COMP 558

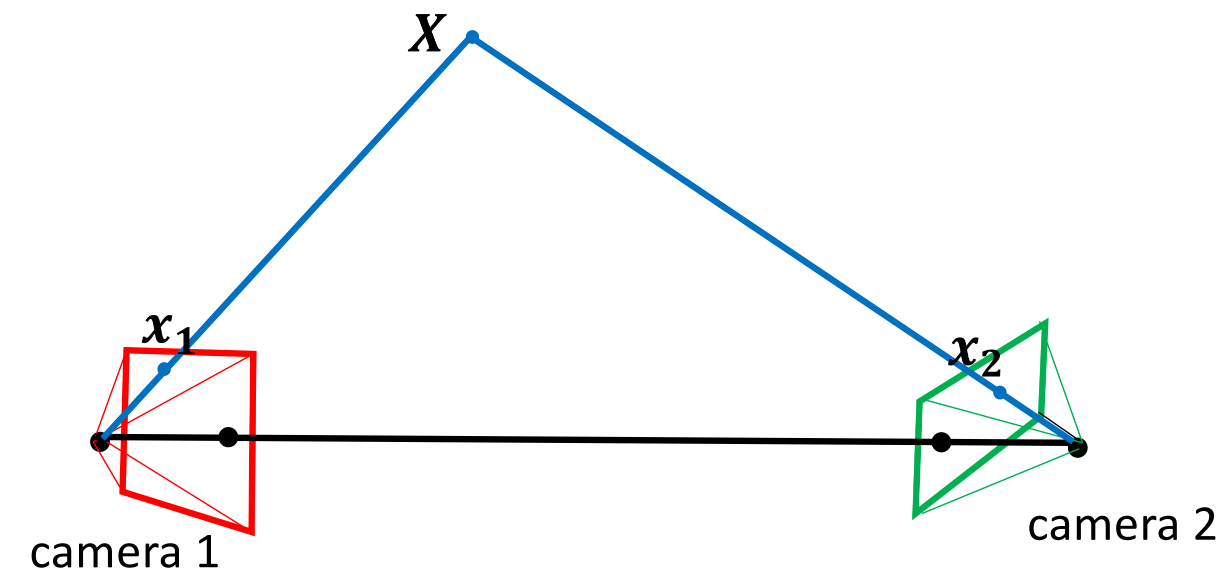
Lecture 21

Stereo 2

(Slides courtesy Mike Langer)

camera 1

camera 2



Recall last lecture:

$$\mathbf{x_2}^{\mathrm{T}} \mathbf{E} \mathbf{x_1} = 0$$

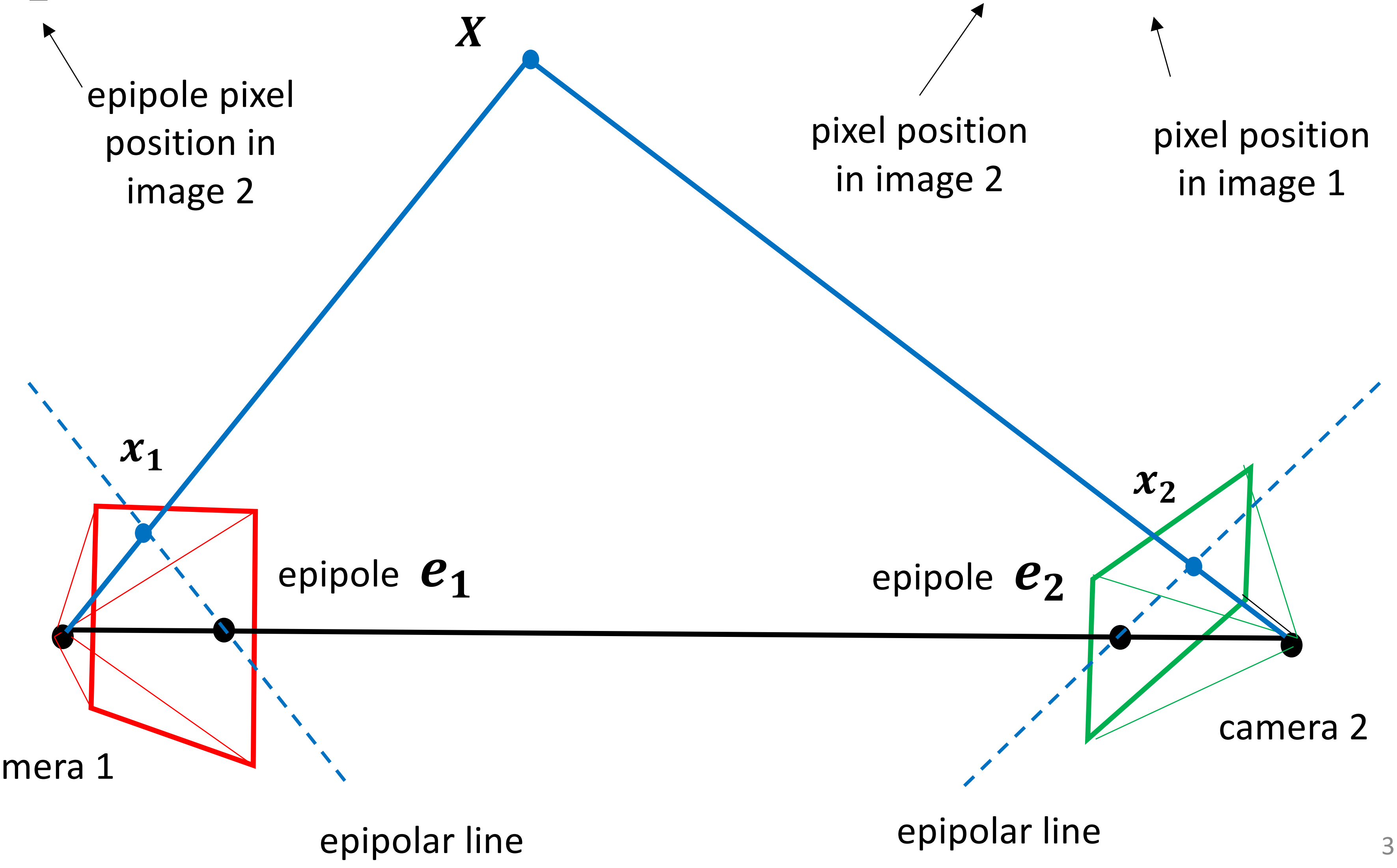
$$\tilde{\mathbf{x}}_2^{\mathrm{T}} \mathbf{F}\, \tilde{\mathbf{x}}_1 = \mathbf{0}$$

$$\tilde{\mathbf{e}}_2^{\mathrm{T}} \mathbf{F} = \mathbf{0}$$

epipole pixel position in image 2

pixel position in image 2

pixel position in image 1



Overview of Today

- Estimating the fundamental matrix
    - 8 point algorithm, least squares
    - Normalization
    - rank 2 constraint
    - RANSAC

- Disparity estimation
    - Image rectification
    - Inherent ambiguities in depth estimation

Today we will only be working with pixel coordinates. So I will drop the $\widetilde{\boldsymbol{x}}$ notation and just work with $\boldsymbol{x}$.

For any matching pairs of points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ in the camera 1 and 2 images, we write $\boldsymbol{x_2}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{x_1} = 0$, that is,

$$[x_2 \quad y_2 \quad 1]\begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix}\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = 0$$

We can write this as :

$$(x_1x_2,\ y_1x_2,\ x_2,\ x_1y_2,\ y_1y_2,\ y_2,\ x_1,\ y_1,\ 1)\cdot$$
$$(F_{11},\ F_{12},\ F_{13},\ F_{21},\ F_{22},\ F_{23},\ F_{31},\ F_{32},\ F_{33}) = 0$$

8 point algorithm for estimating $\boldsymbol{F}$

N matching pairs of points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ in the camera 1 and 2 images give N equations:

$$\underbrace{\begin{bmatrix} x_1^1 x_2^1 & y_1^1 x_2^1 & x_2^1 & x_1^1 y_2^1 & y_1^1 y_2^1 & y_2^1 & x_1^1 & y_1^1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_1^i x_2^i & y_1^i x_2^i & x_2^i & x_1^i y_2^i & y_1^i y_2^i & y_2^i & x_1^i & y_1^i & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_1^N x_2^N & y_1^N x_2^N & x_2^N & x_1^N y_2^N & y_1^N y_2^N & y_2^N & x_1^N & y_1^N & 1 \end{bmatrix}}_{8\times 9} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

If $N = 8$, then we can find an exact solution for $\boldsymbol{F}$.
This is called the *8 point algorithm.*

ASIDE

You might think there are 8 degrees of freedom in $\boldsymbol{F}$.
But in fact there are 7 degrees of freedom. Why?

First, $\boldsymbol{F}$ is of rank 2. Thus the columns are linearly dependent. So if I give you the first two columns and two elements in the third column, you can tell me the last element in the 3rd column. This reduces the number of degrees of freedom to 8. But in addition, scaling $\boldsymbol{F}$ by a constant factor doesn't change the epipolar constraints (namely the left and right null space of $\boldsymbol{F}$). This loses another degree of freedom, bringing us to 7.

So, in theory, one only needs 7 matching points, not 8, to solve for $\boldsymbol{F}$.

(The 7 point solution involves some details, which we omit.)

Least Squares solution for $\boldsymbol{F}$

Suppose we have $N > 8$ matching pairs of points $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ in the camera 1 and 2 images, and suppose these points are noisy. Then we would like to approximate:

$$\underbrace{\begin{bmatrix} x_1^1 x_2^1 & y_1^1 x_2^1 & x_2^1 & x_1^1 y_2^1 & y_1^1 y_2^1 & y_2^1 & x_1^1 & y_1^1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_1^i x_2^i & y_1^i x_2^i & x_2^i & x_1^i y_2^i & y_1^i y_2^i & y_2^i & x_1^i & y_1^i & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x_1^N x_2^N & y_1^N x_2^N & x_2^N & x_1^N y_2^N & y_1^N y_2^N & y_2^N & x_1^N & y_1^N & 1 \end{bmatrix}}_{N \times 9} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This is the familiar form $\boldsymbol{Ax} \approx 0$, where $\boldsymbol{A}$ is an $N \times 9$ data matrix.

The solution for $\boldsymbol{F}$ is the eigenvector of $\boldsymbol{A^T A}$ with the smallest eigenvalue. Equivalently, take the SVD $\boldsymbol{A} = \boldsymbol{U \Sigma V^T}$.

Least squares solution for $\boldsymbol{F}$

The least squares solution finds the matrix $\boldsymbol{F}$ that minimizes:

$$\sum_{i=1}^{N} \left(\boldsymbol{x}_2^{i\,\mathrm{T}} \boldsymbol{F} \boldsymbol{x}_1{}^{i} \right)^2$$

subject to the constraint $||\boldsymbol{F}|| = \mathbf{1}$, namely the sum of squares of elements of $\boldsymbol{F}$ is 1.

However, if there is noise, then this alone does not work so well, for a few reasons…

Data Normalization

The range of magnitudes is very different across columns of $\boldsymbol{A}$, which causes numerical problems.

Therefore, normalize the $\{\,(x_1^i, y_1^i)\,\}$ and $\{\,(x_2^i, y_2^i)\,\}$ values in images 1 and 2.

$$\mathbf{M_1} : (x_1^i, y_1^i) \rightarrow \left(\frac{x_1^i - \bar{x}_1}{\sigma_1}, \frac{y_1^i - \bar{y}_1}{\sigma_1}\right)$$

$$\mathbf{M_2} : (x_2^i, y_2^i) \rightarrow \left(\frac{x_2^i - \bar{x}_2}{\sigma_2}, \frac{y_2^i - \bar{y}_2}{\sigma_2}\right)$$

Solve for $\boldsymbol{F}_{normalized}$ using these normalized values instead, that is,

$$(\mathbf{M_2}\boldsymbol{x_2})^T \boldsymbol{F}_{normalized} \mathbf{M_1} \boldsymbol{x_1} = \mathbf{0}.$$

So, $\boldsymbol{F} \equiv \mathbf{M_2}^T \boldsymbol{F}_{normalized} \mathbf{M_1}$.

We will need epipolar constraints, and so we will need epipoles. So we need $\boldsymbol{F}$ to have rank 2.

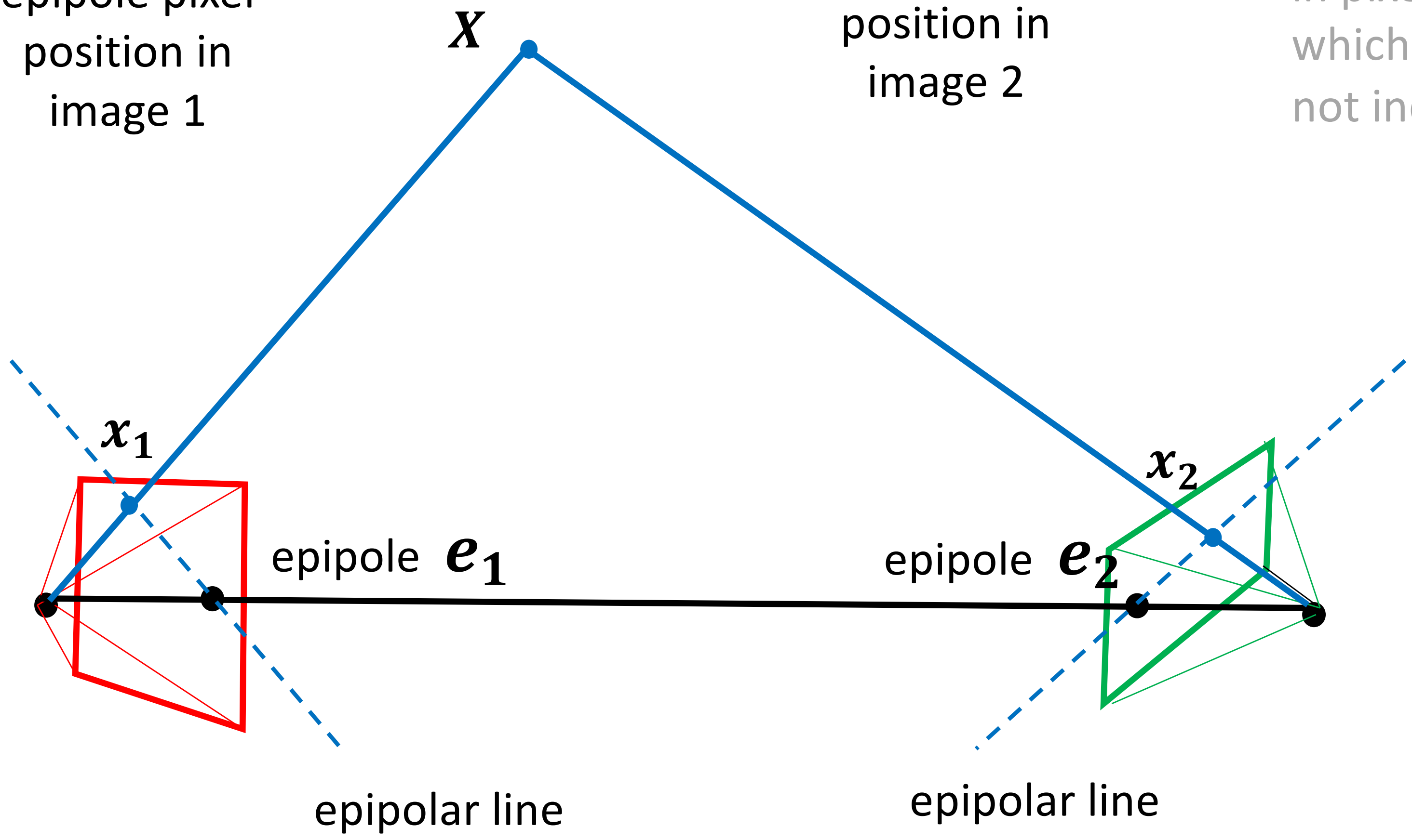
$$\boldsymbol{F}\,\tilde{\boldsymbol{e}}_1 = \mathbf{0}$$

epipole pixel position in image 1

$$\tilde{\boldsymbol{e}}_2^{\mathrm{T}}\boldsymbol{F} = \mathbf{0}$$

epipole pixel position in image 2

The tilde notation on the epipoles is a reminder that we are in pixel coordinates, which the figure does not indicate.



How to constrain $\boldsymbol{F}$ to be rank 2 ?

We obtain the best rank 2 approximation of $\boldsymbol{F}_{normalized}$:

- Compute the SVD of $\boldsymbol{F}_{normalized} = \boldsymbol{U\Sigma V}^{\mathbf{T}}$.

  (This has nothing to do with the SVD of matrix A earlier.)

- Replace $\boldsymbol{\Sigma}$ by $\boldsymbol{\Sigma}'$ by setting Σ_{33} = 0.

The new matrix $\boldsymbol{F}_{normalized} = \boldsymbol{U\Sigma' V}^{\mathbf{T}}$ is of rank 2.

Least Squares solution for $\boldsymbol{F}$
(putting it all together)

Given $N \gg 8$ matching pairs in the two images:

- Normalize the $\{\, (x_1^i, y_1^i) \,\}$ and $\{\, (x_2^i, y_2^i) \,\}$ values in each images.

- Use least squares to solve for $\boldsymbol{F}_{normalized}$.

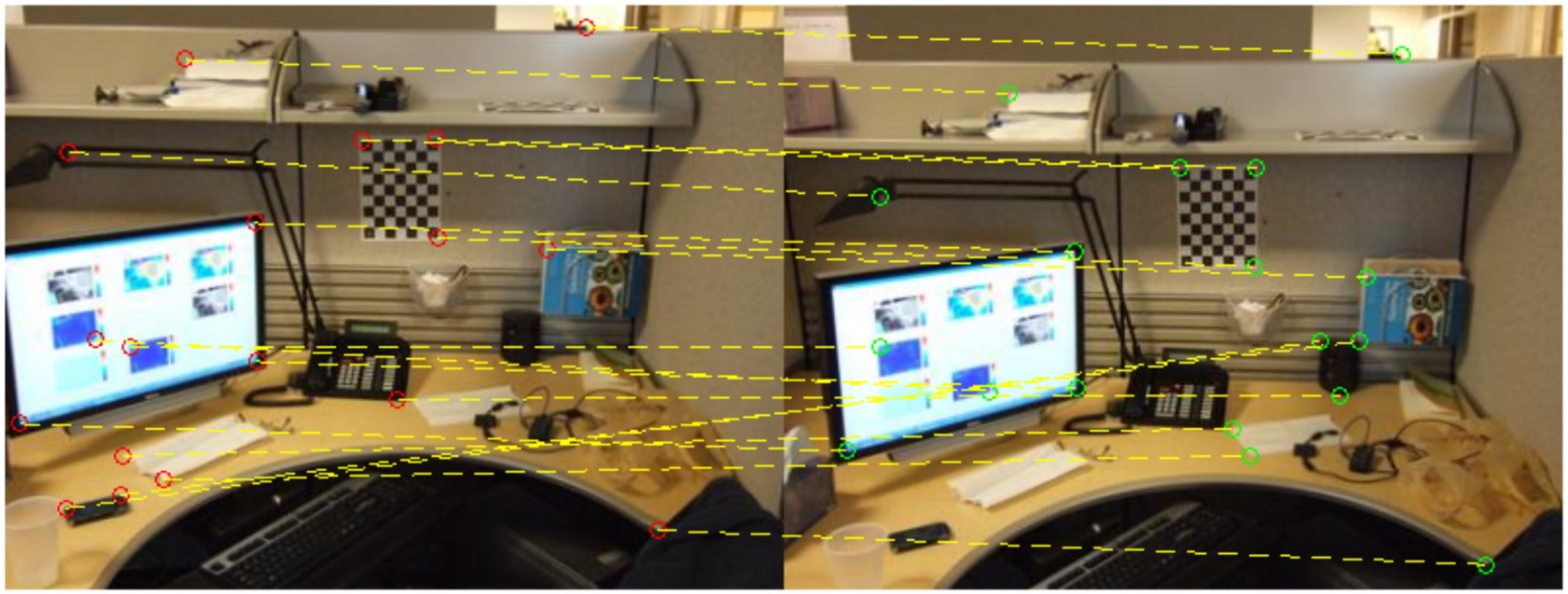
- Enforce the rank 2 constraint on $\boldsymbol{F}_{normalized}$.

- Denormalize: $\boldsymbol{F} \equiv \mathbf{M_2}^{\boldsymbol{T}} \boldsymbol{F}_{normalized} \mathbf{M_1}$

i.e. Matching constraint for the original data is

$$(\mathbf{M_2}\boldsymbol{x_2})^{\boldsymbol{T}} \boldsymbol{F}_{normalized} \mathbf{M_1} \boldsymbol{x_1} = \mathbf{0}.$$

Given two images, how do we choose matching points ?



https://www.mathworks.com/help/vision/ref/estimatefundamentalmatrix.html

Use SIFT features. For each SIFT keypoint in image 1, find the most similar feature in image 2, and vice versa, where similarity is defined by the 128-D SIFT descriptor.

This gives us a set of candidate matching pairs $\{ (x_1^i, y_1^i, x_2^i, y_2^i) \}$.

Robust estimation of $\boldsymbol{F}$ using RANSAC

```
repeat {

} until some condition is satisfied
```

Robust estimation of $\boldsymbol{F}$ using RANSAC

repeat {

- randomly sample $N = 8$ matching pairs of pixel points from the two images, $\boldsymbol{x_1}^{\boldsymbol{i}}, \boldsymbol{x_2}^{\boldsymbol{i}}$ is the $\boldsymbol{i}^{th}$ matching pair
- fit a fundamental matrix (*exact* model fit) that satisfies $\boldsymbol{x_2^i}^{\mathrm{T}} \boldsymbol{F} \boldsymbol{x_1}^{\boldsymbol{i}} = \boldsymbol{0}$ for all $\boldsymbol{i}$ in 1 to 8

} until some condition is satisfied

Robust estimation of $\boldsymbol{F}$ using RANSAC

repeat {

- randomly sample $N = 8$ matching pairs of pixel points from the two images, $\boldsymbol{x_1}^{\boldsymbol{i}}, \boldsymbol{x_2}^{\boldsymbol{i}}$ is the $\boldsymbol{i}^{th}$ matching pair
- fit a fundamental matrix (*exact* model fit) that satisfies $\boldsymbol{x_2^i}^{\mathrm{T}} \boldsymbol{F} \boldsymbol{x_1}^{\boldsymbol{i}} = \boldsymbol{0}$ for all $\boldsymbol{i}$ in 1 to N
- examine remaining matching pairs and count how many are a *good* fit for the model $\boldsymbol{F}$. (C is the "consensus set").
- if C is sufficiently large, then refit $\mathbf{F}$ using *all* matching pairs in the consensus set, using least squares. If the model fit is the best so far, then save it.

} until some condition is satisfied

RANSAC computes a fundamental matrix F in two ways:

- Using the minimal number of points
- Using the least squares (on the consensus set)

Data normalization is only used for the latter.

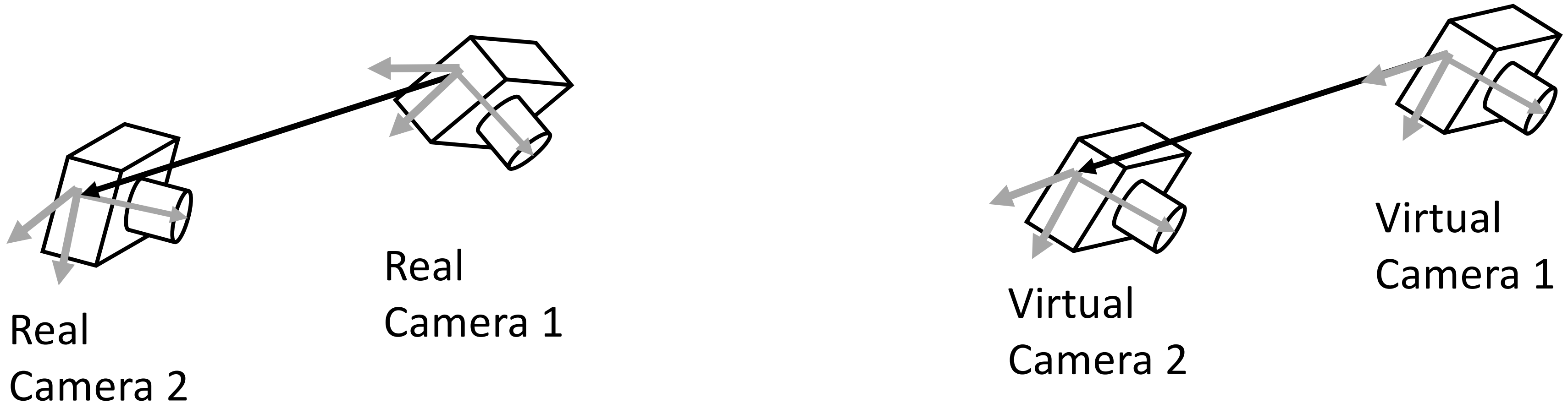
Overview of today

- Estimating the fundamental matrix
    - 8 point algorithm, least squares
    - Normalization
    - rank 2 constraint
    - RANSAC

- Disparity estimation
    - Image rectification
    - Inherent ambiguities in depth estimation

Image rectification

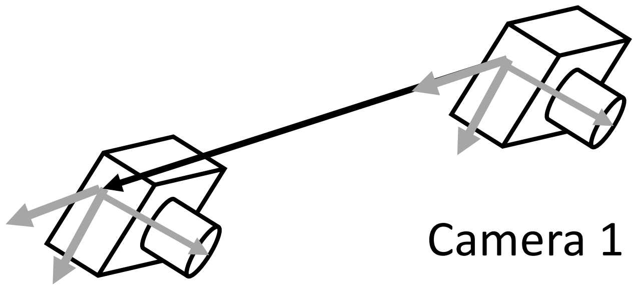
When we discussed homographies (lecture 17) , we briefly considered the problem of rectifying two cameras, *when we knew the camera matrices* ***P***.

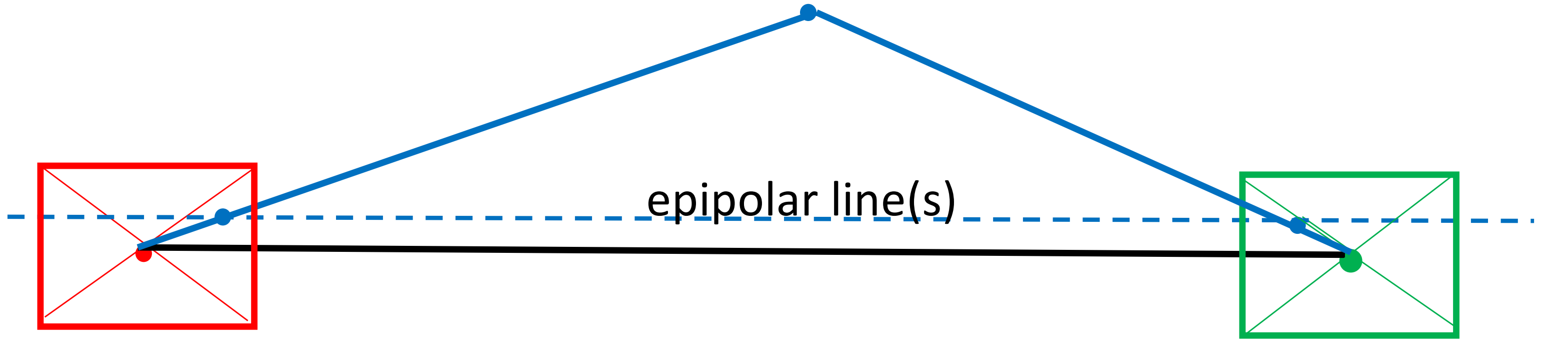


Now, let's briefly consider the *uncalibrated* case.

We have estimated the fundamental matrix $\boldsymbol{F}$.
Can we rectify the two images?

Rectified cameras





When we discussed the special case of rectified cameras last lecture, we observed that the epipolar lines are the image rows.

What did we observe about the epipoles ?

Image rectification

Given two images and a fundamental matrix $\boldsymbol{F}$:

- find the epipoles in images 1 and 2 by solving $\boldsymbol{F}\,\tilde{\boldsymbol{e}}_{\mathbf{1}} = \mathbf{0}$ and $\tilde{\boldsymbol{e}}_{\mathbf{2}}^{\mathbf{T}}\boldsymbol{F} = \mathbf{0}$, respectively

- define homographies $\boldsymbol{H}_{\mathbf{1}}$ and $\boldsymbol{H}_{\mathbf{2}}$ that map the $\tilde{\boldsymbol{e}}_{\mathbf{1}}$ and $\tilde{\boldsymbol{e}}_{\mathbf{2}}$ to the 2D point at infinity in the x axis direction.
  (The homographies also need to ensure that epipolar lines match. We will skip that detail in next few slides.)

- deform the two image intensities $I_1(x, y)$ and $I_2(x, y)$ using the two homographies
  (Note: all pixels are mapped, not just the matching pairs)

Here is an example of a homography that maps $\boldsymbol{e_1} = (e_u, e_v, 1)$ to a point at infinity in the x direction.

$$\begin{array}{cc} \boldsymbol{H_1} & \boldsymbol{e_1} \\ \begin{bmatrix} \frac{1}{e_u} & 0 & 0 \\ -\frac{e_v}{e_u} & 1 & 0 \\ -\frac{1}{e_u} & 0 & 1 \end{bmatrix} & \begin{bmatrix} e_u \\ e_v \\ 1 \end{bmatrix} \end{array} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Here is another example of a homography that maps $\boldsymbol{e_1} = (e_u, e_v, 1)$ to a point at infinity in the x direction.

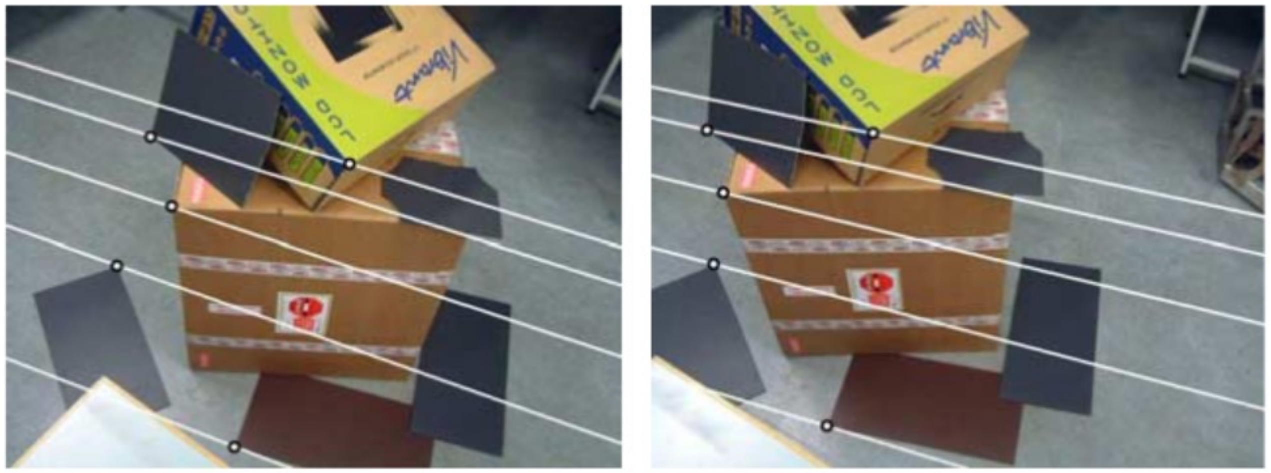
$$\overbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \uparrow & 0 & 1 \end{bmatrix} \begin{bmatrix} e_u & e_v & 0 \\ -e_v & e_u & 0 \\ 0 & 0 & 1 \end{bmatrix}}^{\boldsymbol{H'_1}} \overset{\boldsymbol{e_1}}{\begin{bmatrix} e_u \\ e_v \\ 1 \end{bmatrix}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

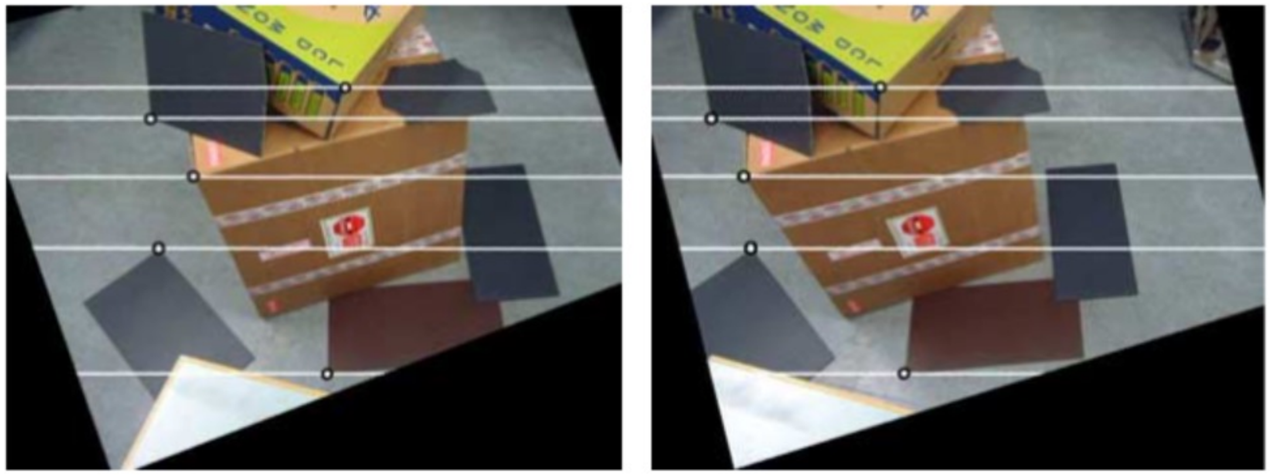
$$\frac{-1}{{e_u}^2 + {e_v}^2}$$

There are many other homographies that achieve this.

We also need to ensure that epipolar lines match. So another homography is needed for that.

Example result


(a) Originals for Boxes example, Left and Right

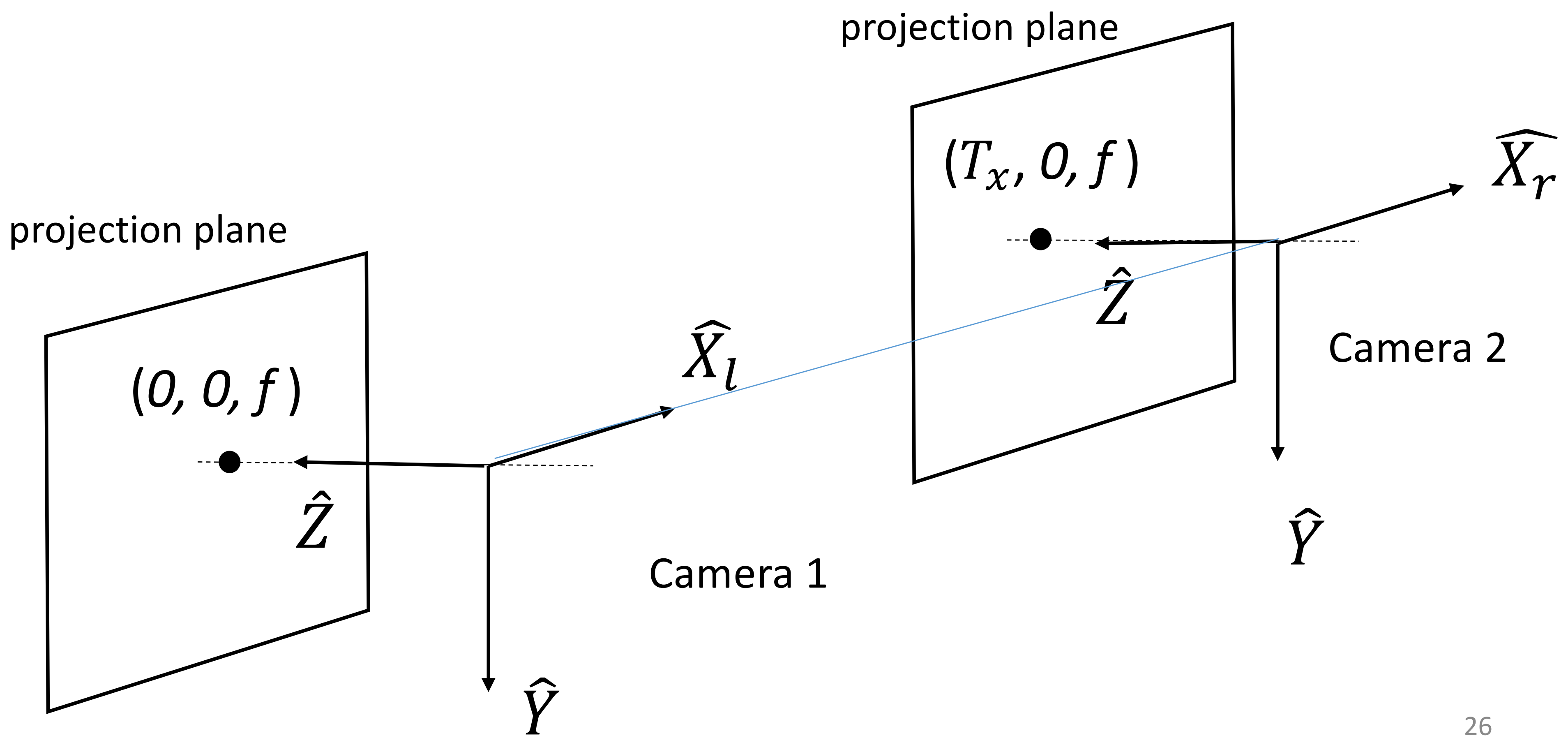

(b) Proposed Rectification Method

Mallon and Whelan, (2005)

Let's now turn to the problem of how to estimate depth.

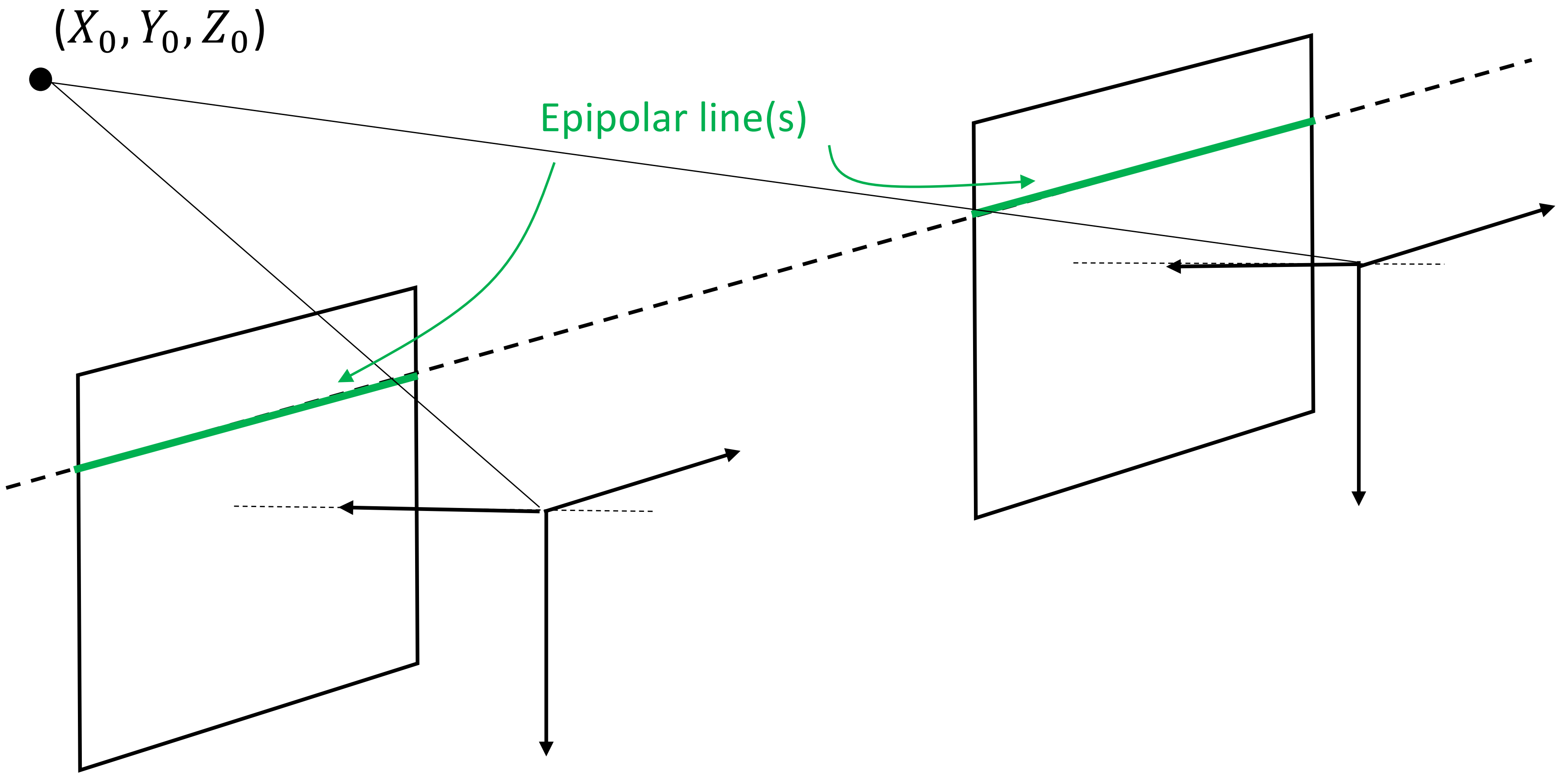
Assume for the moment that we have two *real* cameras that are in a rectified arrangement. (BIG ASSUMPTION).

This is the geometry we considered in lecture 12 for camera translation.



The corresponding image points lie along the same row.

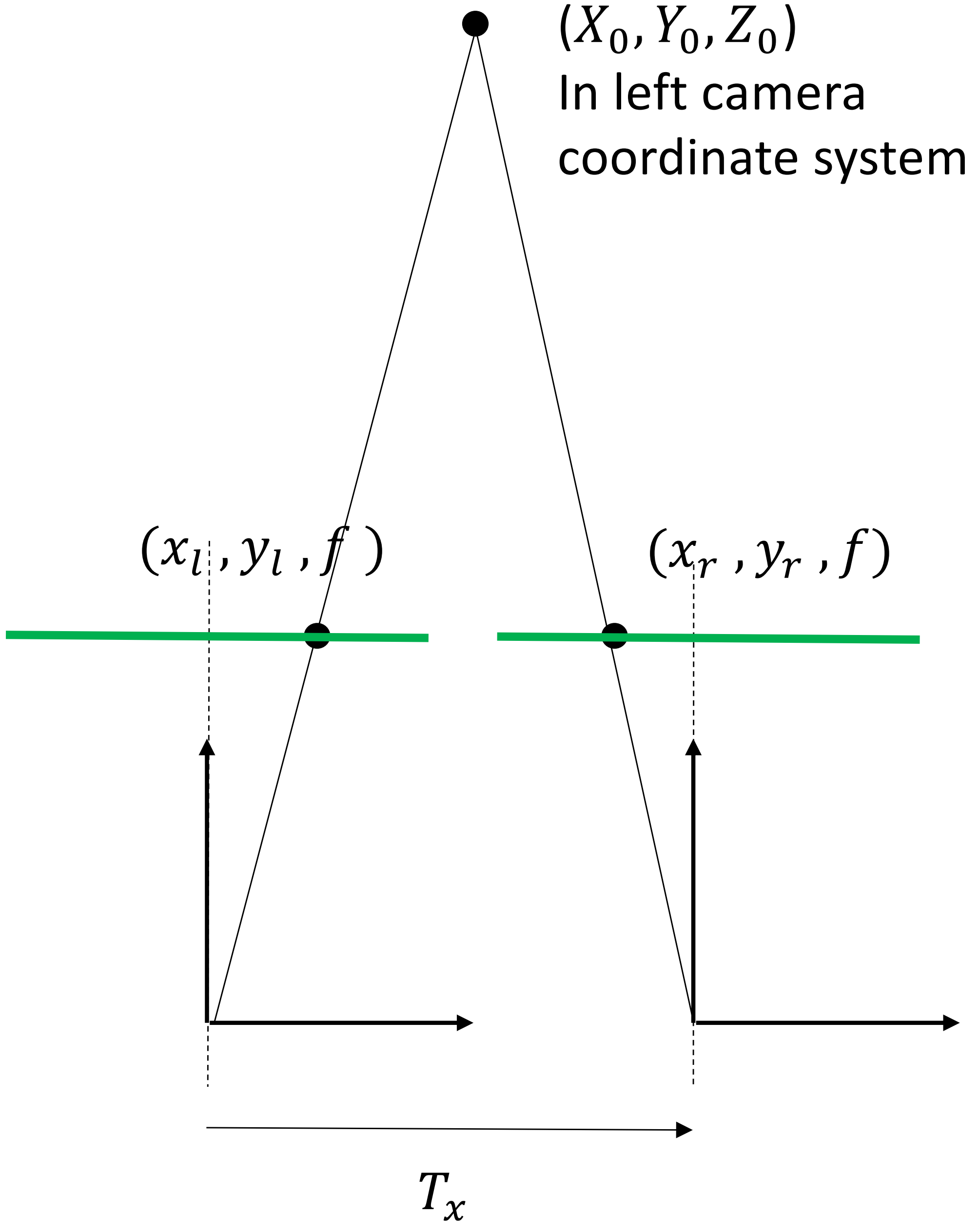
To find correspondences, we only *need* to search within each row.
(It is better to combine information across rows too – see next lecture.)

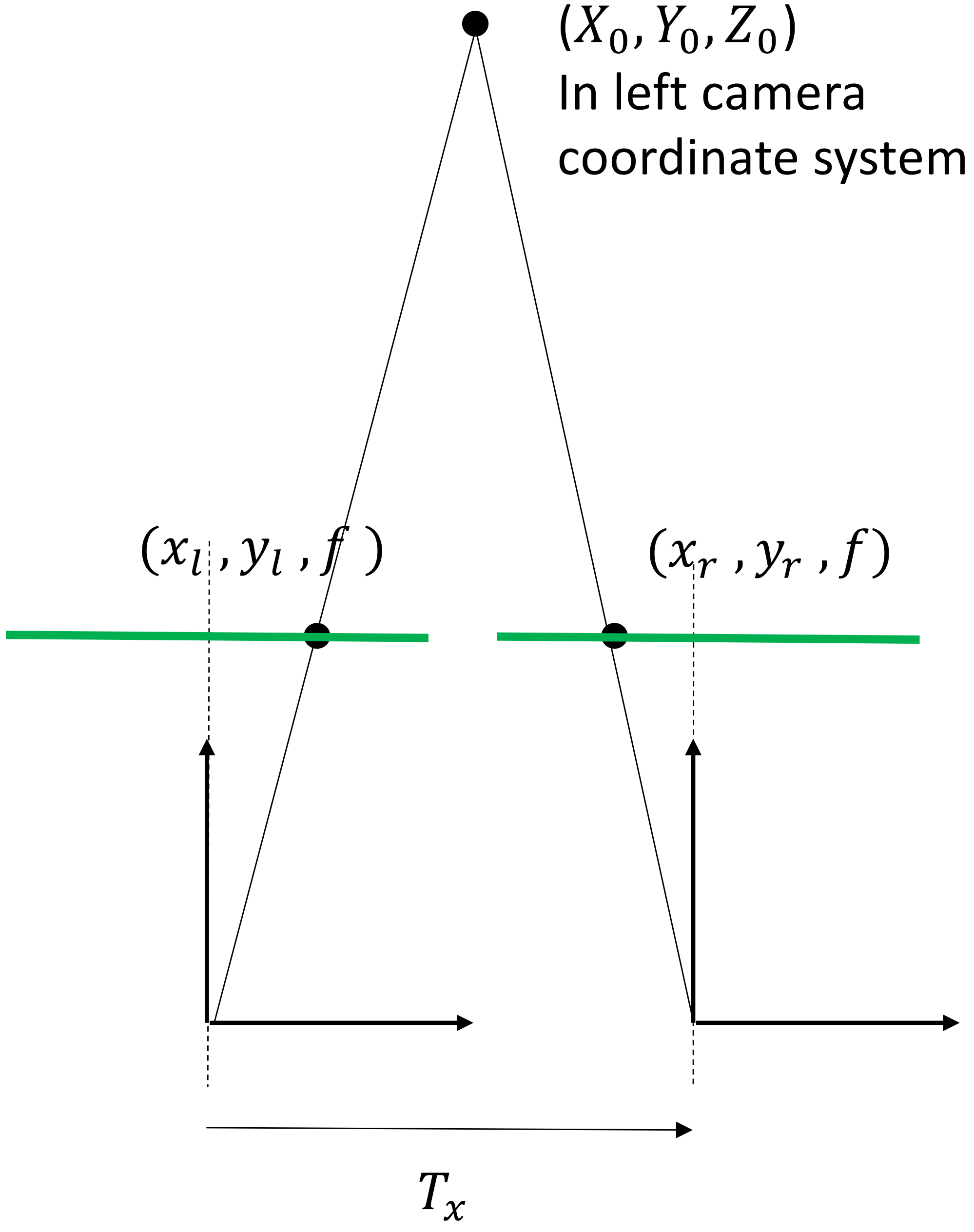


How does the difference in the left and right image position of a point depend on the 3D depth ?

Here we will only consider positions in the projection plane. This can be converted to pixels if certain camera parameters are known.

View from above (XZ)

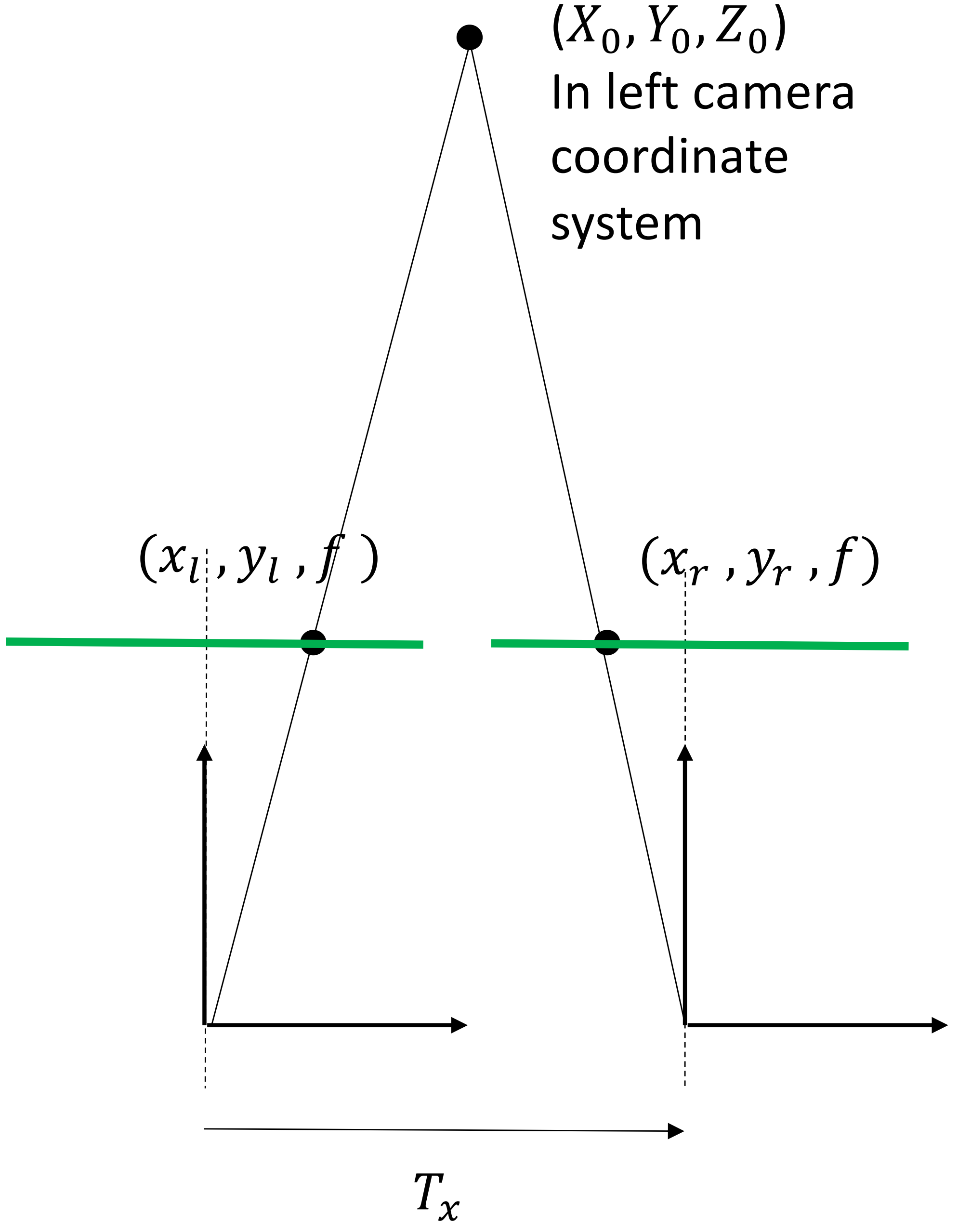




Binocular disparity

$$d \equiv x_l - x_r$$

is the difference in image position of a 3D point as seen by two eye.



Binocular disparity

$$d \equiv x_l - x_r$$

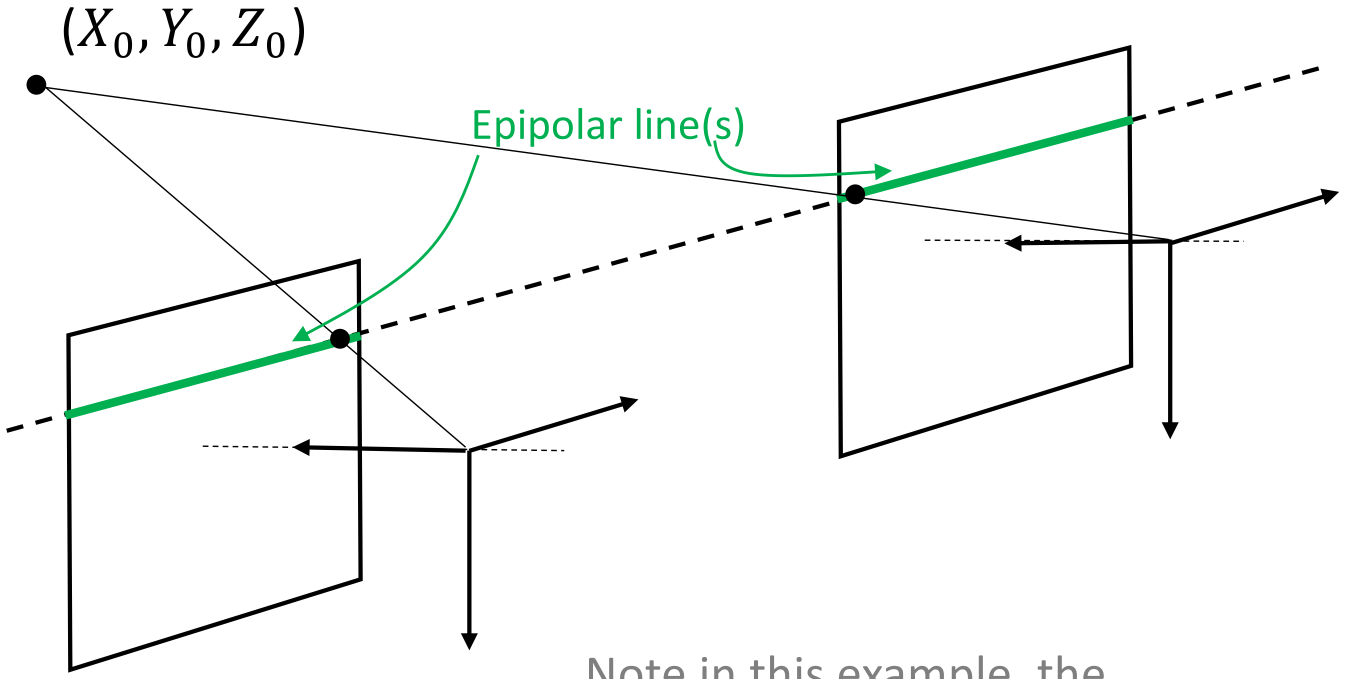
is the difference in image position of a 3D point as seen by two eye.

$$\frac{x_l}{f} = \frac{X_0}{Z_0} \qquad \frac{x_r}{f} = \frac{X_0 - T_x}{Z_0}$$

$$\boldsymbol{d = \frac{f\,T_x}{Z_0}}$$

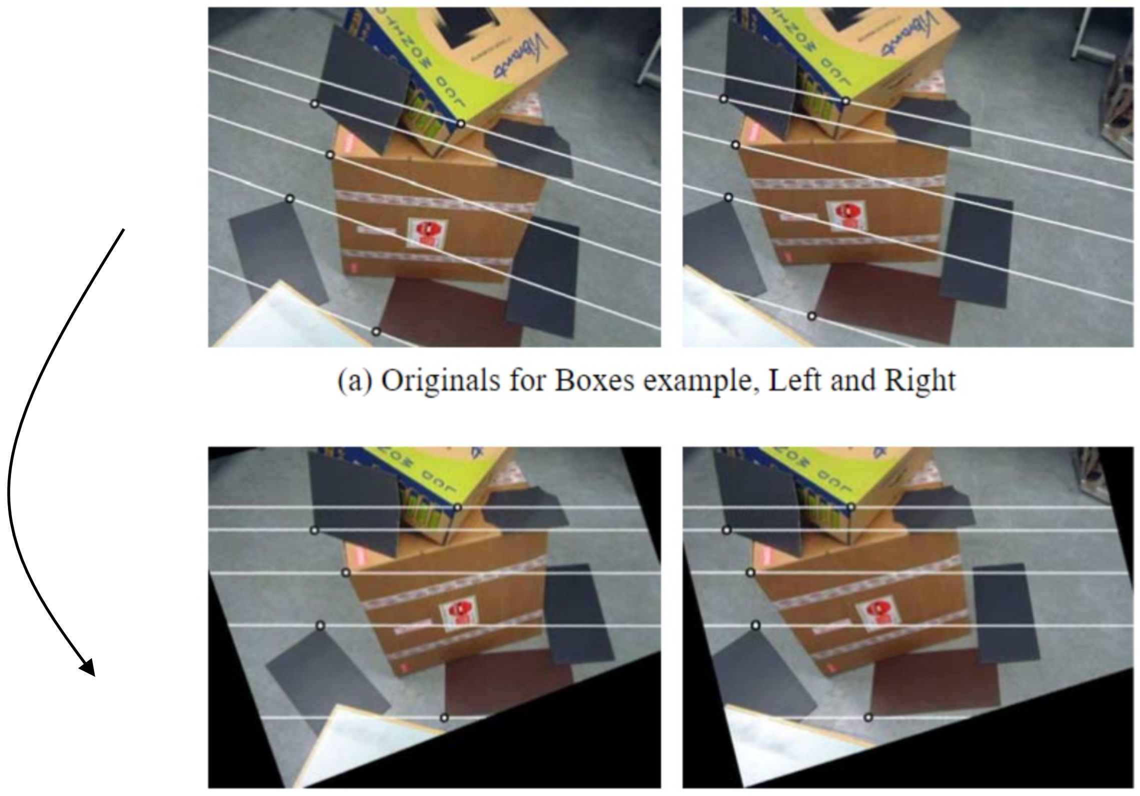
Thus, if we can estimate the binocular disparity of a point, then we get the depth Z of that point.

We will address this problem next lecture.



Note in this example, the disparity is very large!

What if we don't have rectified cameras?   Rather what we have rectified images that were computed using homographies derived the fundamental matrix?


(a) Originals for Boxes example, Left and Right

(b) Proposed Rectification Method

If we now compute the disparities at each pixel, then what can we say about depth ?

ASIDE: “Projective Reconstruction Theorem”

Suppose two cameras view the same scene.

$$\begin{bmatrix} w_1 x_1 \\ w_1 y_1 \\ w_1 \end{bmatrix} = \mathbf{P}_1 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} w_2 x_2 \\ w_2 y_2 \\ w_2 \end{bmatrix} = \mathbf{P}_2 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Then, the same set of image positions $\{ (x_1, y_1) \}$ and $\{ (x_2, y_2) \}$ can be produced by a *different* camera viewing a *different* 3D scene $\{ (X, Y, Z) \}$ ...

ASIDE: "Projective Reconstruction Theorem" (continued...)

$$\begin{bmatrix} w_1 x_1 \\ w_1 y_1 \\ w_1 \end{bmatrix} = \mathbf{P}_1 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} w_2 x_2 \\ w_2 y_2 \\ w_2 \end{bmatrix} = \mathbf{P}_2 \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Let $\boldsymbol{M}$ be any invertible 4x4 matrix. Then,

$$\underbrace{\begin{bmatrix} w_i x_i \\ w_i y_i \\ w_i \end{bmatrix}}_{\text{same image } i \text{ points}} = \underbrace{(\mathbf{P}_i \mathbf{M})}_{\text{new camera matrix}} \underbrace{(\mathbf{M}^{-1} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix})}_{\text{new 3D point(s)}}$$

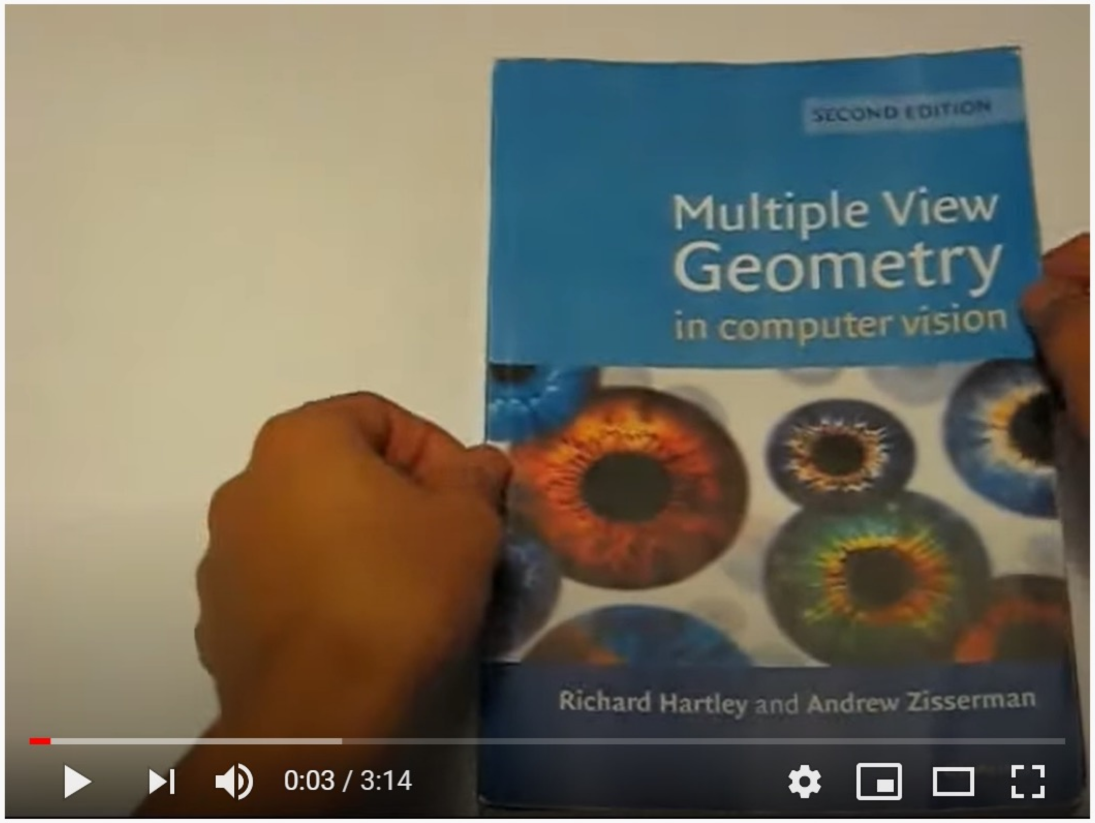
ASIDE: “Projective Reconstruction Theorem” (continued…)

Bottom line:

Given two images from *uncalibrated* cameras, we can *at best* reconstruct the scene up to a “projective ambiguity” (even if we can compute matching points),

Fortunately, in many situations we do know some of the camera parameters (intrinsics, at least).

Check out this funny video


SECOND EDITION
Multiple View Geometry in computer vision
Richard Hartley and Andrew Zisserman
0:03 / 3:14

The Fundamental Matrix Song (Stereo-Image Matching using epipolar lines)

https://www.youtube.com/watch?v=GMil9tpwE_Q

Announcements…

Assignment 3 should be posted by this weekend.

Midterm 2 is in class next Thursday. We will organize a midterm review for Tuesday evening.

Next week   we will have one last look at the stereo problem. We will discuss dense stereo matching, and some subtle aspects.