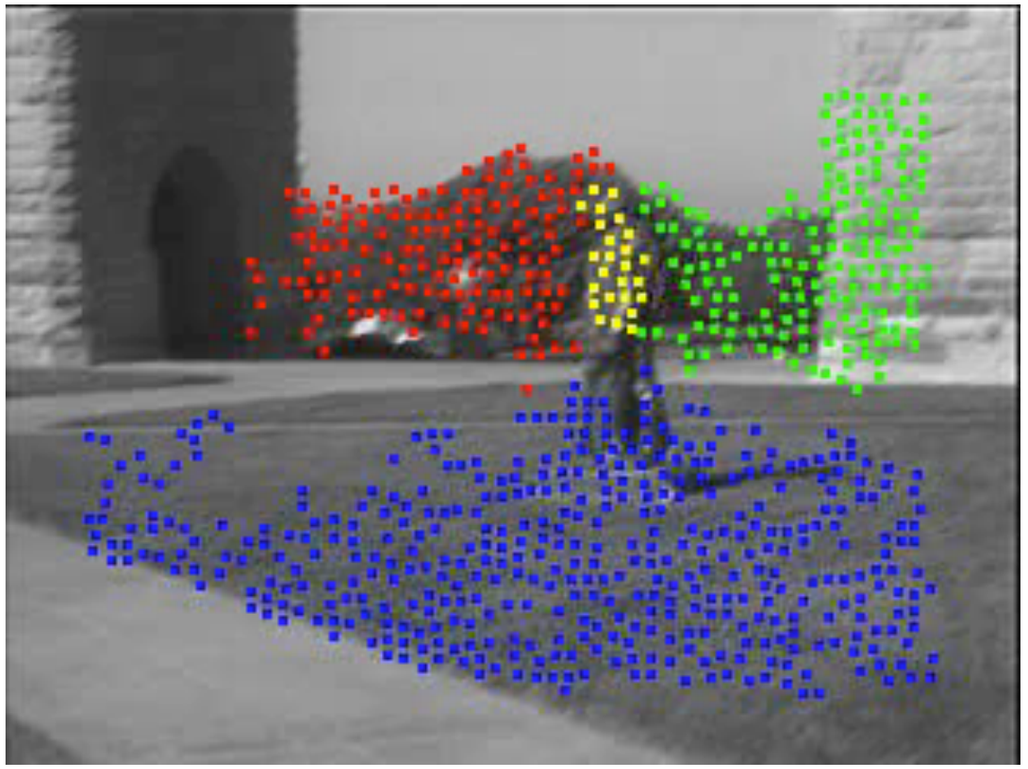
Treat as pixel-clustering problem



pixel color clustering



pixel motion clustering



1. Obtain an initial estimate of flow (sparse or dense)
2. Cluster pixels using feature vectors (consisting of flow, RGB, etc.)

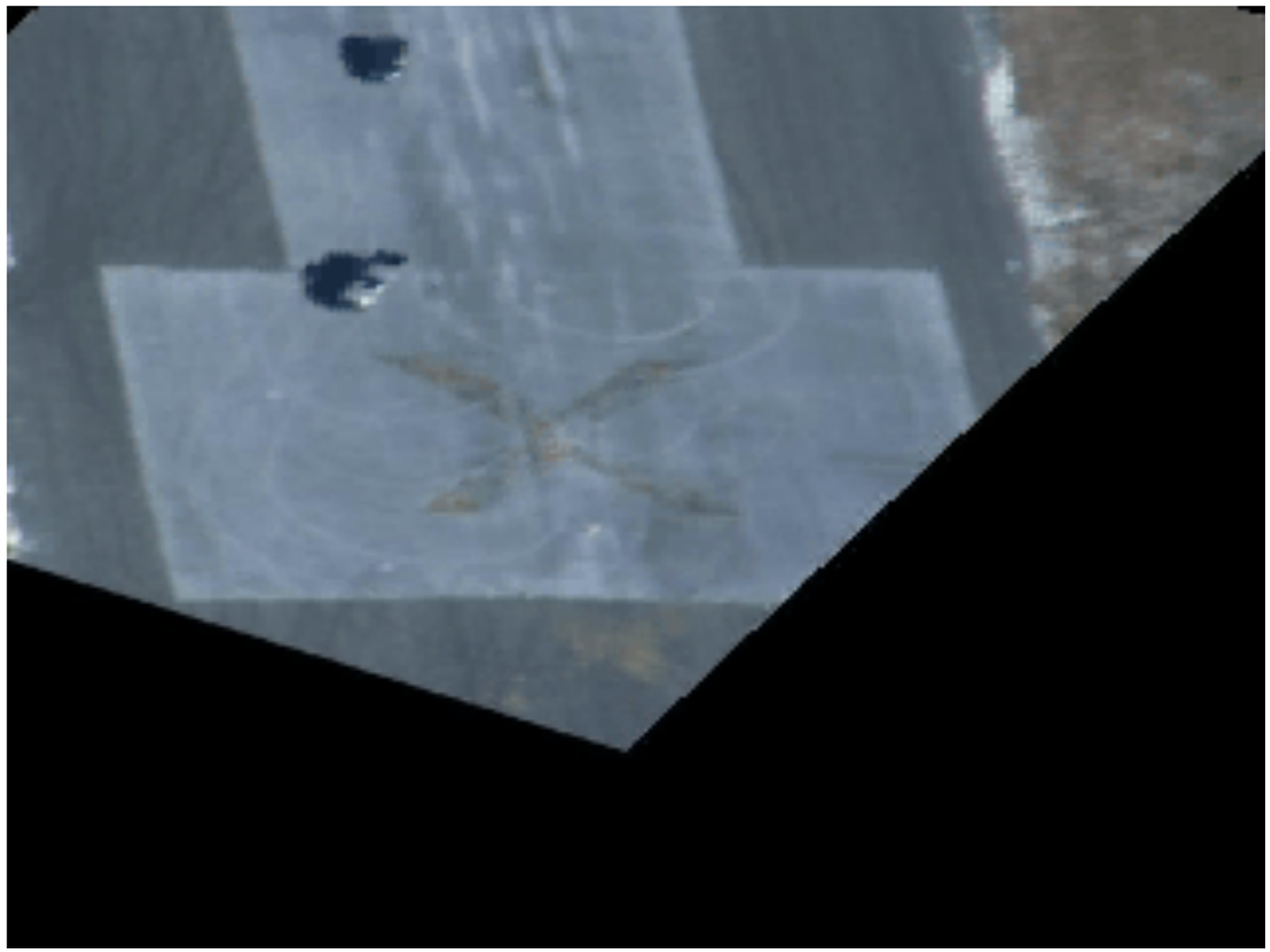
Generalize K-means to fit a parametric model (e.g., affine warp) rather than a centroid

$$\begin{bmatrix} R \\ G \\ B \\ x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} R \\ G \\ B \\ x \\ y \\ u \\ v \end{bmatrix}$$

Weiss & Adelson, CVPR 96 use “soft” K-means or EM algorithm

Motion segmentation: special case (I)

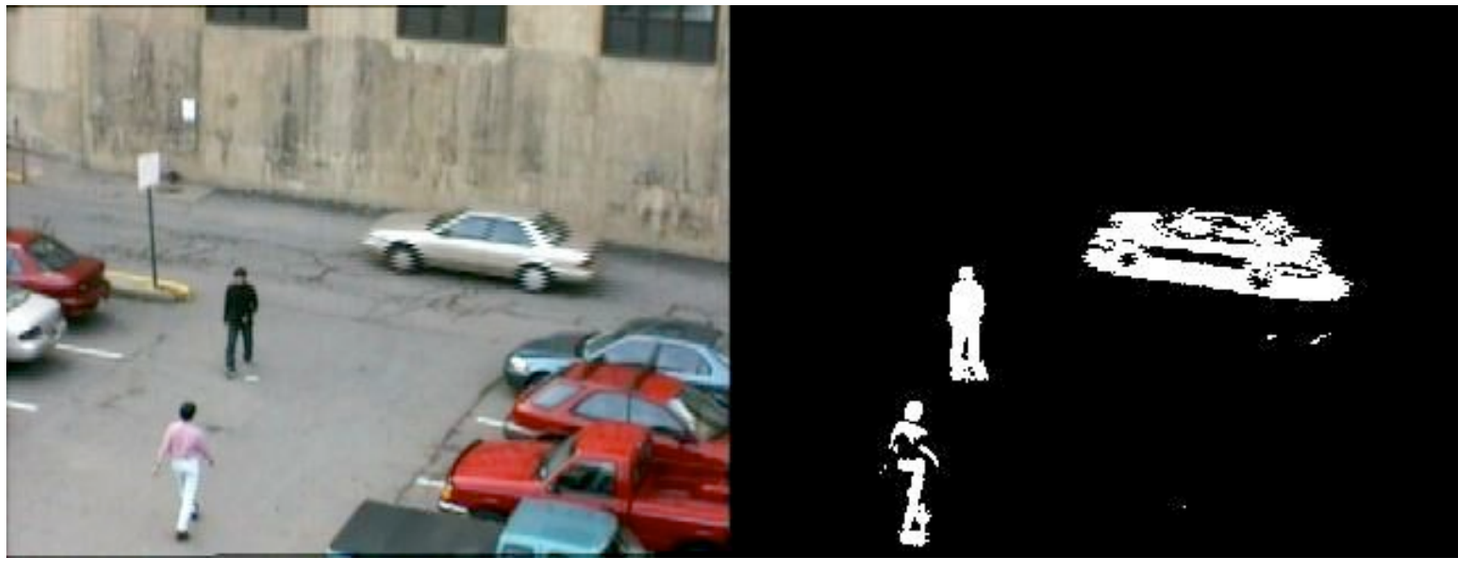
Treat objects as *outliers* when estimating a global homography alignment
(useful approximation for a scene that is mostly the "ground plane")

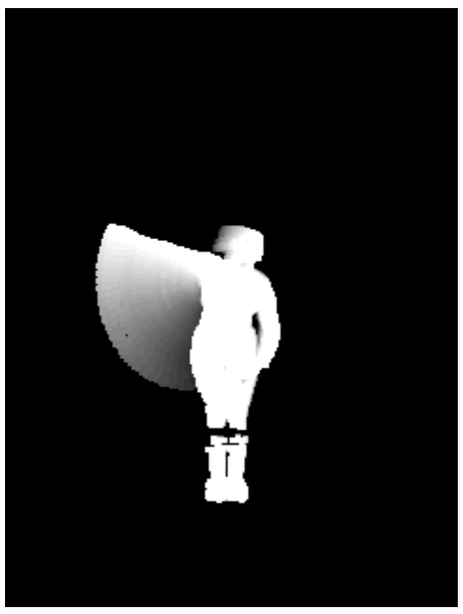


Motion segmentation: special case (II)

Background subtraction

Once we have background image/mosiac (trivial for a stationary camera), how do we identify foreground?

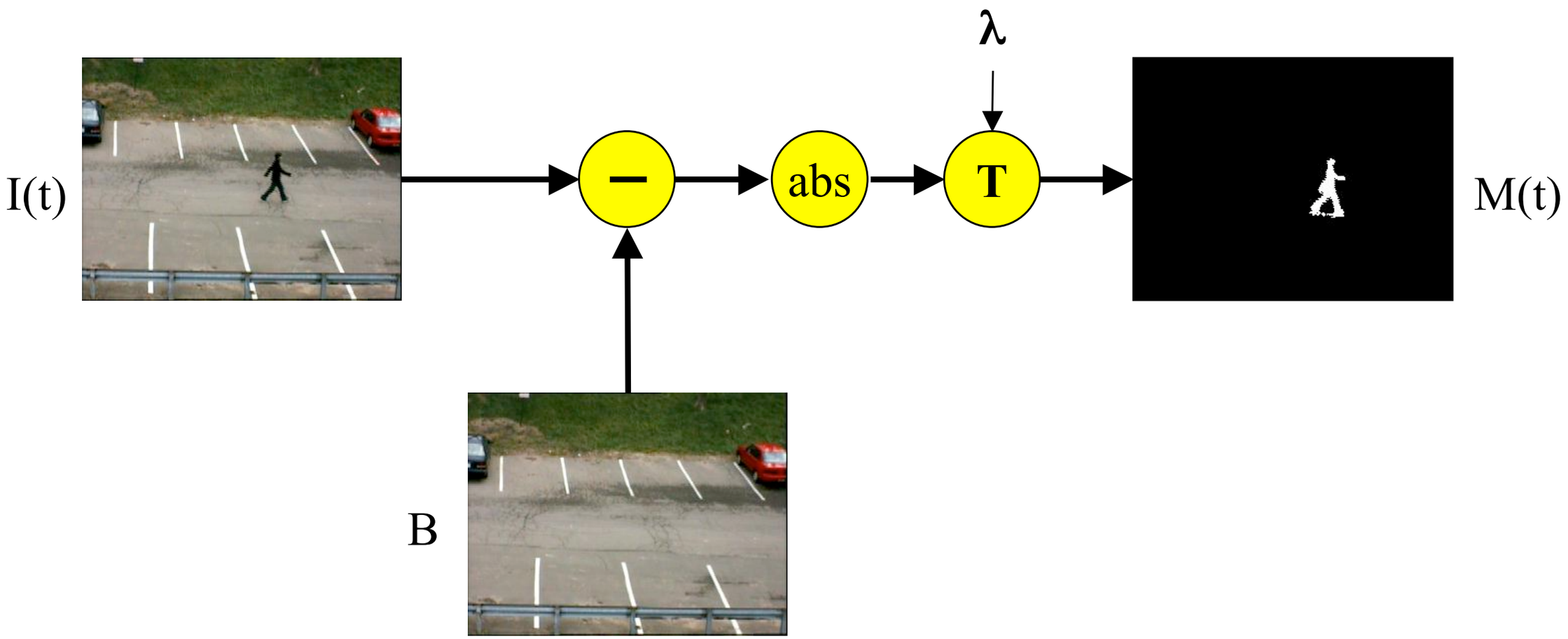




Very commonly-used technique, so we'll spend a few slides on it…

A naive approach

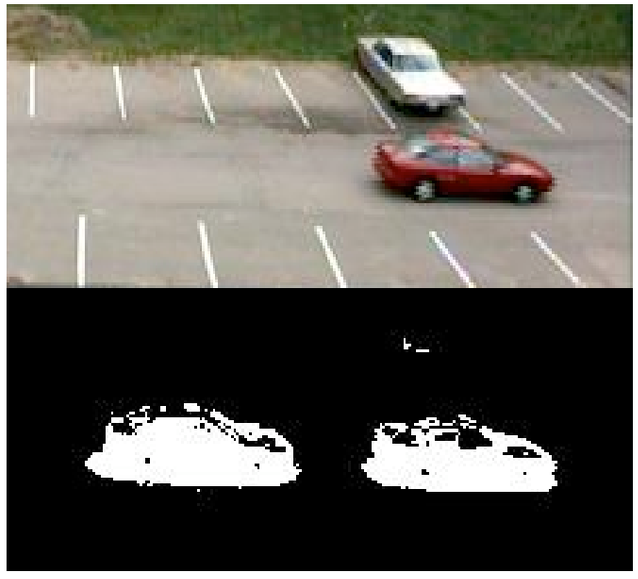
$$M(t) = ||B - I(t)|| > \lambda$$



Simplest approach: assume we have a background image B

Difficulties

Overlapping foreground objects are merged together



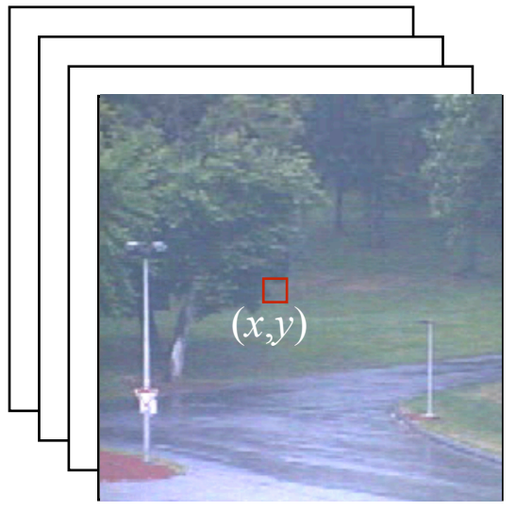
Formerly static objects (that now move) result in ghosting

Senstive to small movements in scene (trees) and changes in illumination (sunlight)

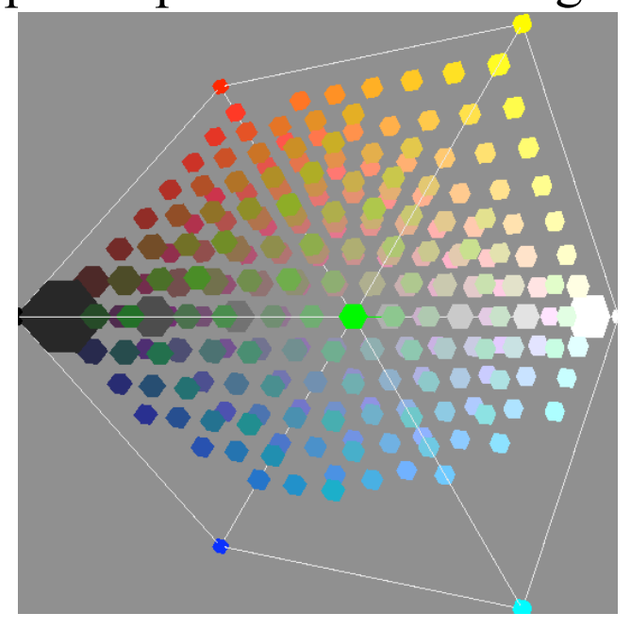
Senstive to small movements of camera

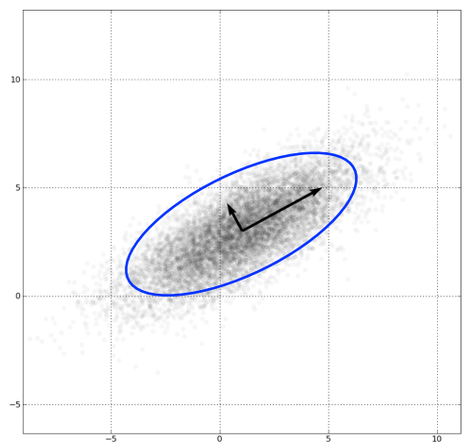
What's a "principled" way to build background model?

Statistical color models: $P(I(x,y)|bg) < \lambda$

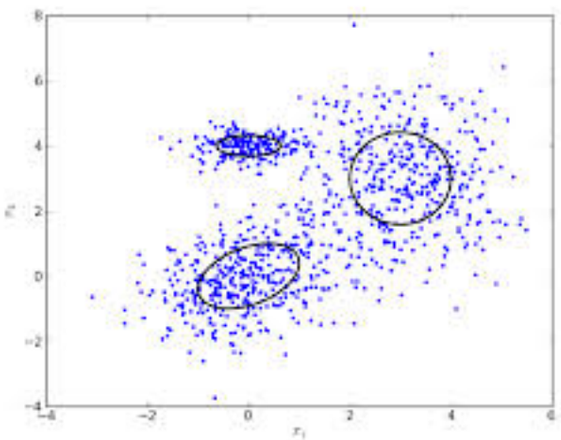


pixel-specific color histogram





$$P(I) = N(I; \mu, \Sigma)$$



$$P(I) = \sum_i \pi_i N(I; \mu, \Sigma)$$

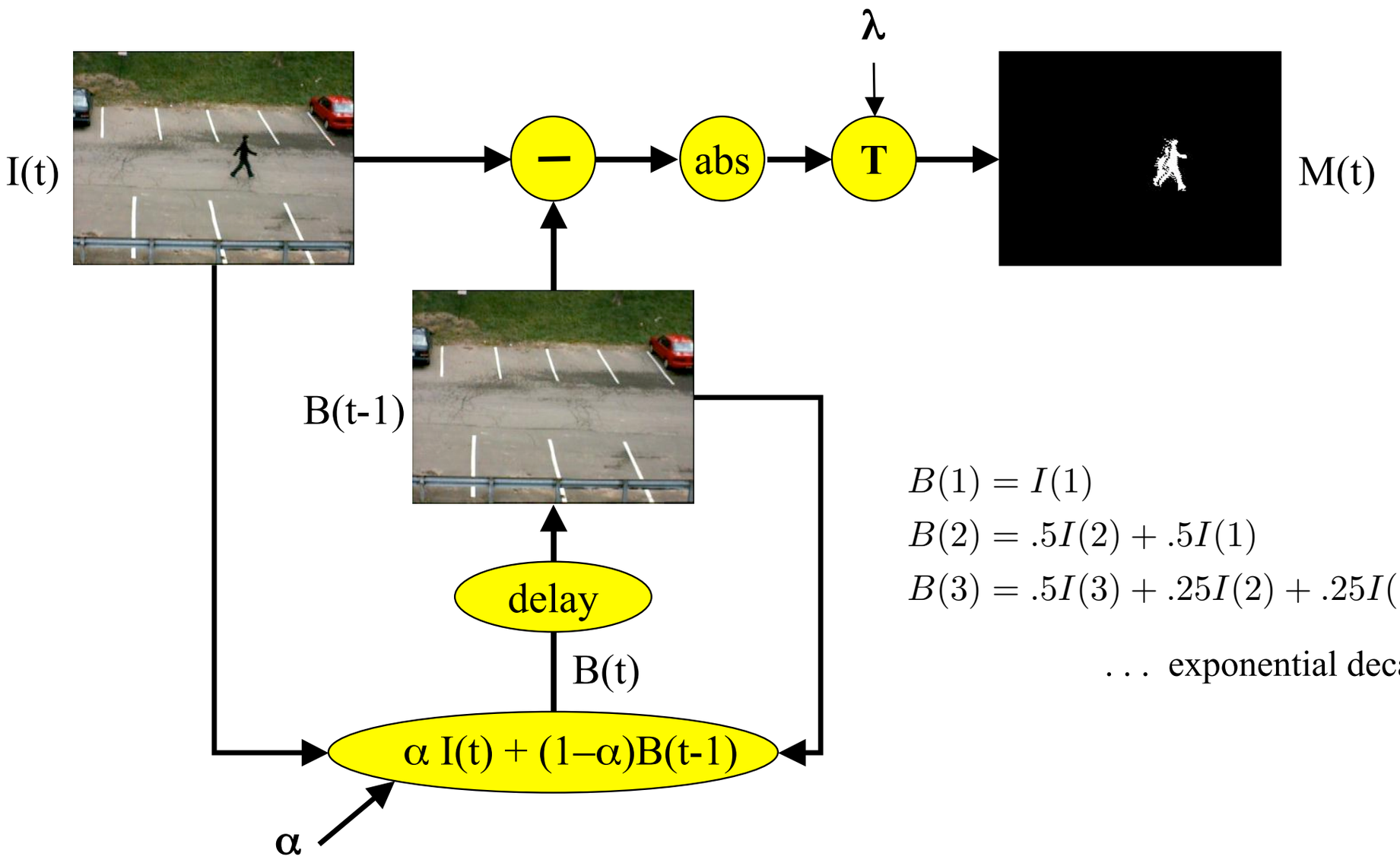
Efficiently implement with o*nline* statistical learning

…of say, mean of distribution at each pixel (x,y)

$$M(t) = [B(t-1) - I(t)] > \lambda$$

$$B(t) = \alpha I(t) + (1-\alpha)B(t-1)$$

alpha=1: frame differencing
alpha=0: fixed (initial) background image



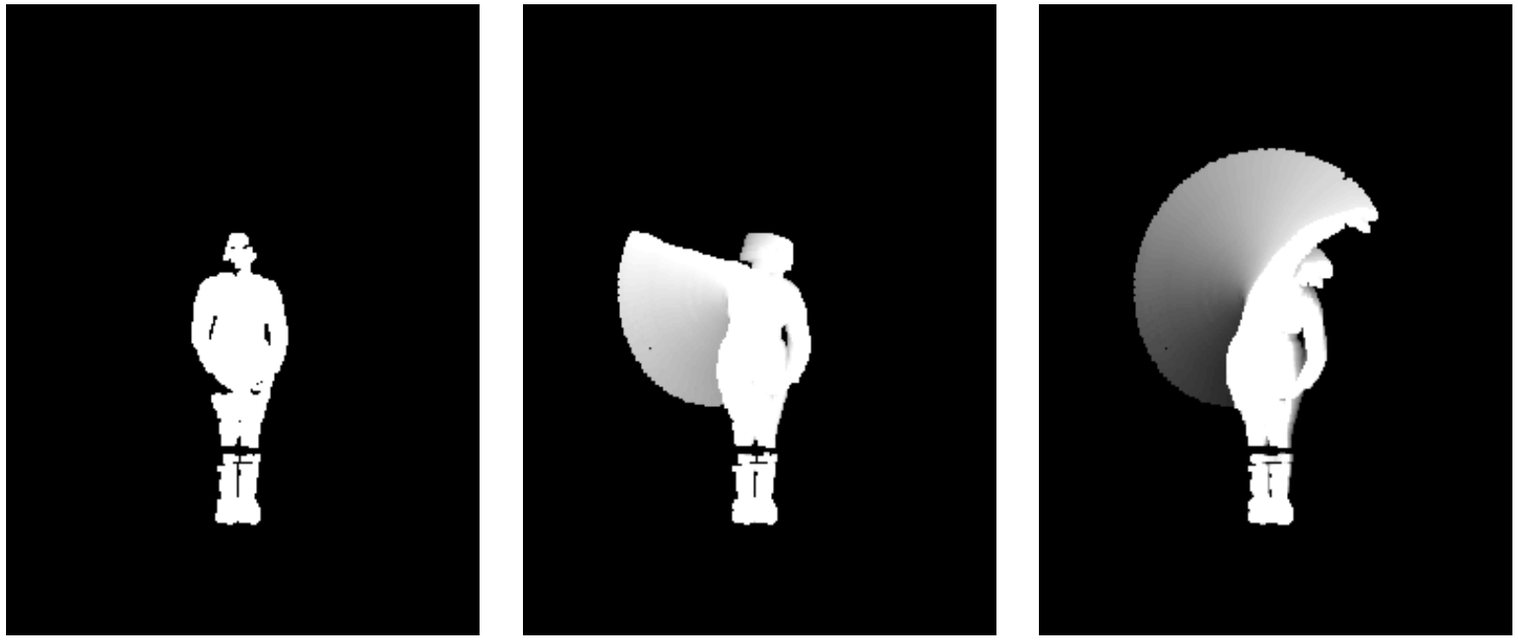
$$B(1) = I(1)$$
$$B(2) = .5I(2) + .5I(1)$$
$$B(3) = .5I(3) + .25I(2) + .25I(1)$$

. . . exponential decay

Adaptive background subtraction

Nifty visualizations: persistant frame differencing

FRAME-0 FRAME-35 FRAME-70



MHI-0 MHI-35 MHI-70

Use some previous method to identify foreground/background pixels

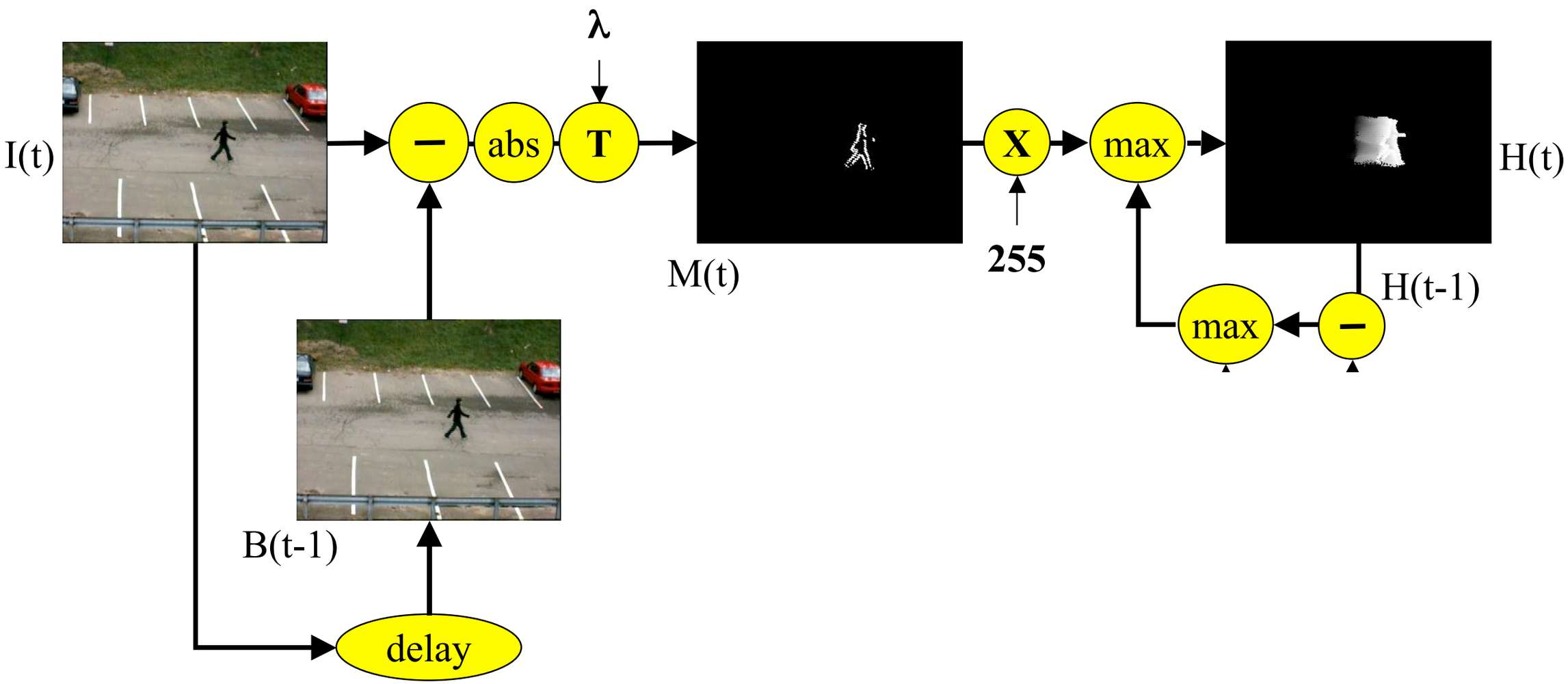
Mark each pixel with the last “time” is was declared foreground

Motion History Images

[Bobick & Davis]

Update history image with current binary mask + old (decremented) history image

$$H(t) = \max(255 * M(t), \max(H(t) - 1, 0))$$

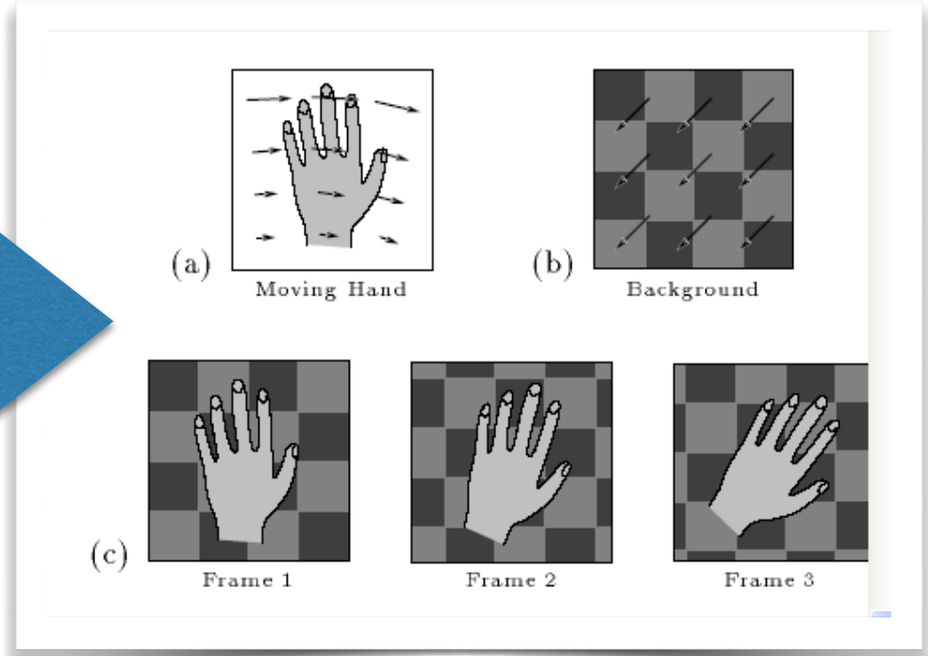


Motion History Images

Outline

- Lucas Kanade
- Egomotion
    - Motivation
    - Time-to-Contact, Parallax, Focus-of-Expansion
- Optical Flow
    - Motivation, aperture problem
    - Sparse (KLT) vs Dense (variational)
    - Optimization tools: varational coarse-to-fine, markov-random feilds
    - Segmentation (dominant motion estimation, background subtraction, **layered models**)

Layered “2.1 D” model

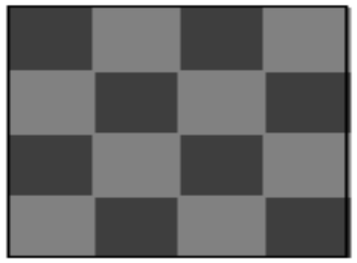


Direct analogy with *layers* in photoshop

Mathematical formalism for layering: alpha compositing

Need an RGB image and alpha-mask (binary *or soft*) for each layer, along with an order of layers back-to-front

Layer 0 (BG)



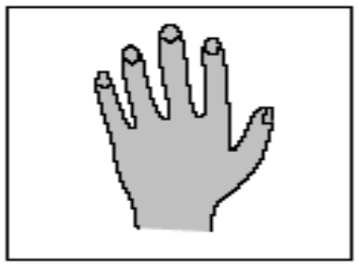
Intensity map

L_0

Alpha map

α_0

Layer 1



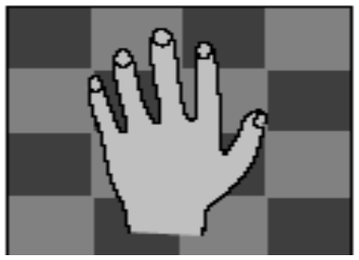
Intensity map

L_1

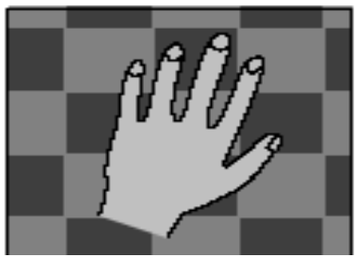
Alpha map

α_1

Alpha composite



I_1

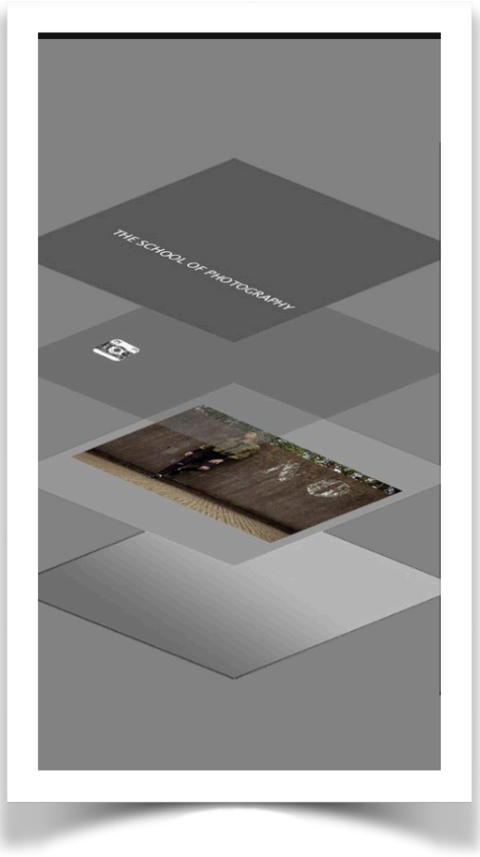


for i = 0:num_layers

$$I_i = \alpha_i L_i + (1 - \alpha_i) I_{i-1}$$

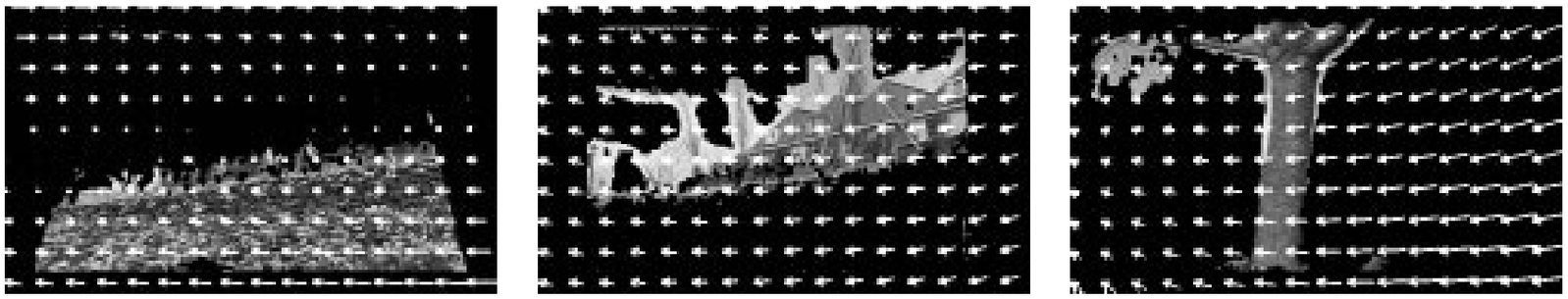
% iterate from back to front

% $bg_{new} = \alpha * fg + (1-\alpha) bg_{old}$



Representing Moving Images with Layers

John Y. A. Wang AND Edward H. Adelson



(a) (b) (c)

Figure 12: The layers corresponding to the tree, the flower bed, and the house shown in figures (a-c), respectively. The affine flow field for each layer is superimposed.

(a) (b) (c)

Figure 13: Frames 0, 15, and 30 as reconstructed from the layered representation shown in figures (a-c), respectively.

(a) (b) (c)

Figure 14: The sequence reconstructed without the tree layer shown in figures (a-c), respectively.

Inferring layers, motion, and appearance with EM clustering

Learning Flexible Sprites in Video Layers

Nebojsa Jojic
Microsoft Research
http://www.ifp.uiuc.edu/~jojic

Brendan J. Frey
University of Toronto
http://www.psi.toronto.edu

Outline (look back)

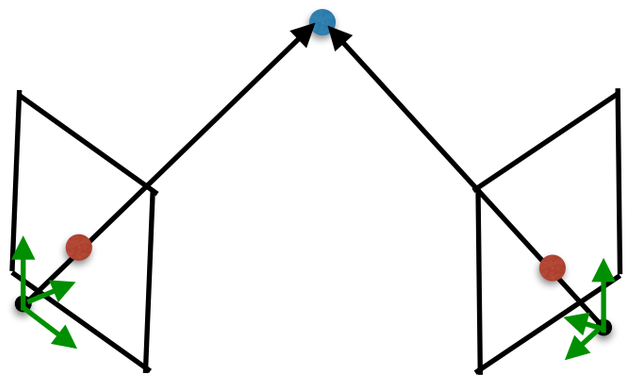
- Lucas Kanade
- Egomotion
    - Motivation
    - Time-to-Contact, Parallax, Focus-of-Expansion
- Optical Flow
    - Motivation, aperture problem
    - Sparse (KLT) vs Dense (variational)
    - Optimization tools: varational coarse-to-fine, markov-random feilds
    - Segmentation (dominant motion estimation, background subtraction, layered models)

Outline (looking forward)

- two-view geometry
- geometric intuition
- essential matrix, fundamental matrix
- properties
- estimation

Multi-view geometry

Three questions:

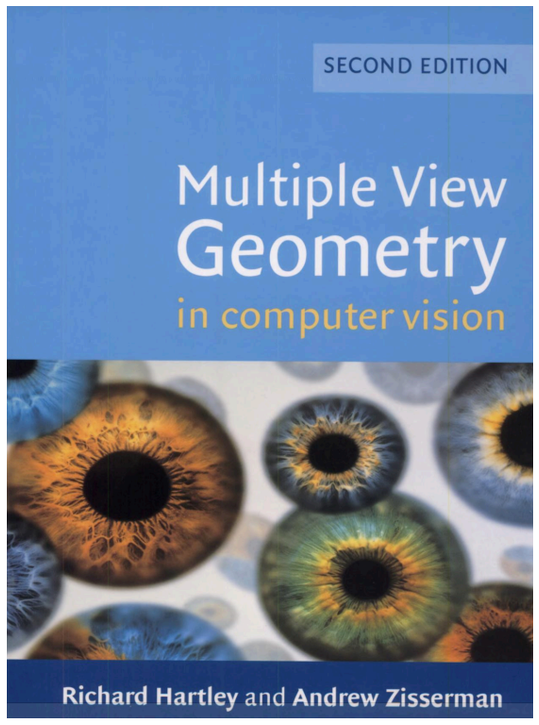


(i) **Correspondence:** Given an image point $\mathbf{x}$ in the first view, find its corresponding point in the second view. (HW3)

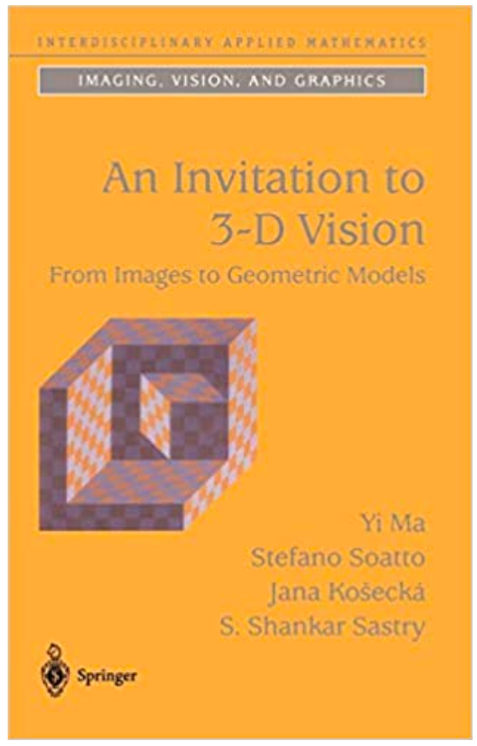
(ii) **Camera pose (motion):** Given a set of corresponding image points $\{x_i \leftrightarrow x'_i\}$, i=1,…,n, what are the cameras M and M' for the two views? (Focus of today)

(iii) **Scene geometry (structure):** Given corresponding image points $x_i \leftrightarrow x'_i$ and cameras M, M', what is the position of the underlying 3D point X? (Focus of next lecture)

References



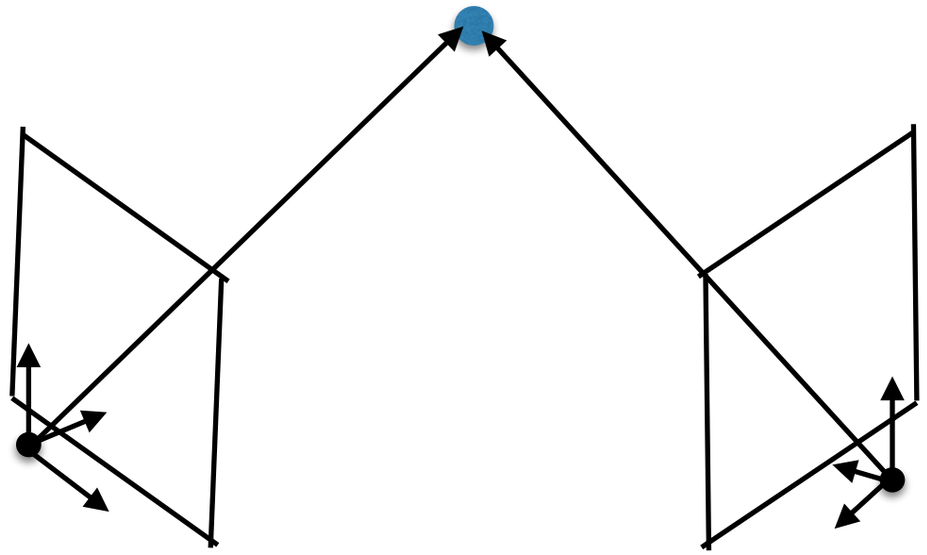
Geometry “bible”



My favorite reference
for “geometry-vision”

Two-view (stereo)

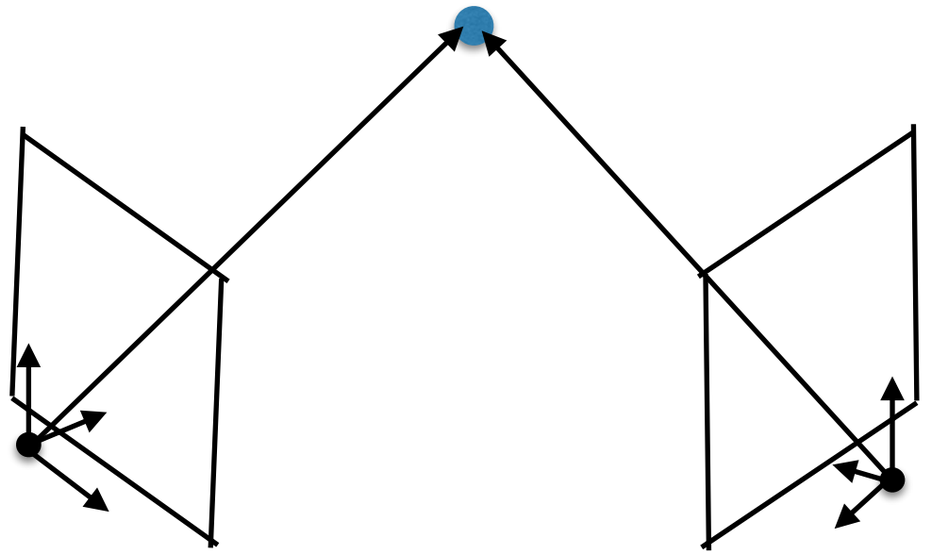
Goal: let's build intuition with pictures rather than math



Given 2 corresponding points in 2 cameras, see where the cast rays meet

Much of basics can be derived from this picture

Triangulation

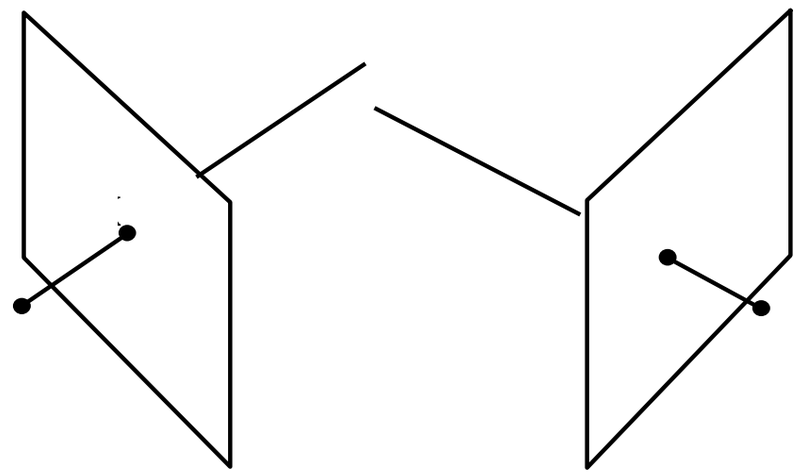


$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \equiv \begin{bmatrix} m'_{11} & m'_{12} & m'_{13} & m'_{14} \\ m'_{21} & m'_{22} & m'_{23} & m'_{24} \\ m'_{31} & m'_{32} & m'_{33} & m'_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Given points (x,y) and (x',y') and cameras M and M',
solve for (X,Y,Z) with homonogenous least squares
(similar to Direct Linear Transform; you'll solve for homework!)

An annoying “detail”

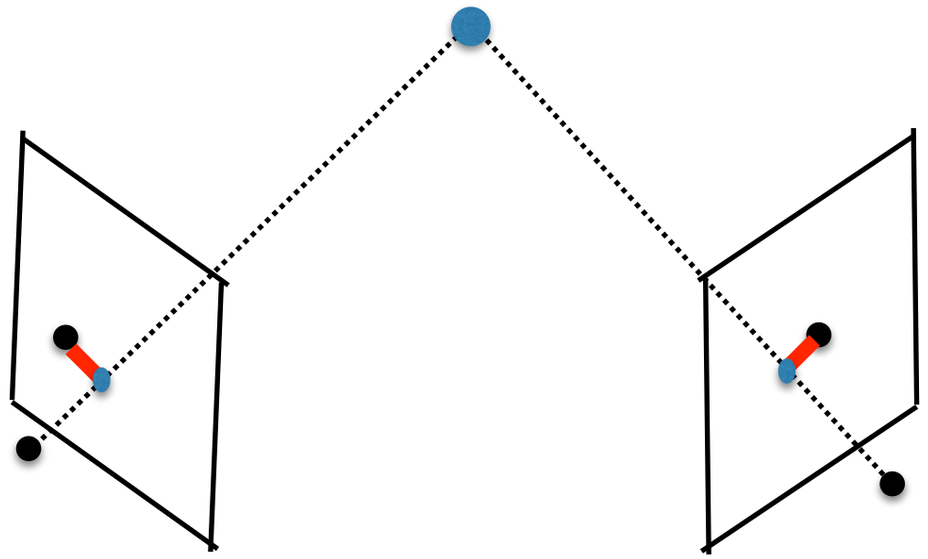


What if rays don’t intersect?

Option 1. Find algebraic solution with Direct Linear Transform (previous slide)

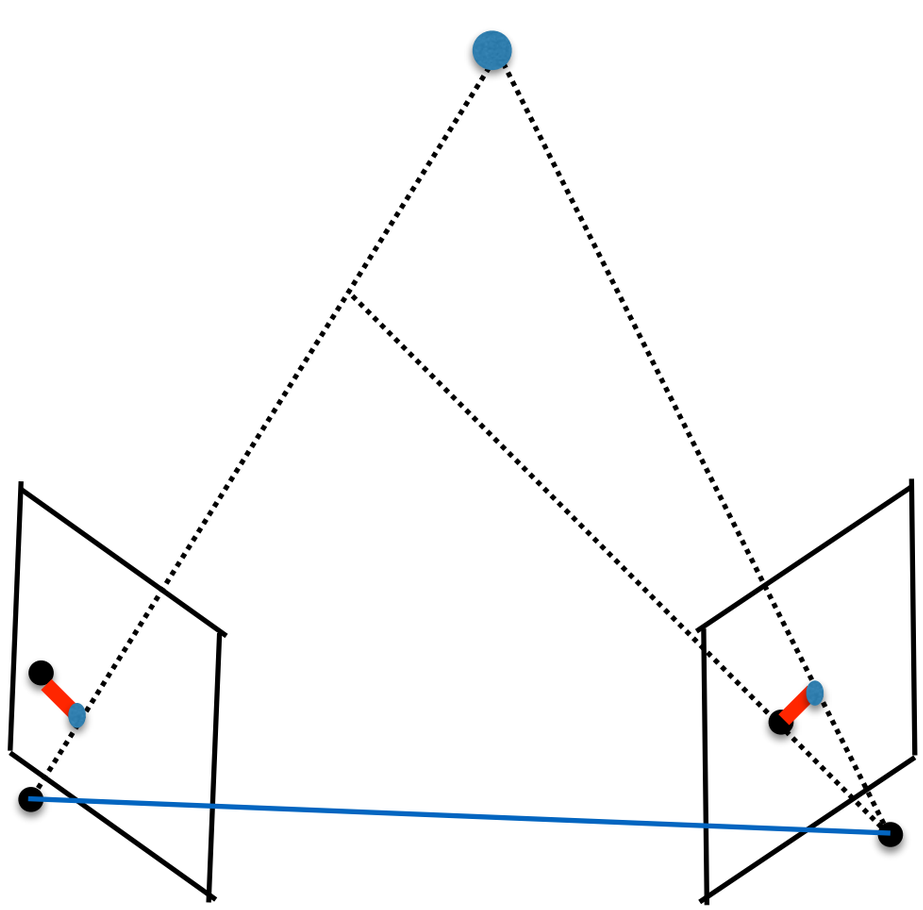
What’s the right error to minimize?

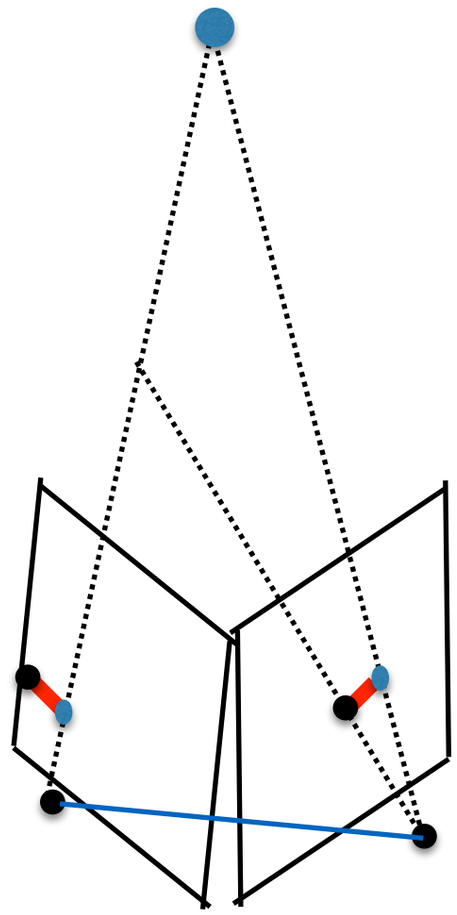
“Right” solution



Find 3D point with 2D low reprojection error

Numerical stability

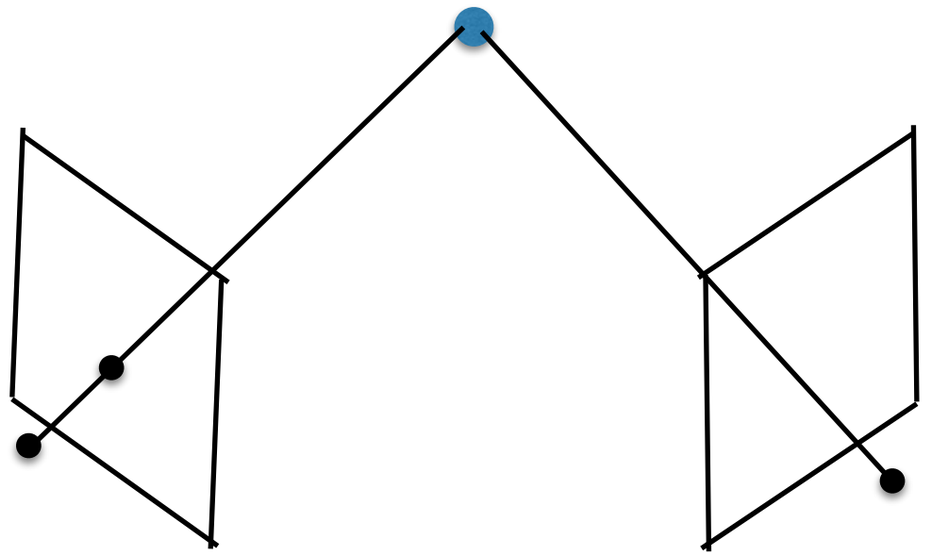




Which setup produces noisier estimates of depth (as a function of image noise)?
small distances between cameras or *baselines* (right)

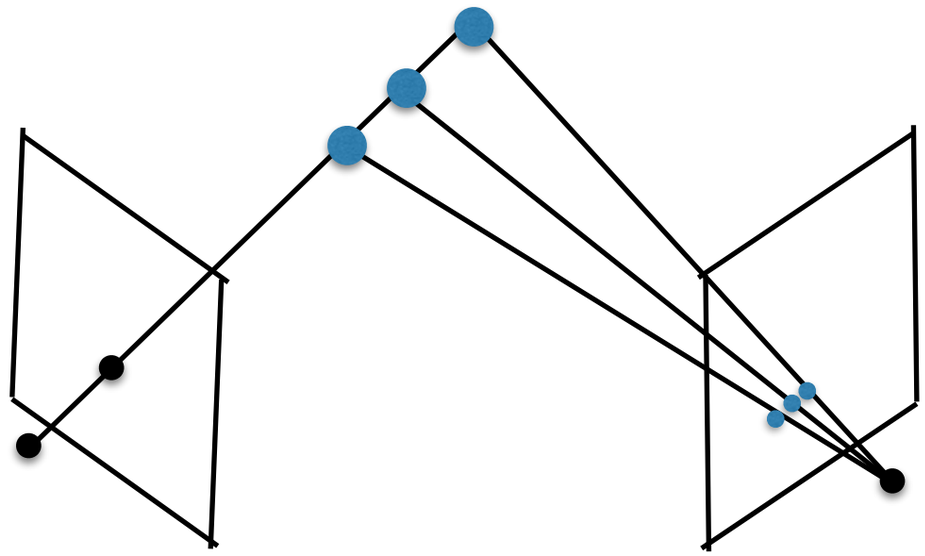
Which setup will produce points that are harder to match?
large baselines (left)

Questions



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

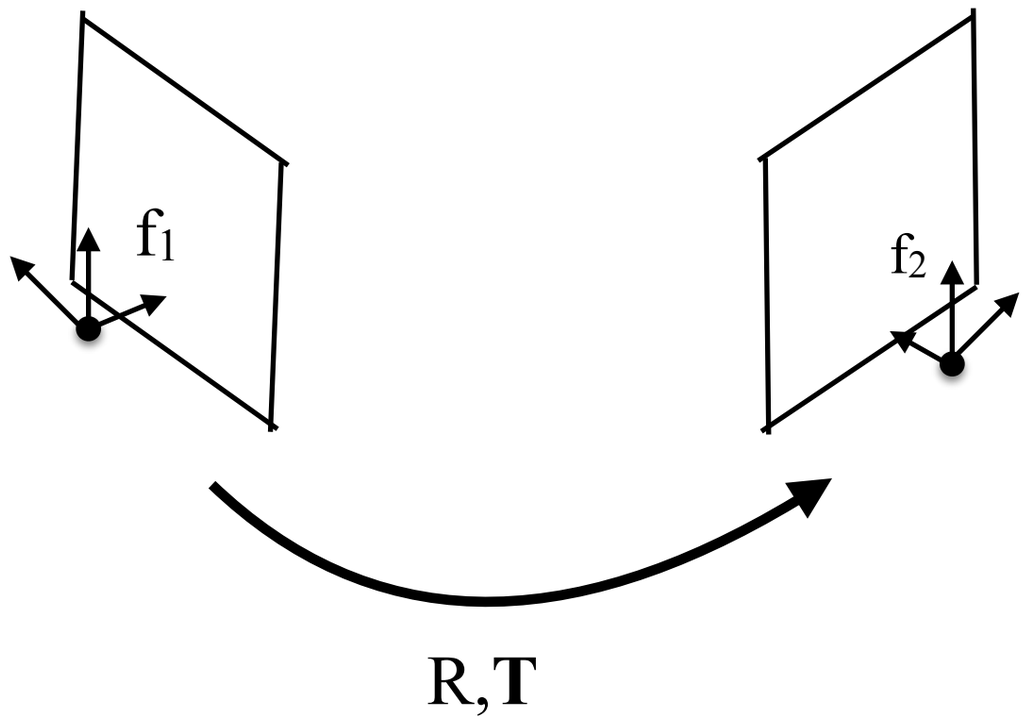
Questions



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

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

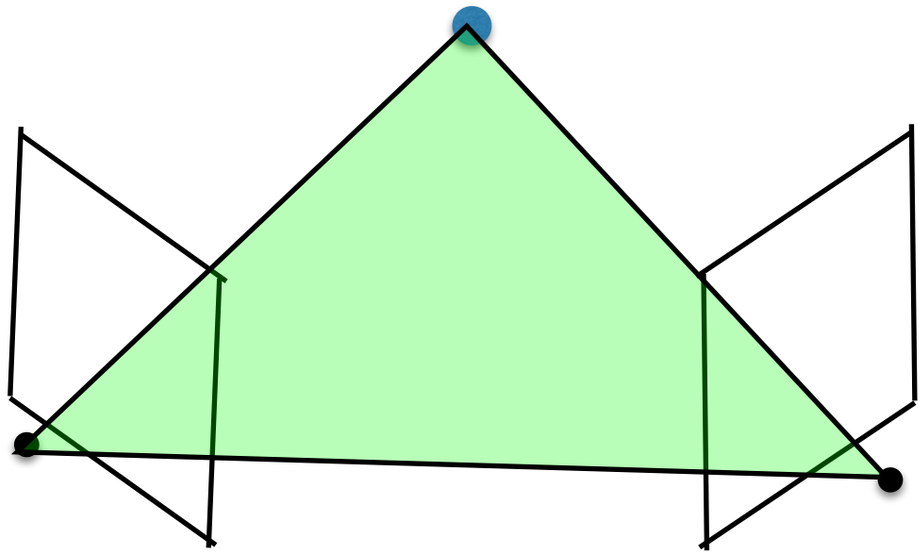
Epipolar geometry



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
(I'll repeat this alot)

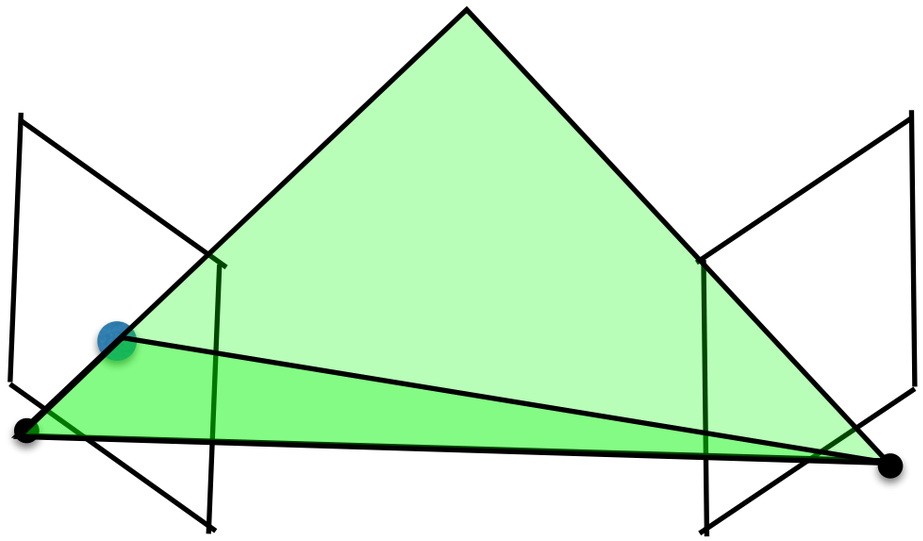
Definitions



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



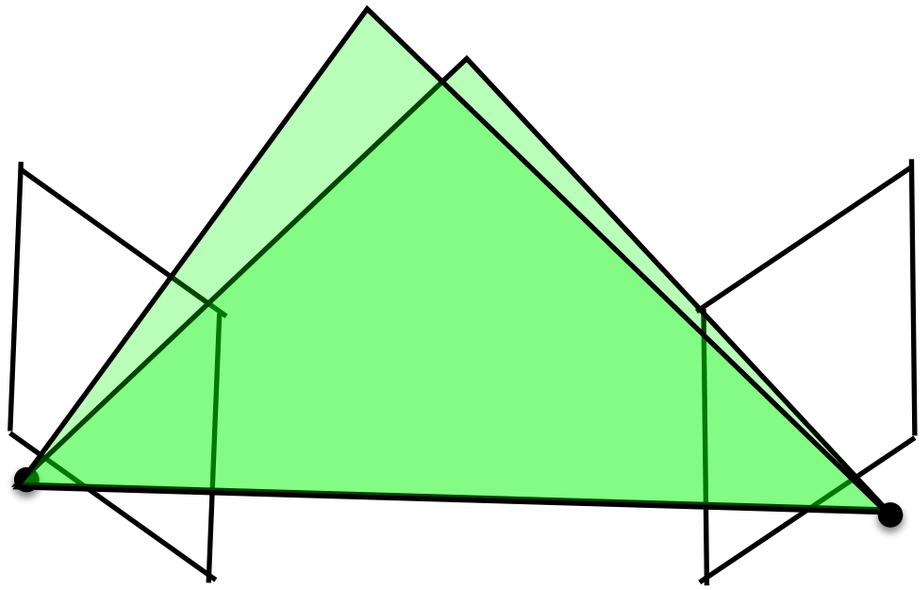
Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

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

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

Epipolar geometry depends on direction of T but not it's length

Definitions

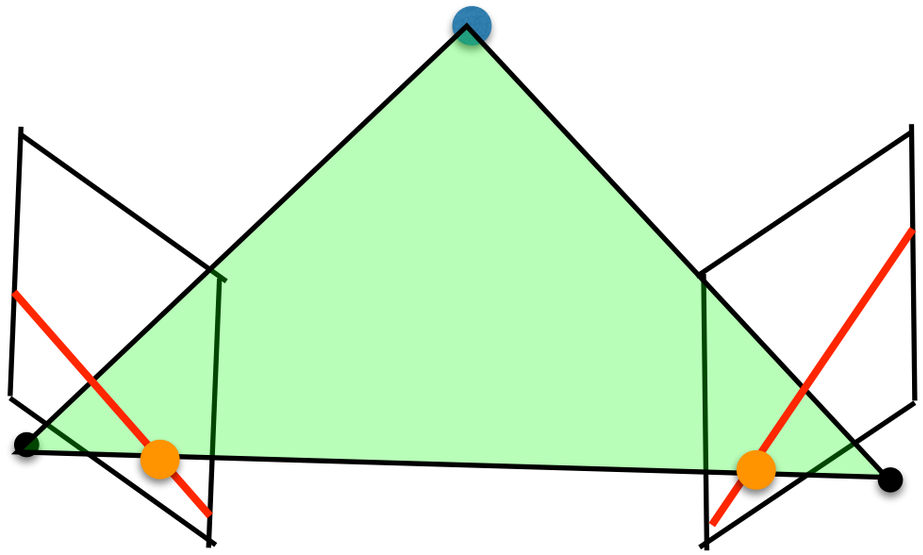


Epipolar plane: plane defined by 2 camera centers & candidate 3D point (green)

How large is the *family* of epipolar planes?

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

Definitions



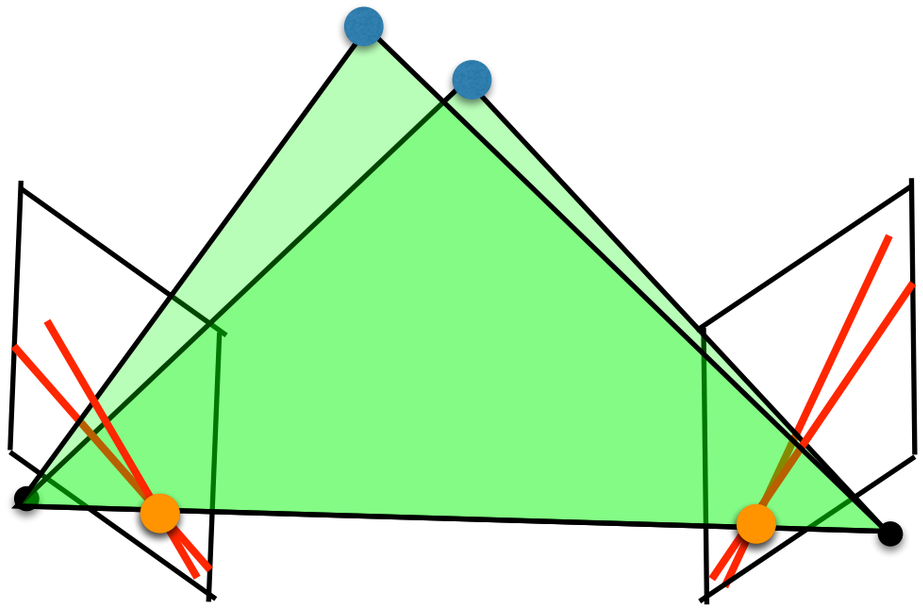
Epipolar plane: plane defined by 2 camera centers & candidate 3D point

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

Epipolar lines: intersection of epipolar plane and image planes

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

Definitions



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

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

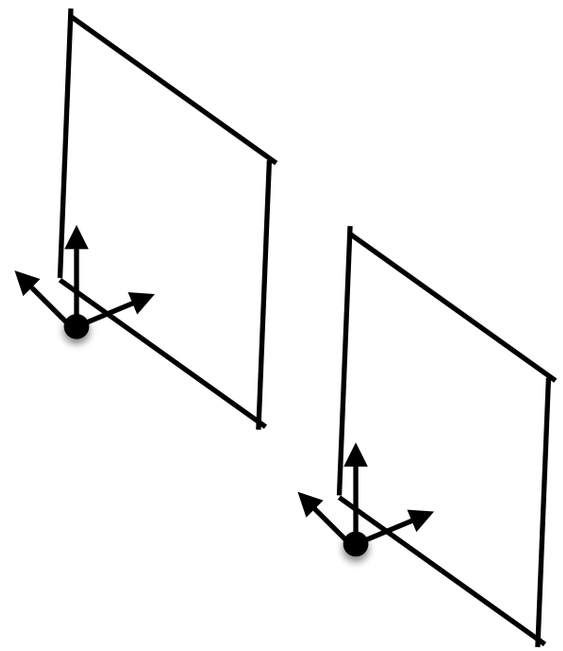
Epipolar lines: intersection of epipolar plane and image planes

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

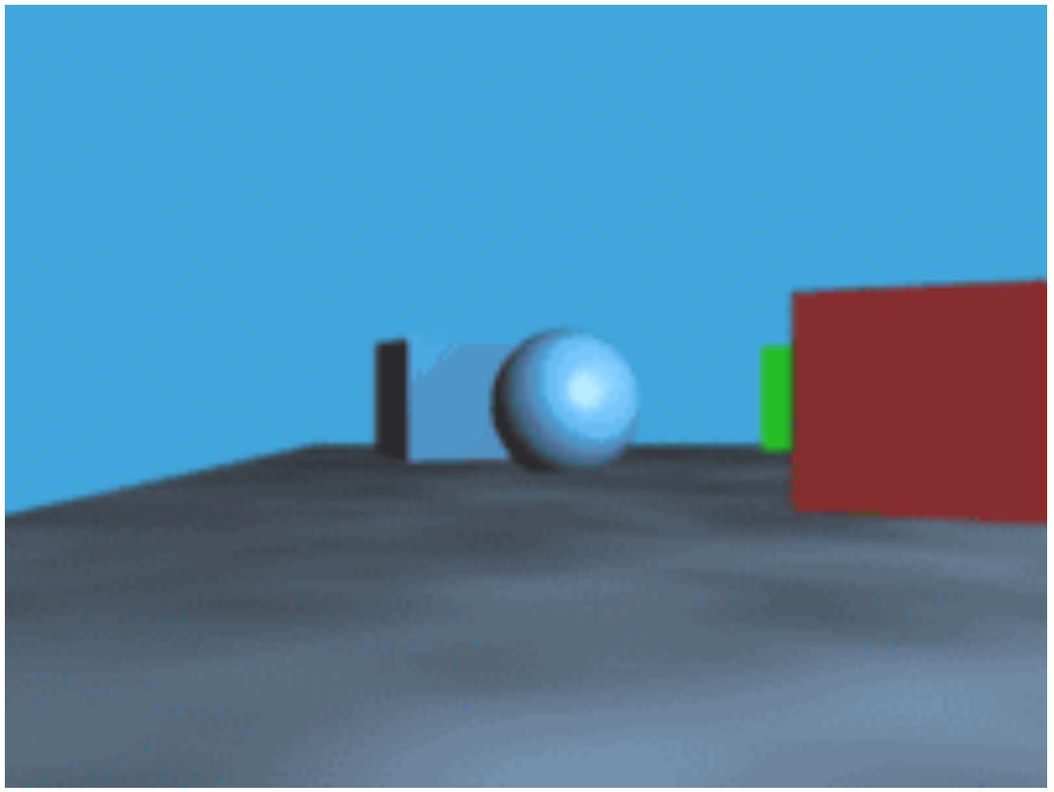
(set of all epipolar lines intersect at the epipoles)

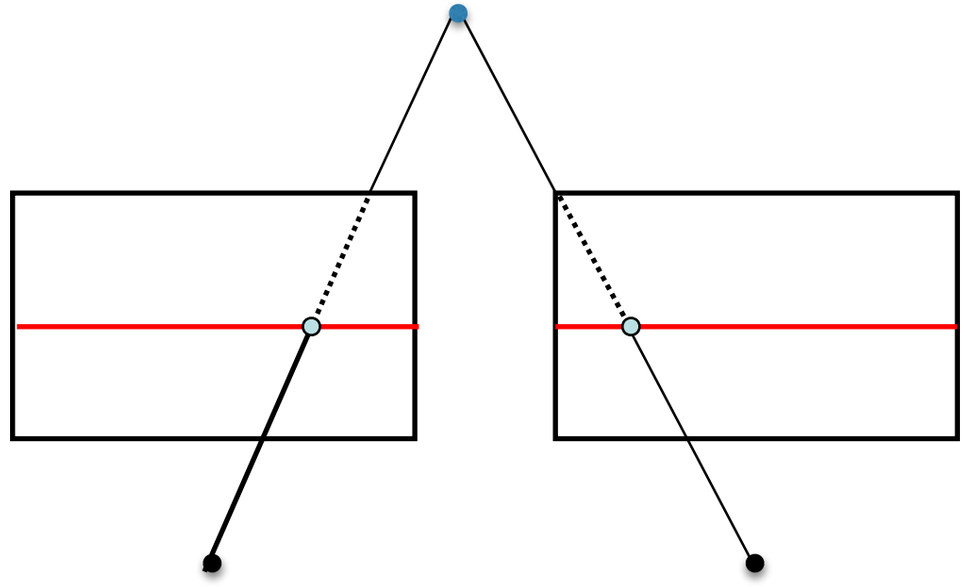
Special case (I)

Parallel, offset cameras (T,R = 0 except for T_x)



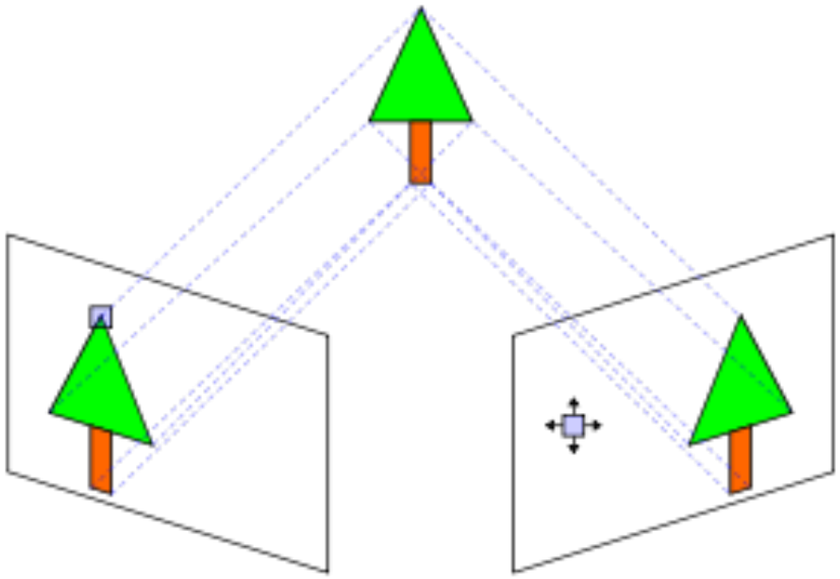
What would epipolar lines look like?



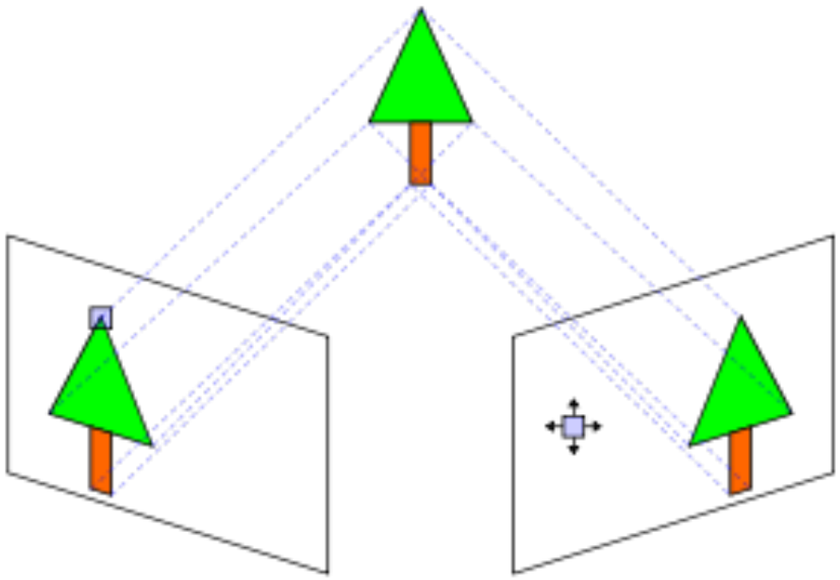


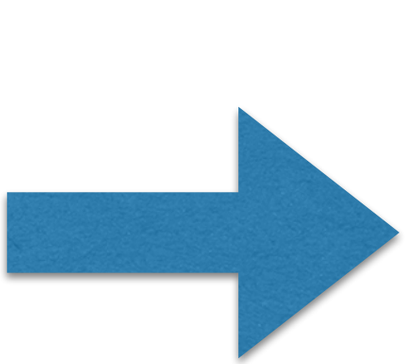
Epipolar lines don’t intersect (are parallel)
Epipoles are at *infinity* (derive by rotating image planes)

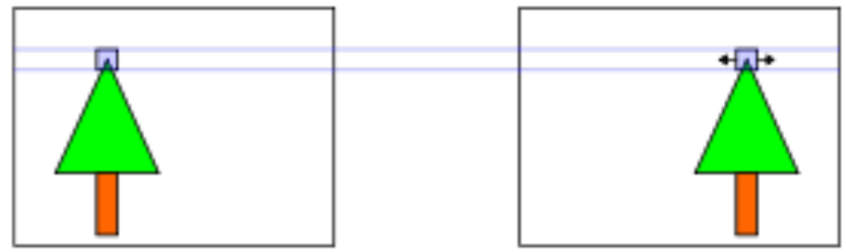
What happens to epipolar lines when we double the distance between two camera views?



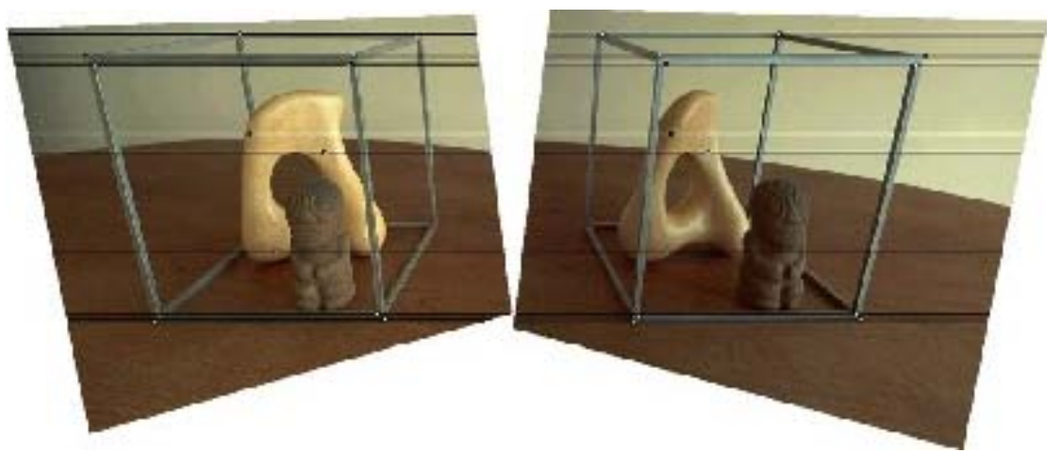
Stereo Pair







Stereo Pair

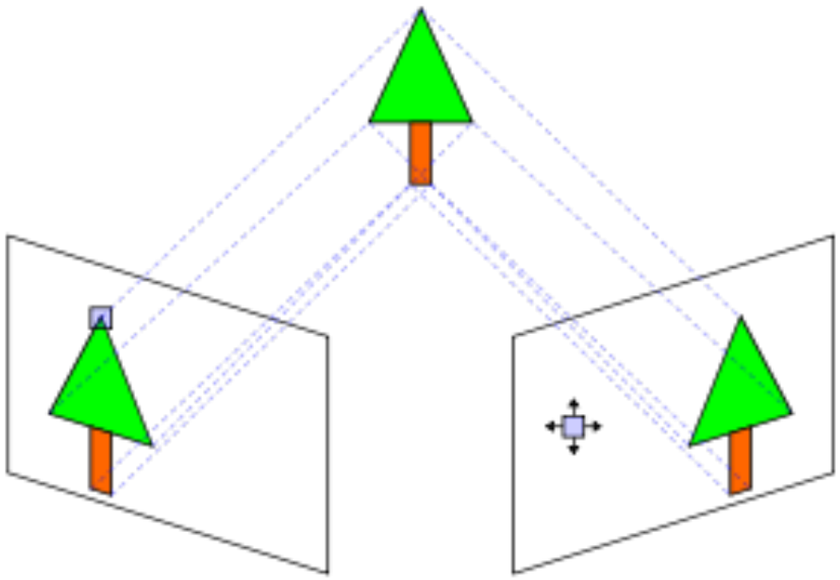


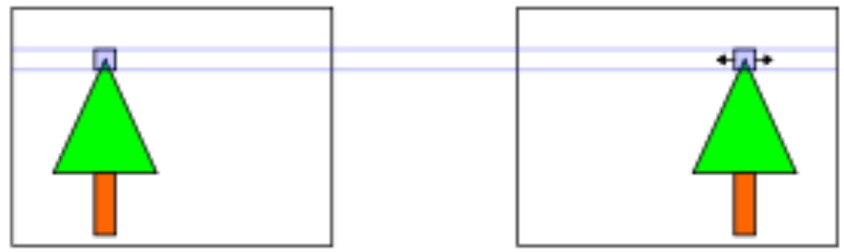
Rectified Stereo Pair

Rotate image plane about fixed camera centers

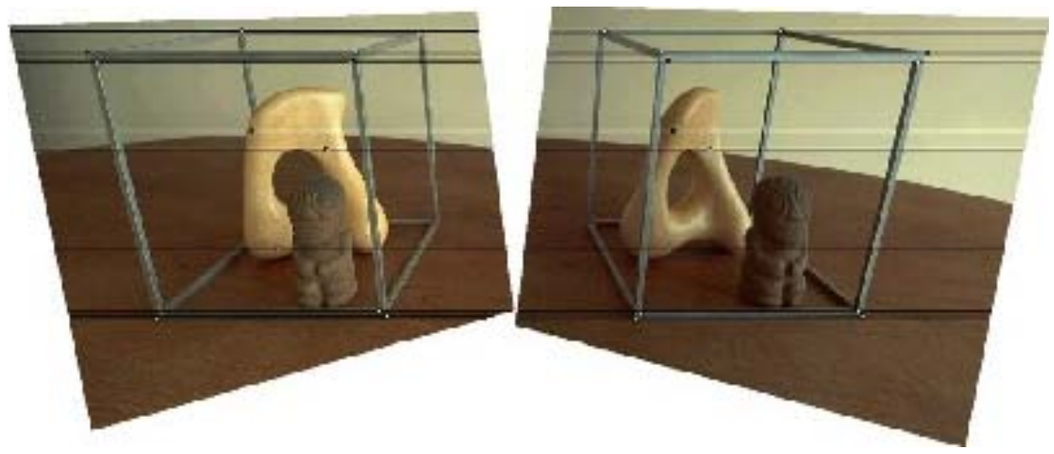
Aside: what kind of transformation is this?

Homography





Stereo Pair



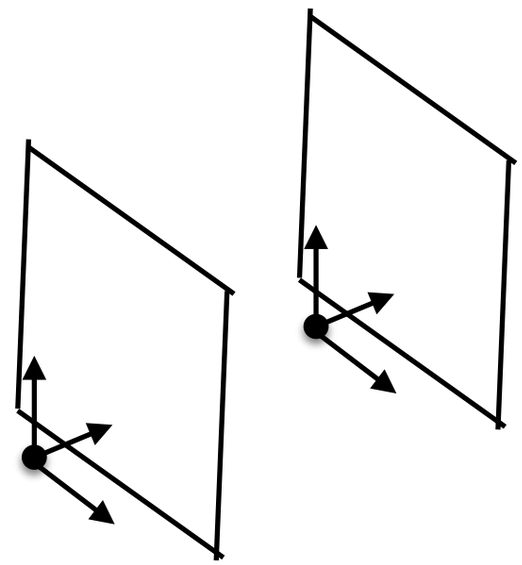
Rectified Stereo Pair

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

Epipolar geometry is purely determined by camera extrinsics and camera instrinics

Special case (II)

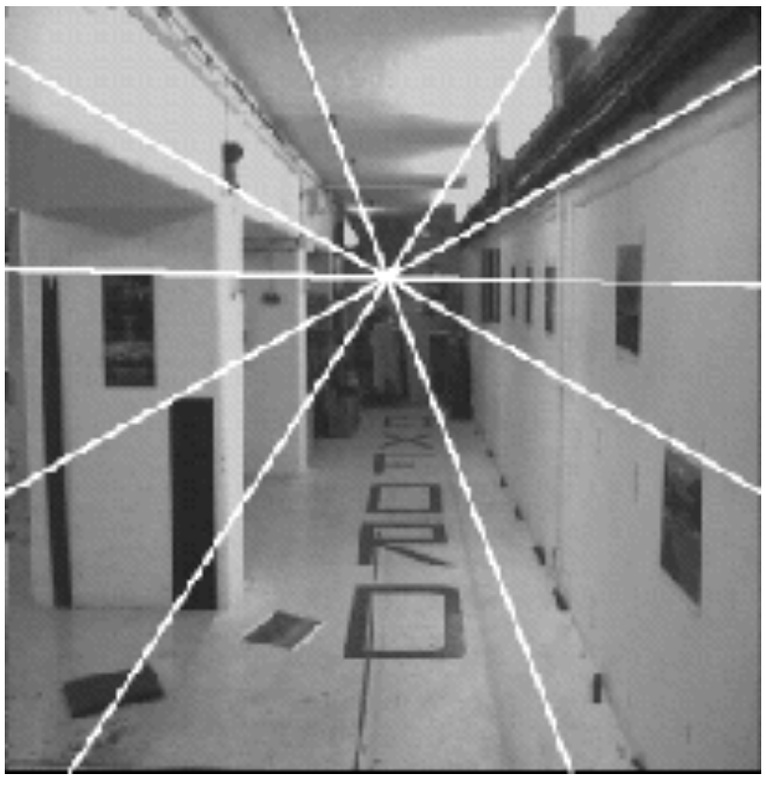
Forward camera motion (T,R = 0 except for T_z)

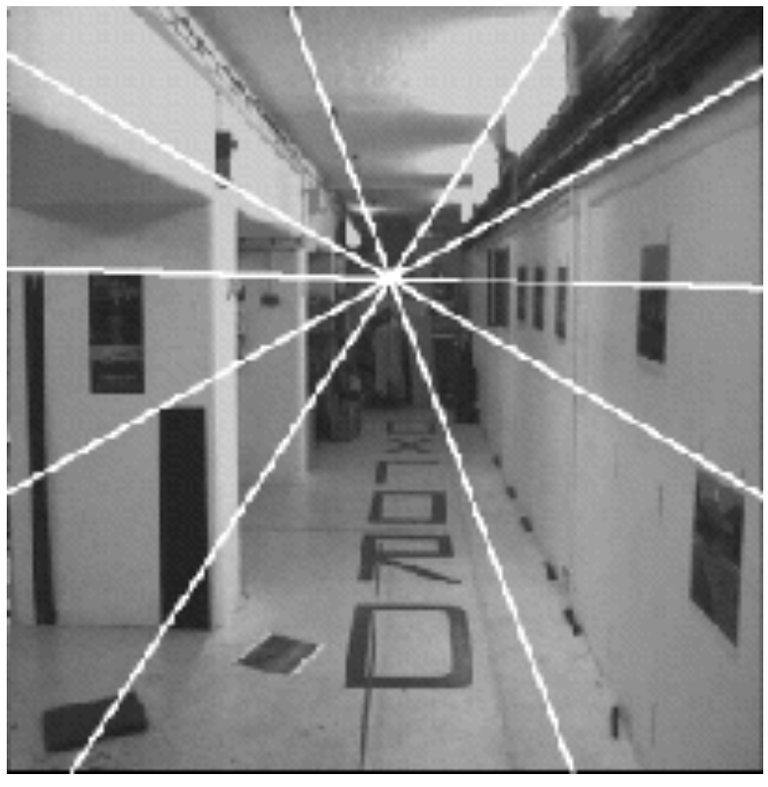


What would epipolar lines look like?

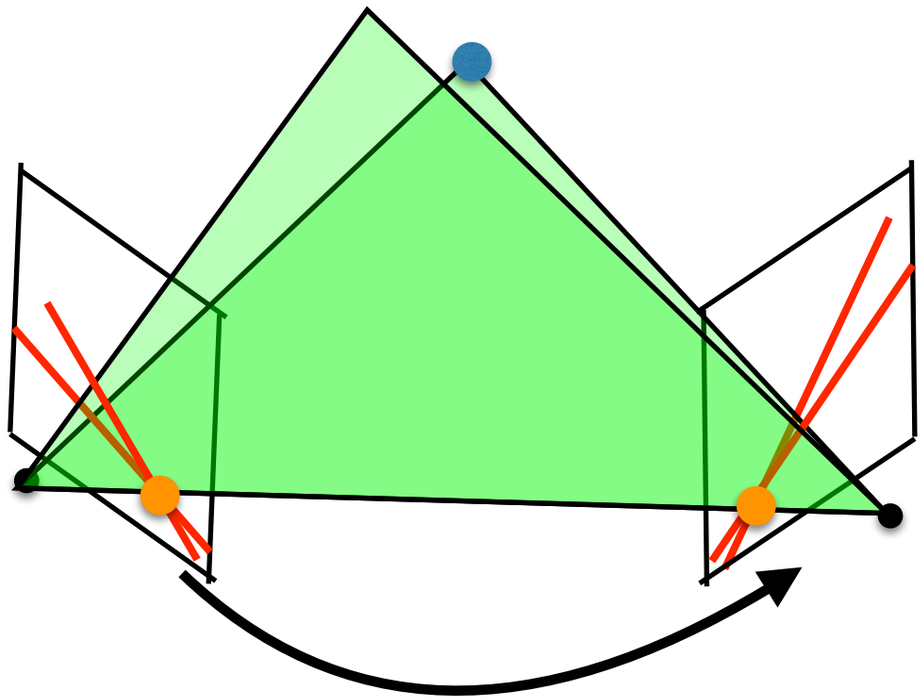
Special case (II)

Forward camera motion





Key observations from geometric intuition



R,**T**

- 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)

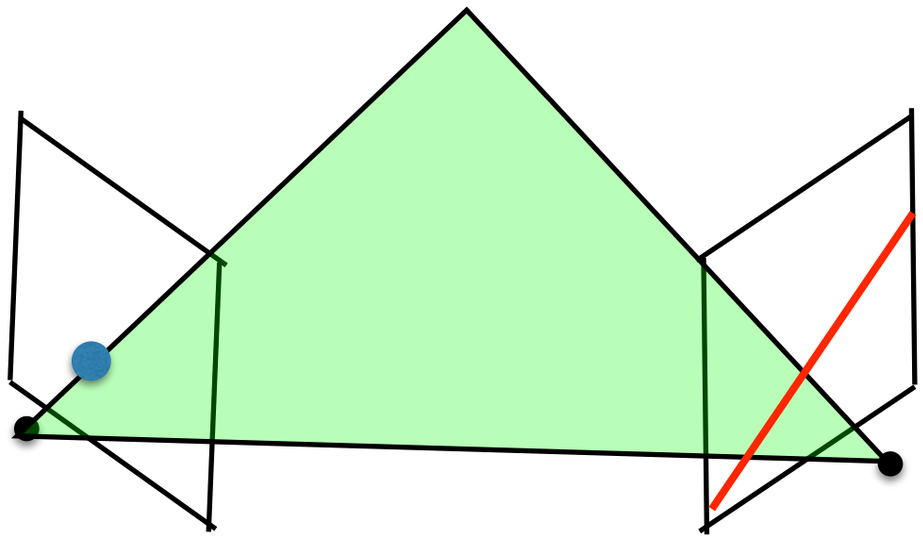
Questions:
Are there *other* points on-the-left that map to the same epipolar line-on-the-right?
If so, how does one characterize the *set* of points-on-the-left that map to the same line-on-the-right?

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

Roadmap

- 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 line on-the-right

Recall notation

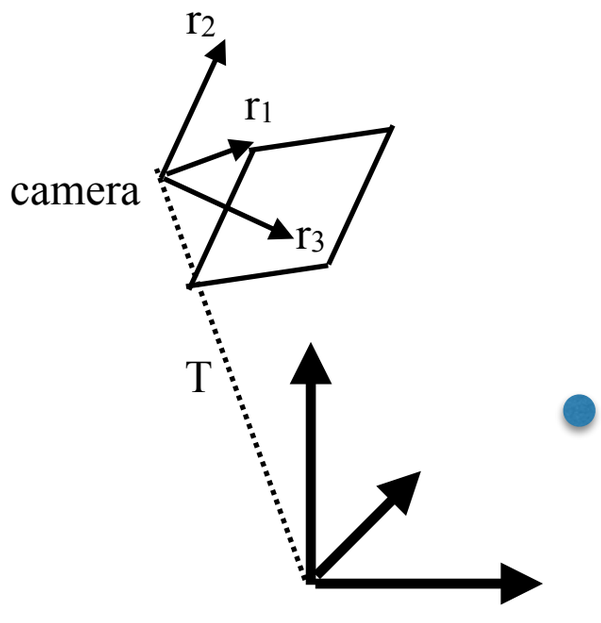
Camera instrinsics K

Camera extrinsics
(rotation and translation)

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} fs_x & fs_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

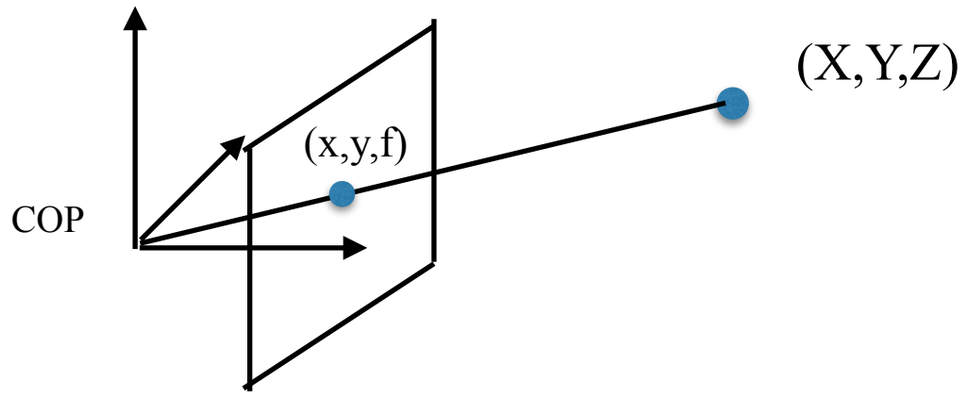
3D point in
world coordinates

$$= K_{3\times3} \begin{bmatrix} R_{3\times3} & T_{3\times1} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



world coordinate frame

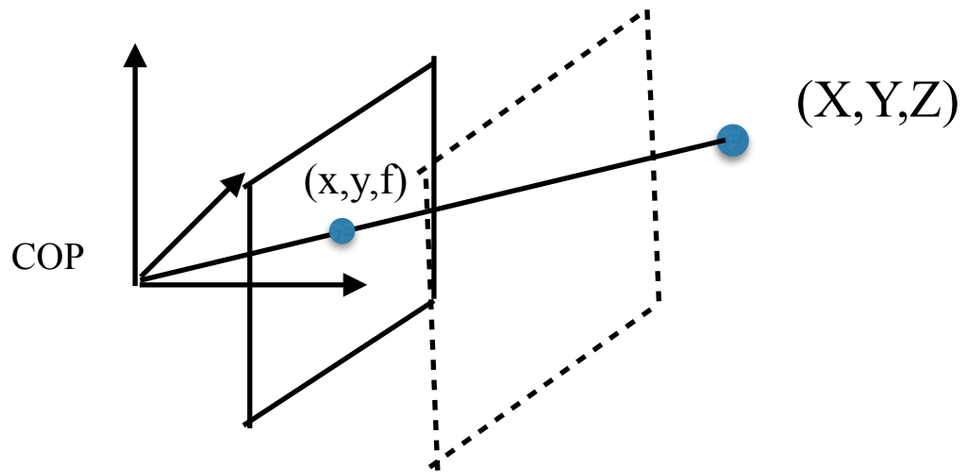
Projecting from camera coordinates to image coordinates



$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} fs_x & fs_\theta & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

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

Recall: from camera coordinates to *normalized* image coordinates



For now, assume K is known.
This means we can work with a *warped* image who's intrinsic matrix is the identity.

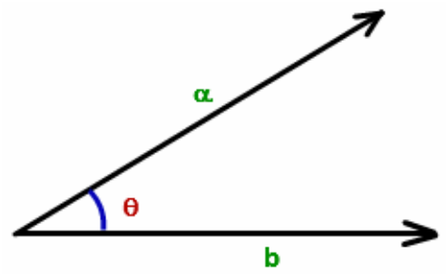
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = K^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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

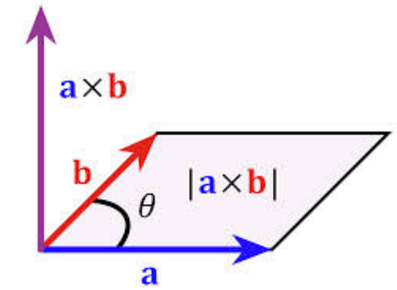
To simplify notation, we'll use **x** instead of **x'**

Recall

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



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



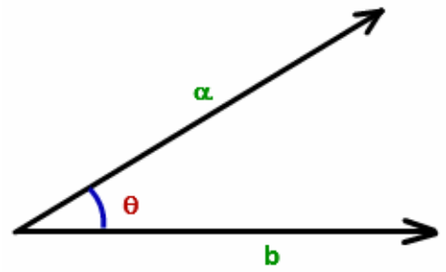
Cross product matrix: $\mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \equiv \hat{\mathbf{a}}\mathbf{b}$

$$\mathbf{b} \times \mathbf{a} = -\hat{\mathbf{a}}\mathbf{b}$$

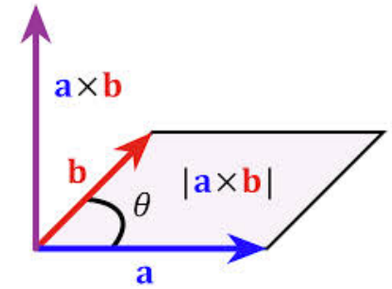
Important property (skew symmetric): $\hat{\mathbf{a}}^T = -\hat{\mathbf{a}}$

Recall

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



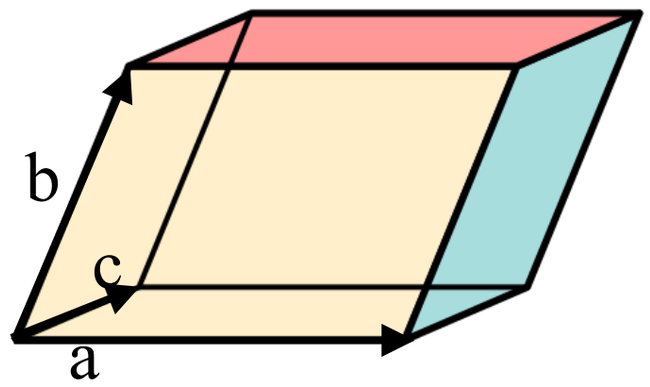
Cross product: $\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \, \|\mathbf{b}\| \sin\theta \, \mathbf{n}$



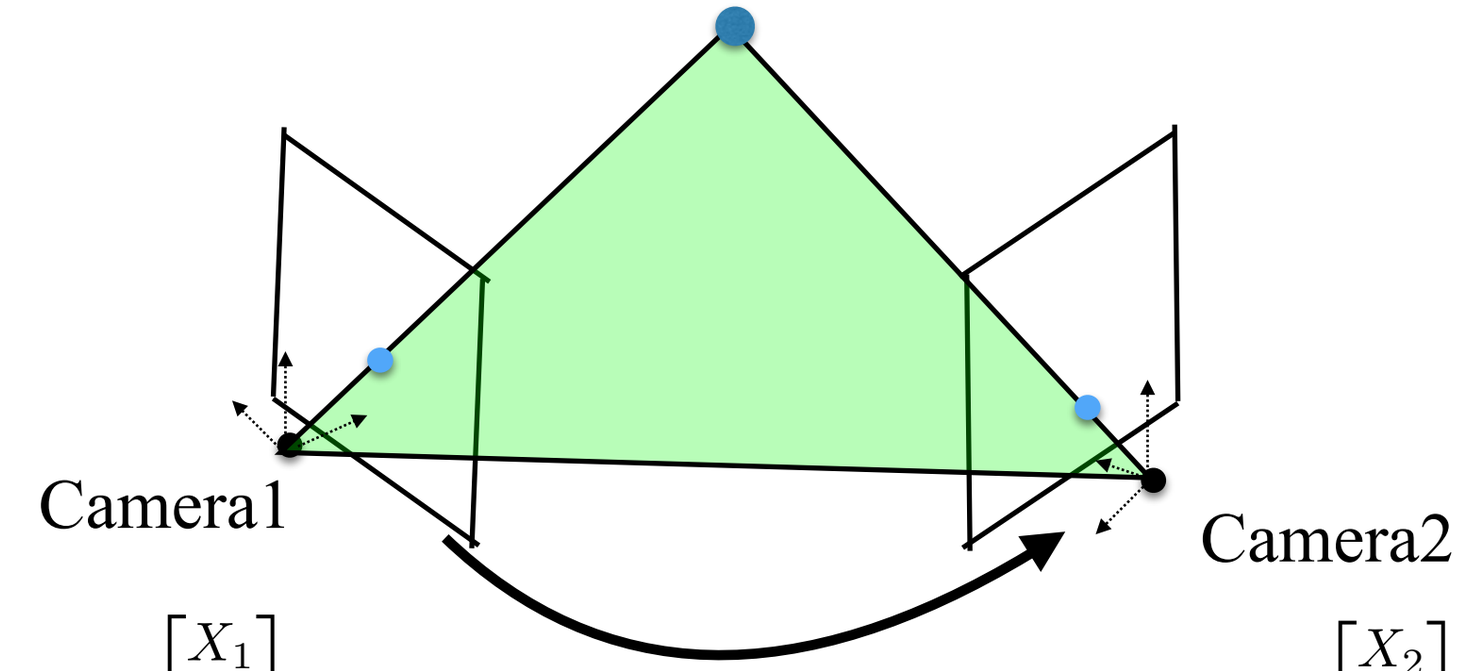
Cross product matrix: $\mathbf{a} \times \mathbf{b} = \hat{\mathbf{a}}\mathbf{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$

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

= 0 for coplanar vectors



Calibrated 2-view geometry



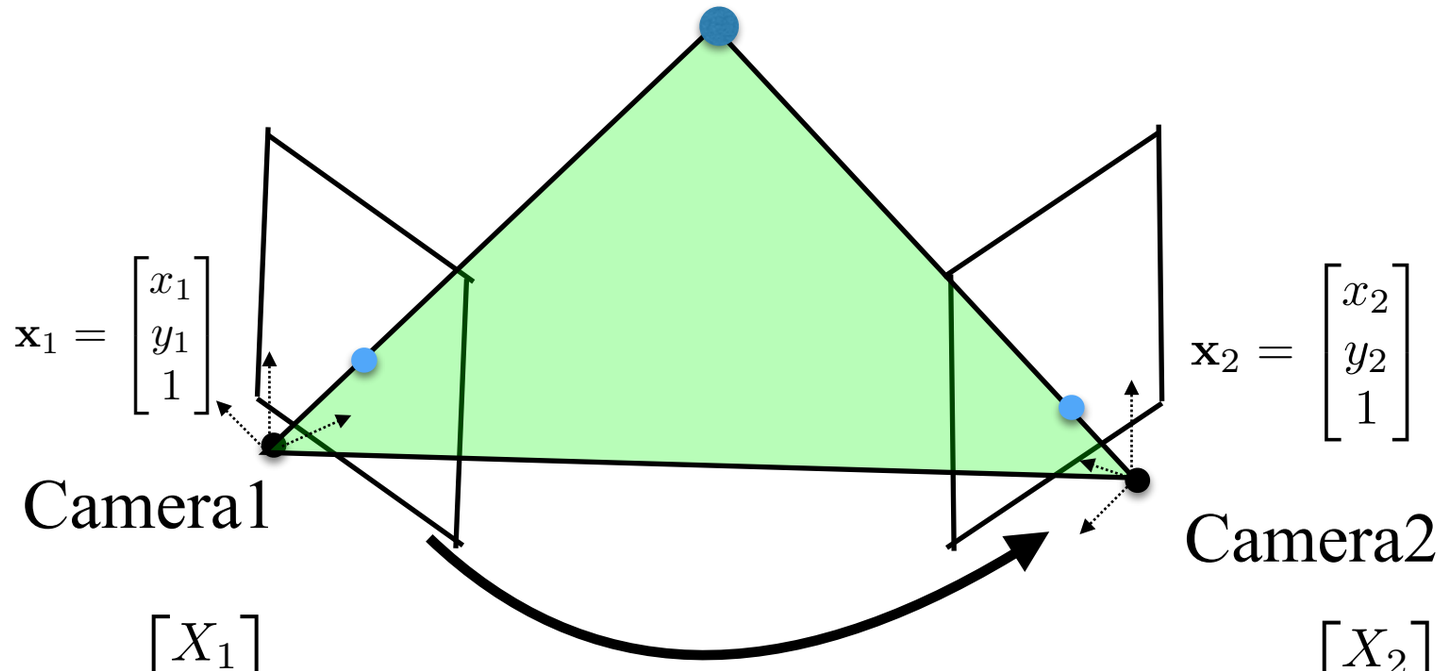
$$\mathbf{X}_1 = \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} \qquad \mathbf{X}_2 = \begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix}$$

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

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

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

Calibrated 2-view geometry



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

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

$\boldsymbol{x}_1$: *homogenous* 2D image coordinate of 3D point in image 1
$\boldsymbol{x}_2$: *homogenous* 2D image coordinate of 3D point in image 2

Epipolar geometry

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

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

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

Take (left) cross product of both sides with ***T***

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

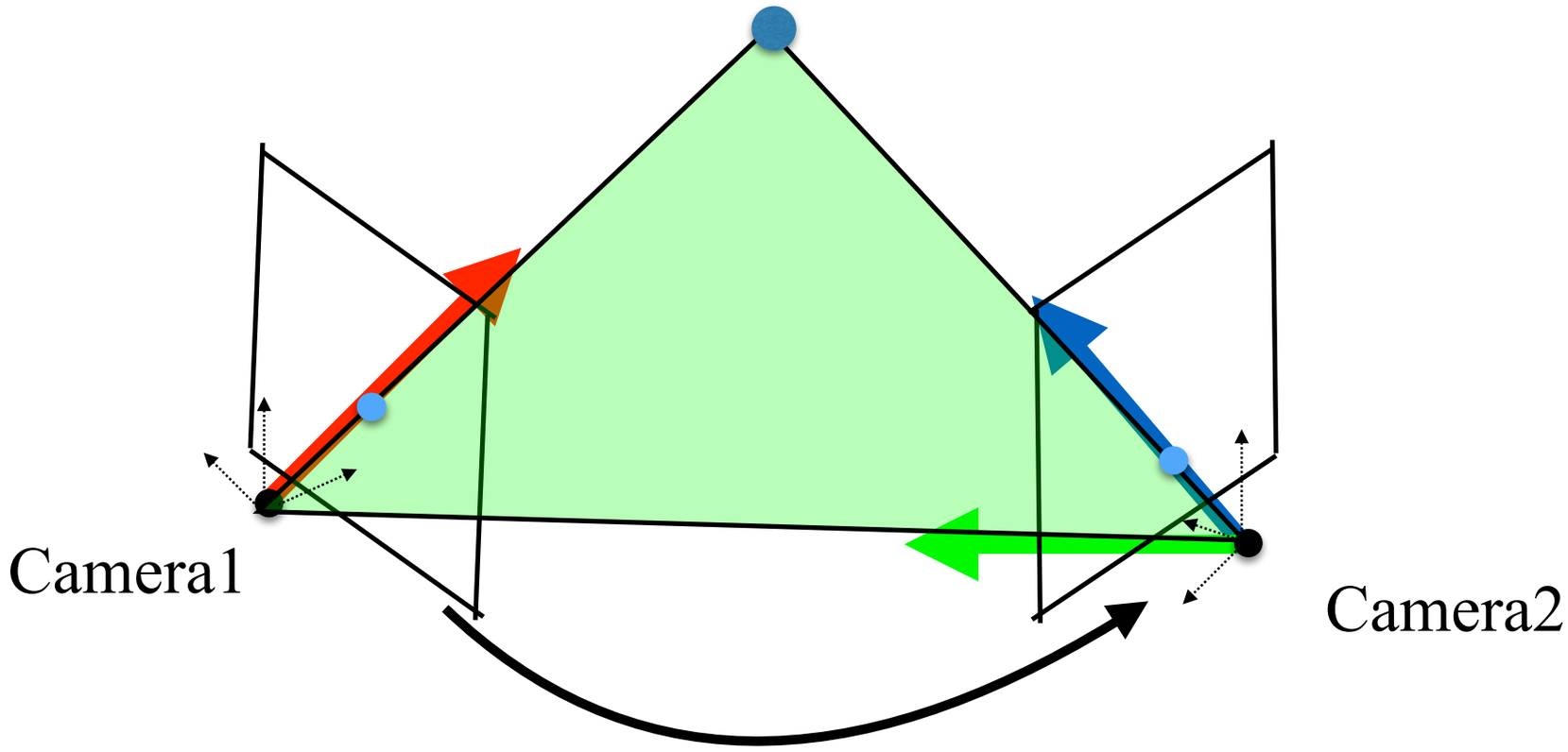
Take (left) dot product of both sides with $\boldsymbol{x_2}$

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

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

Geometric derivation

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



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

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

Epipolar geometry

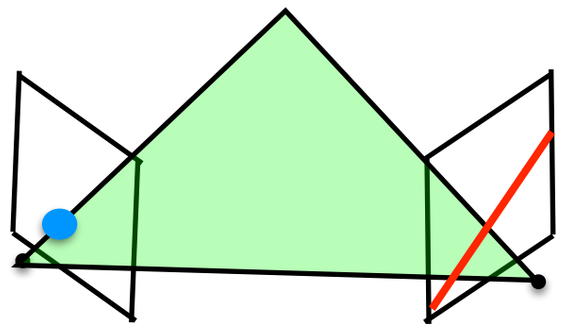
$$\boldsymbol{x}_2^T \hat{\mathbf{T}} R \boldsymbol{x}_1 = 0$$

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

E is known as the *essential* matrix

Essential matrix

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



Let's fix a point on left image $\mathbf{x}_1 = (x_1, y_1)$. What are the set of points from right image $\mathbf{x_2} = (x_2, y_2)$ that satisfy the epipolar constraint?

$$\begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} E_{11} & E_{12} & E_{13} \\ E_{21} & E_{22} & E_{23} \\ E_{31} & E_{32} & E_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = 0 \quad \text{where} \quad \begin{aligned} a &= E_{11}x_1 + E_{12}y_1 + E_{13} \\ b &= E_{21}x_1 + E_{22}y_1 + E_{23} \\ c &= E_{31}x_1 + E_{32}y_1 + E_{33} \end{aligned}$$

$$ax_2 + by_2 + c = 0$$

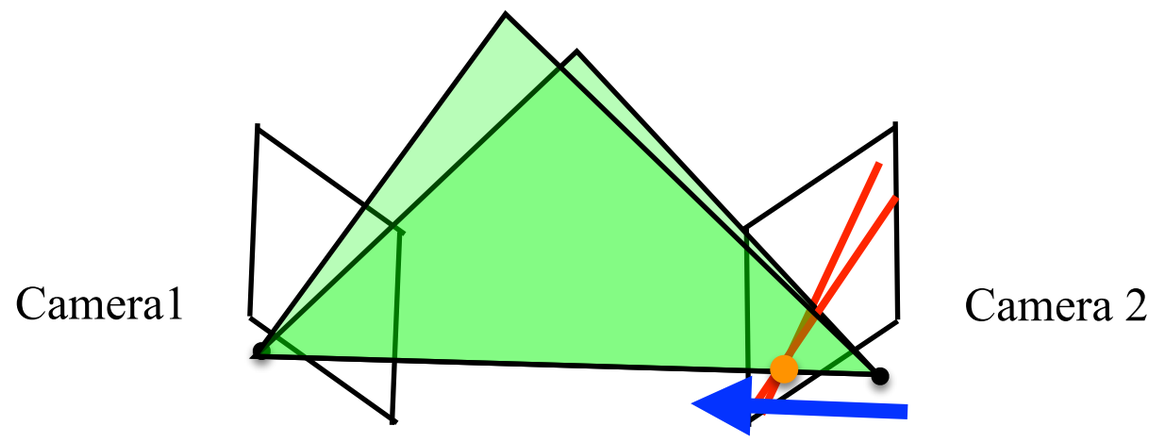
Maps point (x_1,y_1) from left image to line (a,b,c) in right image… and vice versa.

But how is this different from a Homography (also a 3X3 matrix)?

Homographies map points to points; Essential matrix maps points to *lines*

Epipoles

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



Recall that *all* epipolar lines in right image intersect at epipole $\mathbf{e_2}$

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

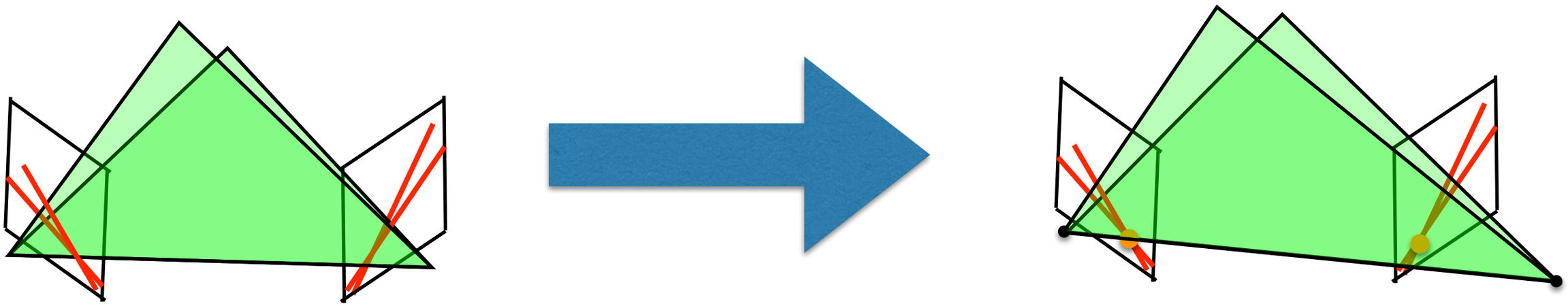
$$\begin{bmatrix} e_x & e_y & 1 \end{bmatrix} \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Implication: E has a singular value of 0 associated with the left singular vector $\mathbf{e_2}$ (and by similar arguments, associated with right singular vector $\mathbf{e_1}$)

Aside: $\mathbf{e_2} = \mathbf{T}$ up-to-scale (the *image* of camera 1's center). Proof by plugging-in: $\mathbf{T}^T \hat{\mathbf{T}} R = -(\mathbf{T} \times \mathbf{T}) R = \mathbf{0}$

Uncalibrated case

(intrinsics K are *not* known and so we can't work with normalized images)



Fortunately, all our geometric insights apply (e.g., epipolar planes still hinge over 1D family, epipolar lines still intersect at epipole…)

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

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

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

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

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

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

We'll call F the *fundamental matrix*

Overview

Essential matrices:

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

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

Fundamental matrices (unknown intrinsics):

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

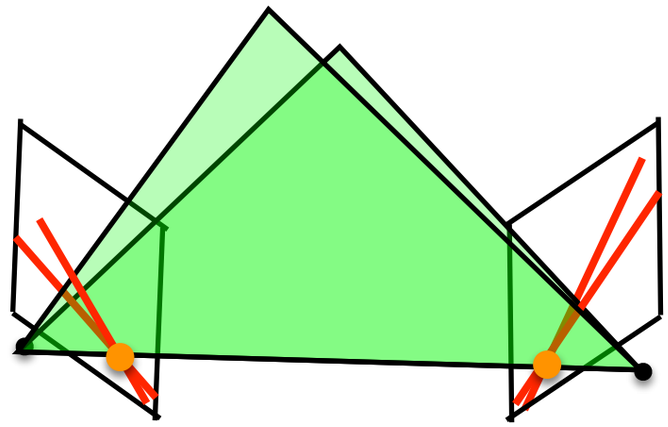
$$\boldsymbol{x}_2^T {K_2}^{-T} \hat{\mathbf{T}} R K_1^{-1} \boldsymbol{x}_1 = 0$$

Roadmap

- logistics
- review of motion segmentation
- two-view
  - geometric intuition
  - essential matrix E, fundamental matrix F
  - **properties of E,F** [rough intuitions without formal proofs]
  - estimating E,F from correspondences
  - inferring R,T from E,F

Where we are headed..

Both E,F can be characterized by their SVD



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

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

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

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

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

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

where $\mathbf{e_1}$, $\mathbf{e_2}$ *are epipoles in right and left images and singular values can be arbitrarily scaled by homogenous factor*

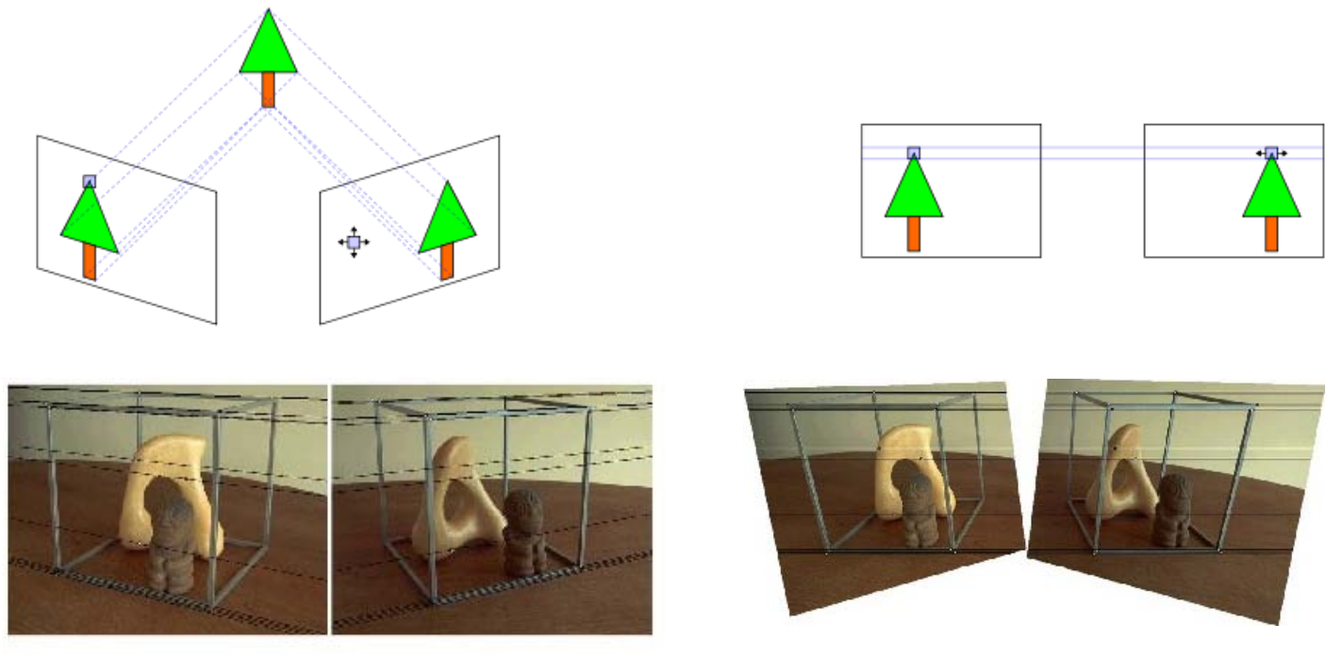
Deriving properties of E

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

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

More-or-less behaves like a cross-product (skew symmetric matrix)

Crucial special case: think of scenario where R = Identity

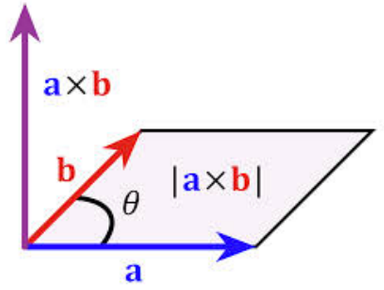


All skew symmetric matrices $\hat{T}$ are essential matrices, but not vice versa

Background: SVDs of skew symmetric matrices

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

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



SVD of a skew-symmetric matrix:

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

Note: $\mathbf{e}_i$ is a dummy variable. Does not mean epipole!

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

Crucial properties:

- $\mathbf{a} \times \mathbf{a} = \mathbf{0}$ => $\mathbf{a} / \|\mathbf{a}\|$ is left and right singular vector with 0 singular value
- $\mathbf{a} \times \mathbf{b} = ||\mathbf{a}|| R_{90°} \mathbf{b}$ **for** $\mathbf{a} \perp \mathbf{b}$ => other 2 singular vectors are orthogonal to $\mathbf{a}$ with singular values equal to $\|\mathbf{a}\|$
- SVD looks like a spectral eigendecomposition with an additional rotation $R_{90°}$

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?

3 (rotation) + 2 (translation direction)

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

No…

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

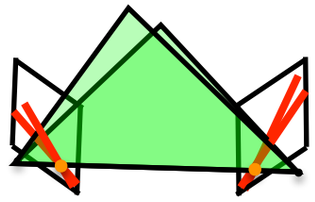
Singular values of E = singular values of $\hat{\mathbf{T}}$ = $(\sigma,\sigma,0)$

[rotations do not effect singular values]

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

Almost. It is unique up to easy-to-deal with 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?

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

Proof: Let $\mathbf{e}_2 = K_2\mathbf{T}$ where $\mathbf{T}$ is a 3-vector

$$\mathbf{e}_2{}^T F = \mathbf{T}^T K_2^T K_2^{-T} \hat{\mathbf{T}} R K_1^{-1} = \mathbf{0}$$

(similar argument for $\mathbf{e}_1$; c.f. Invitation to 3D Vision, Chap 6.2)

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 $E \in \mathbb{R}^{3\times 3}$ is an essential matrix* if and only if *E has a singular value decomposition (SVD): $E = U\Sigma V^T$ with*

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

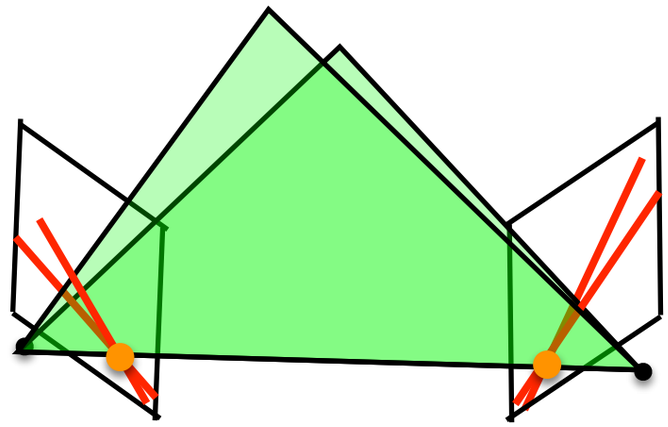
for some $\sigma \in \mathbb{R}_+$ and $U, V \in SO(3)$.

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

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

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

Essential and Fundamental Matrices



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

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

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

Intuitive proof: SVDs of skew-symmetric matrices

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

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

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

Intuitive proof: epipoles

where ***e₁, e₂*** *are epipoles in right and left images and singular values can be arbitrarily scaled by homogenous factor*

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

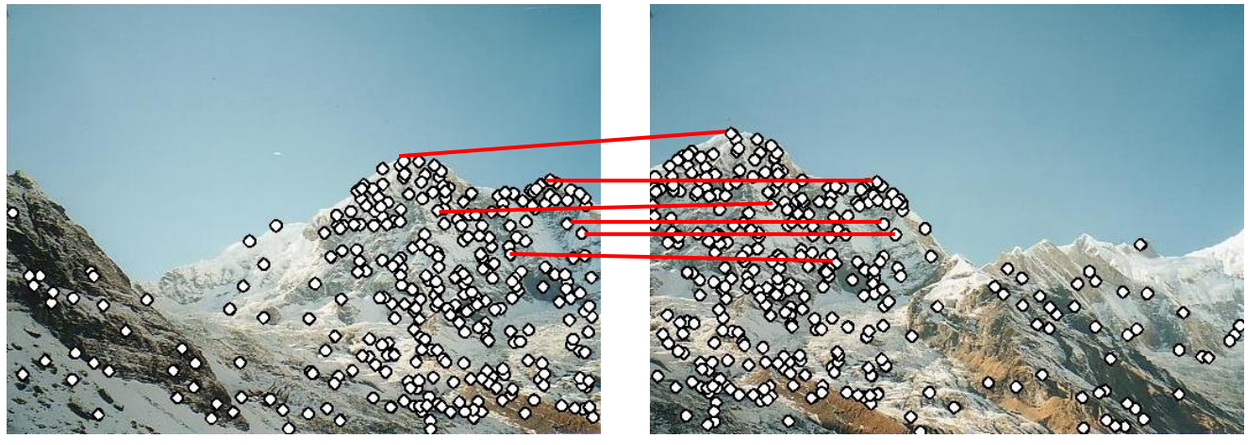
F: zero out smallest singular value

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

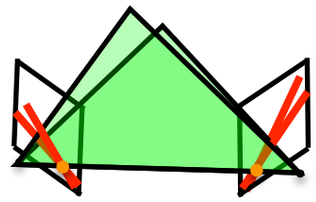
Roadmap

- 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)

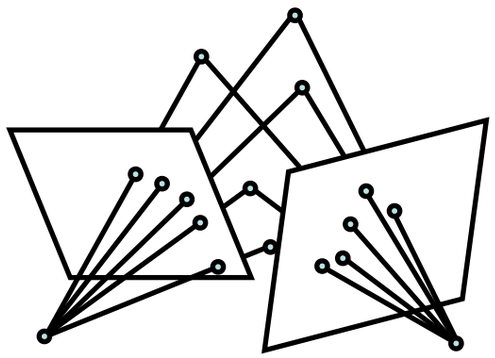


Assume we have N pairs of corresponding points: in noise-free case….



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

Estimation (fundamental matrix)

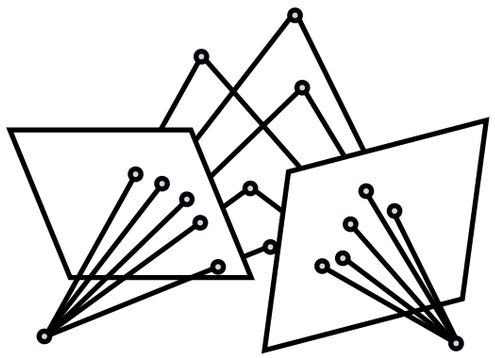


Given N point correspondences (x_i,y_i) and (x'_i,y'_i):

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_Nx'_N & x_Ny'_N & x_N & y_Nx'_N & y_Ny'_N & y_N & x'_N & y'_N & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

$$\mathrm{AF}(:) = 0$$

Estimation (fundamental matrix)



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

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_Nx'_N & x_Ny'_N & x_N & y_Nx'_N & y_Ny'_N & y_N & x'_N & y'_N & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

$$\mathrm{AF}(:) = 0$$

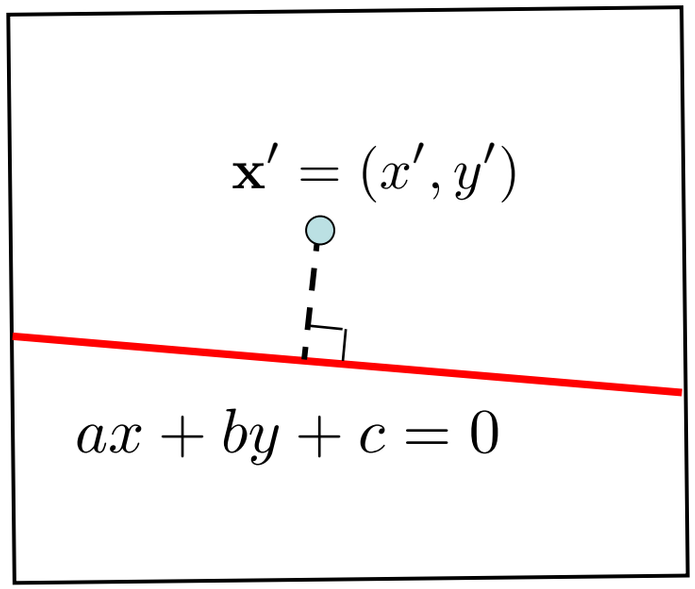
noisy case: $\min_{||F||=1} ||AF(:)||^2 = \min_F \sum_i (\mathbf{x'}_i^T F \mathbf{x}_i)^2$

Is this a reasonable error to minimize?

Recall: distance of point from a line

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

$$\text{dist}(ax + by + c = 0, (x', y')) = \frac{|ax' + by' + c|}{\sqrt{a^2 + b^2}}$$



$\mathbf{x'}_i^T F \mathbf{x}_i$ is *almost* the right error; its the *scaled* euclidean distance of (x'_i, y'_i) from line defined by (x_i, y_i)

Right error requires squared euclidean distance, which is nonlinear in F because it must *divide* out homogenous scale factor:

$$\sum_i \text{dist}^2(F\mathbf{x}_i, \mathbf{x}'_i) + \text{dist}^2(F\mathbf{x}'_i, \mathbf{x}_i)$$

How many points are needed?

m = 8 point algorithm due to Longuet-Higgens

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_Nx'_N & x_Ny'_N & x_N & y_Nx'_N & y_Ny'_N & y_N & x'_N & y'_N & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

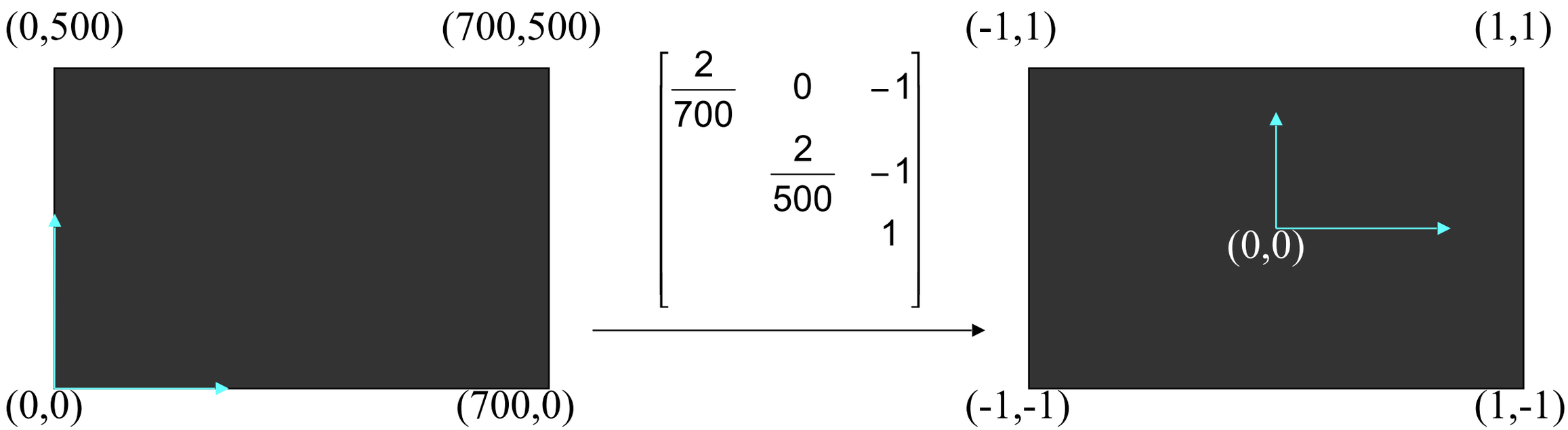
~10000 ~10000 ~100 ~10000 ~10000 ~100 ~100 ~100 1

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

“In Defense of the 8-point Algorithm”

(Hartley, PAMI´97)

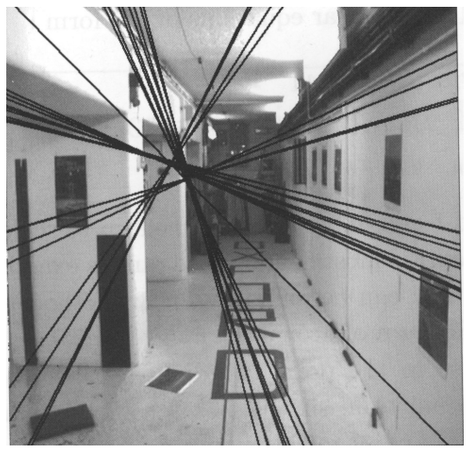
Transform image to [-1,1]x[-1,1]



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)



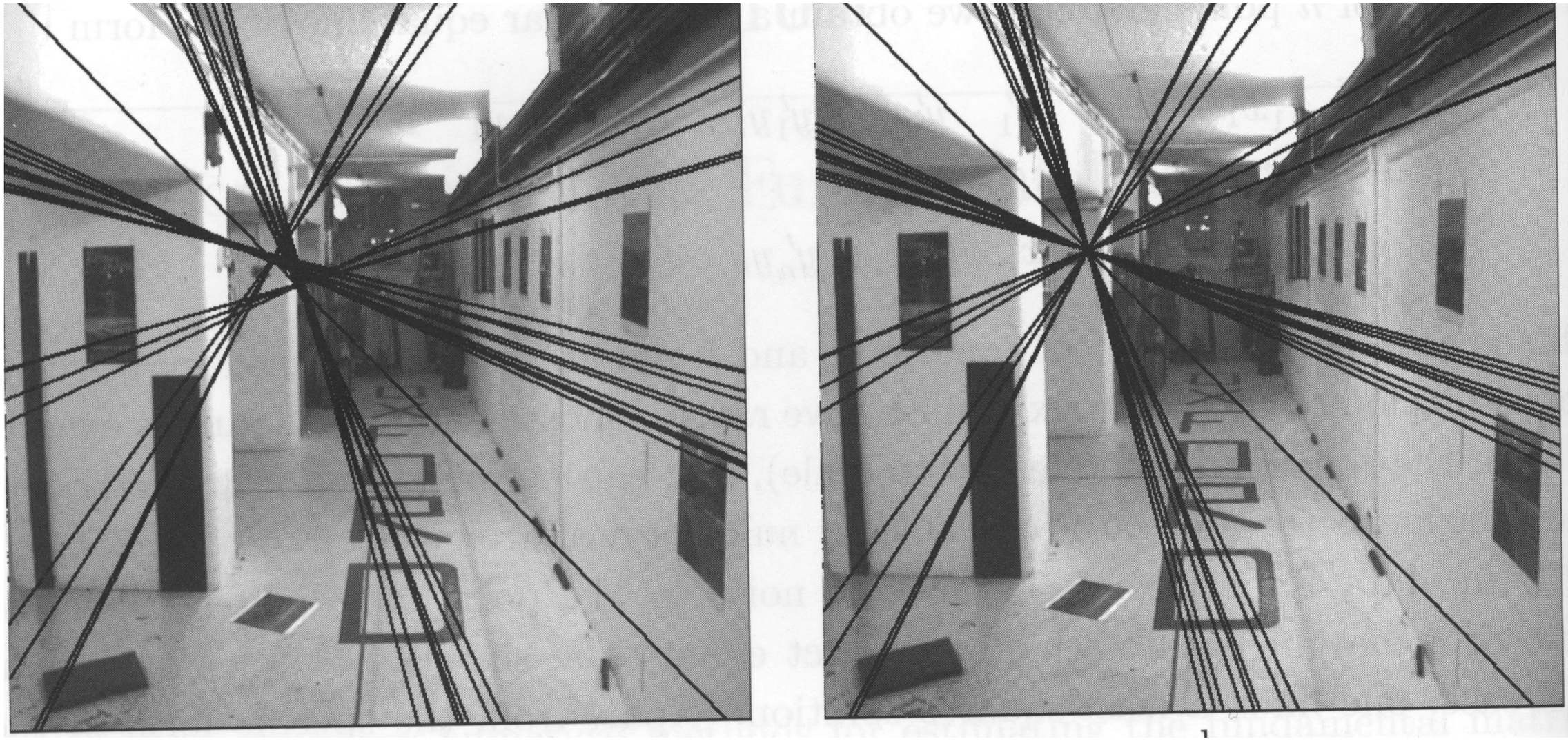
Solution: find the closest F/E (Frebonius norm) with SVD

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

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 $\sigma_3 = 0$

7-point algorithm

Since F are rank-difficient, we can estimate them with m=7 correspondences

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

AF(:)=0

Idea: search for null vector of A_{7x9} that satisfies additional contraints
(reshaping null vector into 3x3 matrix should produce matrix of rank 2)

1) Find 2 vectors that span 2D *null space* of A_{7x9}, F_1 and F_2 (with SVD of A).
2) Reshape F_1 & F_2 into 3x3 matrices
3) Find α such that Determinant$(\alpha F_1 + (1-\alpha)F_2) = 0$

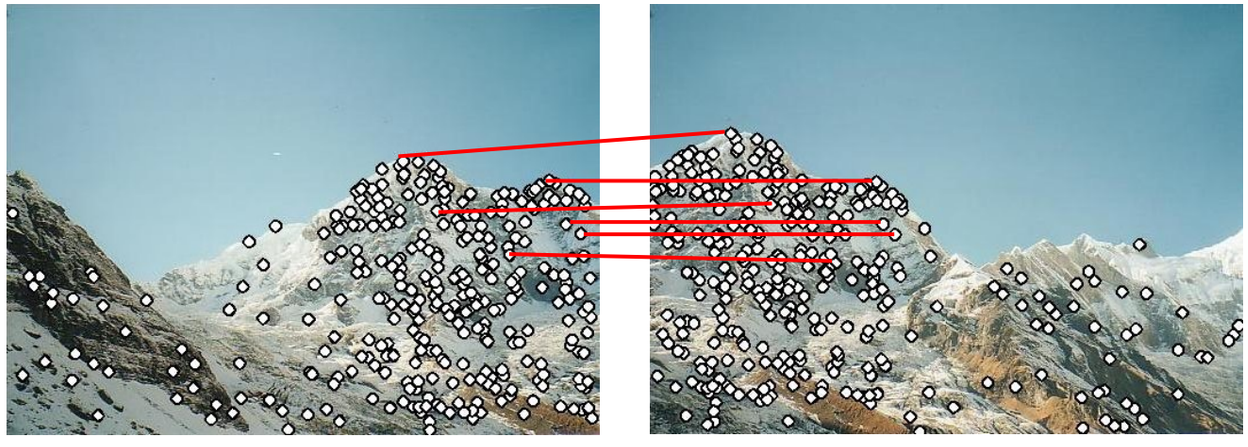
[3rd order polynomial in α with at least one real solution]

Aside: what if cameras are calibrated?

Since there are 5 degrees of freedom, we need only 5 points,
but need to find roots of 10th degree polynomial

[Nister 04]

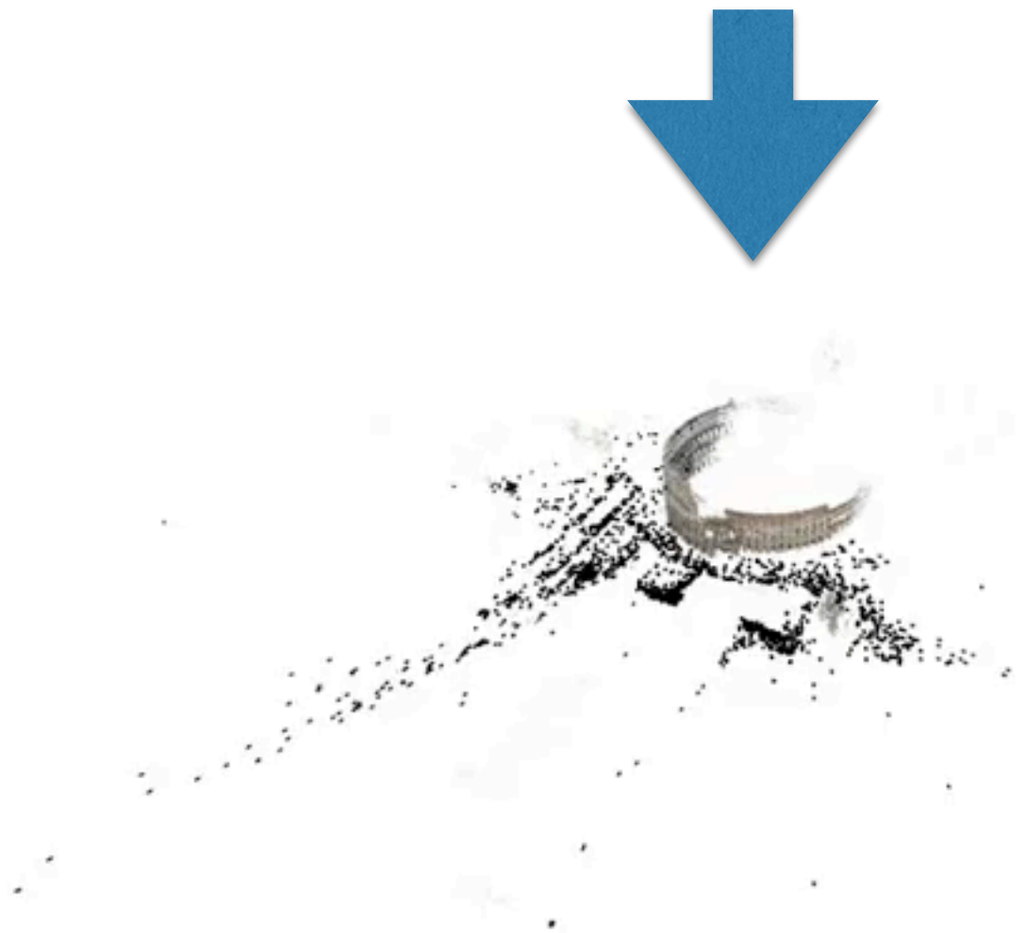
Recall: RANSAC



RANSAC loop:

1. Select 7(or 8) point pairs (at random)
2. Compute fundamental matrix F (exact)
3. Compute *inliers* (point matches where $d(\mathbf{x}_i', F\mathbf{x}_i)^2 \leq \epsilon$)
   Keep largest set of inliers

First step toward 3D reconstruction (after point correspondences): estimate E,F



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**

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?

3 (rotation) + 2 (translation direction)

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

No…

E is the product of a rotation and skew-symmetric matrix

Singular values of E = (sigma,sigma,0)

[rotations do not effect singular values]

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

Almost. It is unique up to easy-to-deal with symmetries

If we don't know K, we can recover R,T from F by *upgrading* structure-from-motion (in next lecture)

Four (camera, 3D point) layouts where image projection (x1,y1) and (x2,y2) are the same

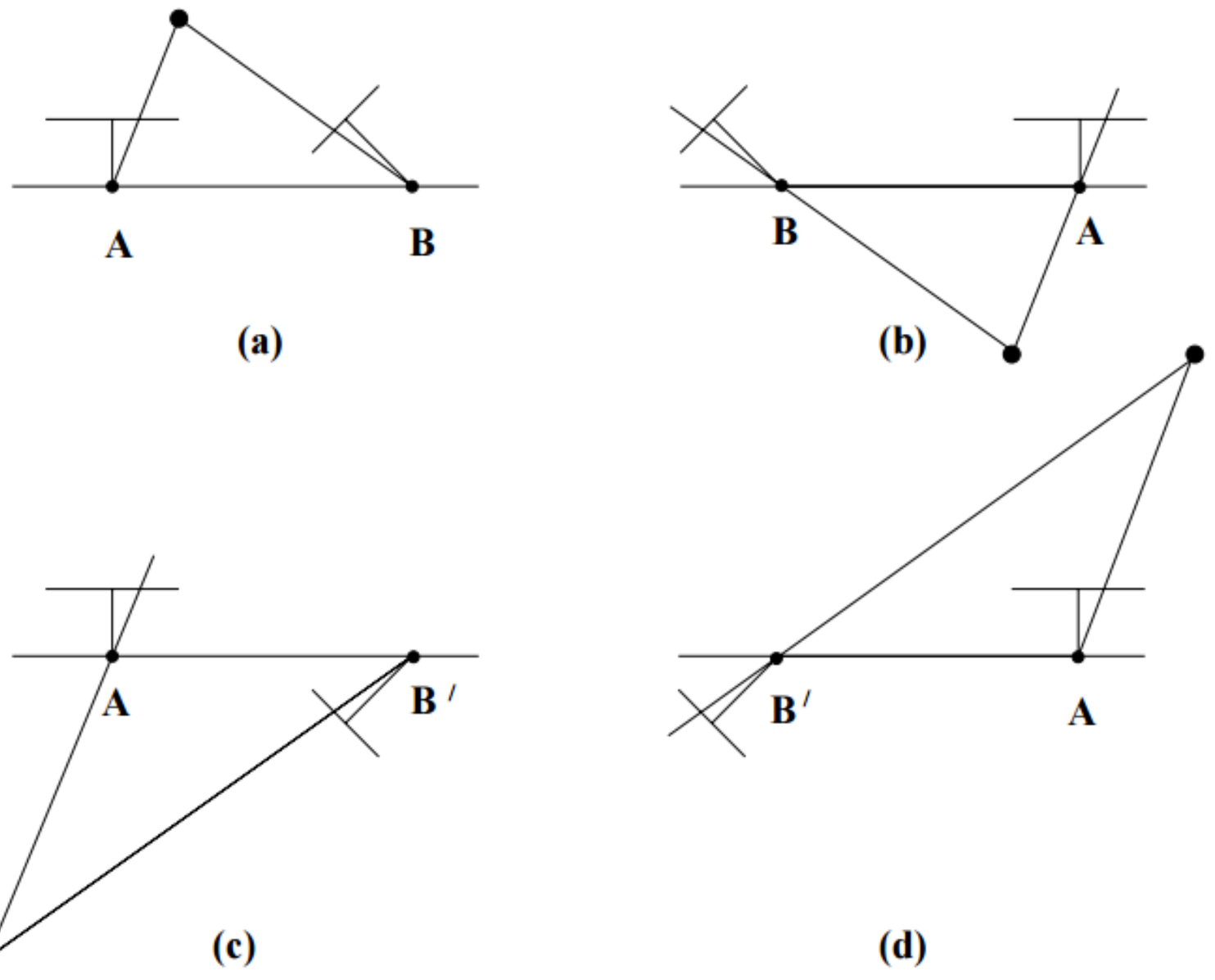
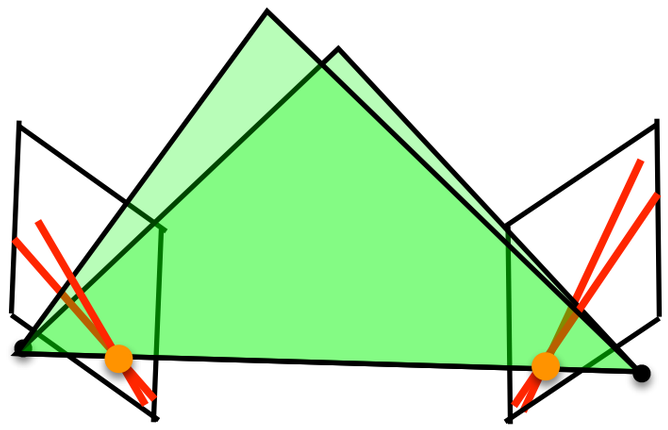


Fig. 8.12. **The four possible solutions for calibrated reconstruction from E.** *Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates* 180° *about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.*

Recovering T,R from E



1. Universal scale ambiguity

Doubling **T** results in same epipolar lines

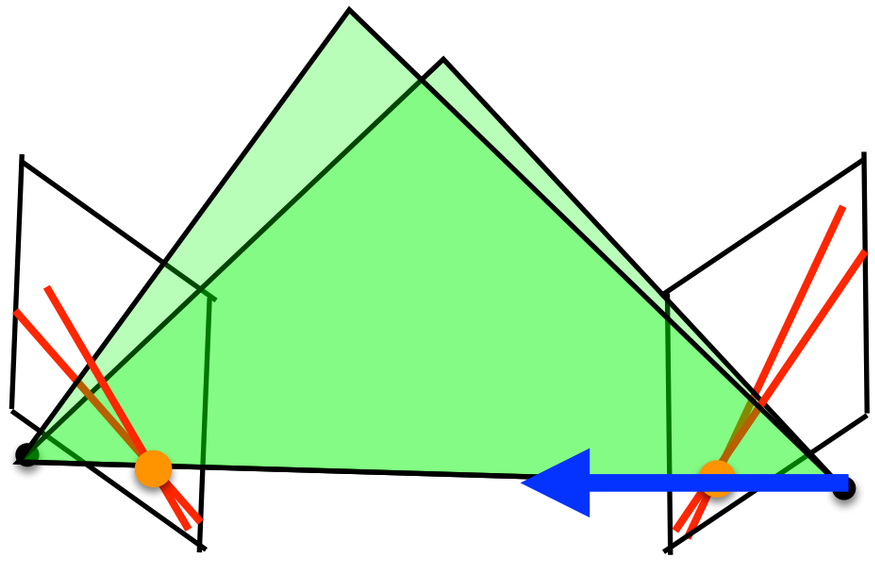
Let's fix ||**T**|| = 1

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 $\mathbf{T}$

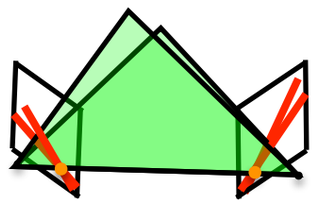
$$\mathbf{T}^T E = -(\mathbf{T} \times \mathbf{T})R = \mathbf{0}$$

Implies that normalized translation vector = left singular *null* vector of E = epipole in right image

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

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)

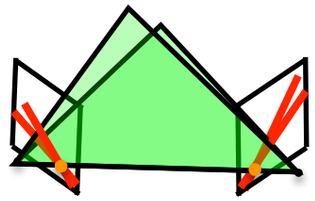
$$E = \hat{\mathbf{T}}R = U\Sigma V^T = \begin{bmatrix} \mathbf{u}_0 & \mathbf{u}_1 & \mathbf{T} \end{bmatrix} \begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_0^T \\ \mathbf{v}_1^T \\ \mathbf{v}_2^T \end{bmatrix}$$

Set translation direction = smallest left singular vector of E

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

Recovering R from E

SVD-based approach (Szeliski Chap 7.2)



Recall skew-symmetric decomposition

$$\hat{\mathbf{T}} = \underbrace{\begin{bmatrix}\mathbf{e}_1 & \mathbf{e}_2 & \mathbf{T}_3\end{bmatrix}}_{\mathrm{S}} \underbrace{\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}}_{\mathrm{Z}} \underbrace{\begin{bmatrix}0 & -1 & 0\\ 1 & 0 & 0\\ 0 & 0 & 1\end{bmatrix}}_{\mathrm{R_{90}}} \underbrace{\begin{bmatrix}\mathbf{e}_1^T\\ \mathbf{e}_2^T\\ \mathbf{T}_3^T\end{bmatrix}}_{\mathrm{S^T}}$$

$$\mathrm{SVD}(\hat{\mathbf{T}}R) = U\Sigma V^T$$

$$\mathrm{SVD}(\mathrm{SZR_{90}S^TR}) = \mathrm{U\Sigma V^T}$$

$$\mathrm{U{=}S,\ \Sigma{=}Z,\ V^T = R_{90}S^TR}$$

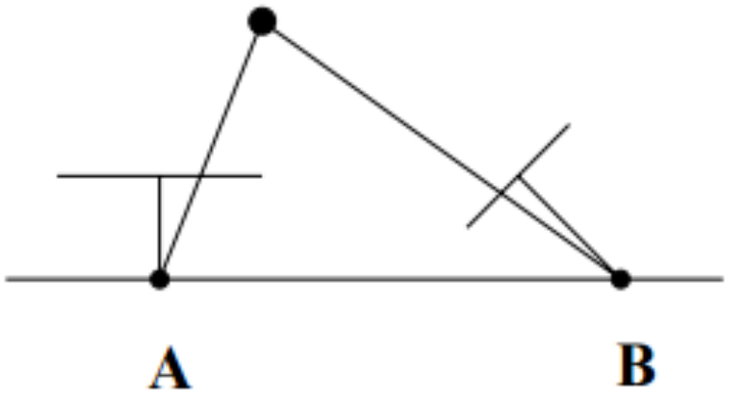
Match orthogonal and diagonal matrices and solve for R:

$$R_{90^\circ} U^T R = V^T$$

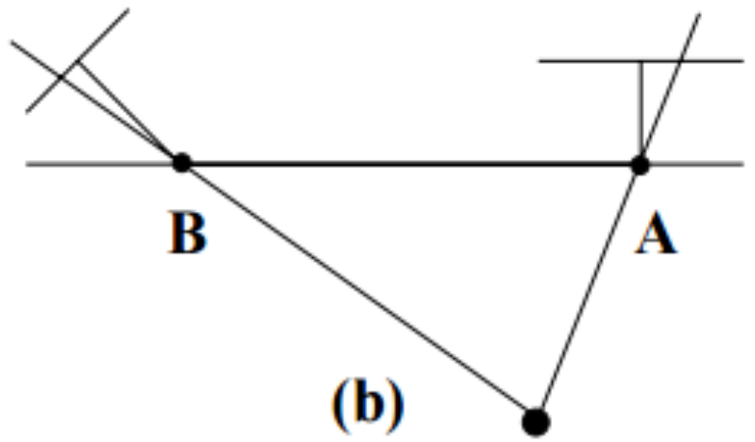
$$R = U R_{90^\circ}^T V^T$$

$$= \pm U R_{\pm 90^\circ}^T V^T$$

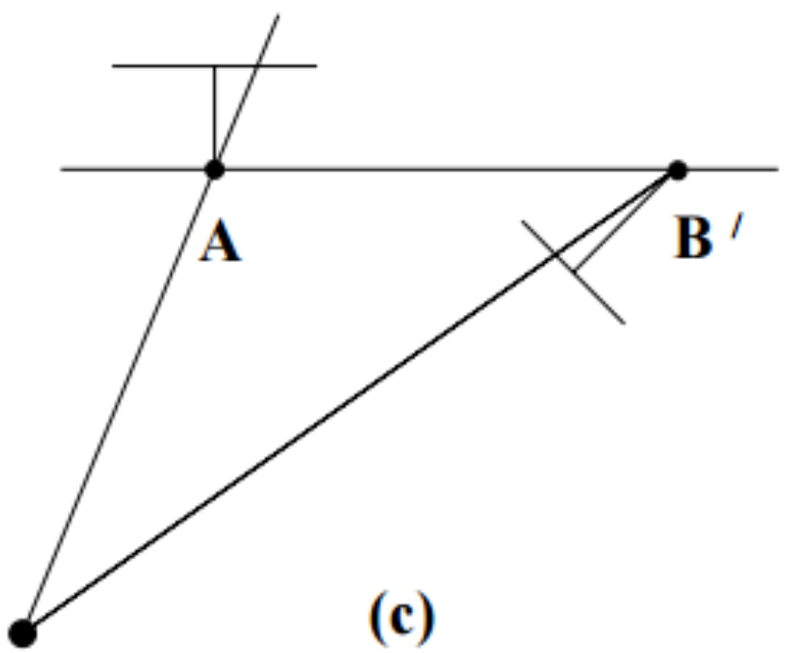
Generate 4 possible rotations and keep 2 with determinant = 1 (non-reflections)



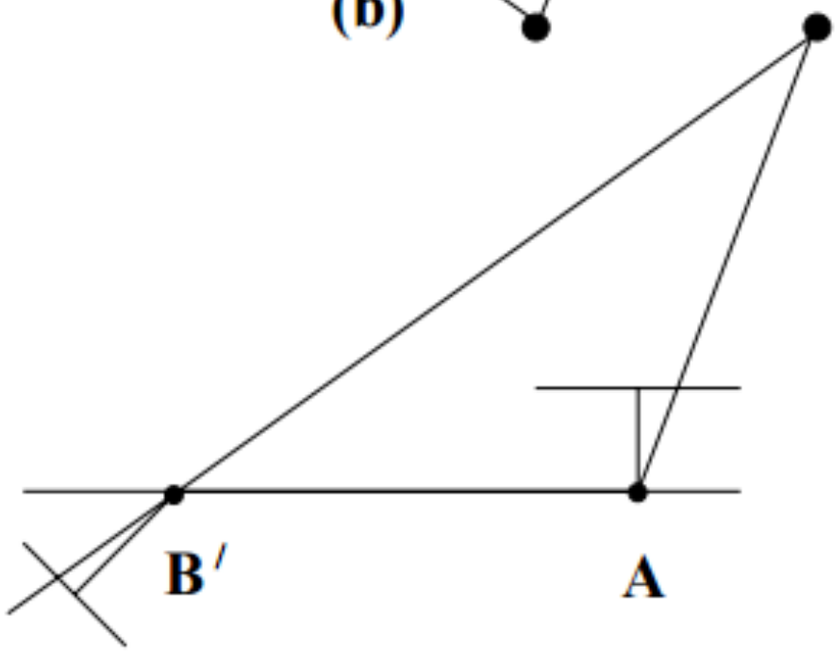
(a)



(b)



(c)



(d)

Fig. 8.12. **The four possible solutions for calibrated reconstruction from E.** *Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates* 180° *about the baseline. Note, only in (a) is the reconstructed point in front of both cameras.*

Roadmap

- 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 & structure from motion (next slide deck)