Stereo Vision

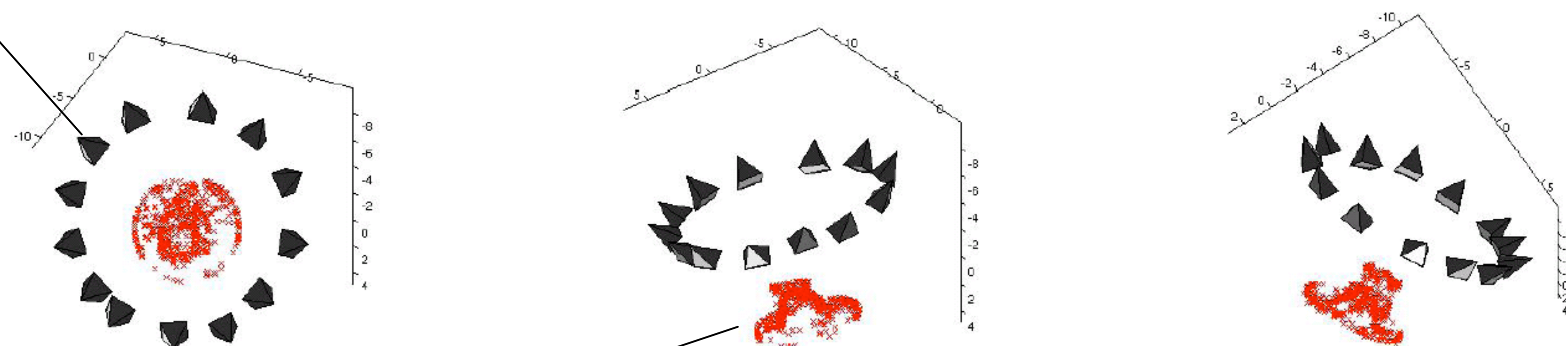
Introduction

- Given two (or more) images of an object, reconstruct 3D geometry
  - WITHOUT knowledge of relative camera locations, but calibrated cameras
  - With uncalibrated cameras
- STRATEGY:
  - Find a set of point matches between images
  - Use geometric constraints to infer camera geometry
  - Triangulate points once relative camera locations are known

Introduction

(a) Input images

Camera locations



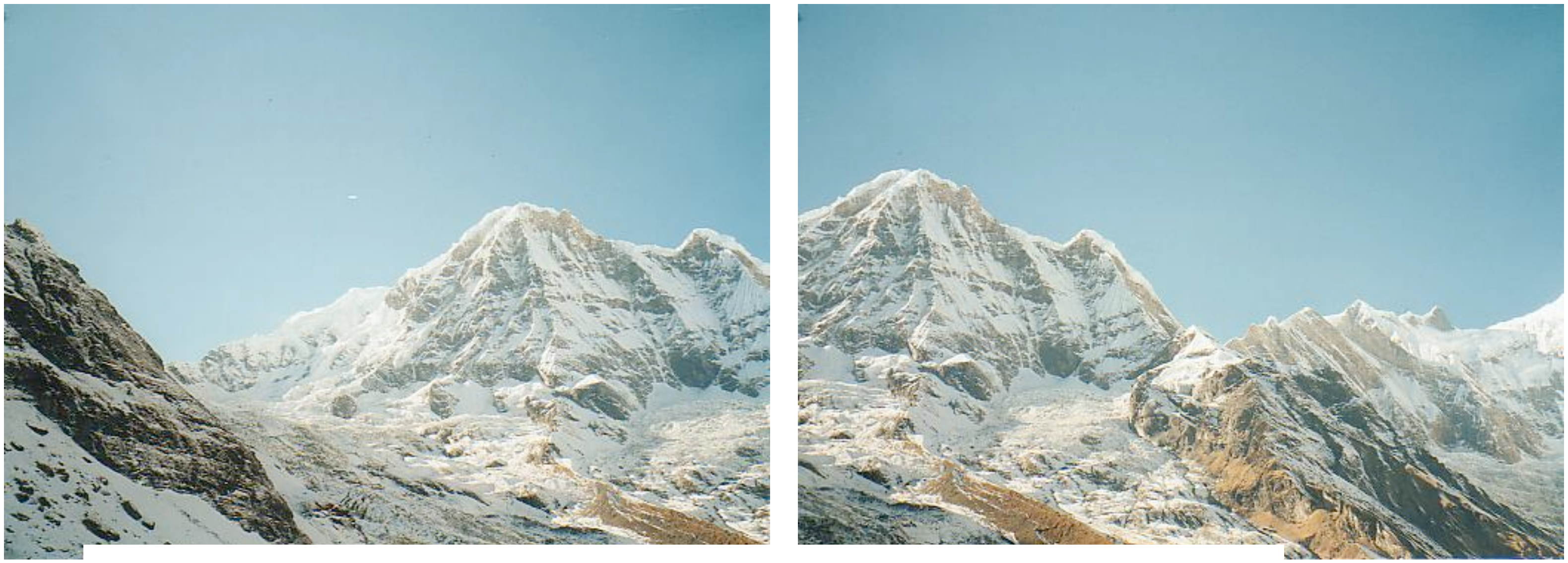
Set of 3D points

(b) Output 3D model

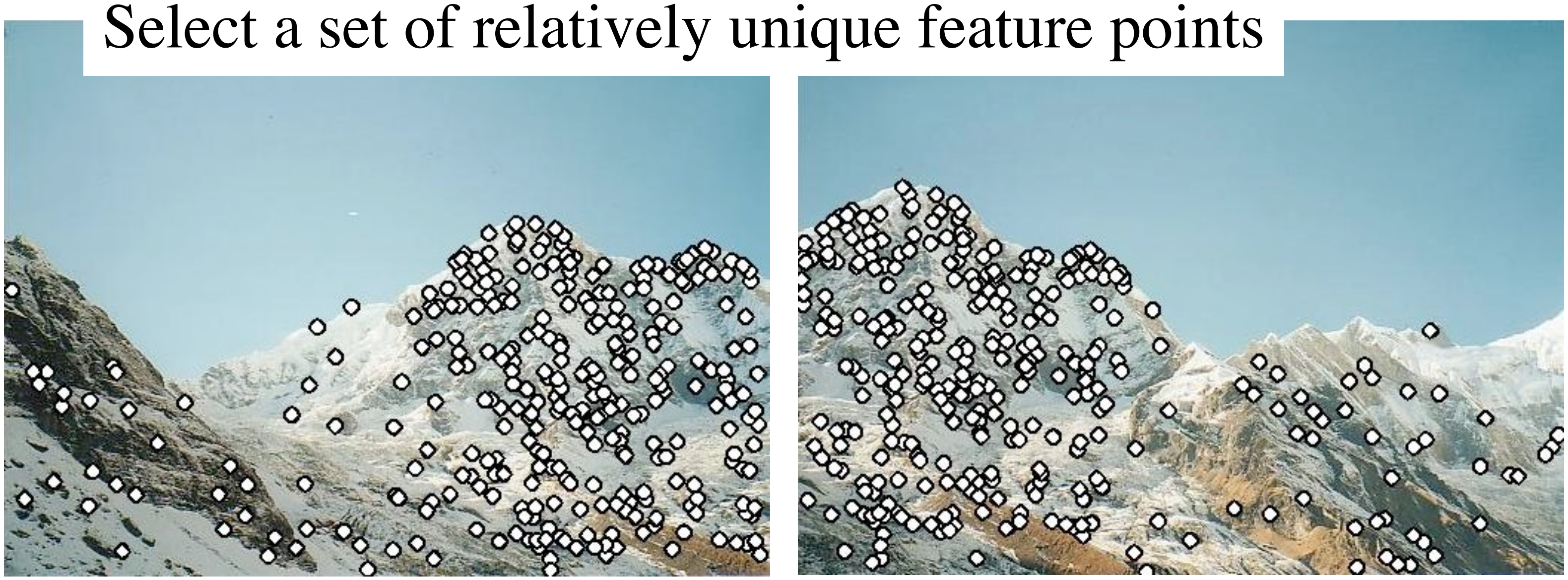
Overview

- Feature Matching
- Image Matching
- Geometric constraints
- 3D Reconstruction

Panoramas: Homography

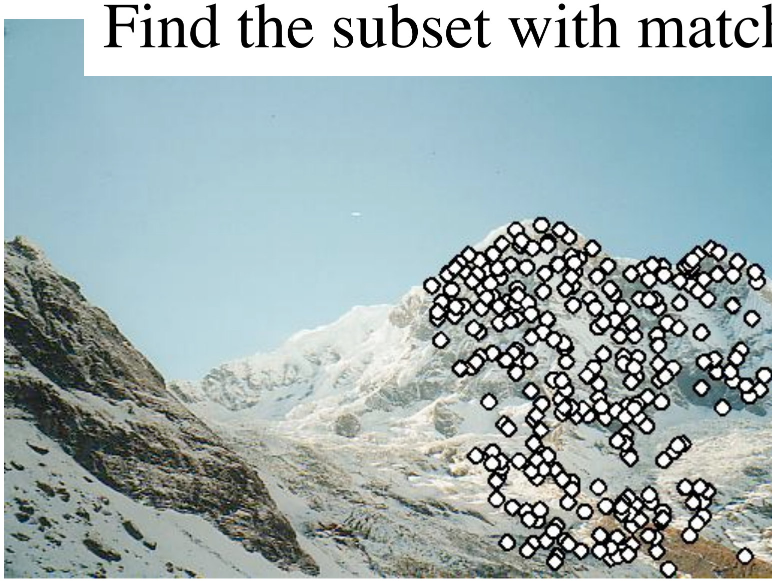


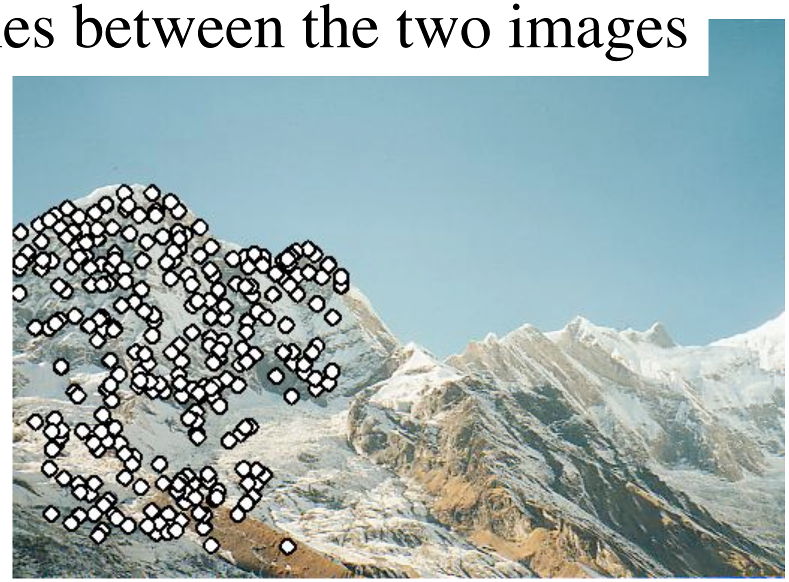
Select a set of relatively unique feature points



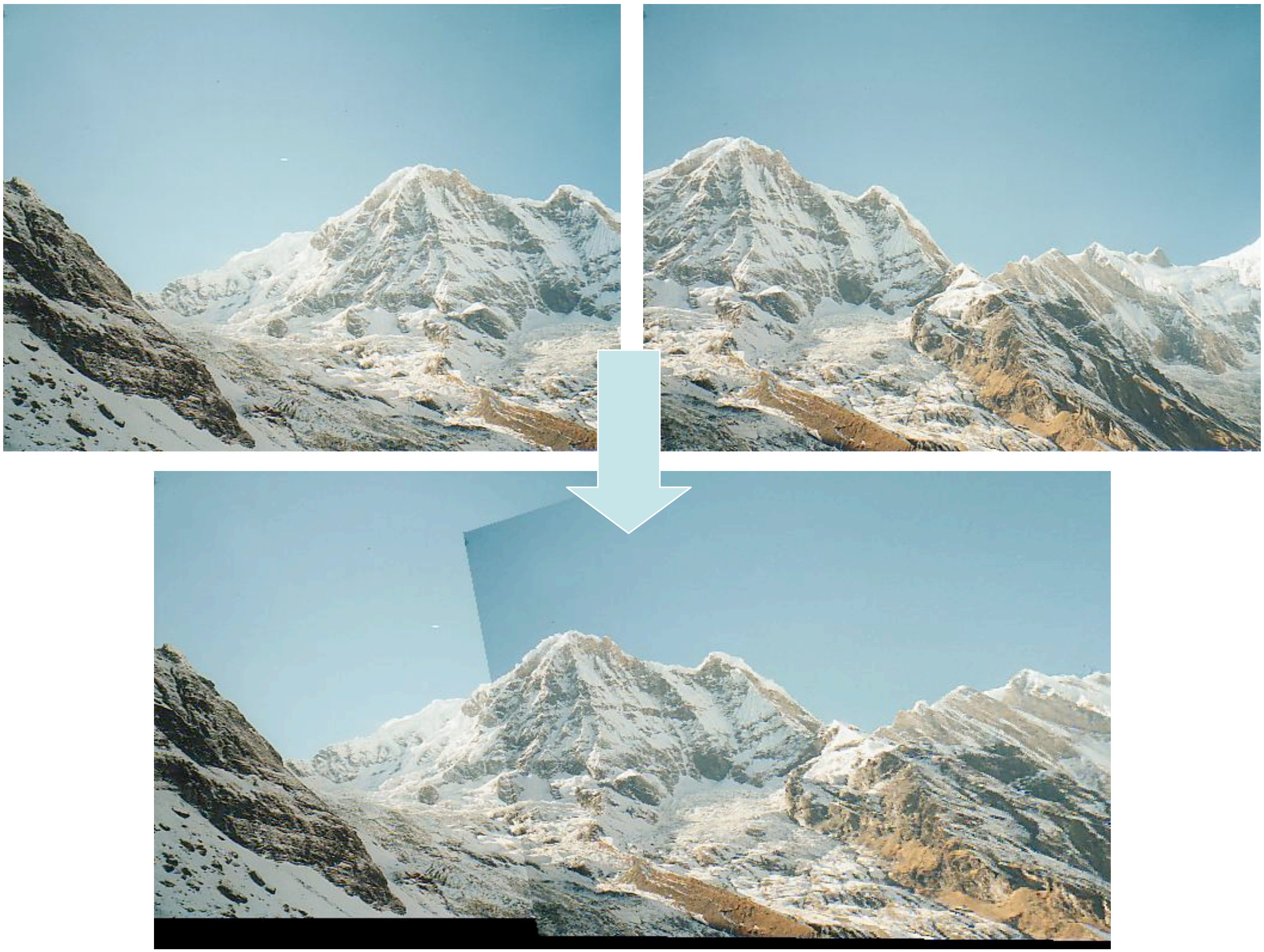
Panoramas: Homography

Find the subset with matches between the two images





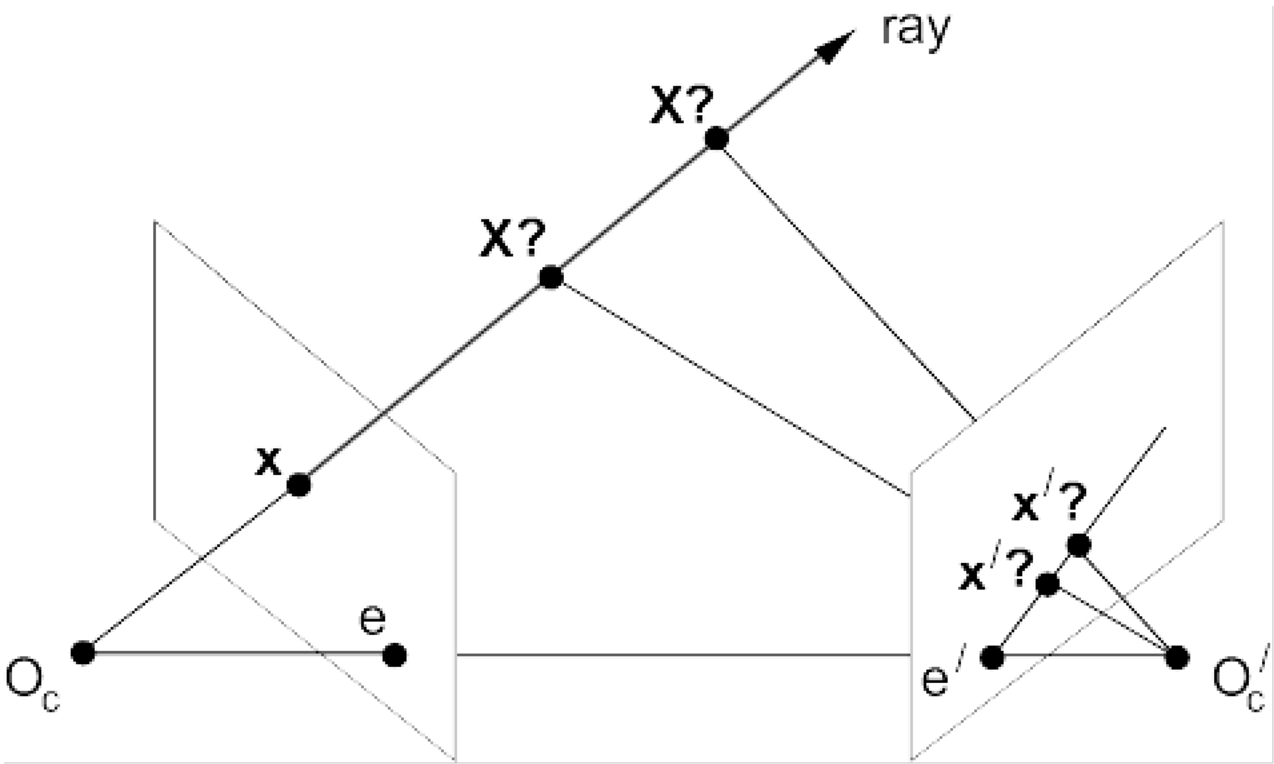
Panoramas: Homography



3D: ???

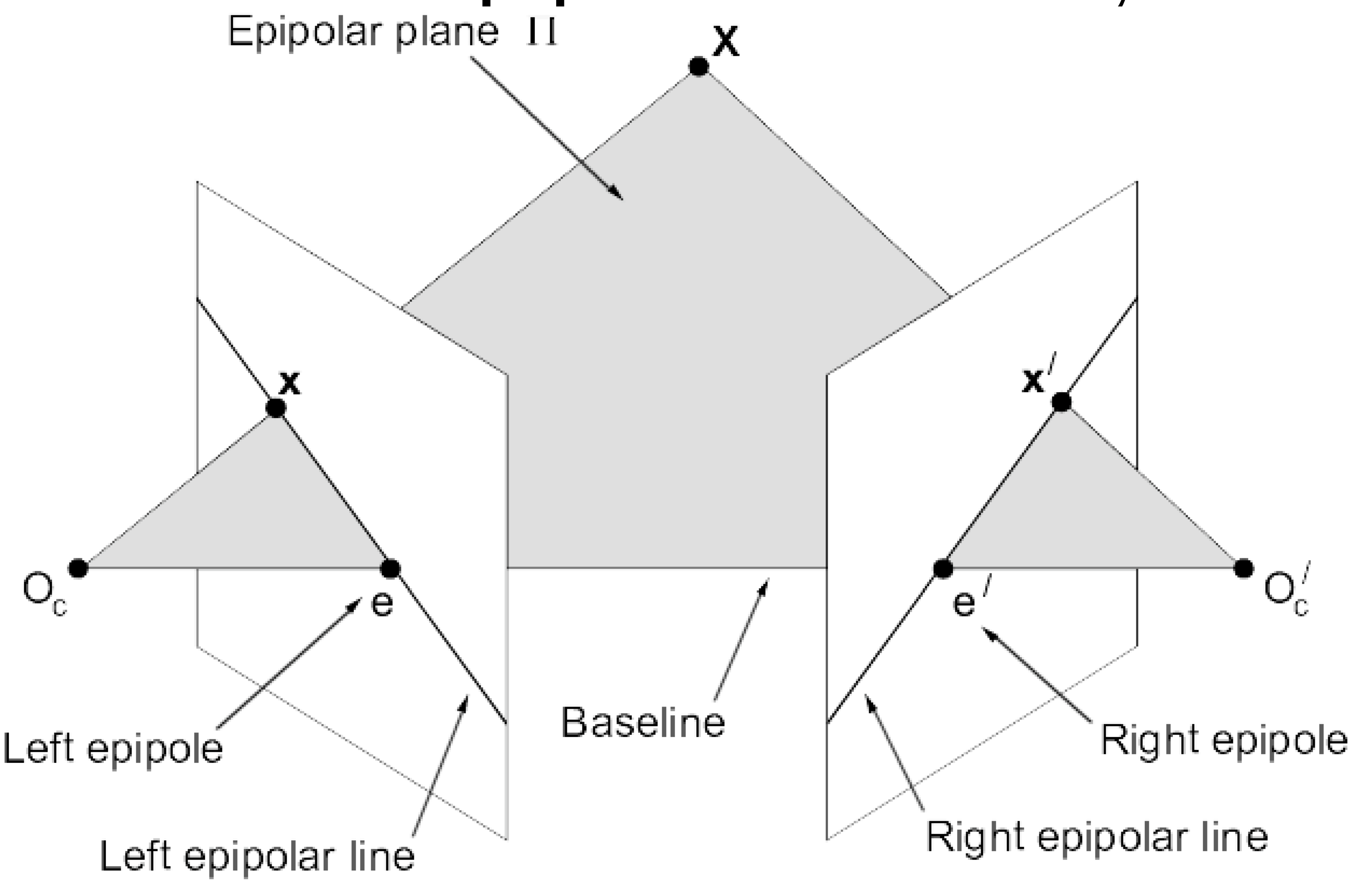
3D: Epipolar Geometry

3D: Epipolar Geometry

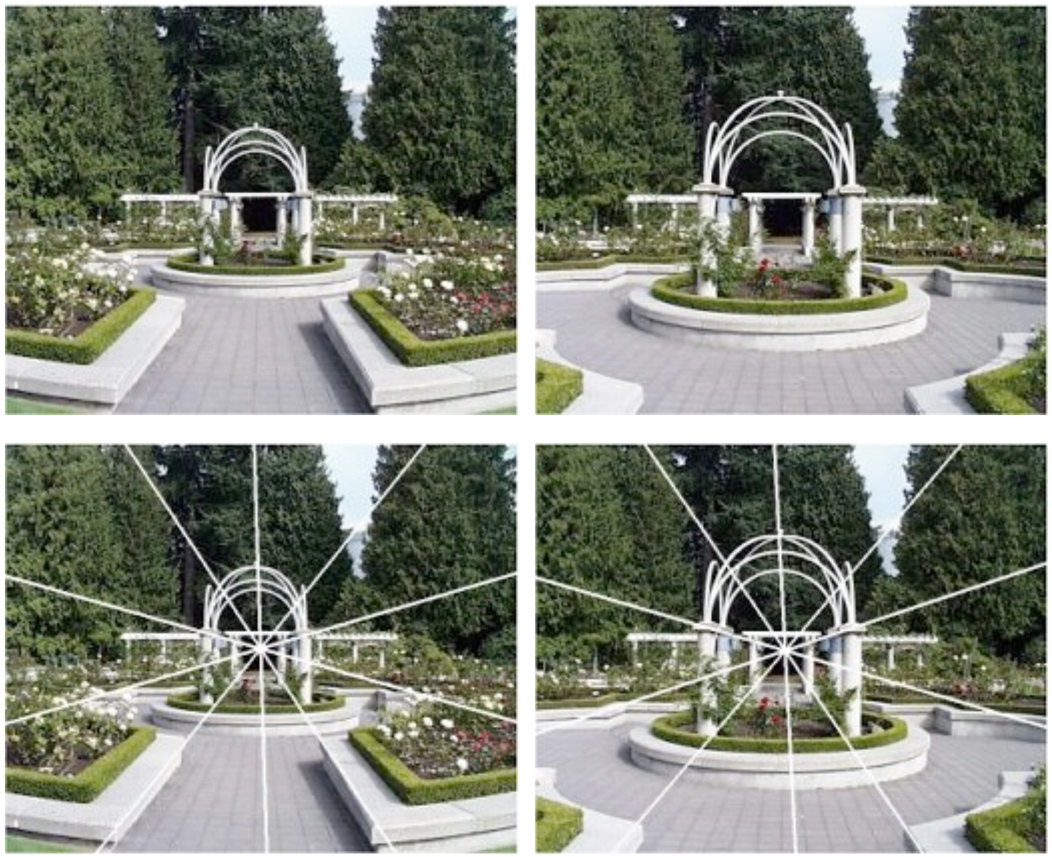


Figures: Andrew Gee

3D: Epipolar Geometry



3D: Epipolar Geometry

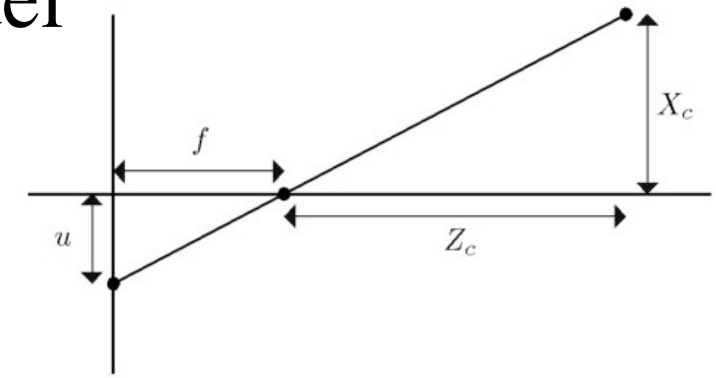


3D: Epipolar Geometry

Camera Models (review)

- Pinhole camera model

$$\begin{bmatrix} u \\ v \end{bmatrix} = f \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix}$$

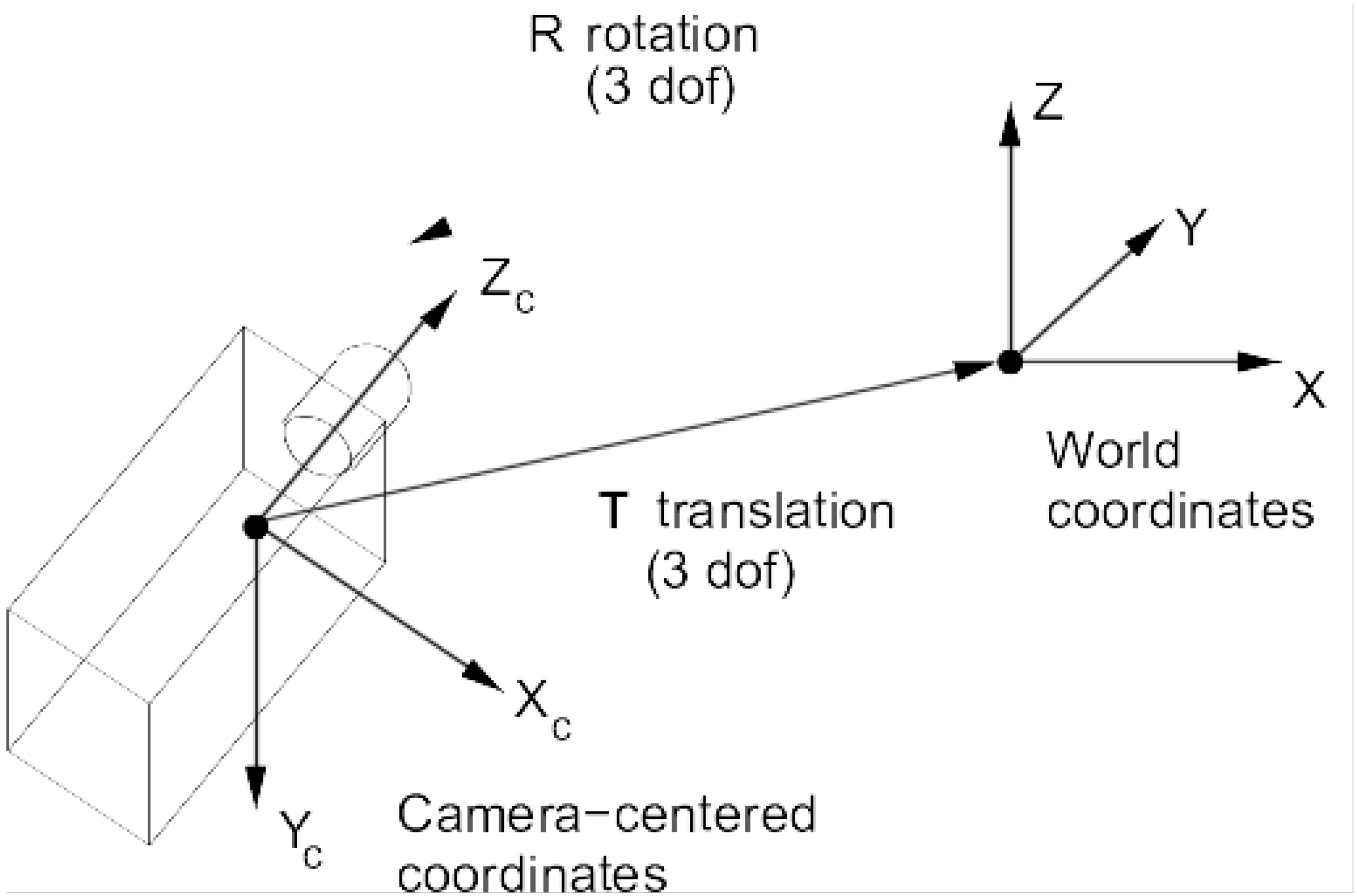


$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$$

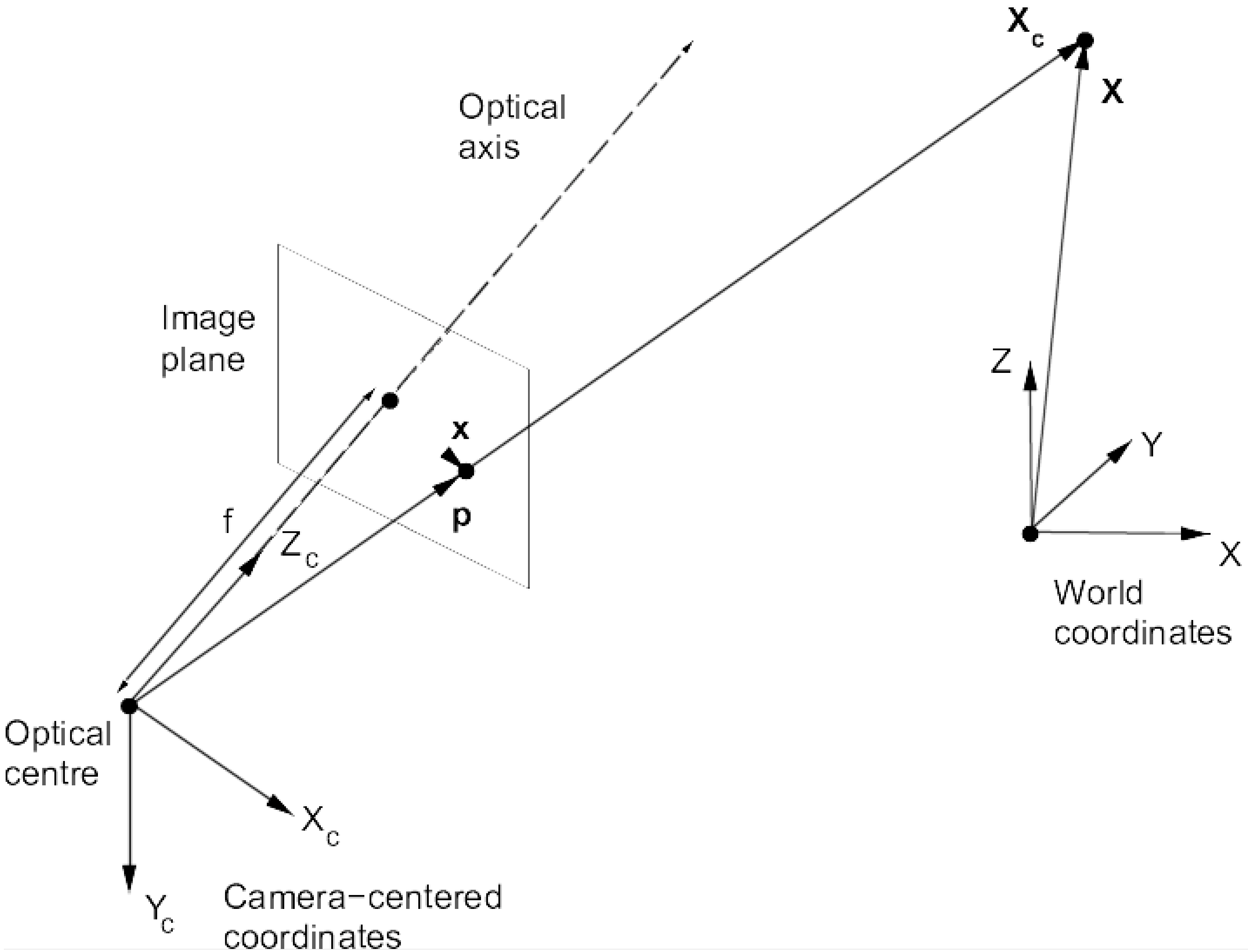
- or equivalently…

$$\tilde{\mathbf{u}} \sim \mathbf{K}\mathbf{X}_c$$

Camera Models



Camera Models



Camera Models

$$\mathbf{X}_c = \mathbf{R}\mathbf{X} + \mathbf{t}$$

- World to camera transform

$$\tilde{\mathbf{u}} \sim \mathbf{K}(\mathbf{R}\mathbf{X} + \mathbf{t})$$

- Projection equation becomes
- or... $\tilde{\mathbf{u}} \sim \mathbf{K}\left[\mathbf{R}|\mathbf{t}\right]\begin{bmatrix}\mathbf{X}\\ 1\end{bmatrix} \sim \mathbf{K}\left[\mathbf{R}|\mathbf{t}\right]\tilde{\mathbf{X}}$

Camera Models

- K(R|t) is a 3x3 matrix…
- …but special because $R^T R = I$
- Relax this constraint

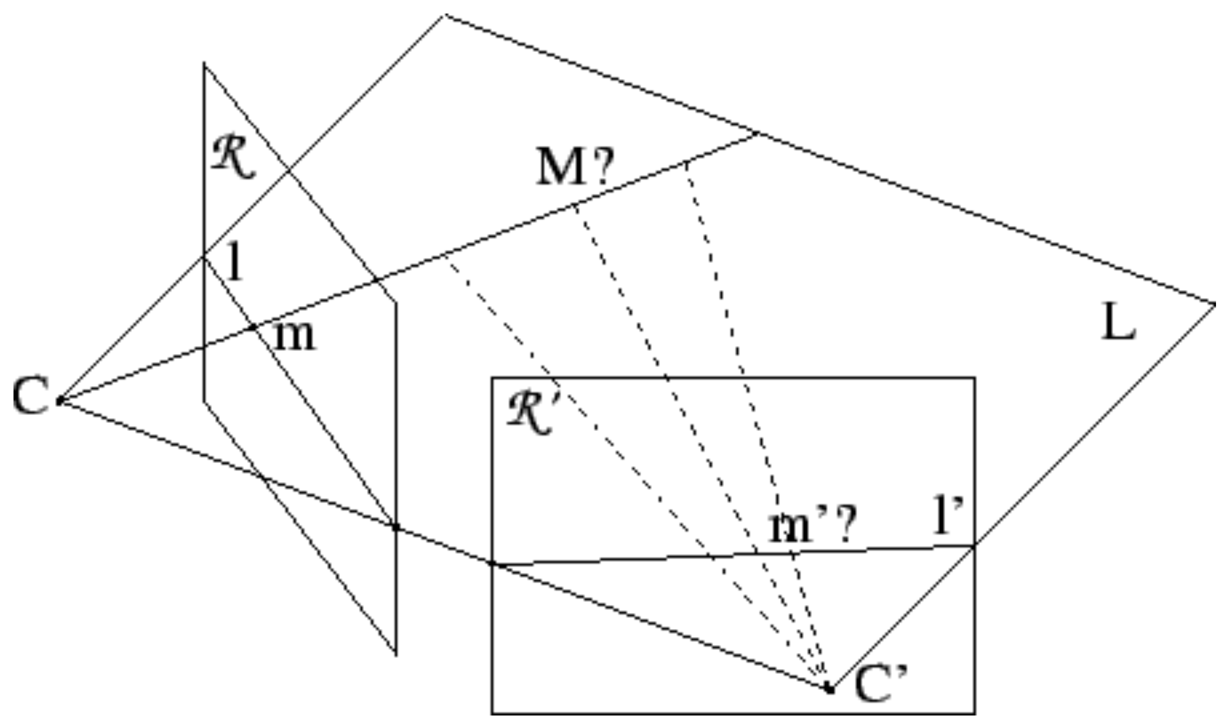
$$\tilde{\mathbf{u}} = \mathbf{P}\tilde{\mathbf{X}}$$

- Where P is an arbitrary 3x4 matrix i.e.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

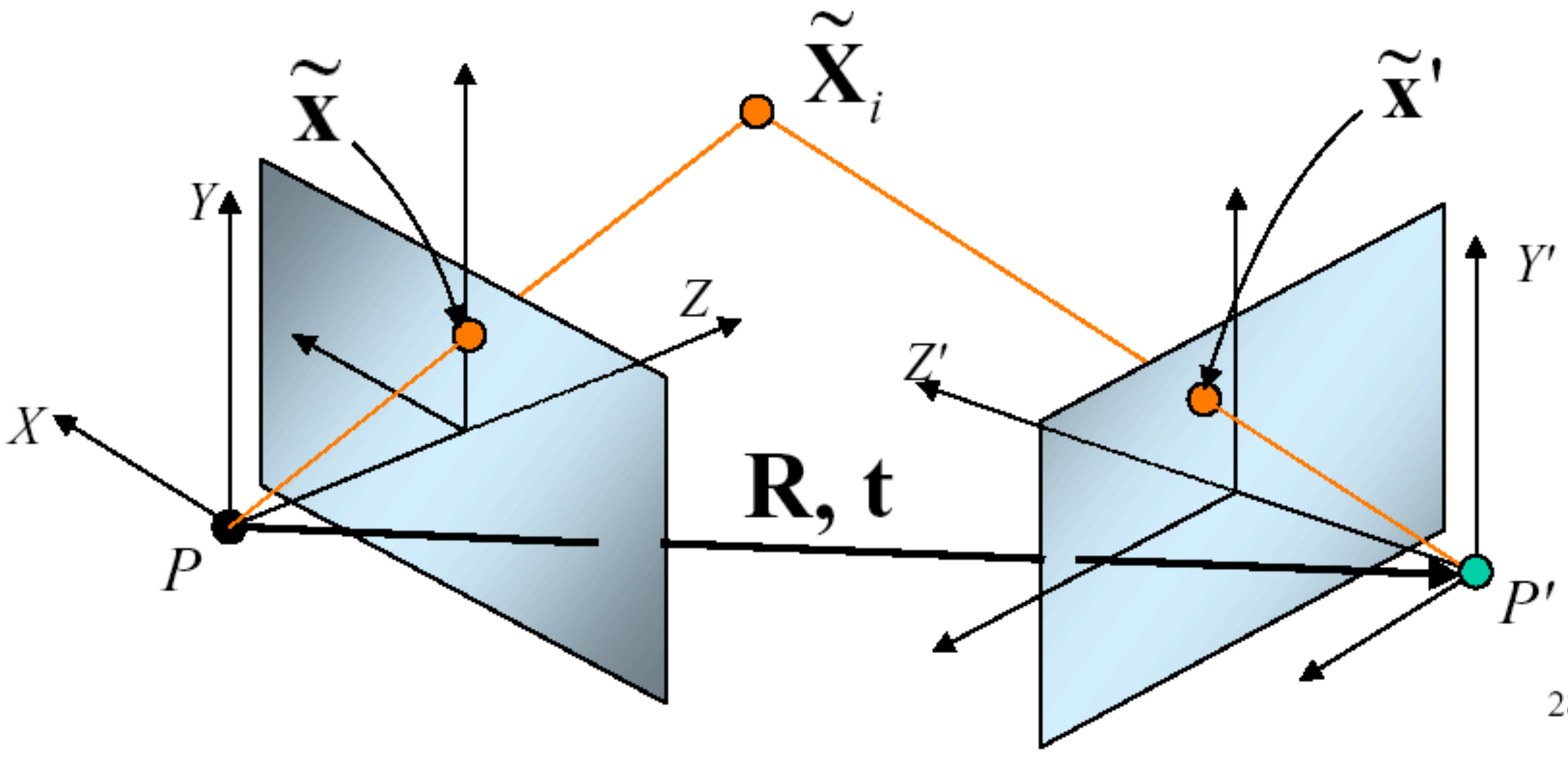
Relations between image coordinates

Given coordinates in one image, and the transformation between cameras T = [R t], what are the image coordinates in the other camera's image.

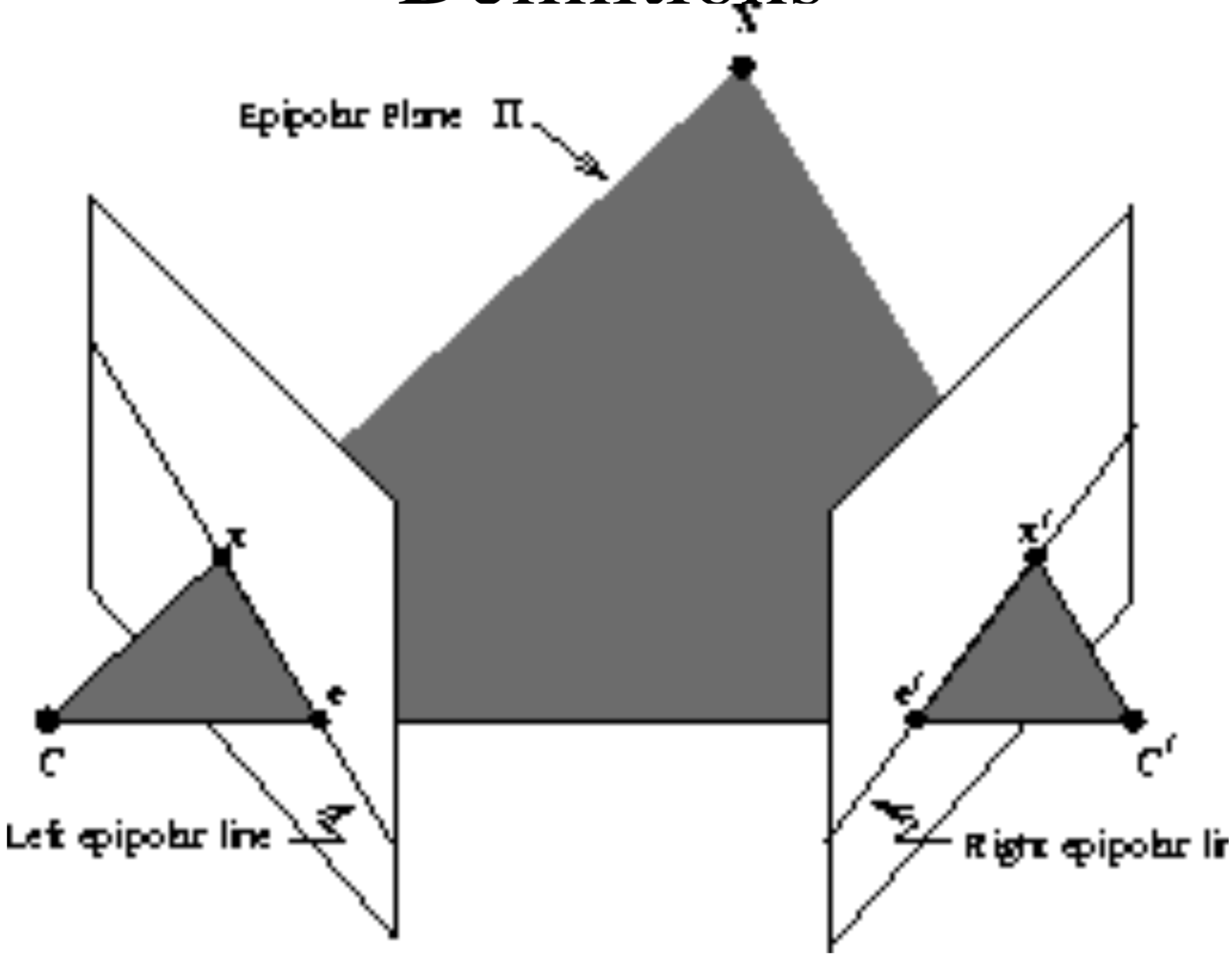


Backprojection to 3D

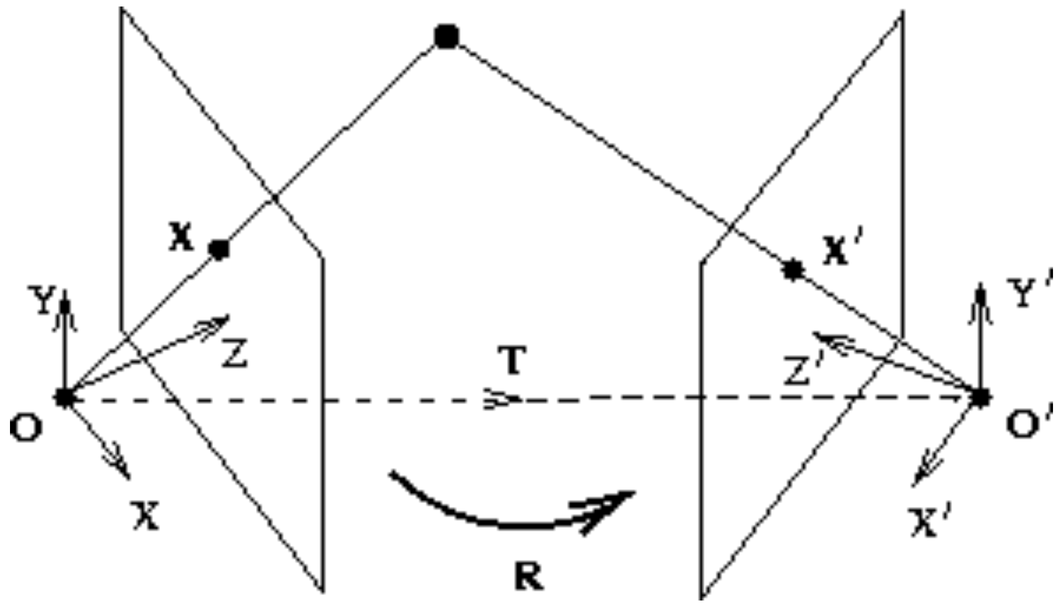
We now know x, x', R, and t
Need X



Definitions



Essential Matrix: Relating between image coordinates



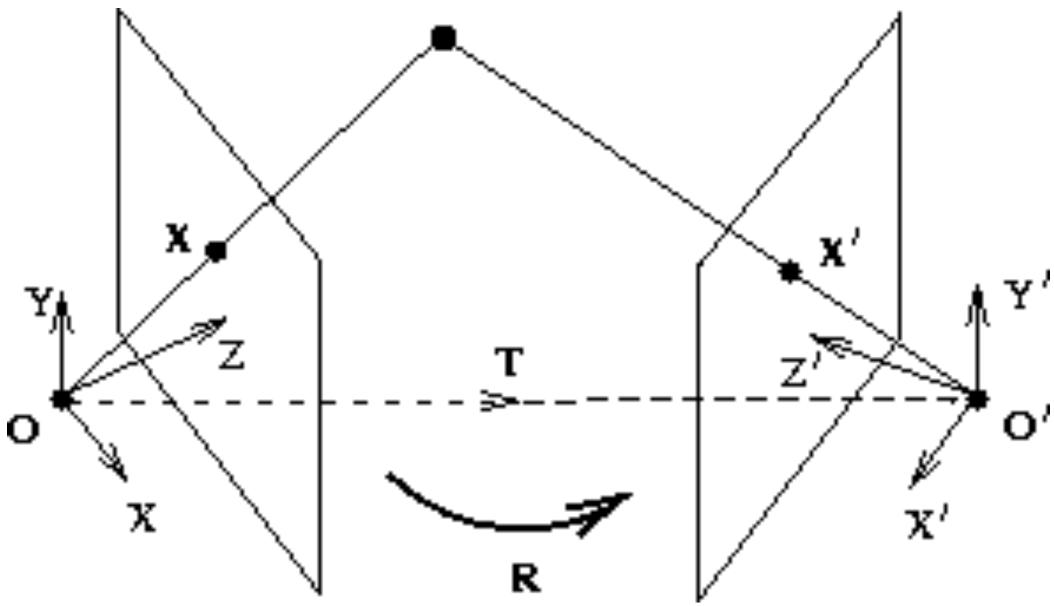
$$\vec{Ox} \quad \vec{O'x'} \quad \vec{OO'}$$

Are Coplanar, so:

$$\vec{O'x'} \bullet \left(\vec{Ox} \times \vec{OO'} \right) = 0$$

camera coordinate systems,
related by a rotation **R** and a translation **T**:

$$x' = \begin{bmatrix} R & t \\ \vec{0}^T & 1 \end{bmatrix} x$$



$$\mathbf{a}\times\mathbf{v} = \begin{bmatrix} a_y v_z - a_z v_y \\ a_z v_x - a_x v_z \\ a_x v_y - a_y v_x \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$$

$$= \mathbf{A}\mathbf{v}$$

$$x' = \begin{bmatrix} R & t \\ \vec{0}^T & 1 \end{bmatrix} x$$

$$\overrightarrow{O'x'} \bullet \left(\overrightarrow{Ox} \times \overrightarrow{OO'} \right) = 0$$

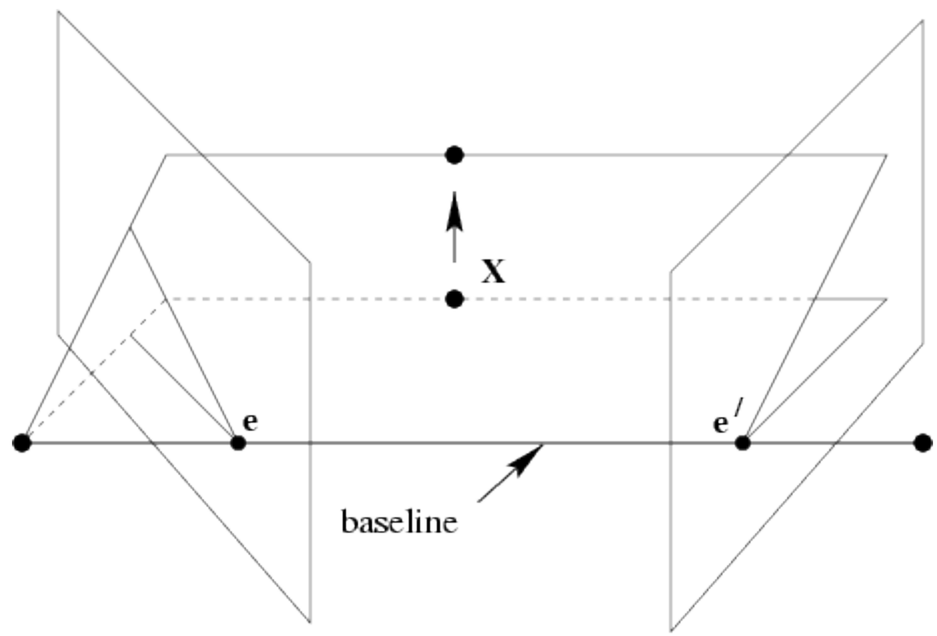
$$x' \bullet \left(\vec{t} \times Rx \right) = 0$$

$$x' \bullet \left(\mathcal{E}\, x \right) = 0$$

$$x' \mathcal{E}\, x = 0$$

$$\mathcal{E} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} R$$

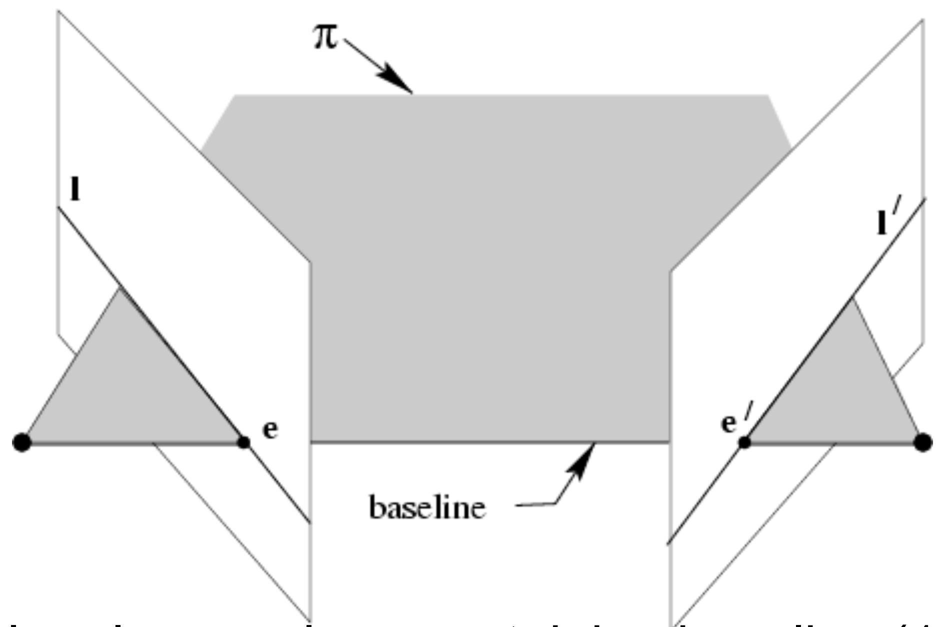
The epipolar geometry



Family of planes π and lines l and l'
Intersection in e and e'

The epipolar geometry

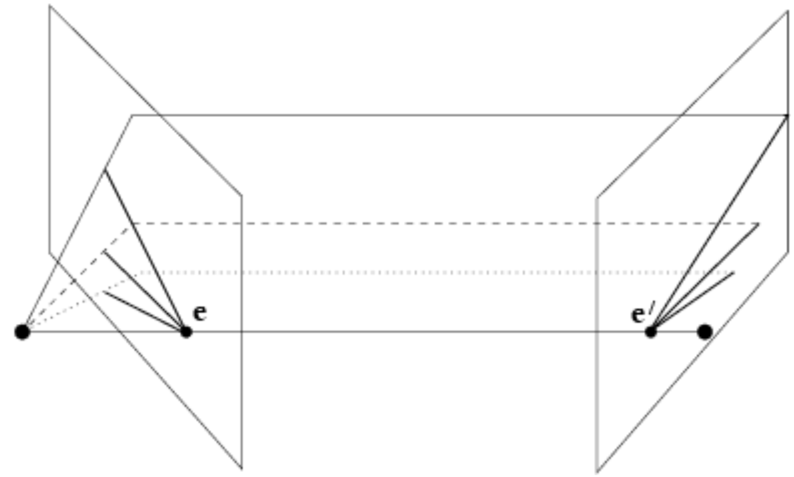
epipoles e,e'
= intersection of baseline with image plane
= projection of projection center in other image
= vanishing point of camera motion direction

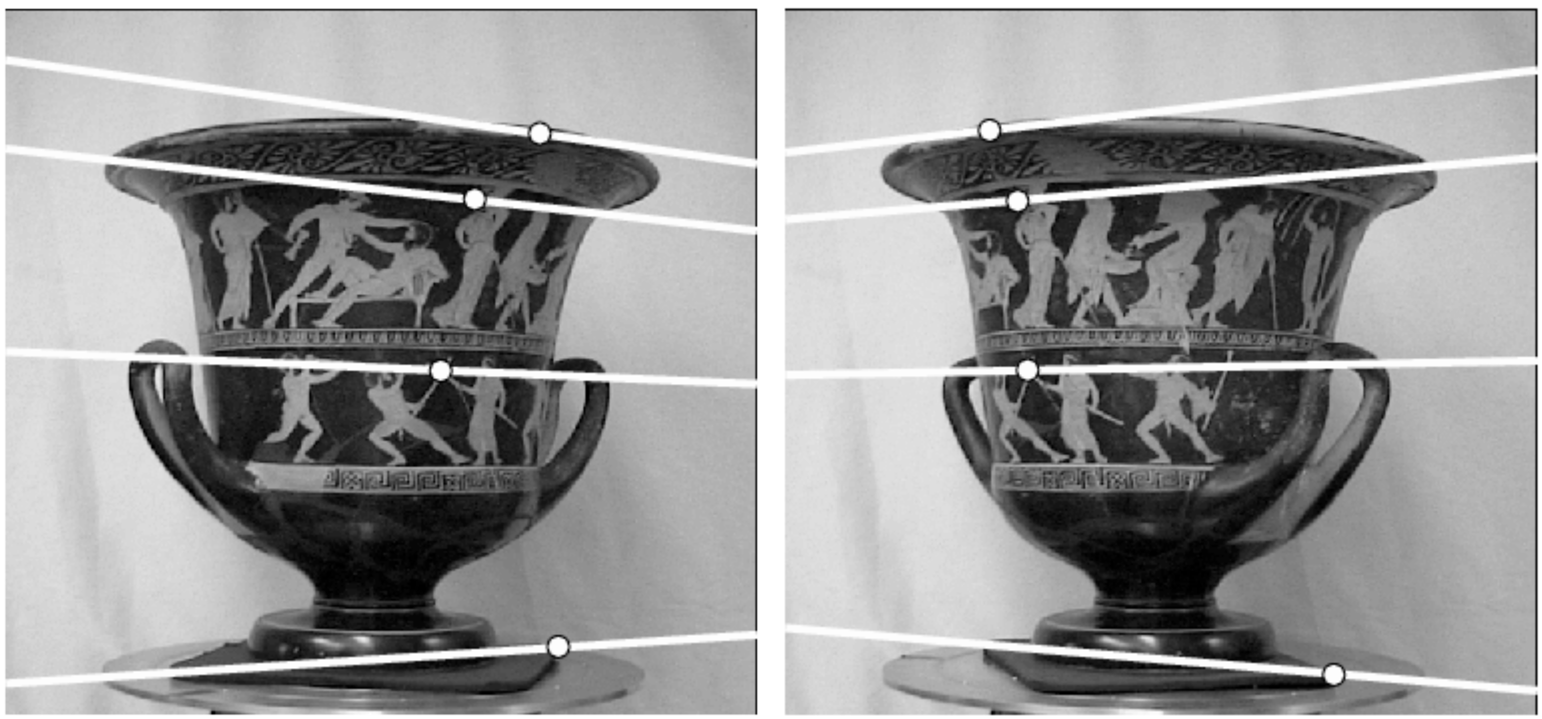


an epipolar plane = plane containing baseline (1-D family)

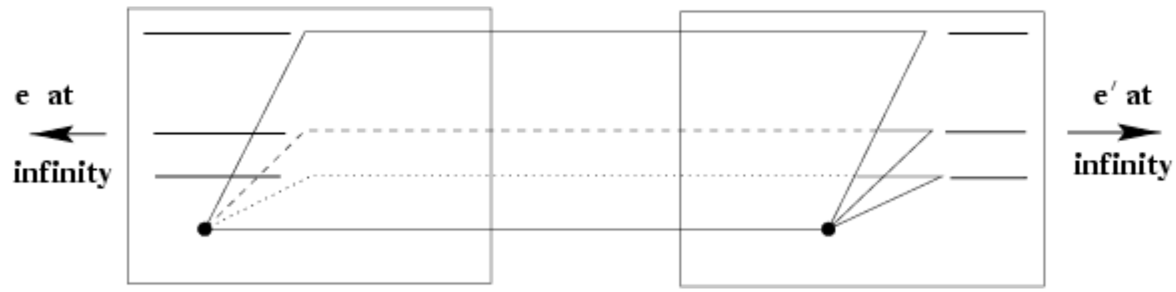
an epipolar line = intersection of epipolar plane with image
(always come in corresponding pairs)

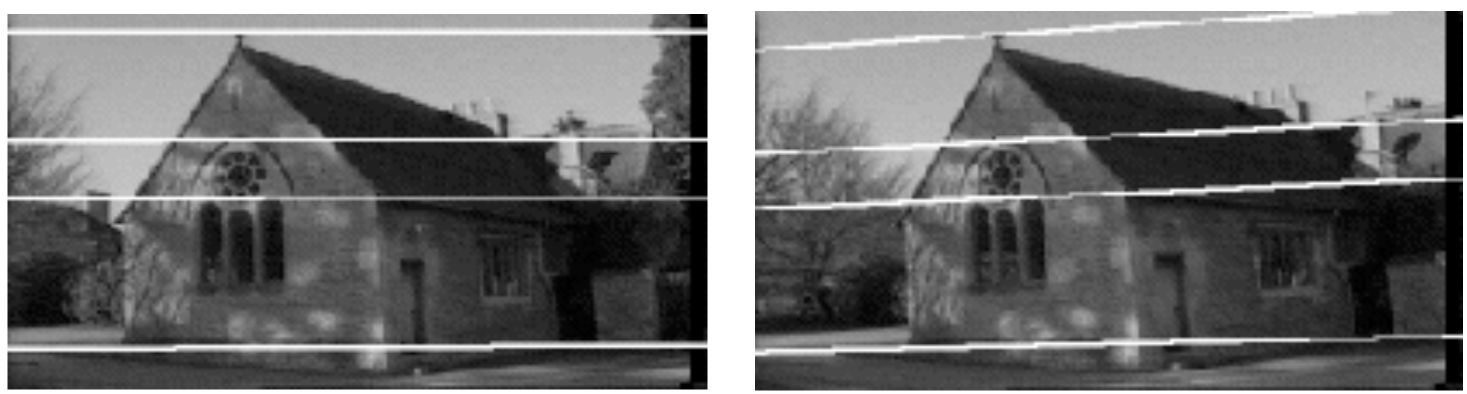
Example: converging cameras



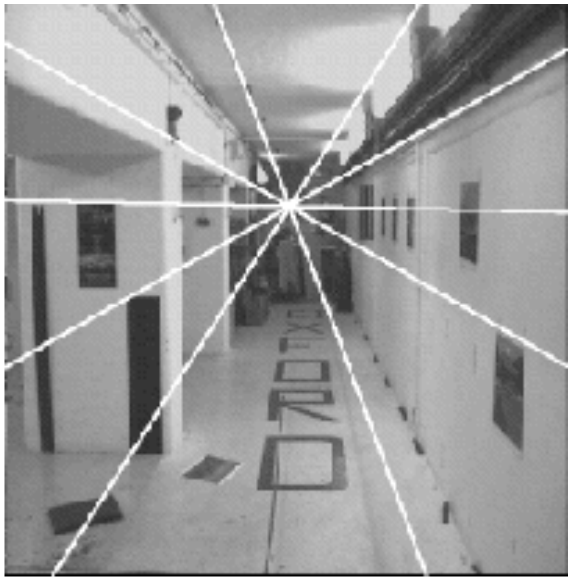


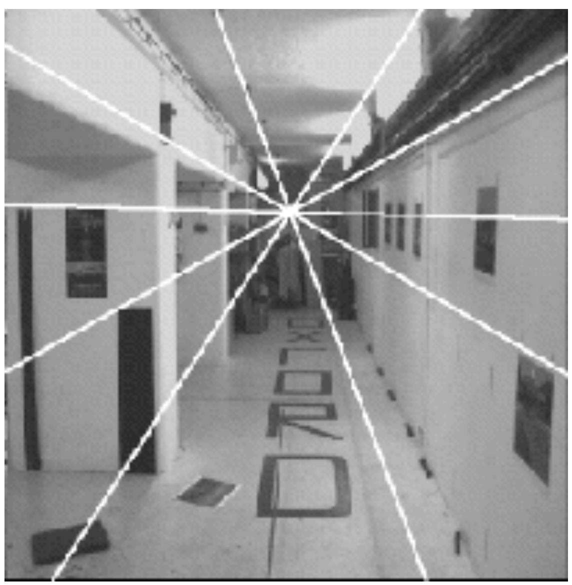
Example: motion parallel with image plane

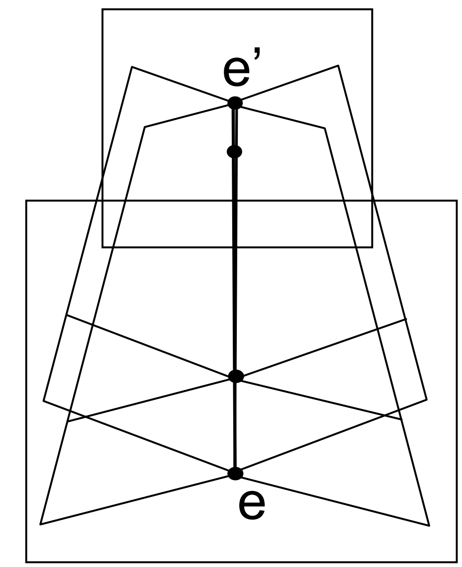




Example: forward motion







What does the Essential matrix do?

It transforms the image point to the normal to the epipolar line *in the other image*

$$n = \mathcal{E}\, x$$

The normal defines a line in image 2:

$$x'_{\text{on epipolar line}} \Rightarrow n \cdot x' = 0$$

$$n_1 x_1 + n_2 x_2 + n_3 1 = 0$$

$$(y = mx + b) \Rightarrow b = -n_3, \quad m = -\frac{n_1}{n_2}$$

What if cameras are uncalibrated? Fundamental Matrix

Choose world coordinates as Camera 1.
Then the extrinsic parameters for camera 2 are just **R** and **t**
However, intrinsic parameters for both cameras are unknown.
Let C_1 and C_2 denote the matrices of intrinsic parameters. Then the pixel coordinates measured are not appropriate for the Essential matrix.
Correcting for this distortion creates a new matrix: the Fundamental Matrix.

$$x'_{measured} = C_2 x' \qquad\qquad x_{measured} = C_1 x$$

$$(x')^t \mathcal{E}\, x = 0 \Rightarrow \left(C_2^{-1} x'_{measured}\right)^t \mathcal{E} \left(C_1^{-1} x_{measured}\right) = 0$$

$$\left(x'_{measured}\right)^t \mathcal{F}\, x_{measured} = 0$$

$$\mathcal{F} = C_2^{-t} \mathcal{E}\, C_1^{-1}$$

Example

$$C = \begin{bmatrix} -f \cdot s_u & 0 & u_0 \\ 0 & -f \cdot s_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Computing the fundamental Matrix

Computing : I Number of Correspondences Given perfect image points (no noise) in general position. Each point correspondence generates one constraint on the fundamental matrix

Constraint for one point

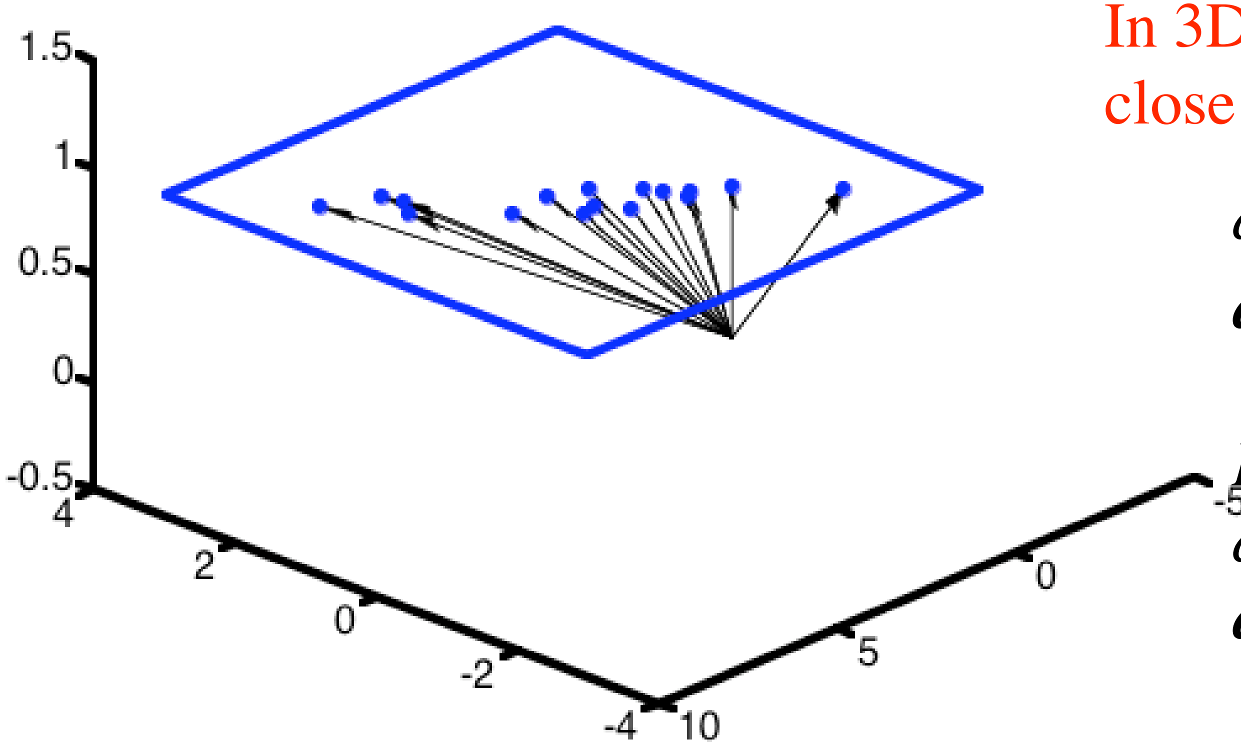
$$\begin{bmatrix} x'_i & y'_i & 1 \end{bmatrix} \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0$$

Each constraint can be rewritten as a dot product.
Stacking several of these results in:

$$\begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{bmatrix} = 0$$

Geometry of Homogenous Coordinates

Data as 3D homogenous vectors $p_i = [x_i \;\; y_i \;\; 1]'$



In 3D, the set of points lies close to a common plane

$a\, x_i + b\, y_i = c$

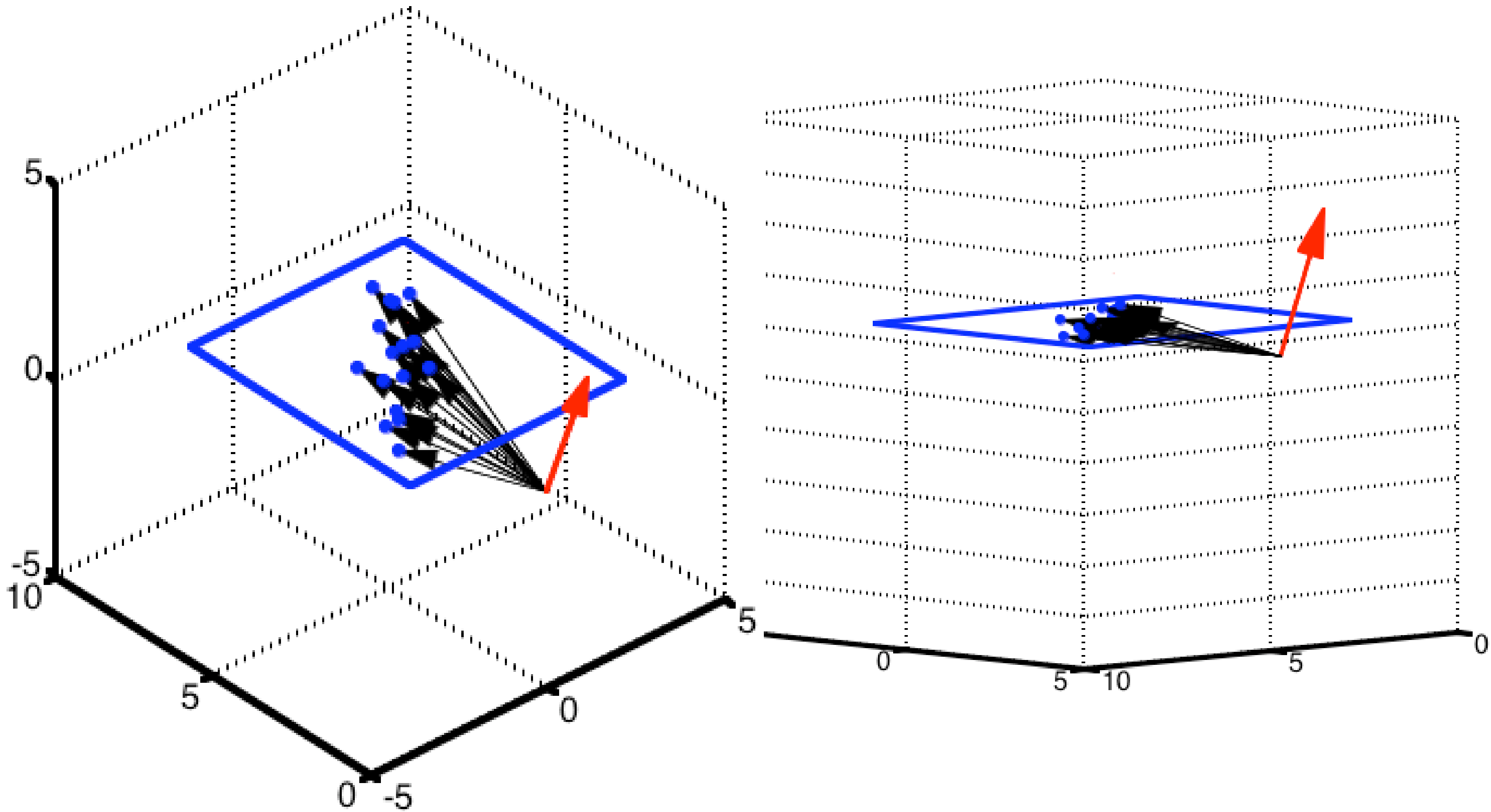
$\boldsymbol{a \cdot x} = \boldsymbol{c} = \|\boldsymbol{x}\| \cos \phi$

Becomes

$a\, x_i + b\, y_i + c\, 1 = 0$

$\boldsymbol{a \cdot p_i = 0}$

Geometry of solution



Enforce both constraints using Lagrangian multipliers

- Proceeding with method of Lagrange multipliers, define V

The function we want to minimise

The constraint, this will be zero so long as $\|\mathbf{x}\| = 1$

$$V = \|\boldsymbol{\varepsilon}\|^2 + \lambda(1 - \|\mathbf{x}\|^2)$$

Lagrange multiplier

$$V = \|\boldsymbol{\varepsilon}\|^2 + \lambda(1 - \|\mathbf{x}\|^2)$$

V can be rewritten as

$$V = \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \lambda(1 - \mathbf{x}^{\mathrm{T}}\mathbf{x})$$

since

- $\|\boldsymbol{\varepsilon}\|^2 = \boldsymbol{\varepsilon}^{\mathrm{T}}\boldsymbol{\varepsilon} = \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}$ and
- $\|\mathbf{x}\|^2 = \mathbf{x}^{\mathrm{T}}\mathbf{x}$

- Find critical points of V, ie. where the derivative $\mathrm{d}V/\mathrm{d}\mathbf{x}$ is zero

$$V = \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} + \lambda(1-\mathbf{x}^T\mathbf{x})$$

$$\mathrm{d}V/\mathrm{d}\mathbf{x} = 2\mathbf{A}^T\mathbf{A}\mathbf{x} - 2\lambda\mathbf{x} = 0$$

$$\Rightarrow \mathbf{A}^T\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$

- This is the eigen equation!
- $\mathbf{x}$ must be an eigen vector of $\mathbf{A}^T\mathbf{A}$
  (so that $\mathrm{d}V/\mathrm{d}\mathbf{x} = 0$) this is a necessary, but not sufficient condition to minimise C.

- Which eigen vector to choose?

- Choose the eigen vector that minimises C.

- Let's substitute in for $\mathbf{x}$ an arbitrary unit eigen vector $\mathbf{e}_n$.

$$
\begin{aligned}
C &= \|\boldsymbol{\varepsilon}\|^2 \\
&= \mathbf{e}_n{}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{e}_n \\
&= \mathbf{e}_n{}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{e}_n) \\
&= \mathbf{e}_n{}^{\mathrm{T}}\mathbf{e}_n\lambda \\
&= \lambda_n
\end{aligned}
$$

This is minimised by choosing

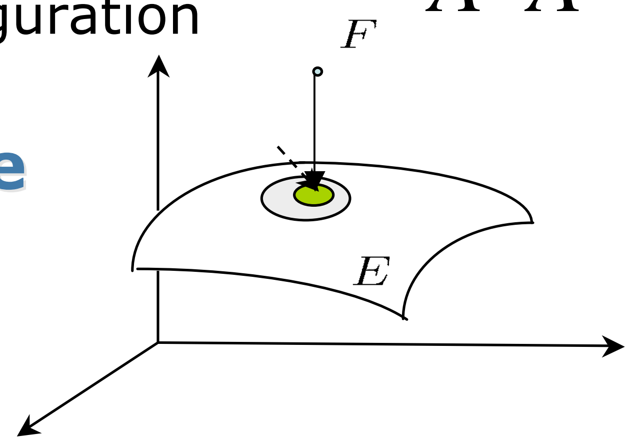
$$\mathbf{x} = \mathbf{e}_{\mathrm{n}}$$

where $\mathbf{e}_n$ is the eigen vector associated with the smallest eigen value λ_n.

Estimating Essential Matrix

Solution Form matrix ***A*** from point matches

- Eigenvector associated with the smallest eigenvalue of $A^T A$
- if $rank(A^T A) < 8$ degenerate configuration



Projection onto Essential Space

(Project to Essential Manifold)

If the SVD of a matrix $F \in \mathcal{R}^{3\times3}$ is given by $F = Udiag(\sigma_1, \sigma_2, \sigma_3)V^T$ then the essential matrix E which minimizes the Frobenius distance $\|E - F\|_f^2$ is given by $E = Udiag(\sigma, \sigma, 0)V^T$ with $\sigma = \frac{\sigma_1 + \sigma_2}{2}$

Stereo Reconstruction

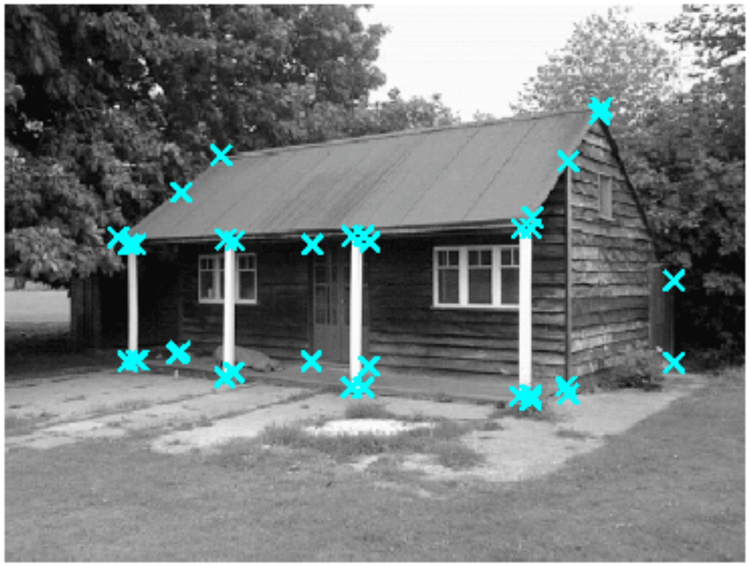
If we know the

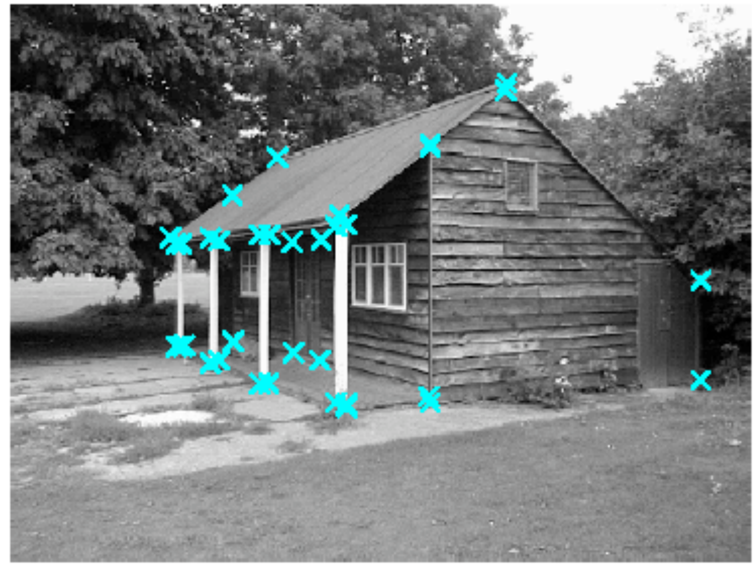
- fundamental matrix, and
- internal camera parameters

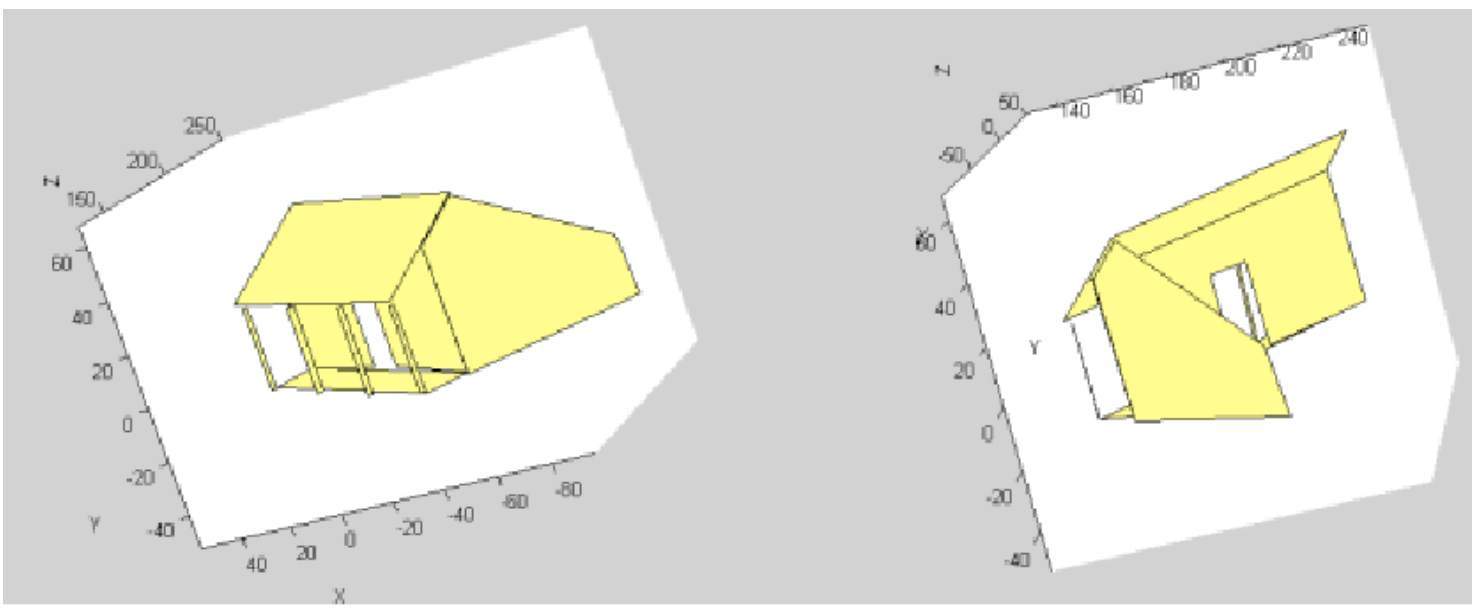
We can solve for the

- external camera parameters, and
- determine 3D structure of a scene

> 8 Point matches







- This is called *calibrated reconstruction*, since it requires the camera(s) be $\underbrace{\textit{internally calibrated}}_{\text{camera calibration matrix } C \text{ must be known.}}$.

- For example:
  - Reconstructing the 3D structure of a scene from multiple views.
  - Determining structure from motion.

Reconstruction Steps

1. Identify a number of (at least 8) point correspondances.
2. Estimate the fundamental matrix using the normalised 8-point algorithm.
3. Determine the external camera parameters (rotation and translation from one camera to the other)
   a. Calculate the essential matrix from the fundamental matrix and the camera calibration matrices.
   b. Extract the rotation and translation components from the essential matrix.

4. Determine 3D point locations.

Determining Extrinsic Camera Parameters

- Want to determine rotation and translation from one camera to the other.
- We know the camera matrices have the form

Why can we use the identity as the external parameters for camera M1?
Because we are only interested in the *relative* position of the two cameras.

$$M_1 = C_1\begin{bmatrix} I & \vec{0} \end{bmatrix}$$
$$M_2 = C_2\begin{bmatrix} R & \vec{t} \end{bmatrix}$$

- First we undo the Intrinsic camera distortions by defining new *normalized* cameras

$$M_1^{norm} = C_1^{-1} M_1 \quad \text{and} \quad M_2^{norm} = C_2^{-1} M_2$$

Determining Extrinsic Camera Parameters

- The *normalized* cameras contain unknown parameters

$$M_1^{norm} = C_1^{-1} M_1 \quad \Rightarrow \quad M_1^{norm} = \begin{bmatrix} I & \vec{0} \end{bmatrix}$$

$$M_2^{norm} = C_2^{-1} M_2 \quad \Rightarrow \quad M_2^{norm} = \begin{bmatrix} R & \vec{t} \end{bmatrix}$$

- However, those parameters can be extracted from the Fundamental matrix

$$F = C_2^{-t} E C_1^{-1}$$

$$E = C_2^{t} F C_1$$

$$E = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} R = \vec{t}_{\times} R$$

Extract t and R from the Essential Matrix

How do we recover t and R? Answer: SVD of $\mathcal{E}$

$$\mathcal{E} = USV^t$$

S diagonal

U,V orthogonal and $\det() = 1$ (rotation)

$$R = UWV^t \quad \text{or} \quad R = UW^tV^t \qquad \vec{t} = u_3 \quad \text{or} \quad \vec{t} = -u_3$$

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

SVD

- For an $n \times m$ matrix there exist unitary* matrices $\mathbf{U}$ and $\mathbf{V}$ such that

$$\mathbf{U} = [\mathbf{u}_1 | \mathbf{u}_2 | \ldots | \mathbf{u}_m] \longleftarrow m \times m \text{ matrix}$$

$$\mathbf{V} = [\mathbf{v}_1 | \mathbf{v}_2 | \ldots | \mathbf{v}_n] \longleftarrow n \times n \text{ matrix}$$

$$\mathbf{A} = \mathbf{USV}^{T}, \text{ where } \quad \mathbf{S} = \begin{bmatrix} \mathbf{S}_1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{S}_1 = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_p \end{bmatrix}$$

and $s_1 \geq s_2 \geq \ldots \geq s_p \geq 0$, $p = \min\{n,m\}$

*a real matrix $\mathbf{U}$ is unitary if $\mathbf{U}^{-1} = \mathbf{U}^{T}$

SVD

- s_i is the i^{th} singular value of $\mathbf{A}$, the vectors $\mathbf{u}_i$ and $\mathbf{v}_i$ are the left and right singular vectors of $\mathbf{A}$.

- s_i^2 is an eigenvalue of $\mathbf{A}\mathbf{A}^{\mathrm{T}}$ or $\mathbf{A}^{\mathrm{T}}\mathbf{A}$,
- $\mathbf{u}_i$ is an eigen vector of $\mathbf{A}\mathbf{A}^{\mathrm{T}}$ and
- $\mathbf{v}_i$ is an eigen vector of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$.

SVD

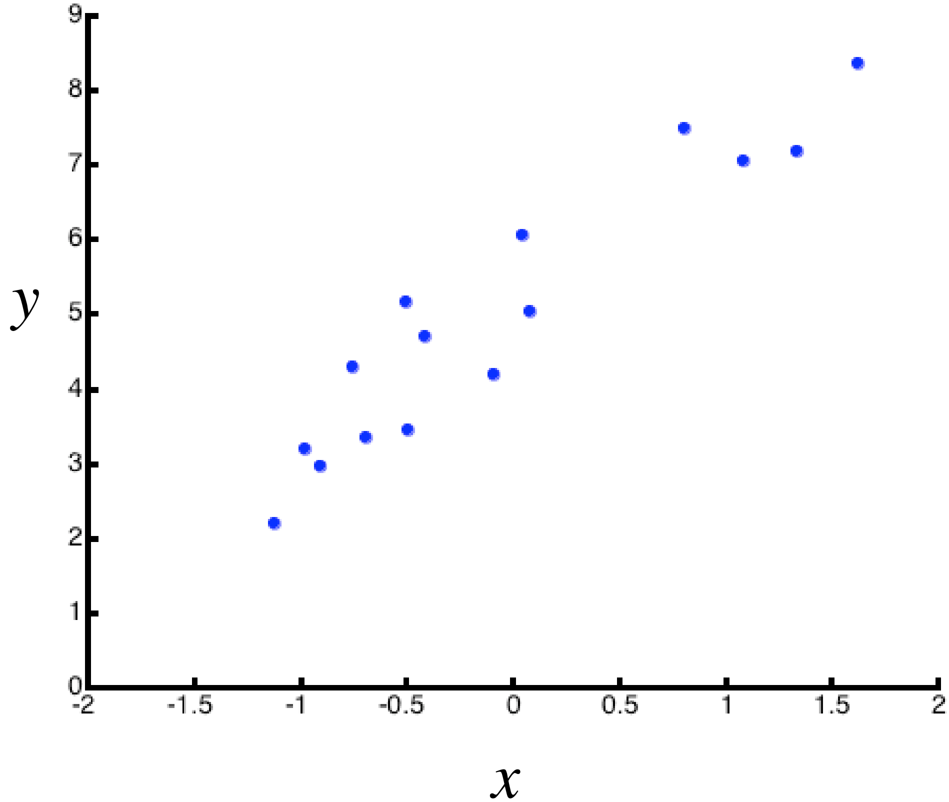
- $\mathbf{V} = [\mathbf{v}_1|\mathbf{v}_2|\ldots|\mathbf{v}_n]$
- is a matrix of eigen vectors of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ with associated eigen values s_i^2. The eigen vector corresponding to the smallest eigen value of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is $\mathbf{v}_n$.
- Hence the non-zero $\mathbf{x}$ that minimises

$$\mathbf{Ax} = \mathbf{0}$$

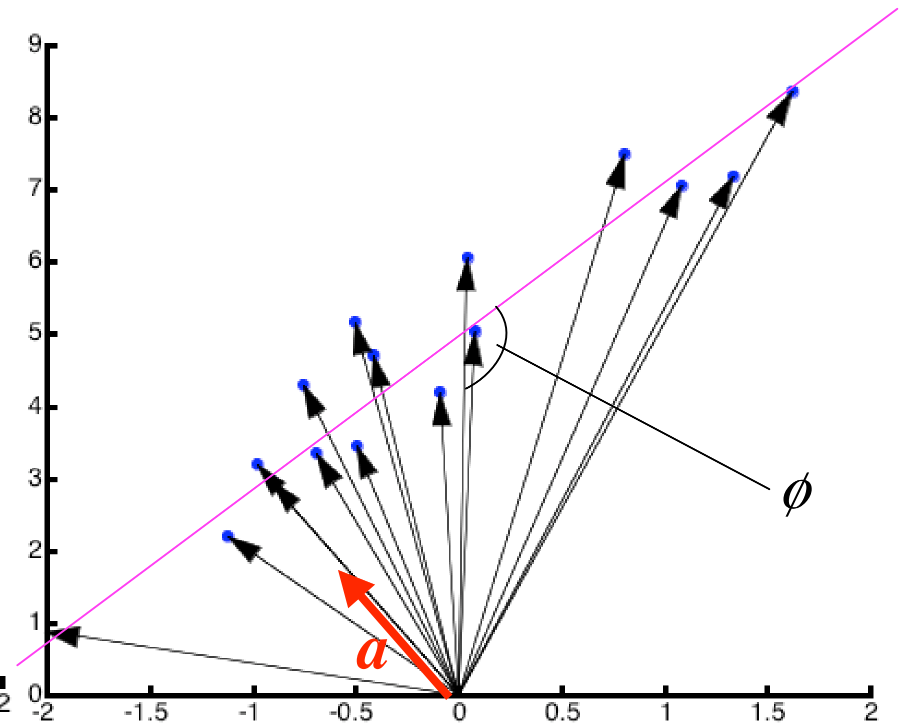
is $\mathbf{x} = \mathbf{v}_n$.

Example: Least Square Line Fitting

Data scatter



Data as 2D vectors



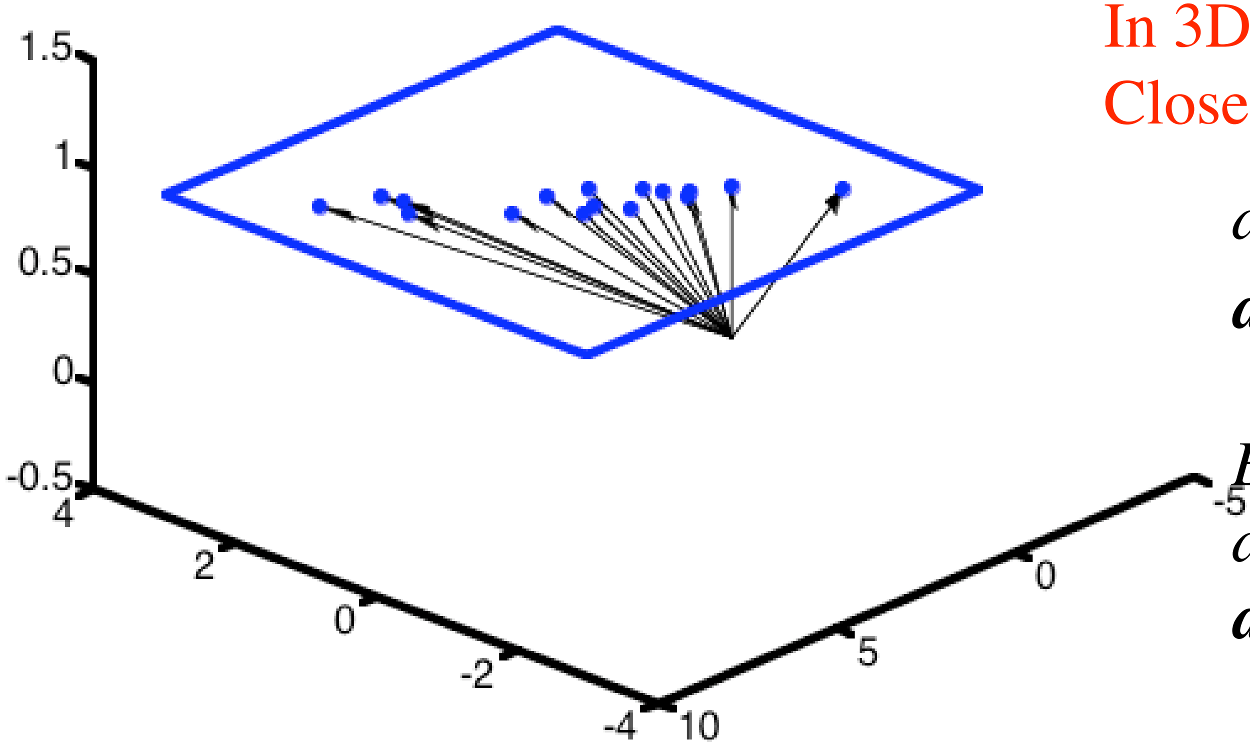
$$a\, x_i + b\, y_i = c$$

$$\boldsymbol{a} \cdot \boldsymbol{x} = \|\boldsymbol{a}\|\ \|\boldsymbol{x}\| \cos \phi = \boldsymbol{c}$$

$$= \|\boldsymbol{x}\| \cos \phi = \boldsymbol{c}$$

Introducing Homogenous Coordinates

Data as 3D homogenous vectors $p_i = [x_i \;\; y_i \;\; 1]'$



In 3D, the set of points lies
Close to a common plane

$a\,x_i + b\,y_i = c$

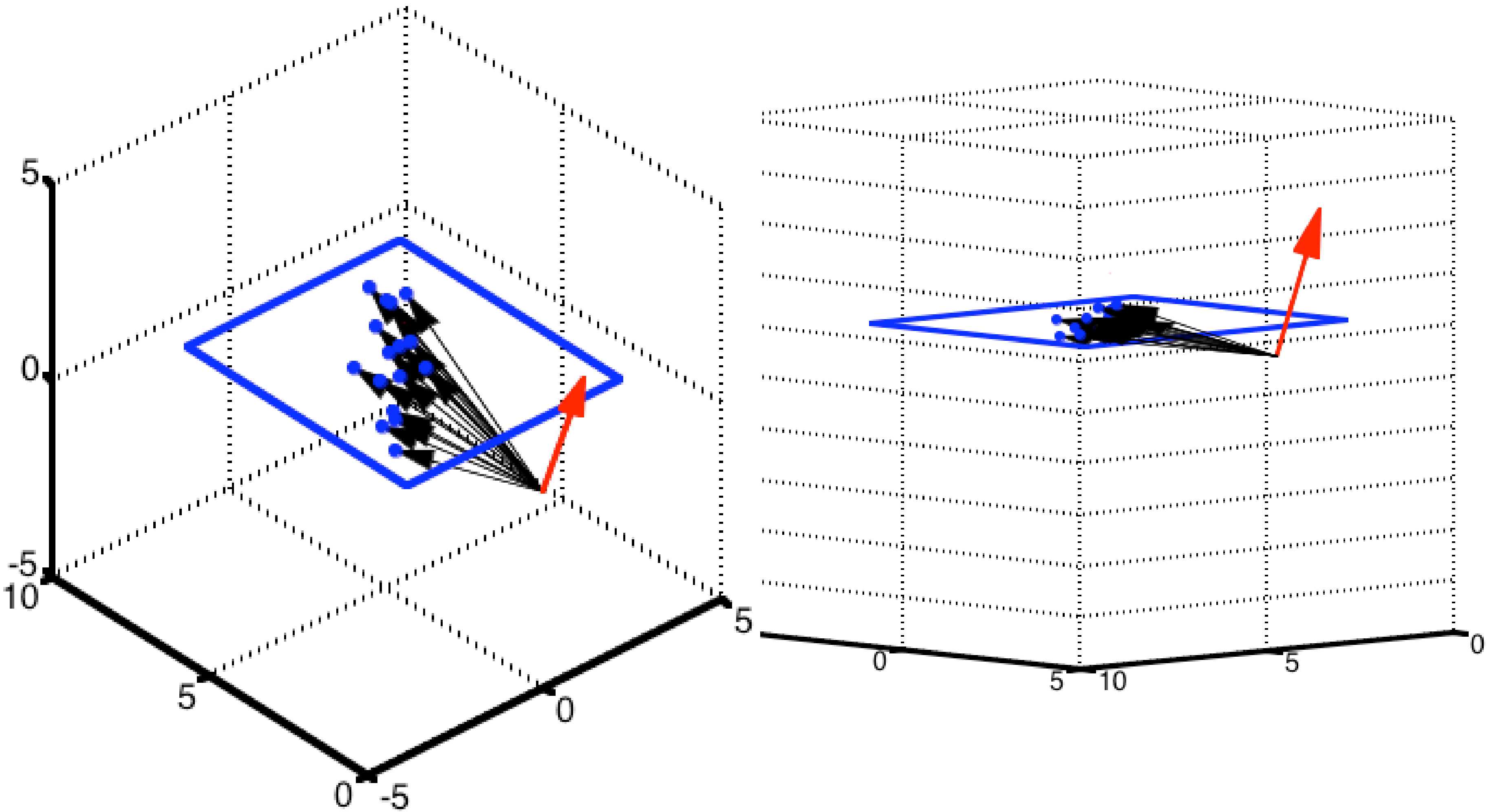
$\boldsymbol{a \cdot x} = \boldsymbol{c} = \|\boldsymbol{x}\| \cos \phi$

Becomes

$a\,x_i + b\,y_i + c\,1 = 0$

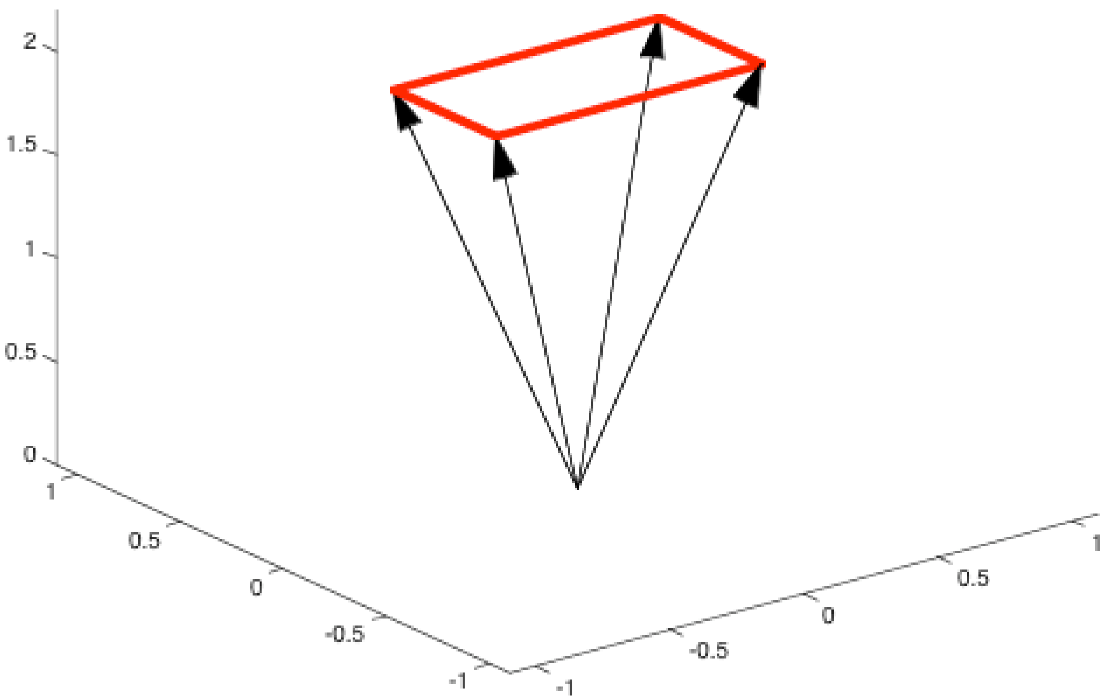
$\boldsymbol{a \cdot p_i = 0}$

Geometry of solution



$$A = \begin{pmatrix} -0.5 & -0.5 & 0.5 & 0.5 \\ 0.25 & -0.25 & -0.25 & 0.25 \\ 2 & 2 & 2 & 2 \end{pmatrix}$$

$[U,S,V] = \mathrm{svd}(A)$



$$U = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$S = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \end{pmatrix}$$

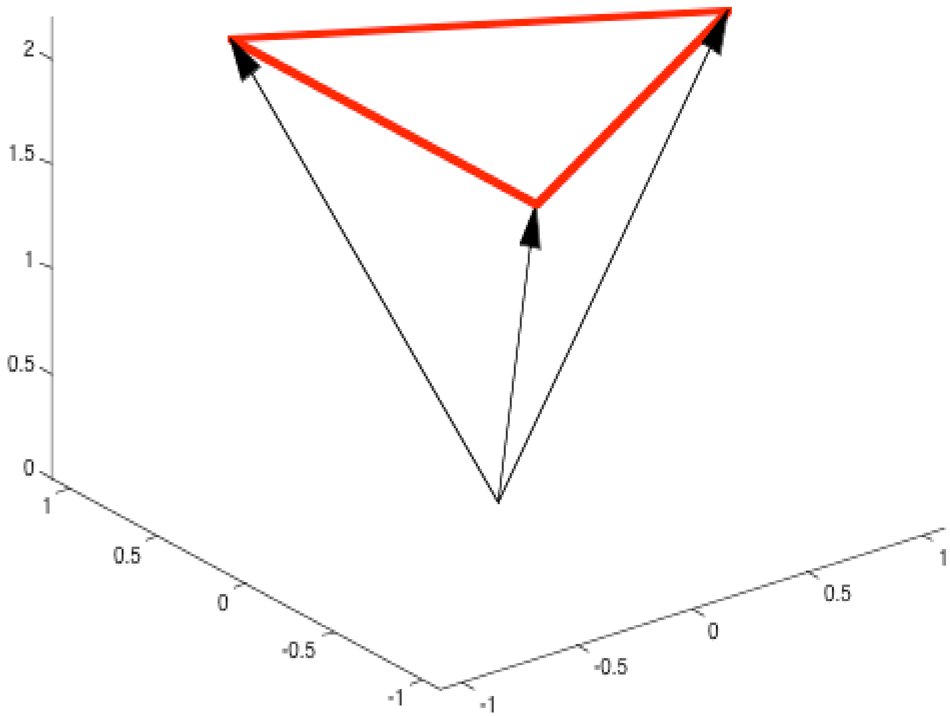
B =

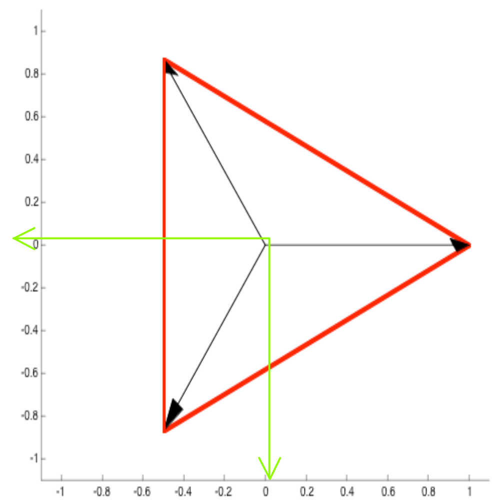
1	-0.5	-0.5
0	0.86603	-0.86603
2	2	2

[U,S,V] = svd(B)

U =

$$\begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{pmatrix}$$





[V,D]=eig(B)
Yields z-axis and
Complex eigenvalues
Representing the ambiguity

Extract t and R from the Essential Matrix

How do we recover t and R? Answer: SVD of $\mathcal{E}$

$$\mathcal{E} = USV^t$$

S diagonal

U,V orthogonal and $\det() = 1$ (rotation)

$$R = UWV^t \quad \text{or} \quad R = UW^tV^t \qquad \vec{t} = u_3 \quad \text{or} \quad \vec{t} = -u_3$$

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

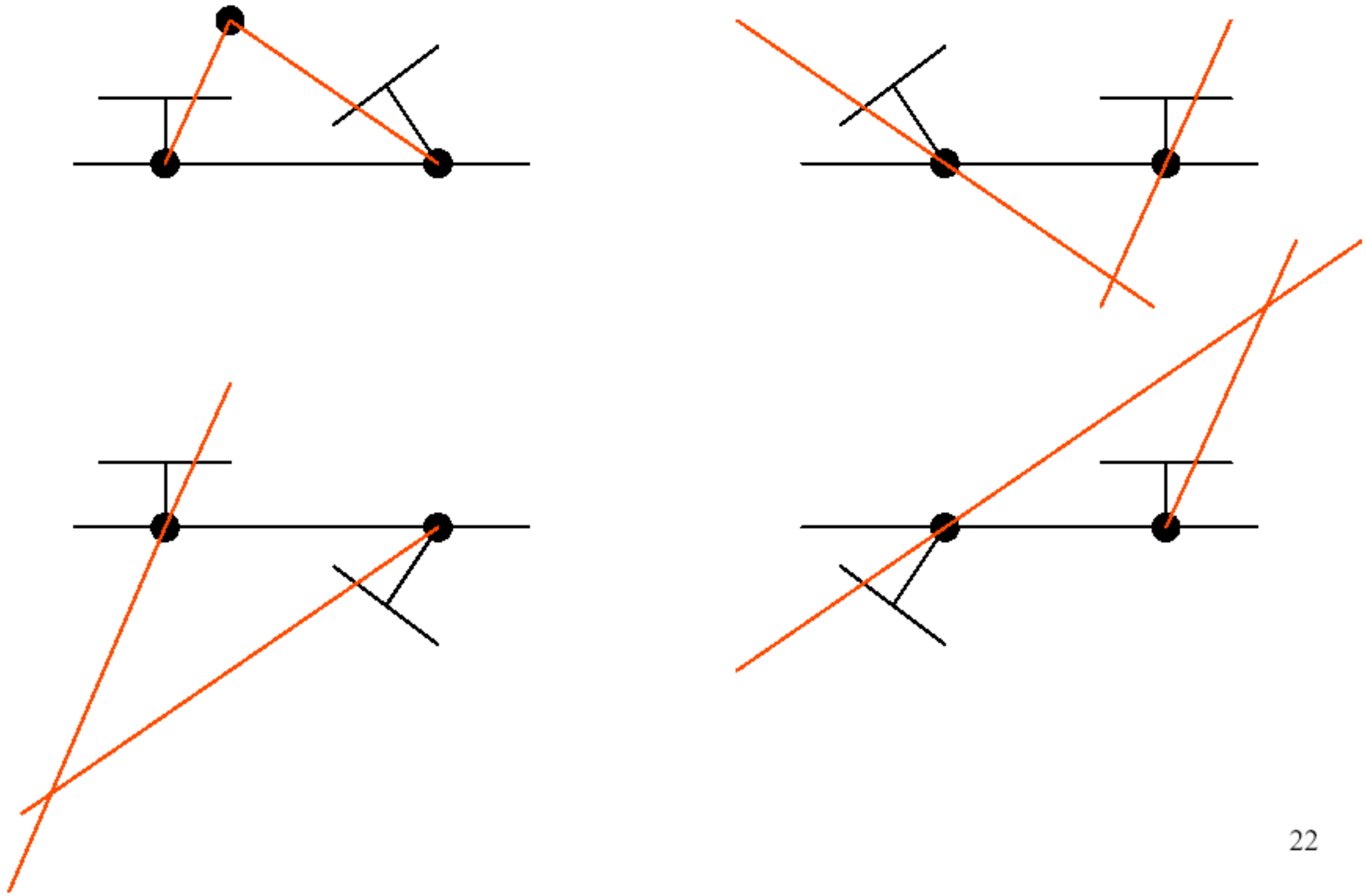
Reconstruction Ambiguity

So we have 4 possible combinations of translations and rotations giving 4 possibilities for $\boldsymbol{M}_2^{norm} = [\boldsymbol{R} \mid \boldsymbol{t}]$

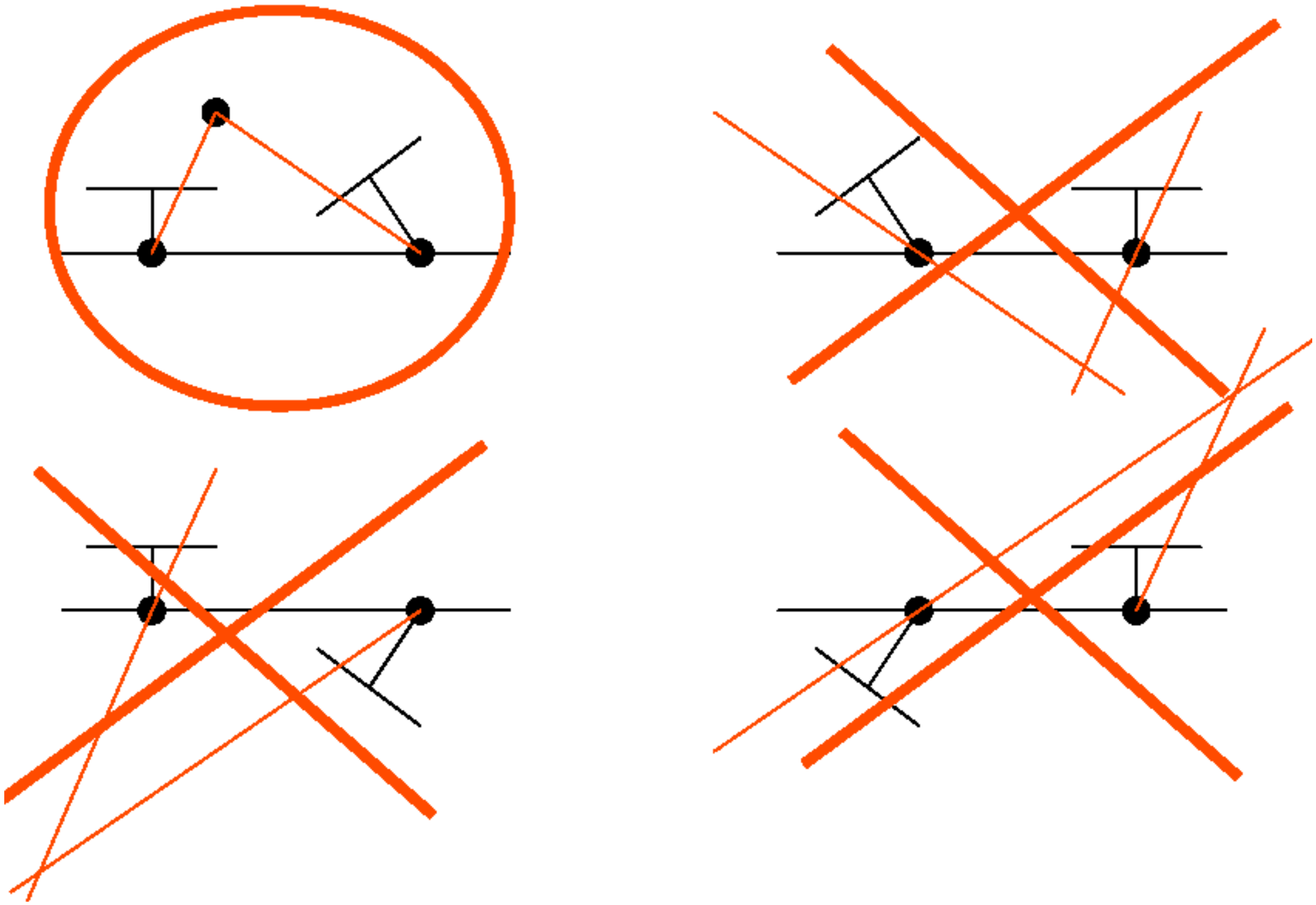
1. $\boldsymbol{M}_2^{norm} = [\boldsymbol{U}\boldsymbol{W}^t\boldsymbol{V}^t \mid \boldsymbol{t}]$
2. $\boldsymbol{M}_2^{norm} = [\boldsymbol{U}\boldsymbol{W}\boldsymbol{V}^t \mid \boldsymbol{t}]$
3. $\boldsymbol{M}_2^{norm} = [\boldsymbol{U}\boldsymbol{W}^t\boldsymbol{V}^t \mid -\boldsymbol{t}]$
4. $\boldsymbol{M}_2^{norm} = [\boldsymbol{U}\boldsymbol{W}\boldsymbol{V}^t \mid -\boldsymbol{t}]$

Which one is right?

- We can determine which of these is correct by looking at their geometric interpretation.



Both Cameras must be facing the same direction



Which one is right?

- The correct pair will have our data points in front of both cameras.
- How do we choose the correct pair?
- Procedure:
  - Take a test point from data
  - Backproject to find 3D location
  - Determine the depth of 3D point in both cameras
  - Choose the camera pair that has a positive depth for both cameras.

How do we backproject?

$$x'_{measured} = C_2 x' \qquad\qquad x_{measured} = C_1 x$$

Knowing C_i allows us to determine the undistorted image points:

$$x' = C_2^{-1} x'_{measured} \qquad\qquad x = C_1^{-1} x_{measured}$$

Recalling the projection equations allows to relate the world point and the image points.

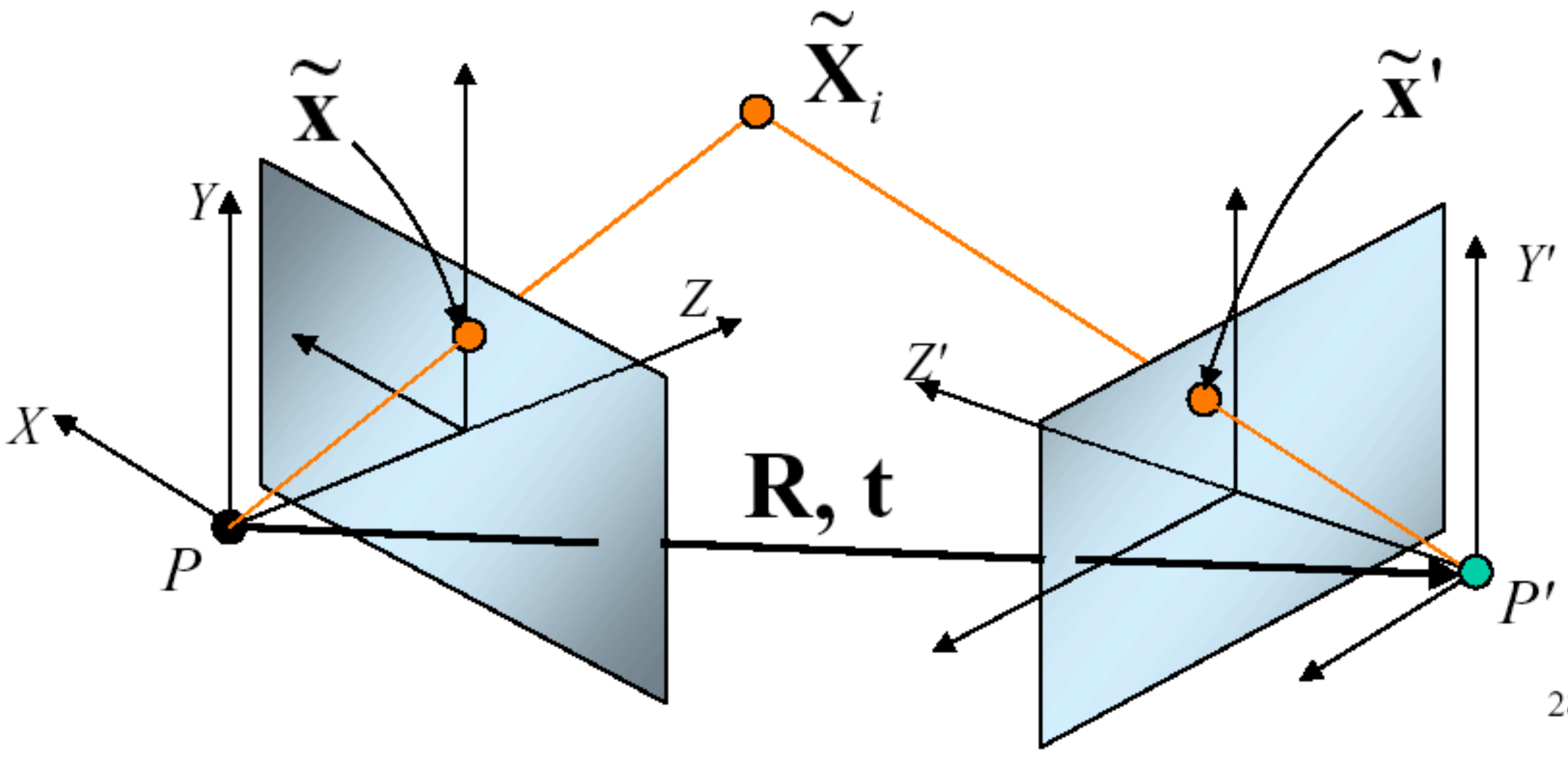
$$x' = C_2^{-1} x'_{measured} \qquad\qquad x = C_1^{-1} x_{measured}$$

$$z' x' = C_2^{-1} C_2 M_2^{norm} X \qquad\qquad zx = C_1^{-1} C_1 \left[\mathbf{I} \mid 0 \right] X$$

$$z' x' = M_2^{norm} X \qquad\qquad zx = \left[\mathbf{I} \mid 0 \right] X$$

Backprojection to 3D

We now know *x, x'*, *R*, and *t*
Need *X*



Solving…

$$z x_i = M^{norm} X_i$$

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} m_1^t \\ m_2^t \\ m_3^t \end{bmatrix} X_i \Rightarrow \begin{cases} z u_i = m_1^t \cdot X_i \\ z v_i = m_2^t \cdot X_i \\ z = m_3^t \cdot X_i \end{cases} \Rightarrow \begin{cases} (m_3^t \cdot X_i) u_i = m_1^t \cdot X_i \\ (m_3^t \cdot X_i) v_i = m_2^t \cdot X_i \end{cases}$$

$$\Rightarrow \begin{cases} (m_3^t \cdot X_i) u_i - m_1^t \cdot X_i = 0 \\ (m_3^t \cdot X_i) v_i - m_2^t \cdot X_i = 0 \end{cases}$$

$$\Rightarrow \begin{bmatrix} u_i (m_3^t) - m_1^t \\ v_i (m_3^t) - m_2^t \end{bmatrix} X_i = 0$$

Solving…

- Similarly for the other camera

$$\begin{bmatrix} u'_i({}^2m_3^t) - {}^2m_1^t \\ v'_i({}^2m_3^t) - {}^2m_2^t \end{bmatrix} X_i = 0$$

Where ${}^2m^t_i$ denotes the ith row of the second camera's normalized projection matrix.

Combining 1 & 2:

$$\begin{bmatrix} u_i(m_3^t) - m_1^t \\ v_i(m_3^t) - m_2^t \\ u'_i({}^2m_3^t) - {}^2m_1^t \\ v'_i({}^2m_3^t) - {}^2m_2^t \end{bmatrix} X_i = 0$$

$$AX_i = 0$$

It has a solvable form! Solve using minimum eigenvalue-Eigenvector approach (e.g. Xi = Null(A))

Finishing up

- Now we have the 3D point $\tilde{\mathbf{X}}_i$
- Determine the location of this point for all 4 possible camera configurations
- Next determine the depths of these points in in each camera.

A little more Linear Algebra

- Given 2 simultaneous equations we can write them in matrix form

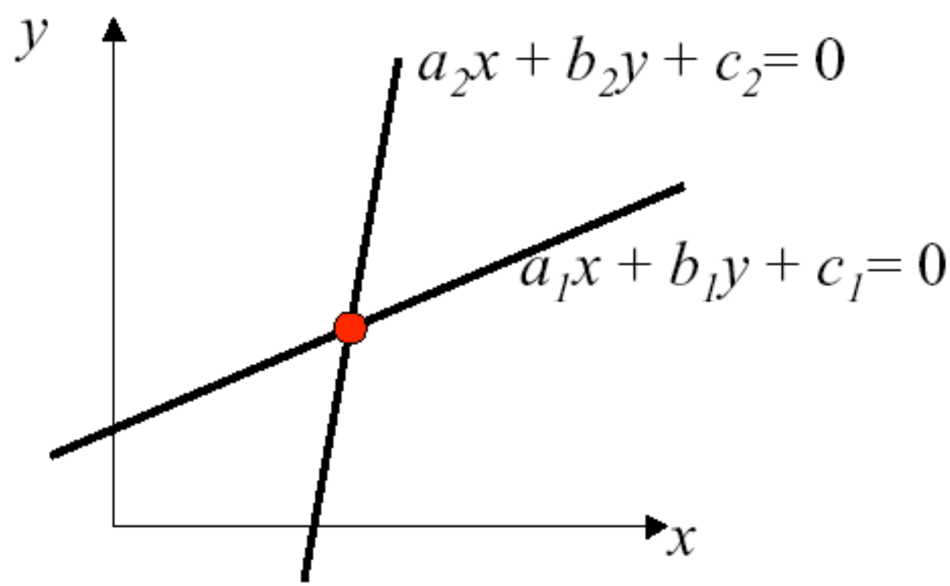
$$\left.\begin{aligned} a_1 x + b_1 y + c_1 = 0 \\ a_2 x + b_2 y + c_2 = 0 \end{aligned}\right\} \quad \Leftrightarrow \quad \mathbf{A}\mathbf{x} + \mathbf{c} = \mathbf{0}$$

where
$$\mathbf{A} = \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix}, \; \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}, \; \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}, \; \mathbf{0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

and the solution is $\mathbf{x} = -\mathbf{A}^{-1}\mathbf{c}$

Interpretation

- We are asking for a single (x,y) point that satisfies both line equations.
- Graphically this amounts to finding the point that lies on both lines.



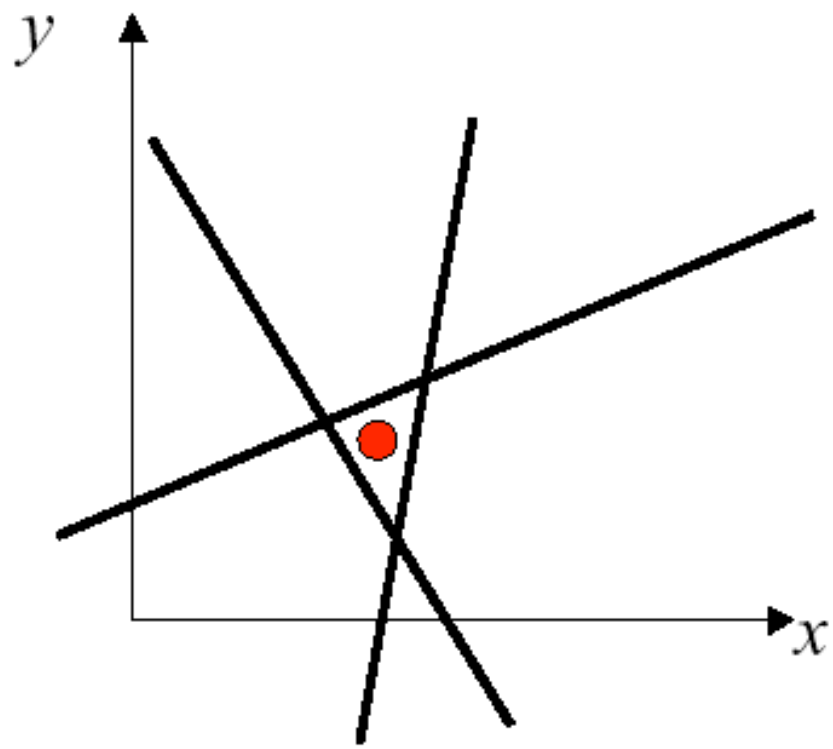
- Given n simultaneous equations in 2D

$$\left.\begin{aligned} a_1 x + b_1 y + c_1 &= 0 \\ a_2 x + b_2 y + c_2 &= 0 \\ &\cdots \\ a_n x + b_n y + c_n &= 0 \end{aligned}\right\} \Leftrightarrow \quad \mathbf{Ax} + \mathbf{c} = \mathbf{0}$$

$$\mathbf{A} = \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \\ \vdots & \vdots \\ a_n & b_n \end{bmatrix}, \ \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}, \ \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \ \mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- For $n>2$ this is an over-constrained system. $\mathbf{A}^{-1}$ does not exist.
- There need not be an exact solution.
- We want to find the 'best' solution.

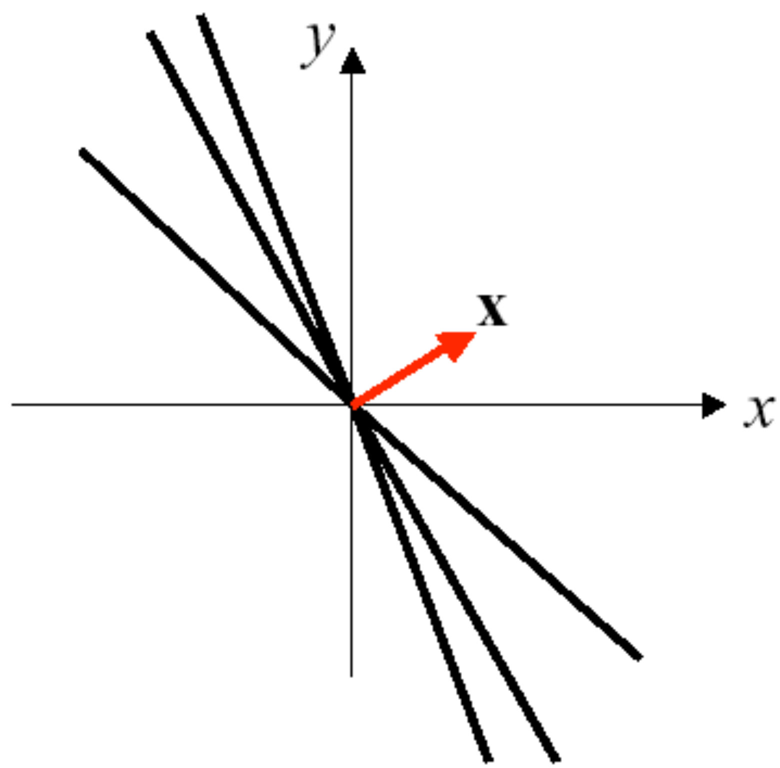
- Graphically we want to find the point that is closest to all n lines at once.



- Note that “closeness” means Euclidean distance for (unweighted) least squares solutions.

What about when c = 0? *Case*: Ax = 0, x≠0

- Solve $\mathbf{Ax} = \mathbf{0}$, for non-zero $\mathbf{x}$,
- Find the *direction* of $\mathbf{x}$ that *minimises* $\mathbf{Ax}$.
- In 2D this can be interpreted as finding the $\mathbf{x}$ that is *most perpendicular* to all *n* lines (ie. most perpendicular to the rows of $\mathbf{A}$).



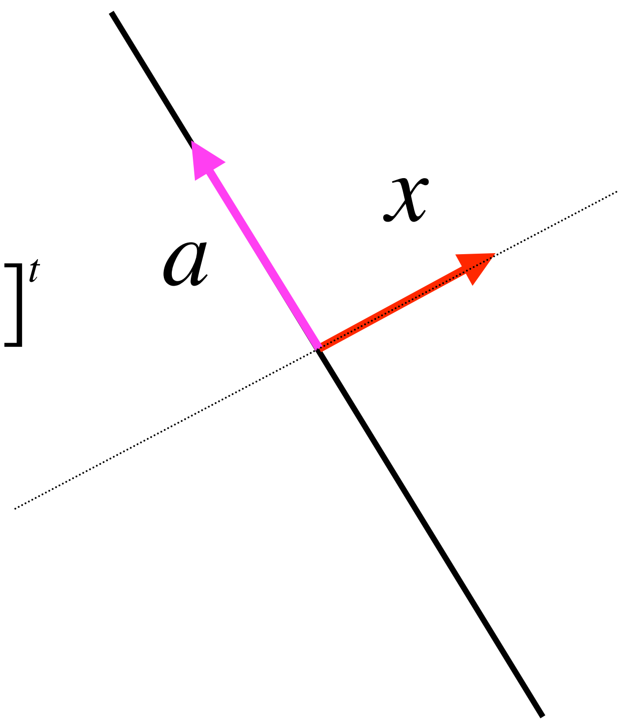
Why does Ax = 0 represent a normal constraint?

The equation of a (2D) line can be written

$$y = mx + b$$
$$\Rightarrow ax + by = -c$$
$$\Rightarrow ax + by + c = 0$$

$$\left[\vec{a}^t \vec{x}\right] = 0, \quad \vec{a} = \left[a \quad b \quad c\right]^t, \quad \vec{x} = \left[x \quad y \quad 1\right]^t$$

For dimension > 2, this is a *hyperplane*

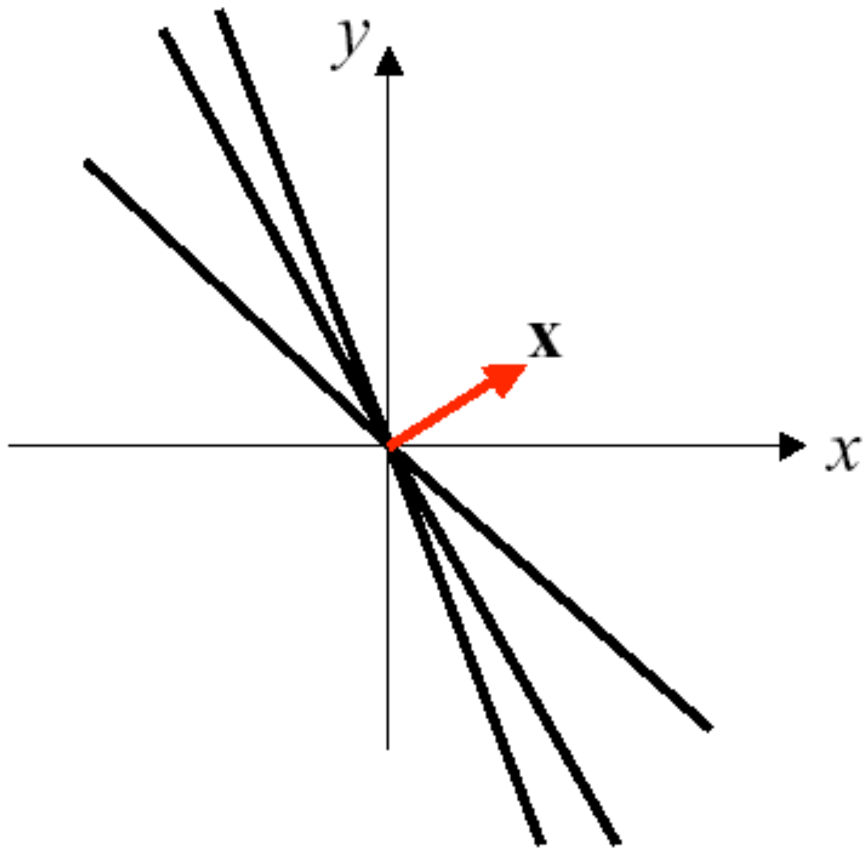


Note x is only defined up to a scale factor, because

$$\vec{a}^t(\lambda\vec{x}) = 0$$
$$\lambda(\vec{a}^t\vec{x}) = 0$$
$$\vec{a}^t\vec{x} = 0$$

Solution

- Choose **x** to be the eigen vector associated with the smallest eigen value of $\mathbf{A}^T\mathbf{A}$. …Why is this?



- $\mathbf{x}$ can only be determined to a scale.
- So, choose $\mathbf{x}$ to be a unit vector,

$$\|\mathbf{x}\| = 1,$$

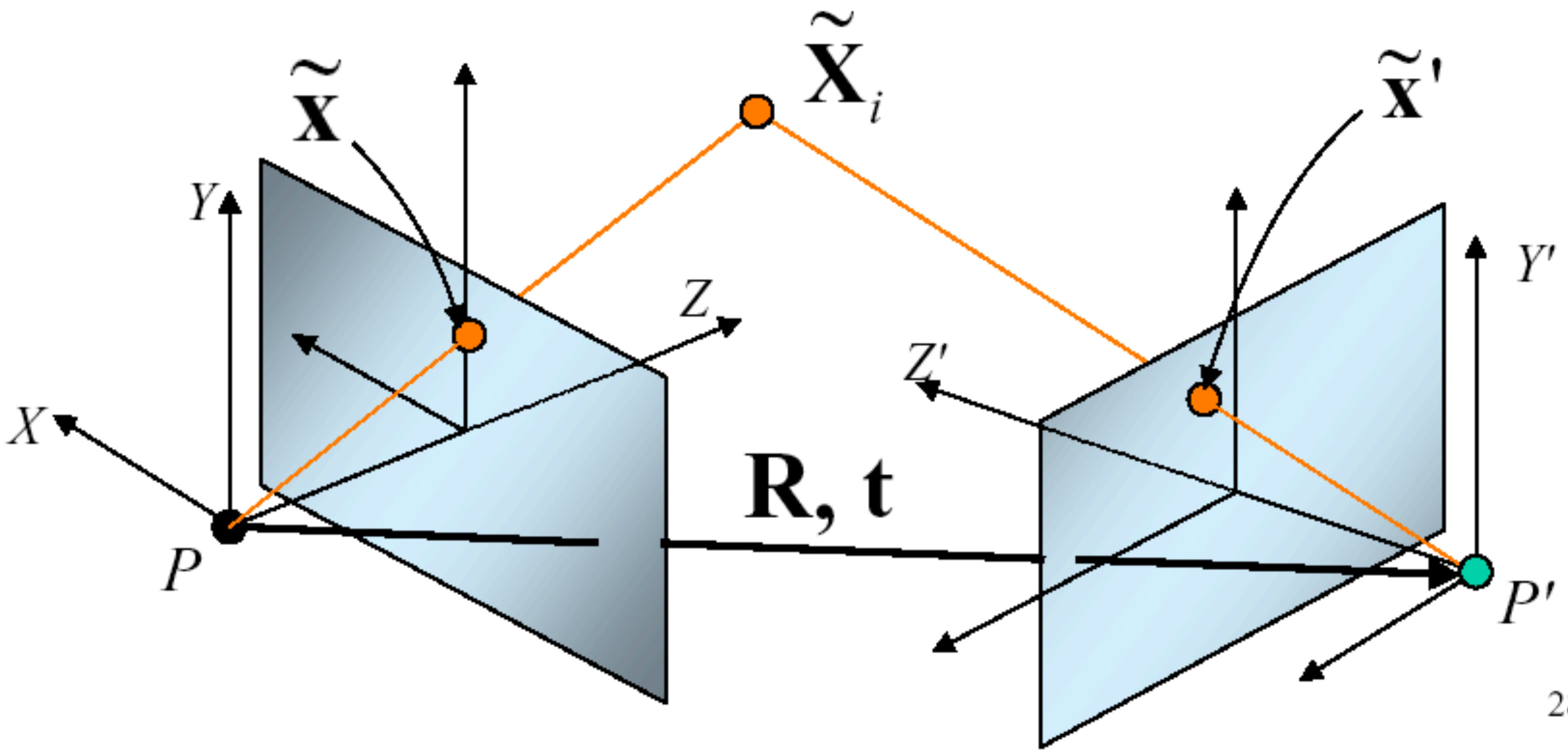
- Define $\boldsymbol{\varepsilon} = \mathbf{A}\mathbf{x}$,

- We want to find an $\mathbf{x}$ so that $\|\boldsymbol{\varepsilon}\|$ is as small as possible, and $\|\mathbf{x}\| = 1$.

- We can achieve this by minimising the positive cost function $C = \|\boldsymbol{\varepsilon}\|^2$ using the method of Lagrange Multipliers.

Backprojection to 3D

We now know *x*, *x'*, *R*, and *t*
Need *X*

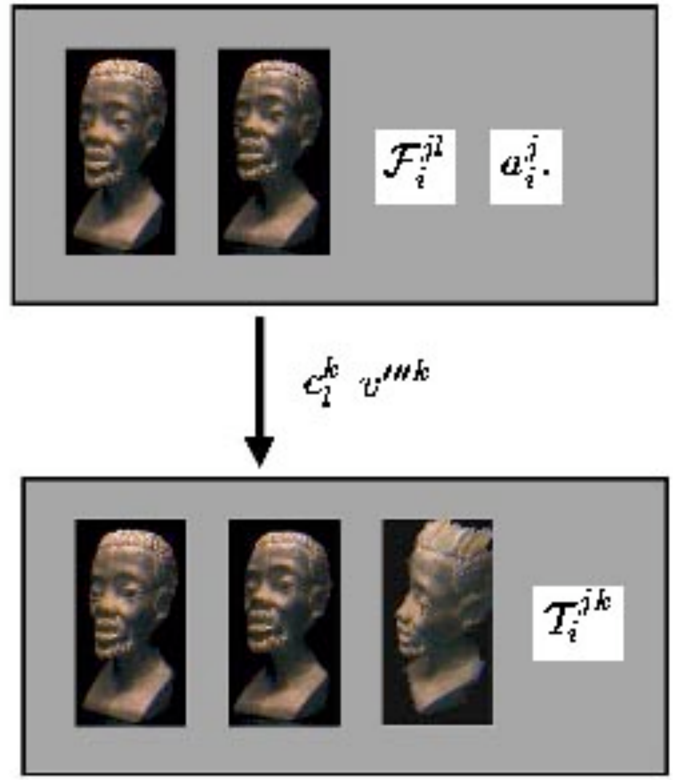


What else can you do with these methods? Synthesize new views

Image 1

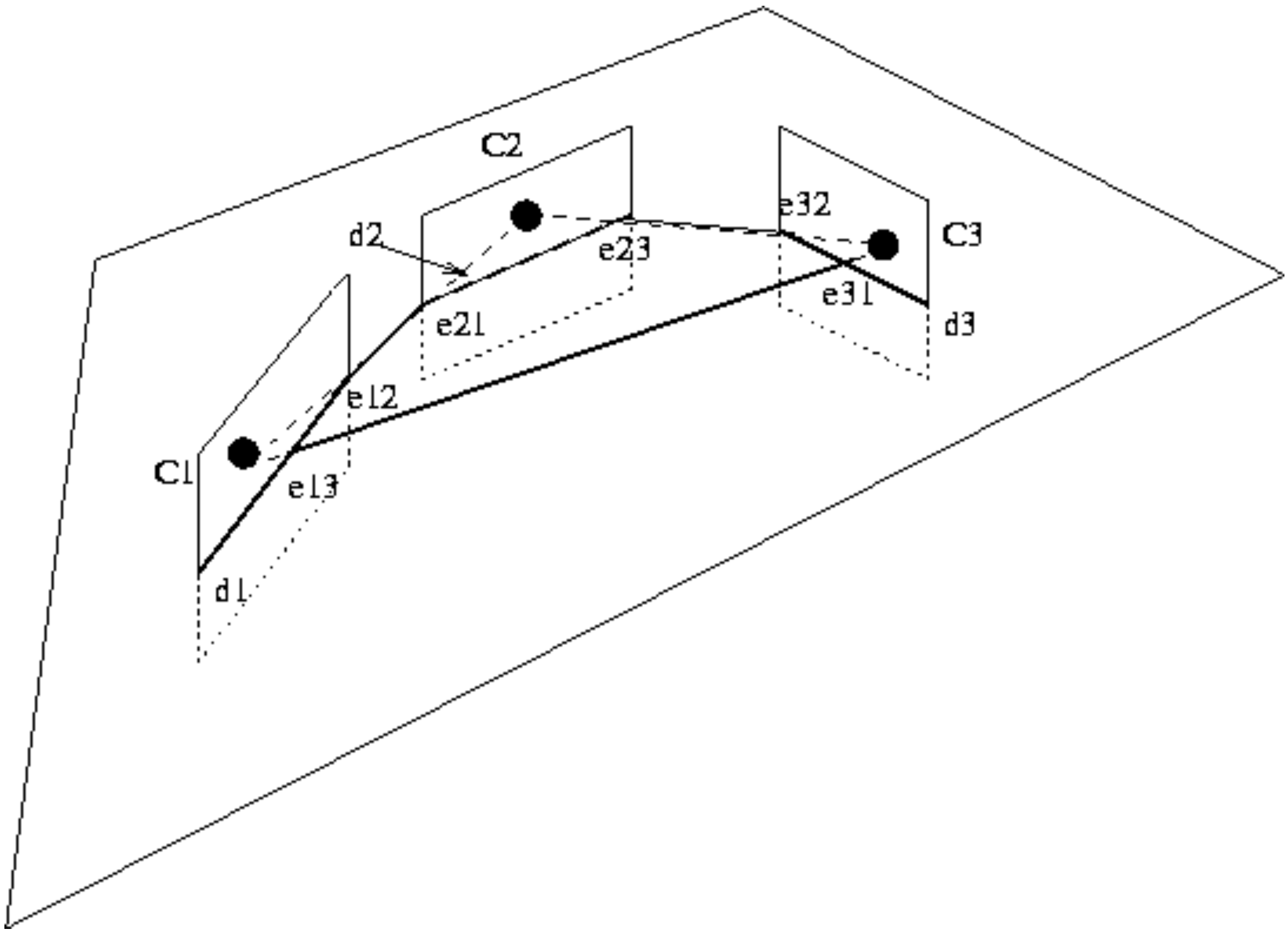
Image 2

60 deg!



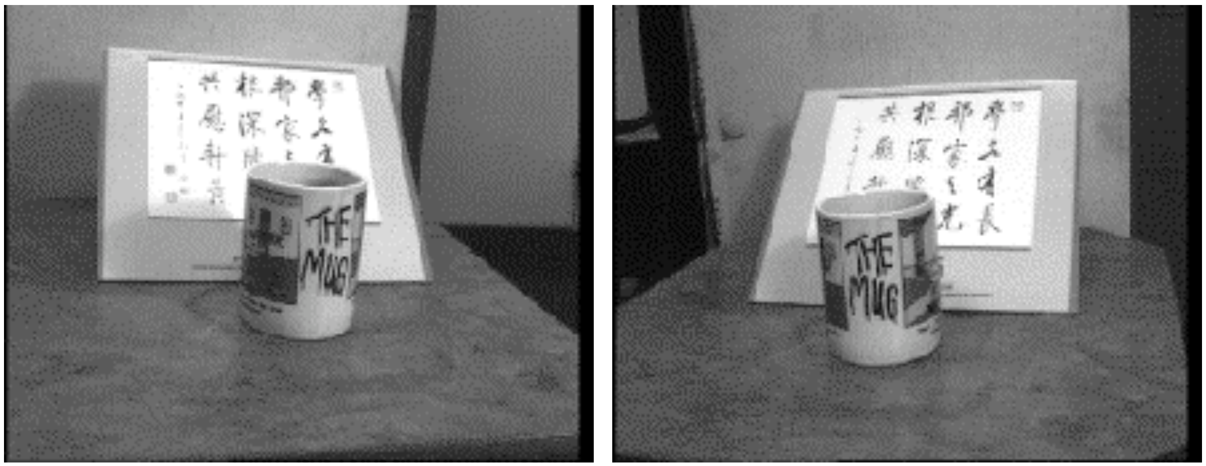
Avidan & Shashua, 1997

Faugeras and Robert, 1996

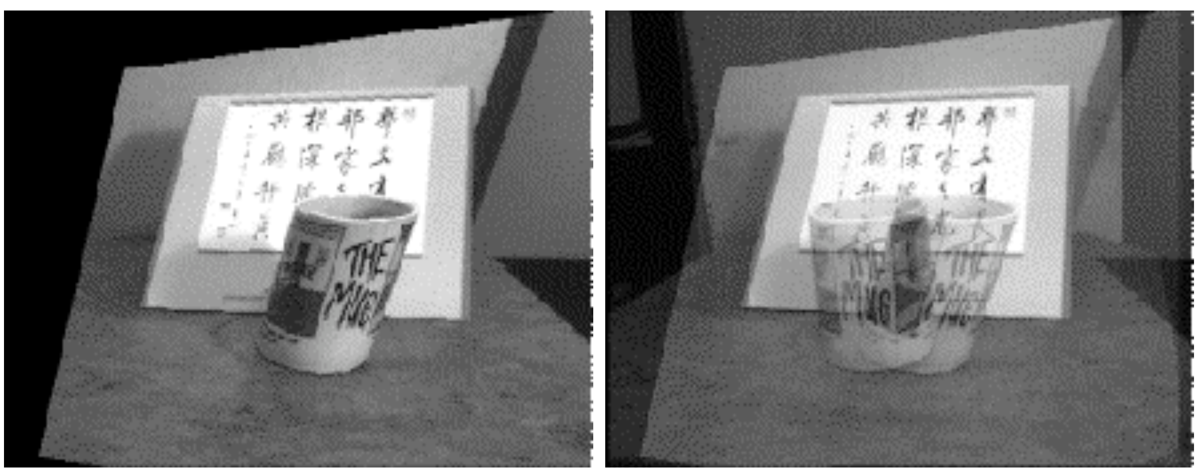


Undo Perspective Distortion (for a plane)

Original images (left and right)



Transfer and superimposed images

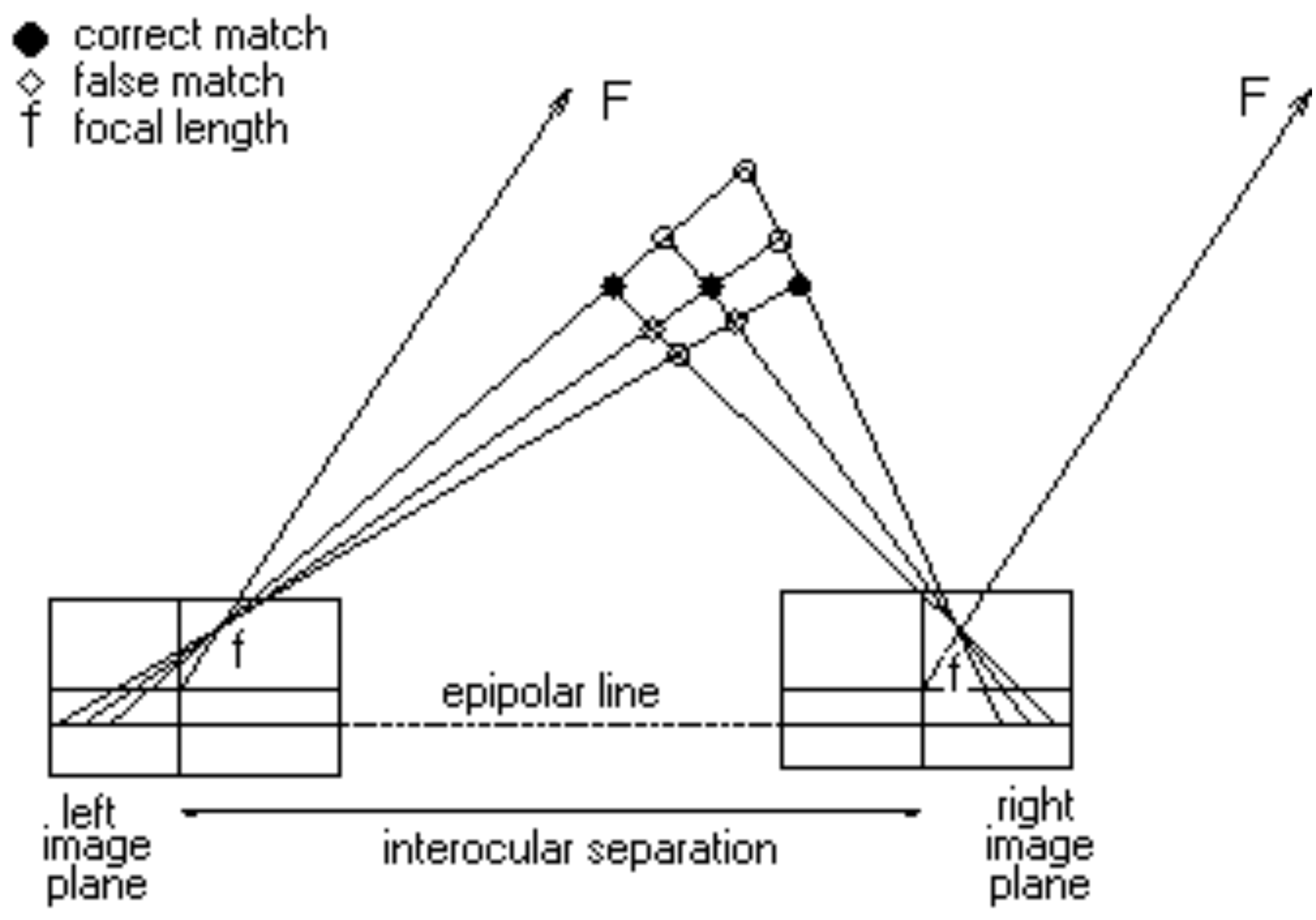


The ``transfer'' image is the left image projectively warped so that points on the plane containing the Chinese text are mapped to their position in the right image.

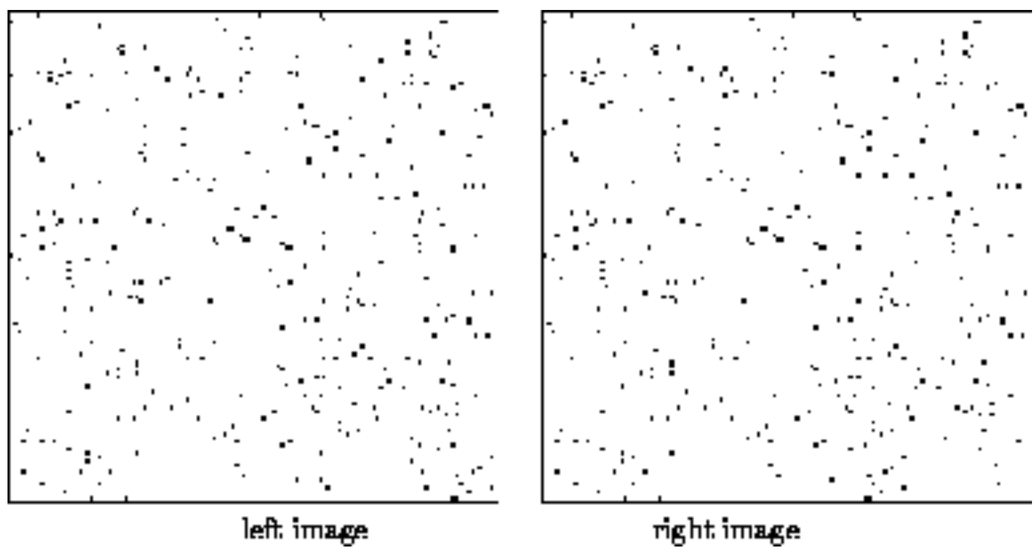
•The ``superimpose'' image is a superposition of the transfer and right image. The planes exactly coincide. However, points off the plane (such as the mug) do not coincide.

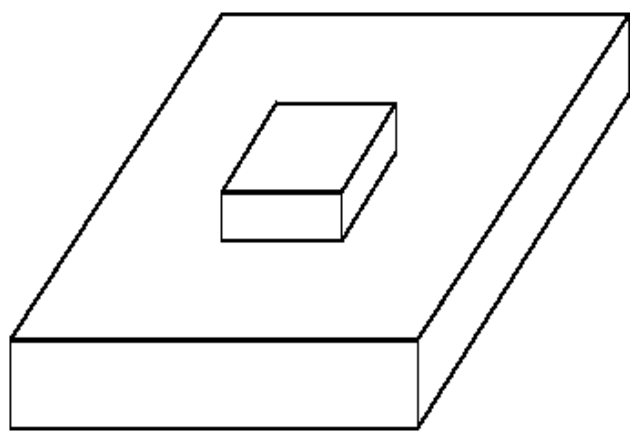
*This is an example of planar projectively induced parallax. Lines joining corresponding points off the plane in the ``superimposed'' image intersect at the epipole.

Its all about point matches



Point match ambiguity in human perception

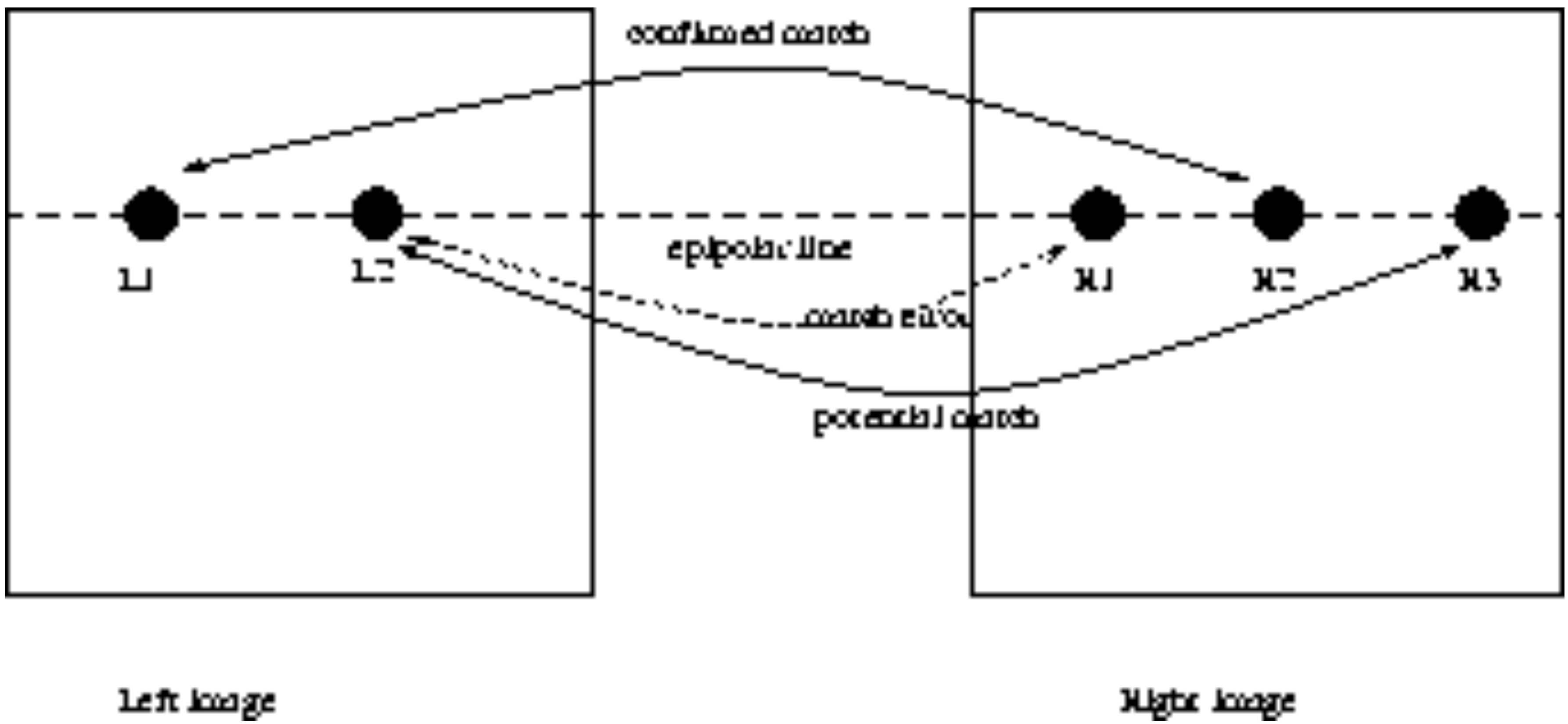




Traditional Solutions

- Try out lots of possible point matches.
- Apply constraints to weed out the bad ones.

Find matches and apply epipolar uniqueness constraint

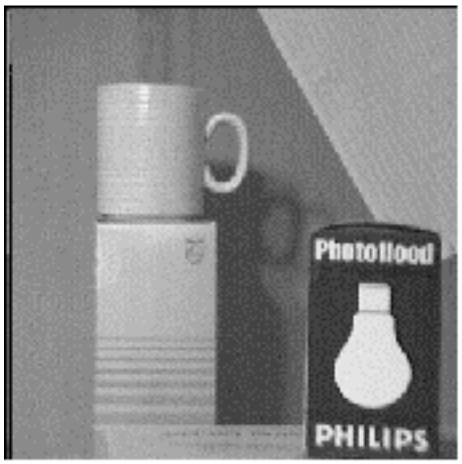


Compute lots of possible matches

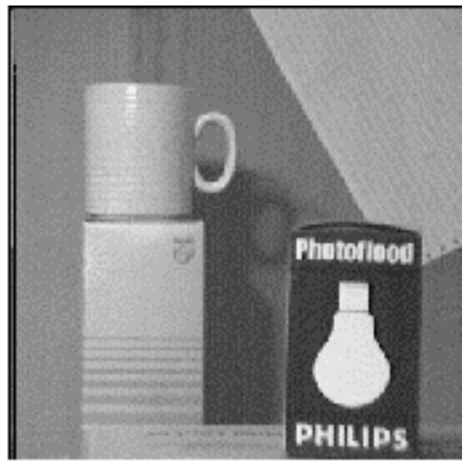
1) Compute “match strength”
2) Find matches with highest strength

Optimization problem with many possible solutions

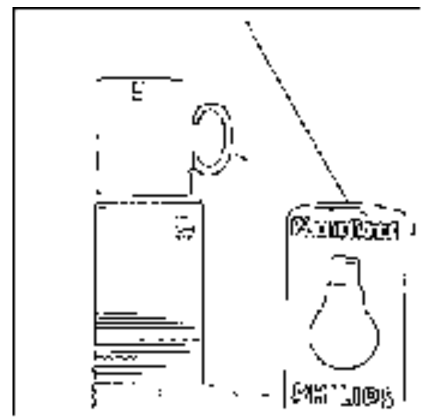
Example



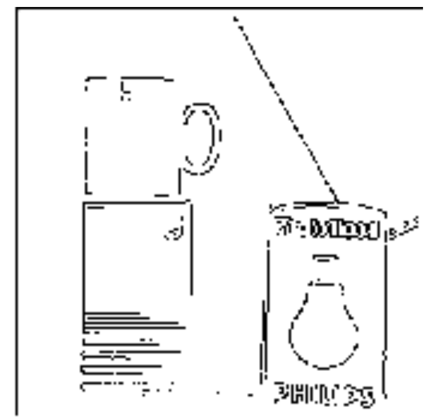
left image



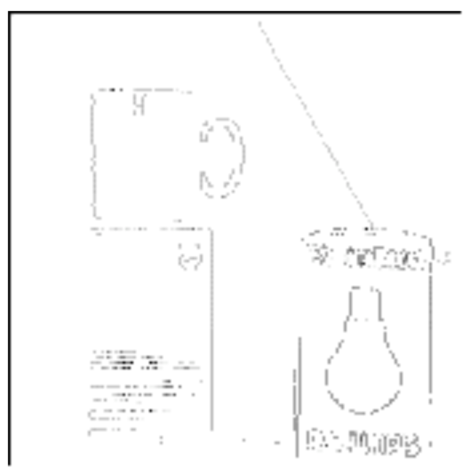
right image

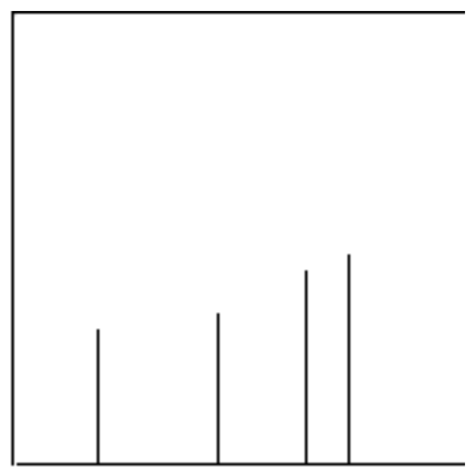


left image



right image





Cyclopean depth image   A scan along row 185 (across the bulb)

- E.g. from image points alone we cannot determine latitude and longitude or which way is north.
- Can only determine the location of the road up to a Euclidean transformation from the world coordinate frame.

- Cannot determine scale from image points either.
- Hence the scaling factor.

- Combining the Euclidean transformation and the scaling factor gives us a similarity transform.

rotation

translation

$$\mathbf{T}_{Sim} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & k \end{bmatrix}$$

scaling by k^{-1}

Fundamental Matrix, why are 8 point matches enough?

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \Rightarrow f_{33} = 1$$

Thus only 8 free parameters =>
Need 8 or more constraints.

Stereo Reconstruction Ambiguity

- Without knowledge of scene's placement with respect to a 3D coordinate frame it is not possible to determine the absolute position and orientation of the scene from 2 (or any number of) views.

- What does this mean mathematically?
- Given
  - a set of 3D points $\tilde{\mathbf{X}}_i$,
  - two cameras $\mathbf{P}$, $\mathbf{P'}$ and
  - image points $\tilde{\mathbf{x}}_i$, $\tilde{\mathbf{x}}'_i$.

- Remember these are related by:

$$\tilde{\mathbf{x}}_i = \mathbf{P}\tilde{\mathbf{X}}_i$$

$$\tilde{\mathbf{x}}'_i = \mathbf{P'}\tilde{\mathbf{X}}_i$$

- Replacing

  $\tilde{\mathbf{X}}_i$ with $\mathbf{T}_{\text{sim}}\tilde{\mathbf{X}}_i$,

  $\mathbf{P}, \mathbf{P}'$ with $\mathbf{P}\mathbf{T}_{\text{sim}}^{-1}, \mathbf{P}'\mathbf{T}_{\text{sim}}^{-1}$

- Does not change the observed image points

$$
\begin{aligned}
\tilde{\mathbf{x}}_i &= \mathbf{P}\tilde{\mathbf{X}}_i \\
&= (\mathbf{P}\mathbf{T}_{\text{sim}}^{-1})(\mathbf{T}_{\text{sim}}\tilde{\mathbf{X}}_i) \\
\tilde{\mathbf{x}}'_i &= \mathbf{P}'\tilde{\mathbf{X}}'_i \\
&= (\mathbf{P}'\mathbf{T}_{\text{sim}}^{-1})(\mathbf{T}_{\text{sim}}\tilde{\mathbf{X}}'_i)
\end{aligned}
$$

Extrinsic Parameter ambiguity

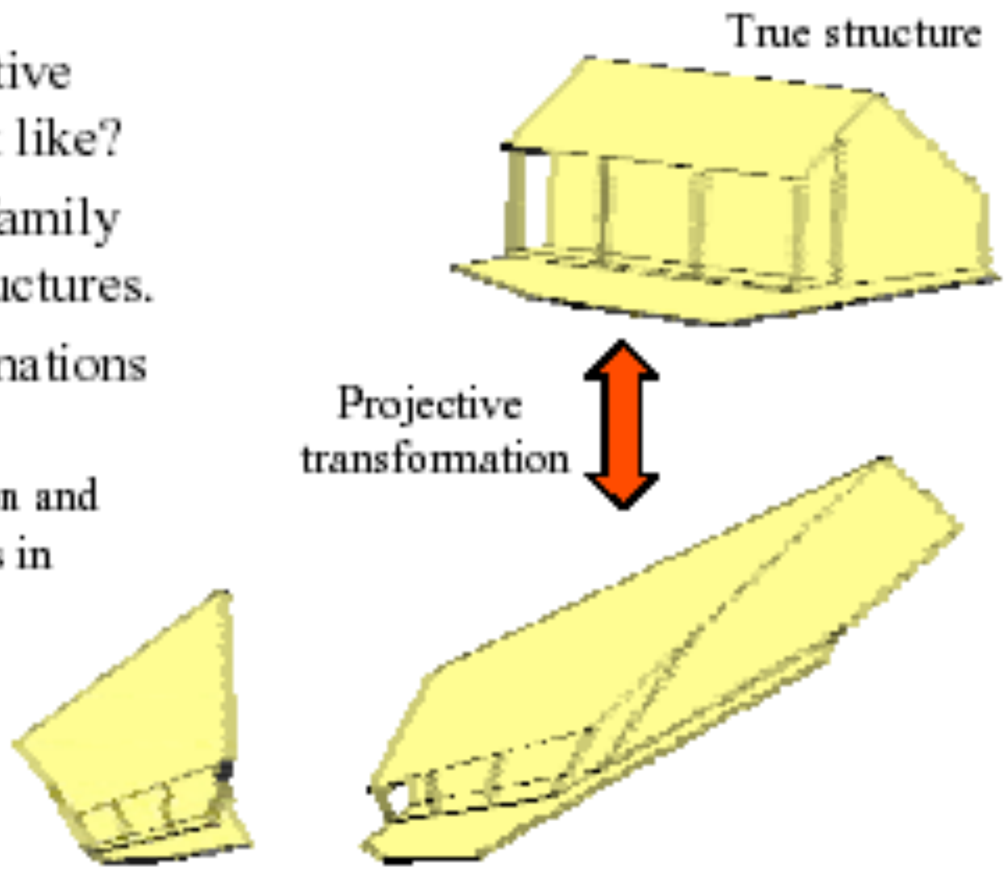
- If the camera calibration matrices are not known then the scene can only be constructed up to a *projective transformation* of the actual structure.
- A projective transformation is a homogeneous transformation of the form

$$\tilde{\mathbf{X}}_{\text{new}} = \mathbf{T}_{\text{proj}}\,\tilde{\mathbf{X}}$$

- where $\mathbf{T}_{\text{proj}}$ is any 4×4 invertible matrix.

Projective structure

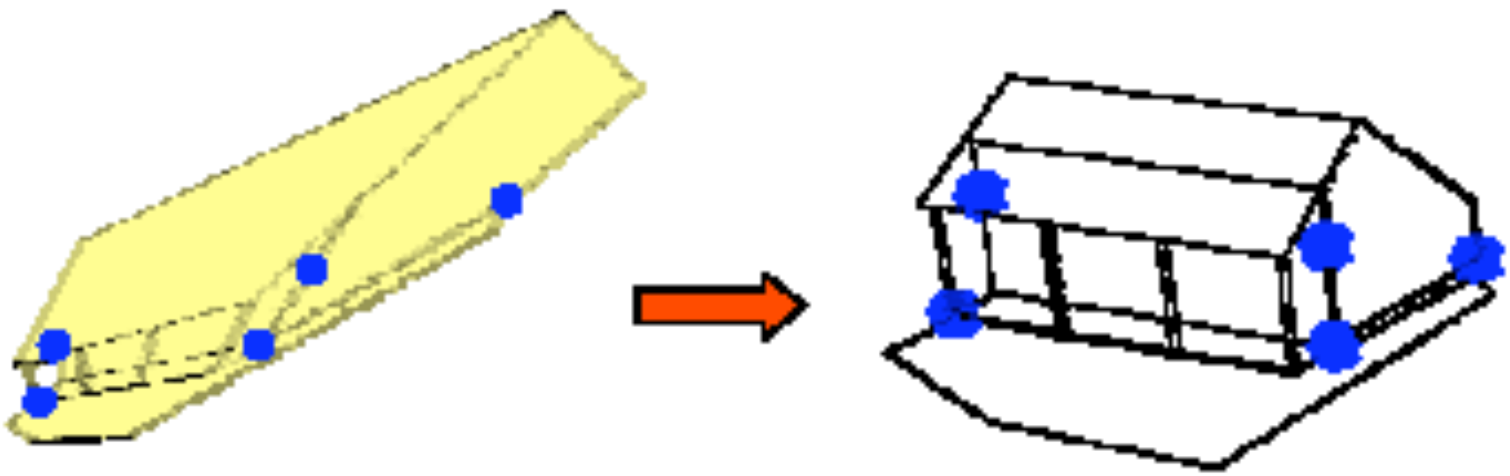
- What does a projective transformation look like?
- There are a whole family of these warped structures.
- Projective transformations
  - Map lines to lines
  - Preserve intersection and tangency if surfaces in contact.



2 views of a structure projectively equivalent to the true structure

Metric Structure

- If *control points* are available we can go from a projective to a true metric reconstruction.
- A projective reconstruction can be upgraded to a true metric reconstruction by specifying the 3D locations of 5 (or more) world points.



Projective reconstruction

Metric reconstruction