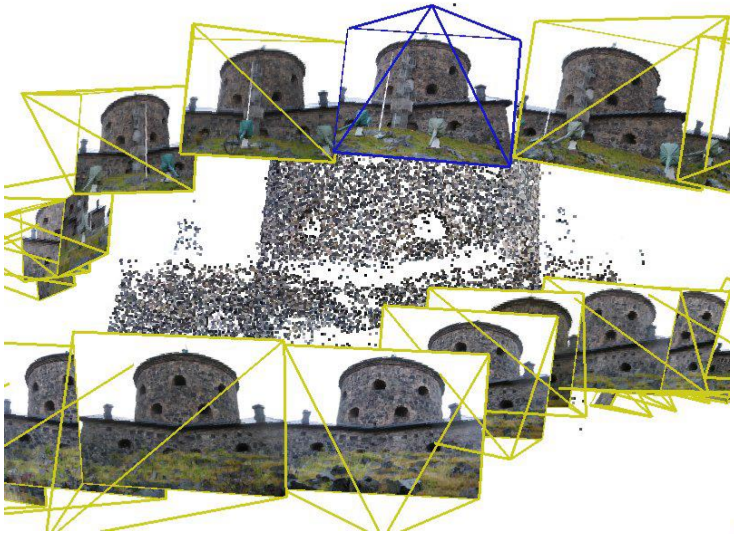
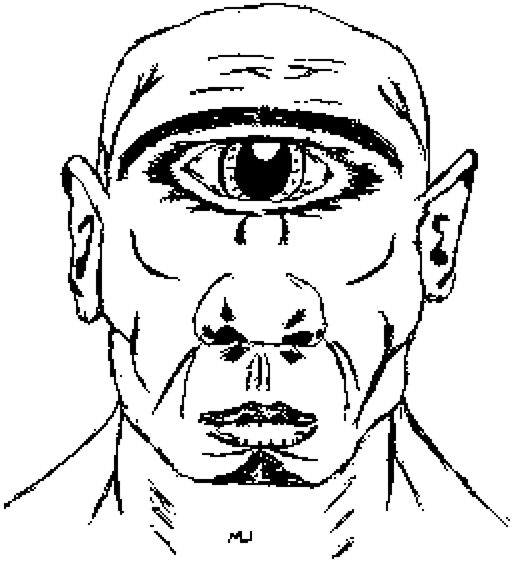
CS484/684 Computational Vision

# Multi-View Geometry



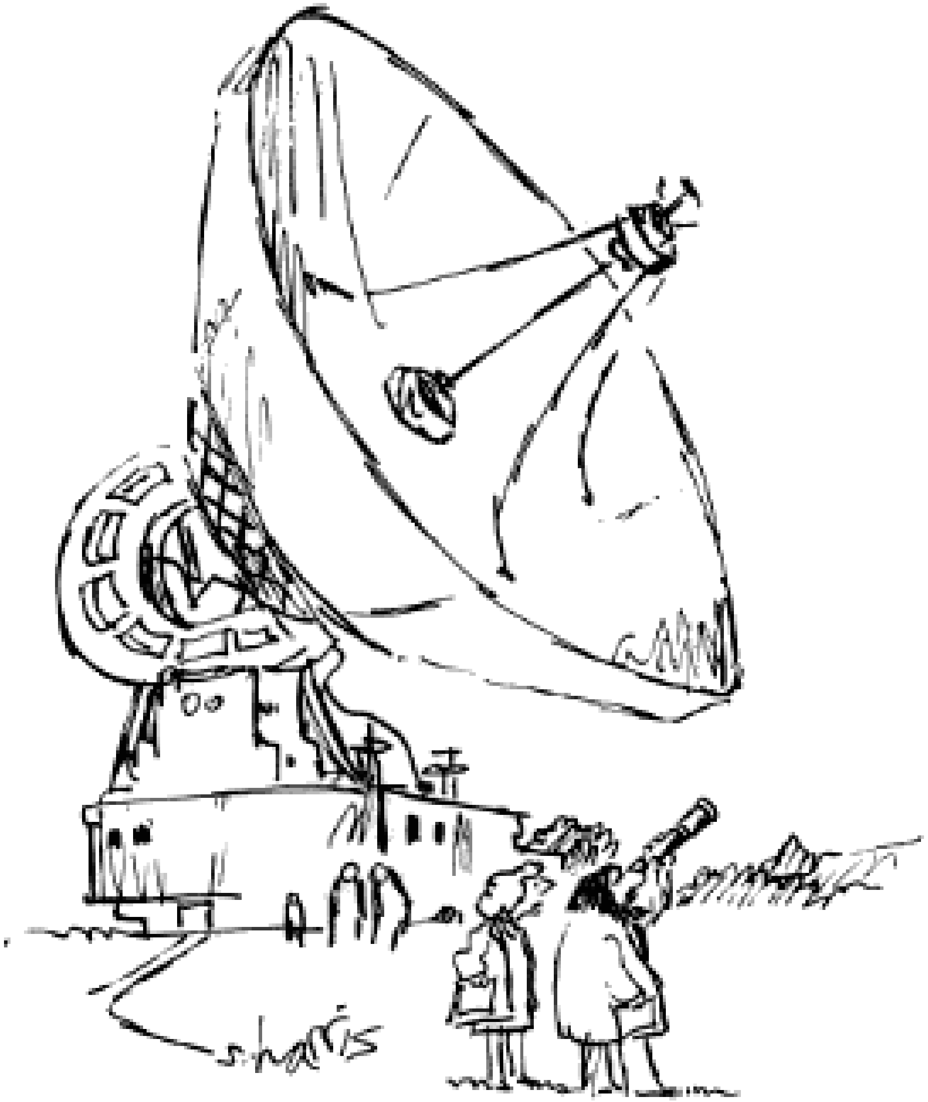
*…with materials from H&Z and Carl Olsson*

# Motivation: why do we have two eyes?



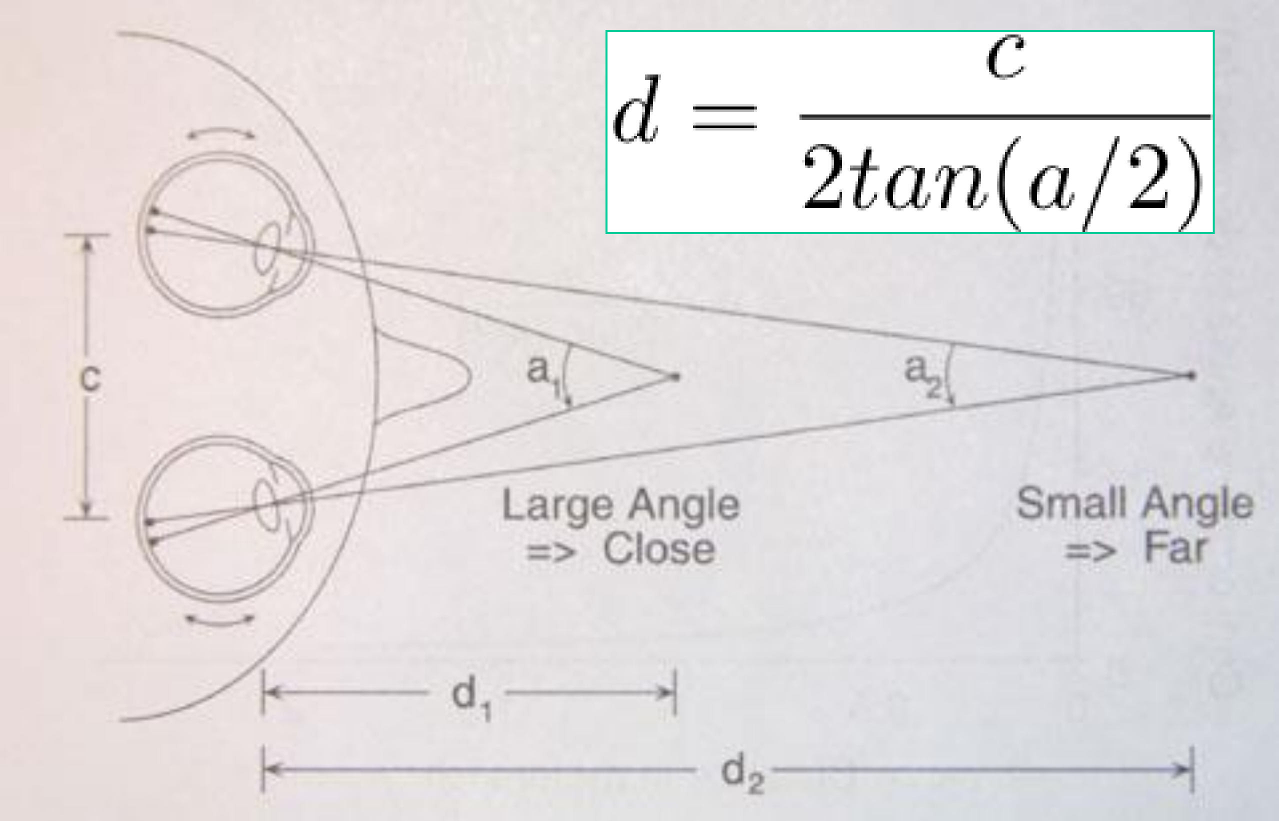
**Cyclope** **vs.** **Odysseus**

# Motivation: two is better than one



"Just checking."

# Motivation: **triangulation** gives depth

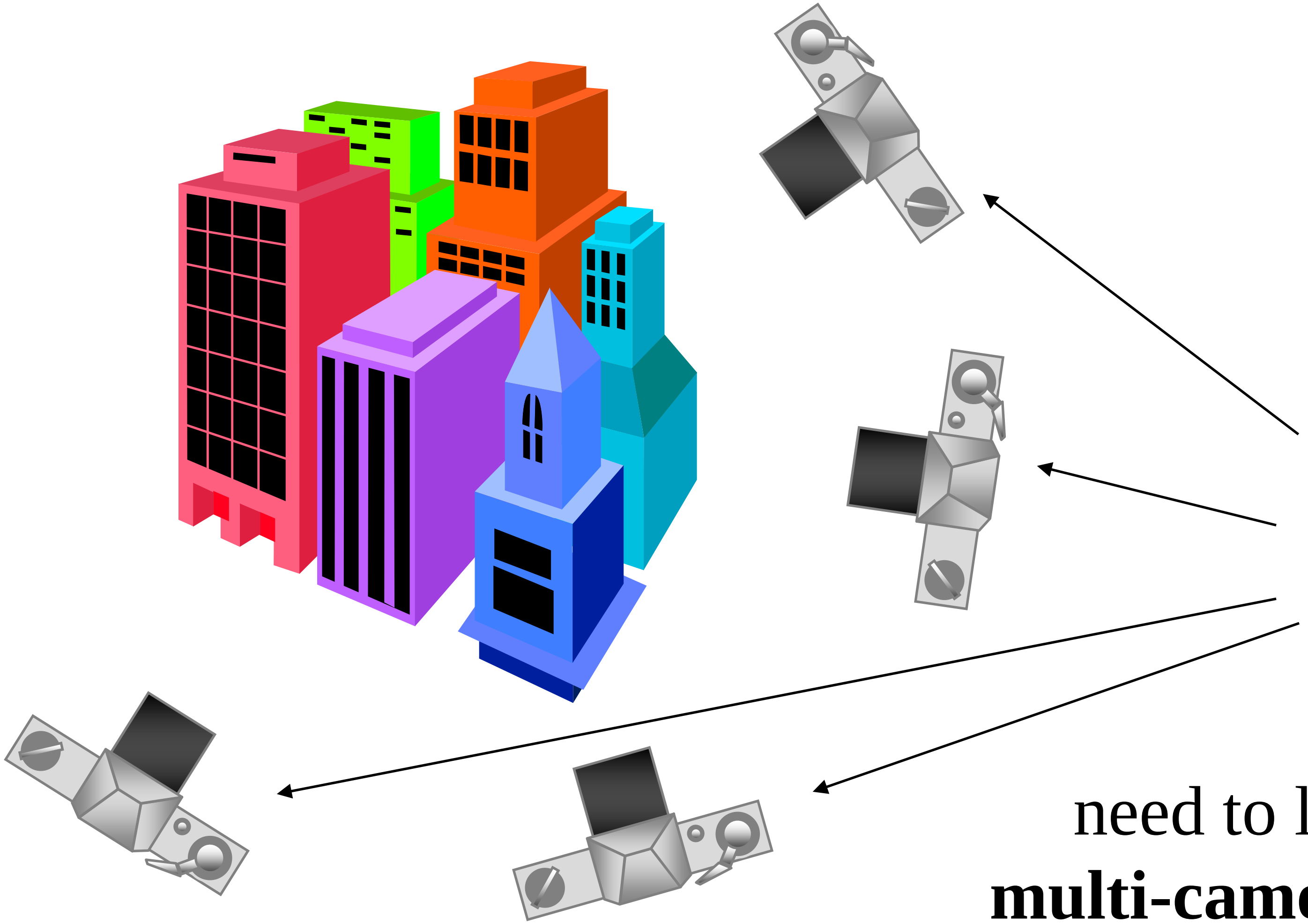


$$d = \frac{c}{2tan(a/2)}$$

*Human performance: up to 6-8 feet*

# Motivation: reconstruction problems

Multi-view reconstruction: **shape from two or more images**



**to "triangulate" should estimate camera viewpoints**

need to learn about **multi-camera geometry**

# Summary:

- Projective Camera Model
  - intrinsic and extrinsic parameters
  - projection matrix (a.k.a. camera matrix)
  - camera calibration (from known 3D points)
    - resection problem
    - estimating intrinsic/extrinsic parameters

- Two cameras (*epipolar* geometry)
  - essential and fundamental matrices: $E$ and $F$
  - estimating $E$ (from matched features)
  - computing projection matrices from $E$

- *Structure-from-Motion (SfM)* problem - quick overview
  - estimating "**motion**": camera positions (projection matrices)
  - estimating "**structure**": scene points in 3D space

at the same time
(both are unknown)

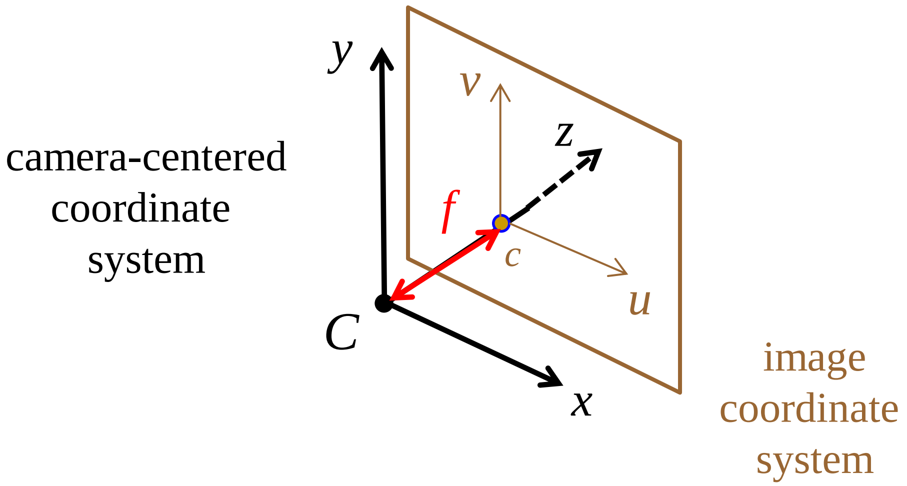
# Additional readings:

- Hartley and Zisserman *“Multiple View Geometry”*
  *Cambridge University Press, Ed.2*

- Heyden and Pollefeys *“Multiple View Geometry”*
  short course at CVPR 2001
  https://inf.ethz.ch/personal/marc.pollefeys/pubs/HeydenPollefeysCVPR01.pdf

# Towards projective camera model

First, if there is only one camera, can use a **camera-centered 3D coordinate system** (*x*,*y*,*z*):

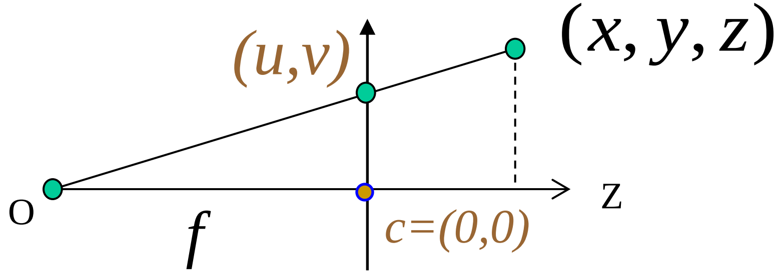


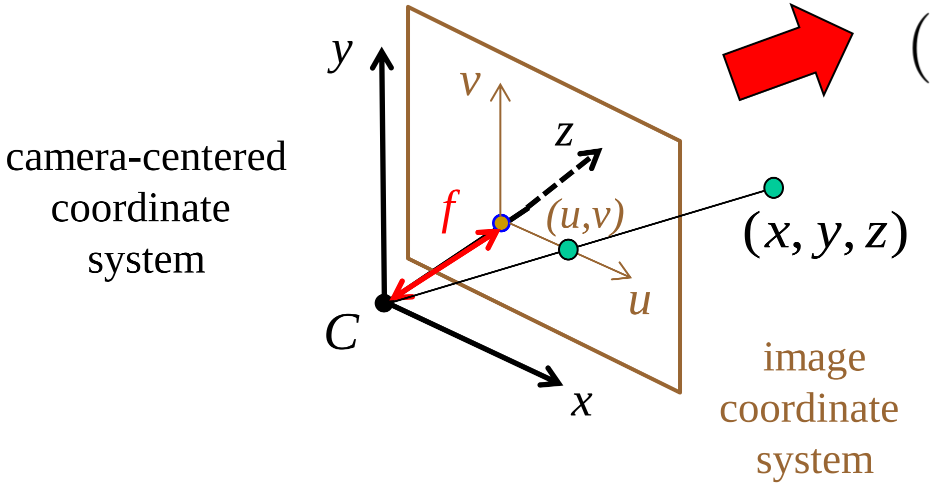
**as seen in topic 2**

- optical center is point (0,0,0)
- *x* and *y* axis are parallel to the image plane
- *x* and *y* axis parallel to *u* and *v* axis of the image coordinate system
- optical axis (*z*) intersects image plane at image point $c = (0,0)$

# Camera-centered coordinate system

For simplicity,
illustration below assumes
world point (x,y,x)
is inside x-z plane





$$(x, y, z) \rightarrow (\underbrace{f\frac{x}{z}}_{u}, \underbrace{f\frac{y}{z}}_{v})$$

image-based coordinates
of the **projection point**

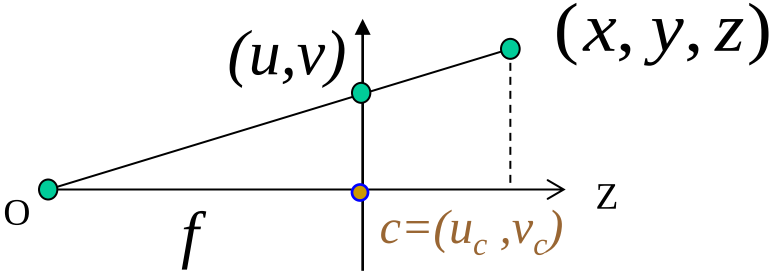
**as seen in topic 2**

- optical center is point (0,0,0)
- $x$ and $y$ axis are parallel to the image plane
- $x$ and $y$ axis parallel to $u$ and $v$ axis of the image coordinate system
- optical axis ($z$) intersects image plane at image point $c = (0,0)$

# Camera-centered coordinate system

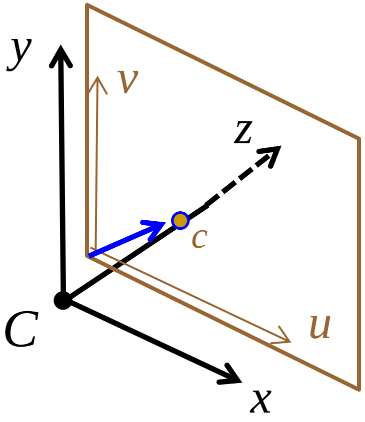
In general, image coordinate center can be anywhere (often in image corner).

Thus, optical axis may intersect image plane at a point with image coordinates $c=(u_c, v_c)$ contributing **additional shift**



$$(x, y, z) \rightarrow (\underbrace{f\frac{x}{z} + u_c}_{u}, \underbrace{f\frac{y}{z} + v_c}_{v})$$

image-based coordinates of the projection point



camera-centered coordinate system

image coordinate system

# Camera-centered coordinate system

camera projection
can be represented as
**matrix multiplication**

$$(x, y, z) \rightarrow (\underbrace{f\frac{x}{z} + u_c}_{u}, \underbrace{f\frac{y}{z} + v_c}_{v})$$

image-based coordinates
of the **projection point**

using **homogeneous representation**
for image points

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \underbrace{\begin{bmatrix} f & 0 & u_c \\ 0 & f & v_c \\ 0 & 0 & 1 \end{bmatrix}}_{K} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

**matrix of intrinsic**
**camera parameters**

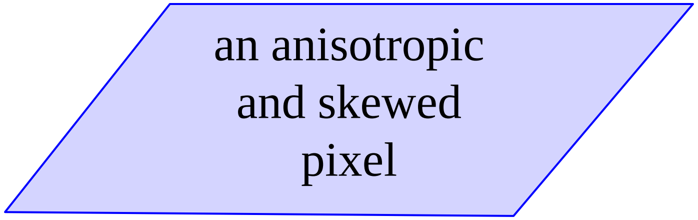
**camera centered coordinates**
for 3D world points

**NOTE: $w = z$ (depth)**

# Camera-centered coordinate system

Generally, **anisotropic** or **skewed** pixels result in

- different $f_x$ and $f_y$
- skew coefficient $s$


an anisotropic and skewed pixel

$s$ - skew/tilt

$\frac{f_x}{f_y}$ - aspect ratio

using **homogeneous representation** for image points

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & s & u_c \\ 0 & f_y & v_c \\ 0 & 0 & 1 \end{bmatrix}}_{K} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

**matrix of intrinsic camera parameters**

**camera centered coordinates** for 3D world points

# Camera-centered coordinate system

**In general**, matrix *K* of intrinsic camera parameters is 3x3 **upper triangular**. It has 5 degrees of freedom. For square pixels, *K* has 3 d.o.f.

using **homogeneous representation** for image points

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & s & u_c \\ 0 & f_y & v_c \\ 0 & 0 & 1 \end{bmatrix}}_{K} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

**matrix of intrinsic camera parameters**

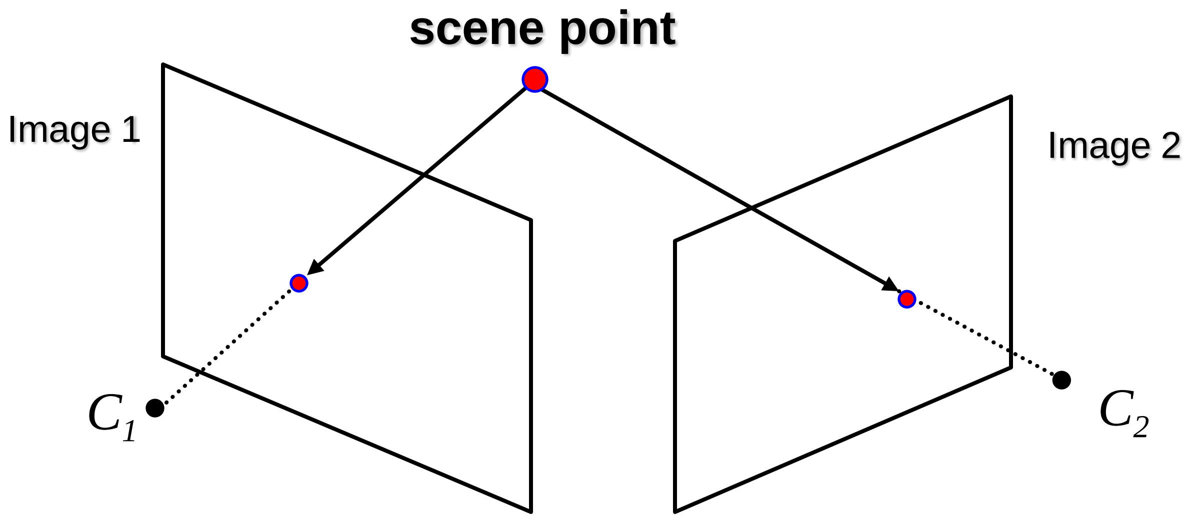
**camera centered coordinates** for 3D world points

NOTE: here matrix *K* maps $\mathbb{R}^3$ to $\mathbb{R}^2$ ($\mathbb{P}^2$)

(**not a homography** $\mathbb{P}^2 \rightarrow \mathbb{P}^2$)

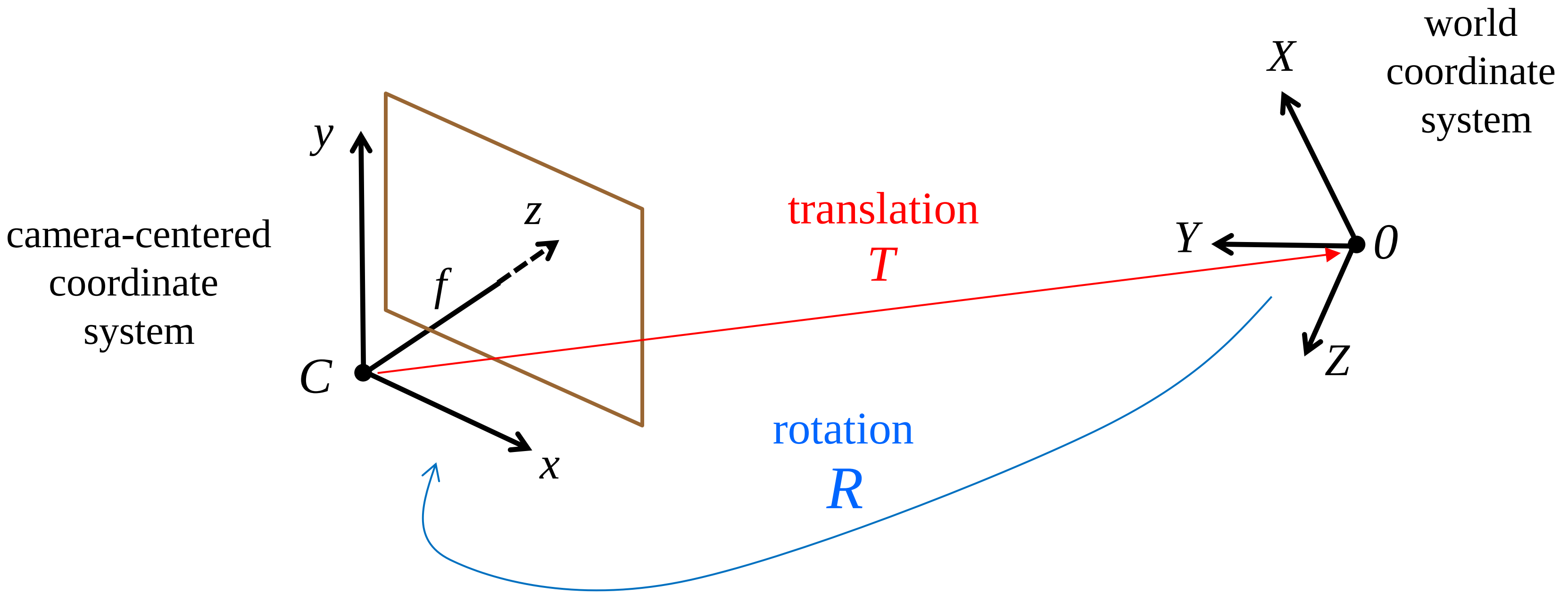
# What if there are more than one camera?

Projecting 3D scene onto images with different view-points



Only one camera can serve for world coordinate system.
Other cameras will have their **camera-centered 3D coordinates different from the world coordinate system**.

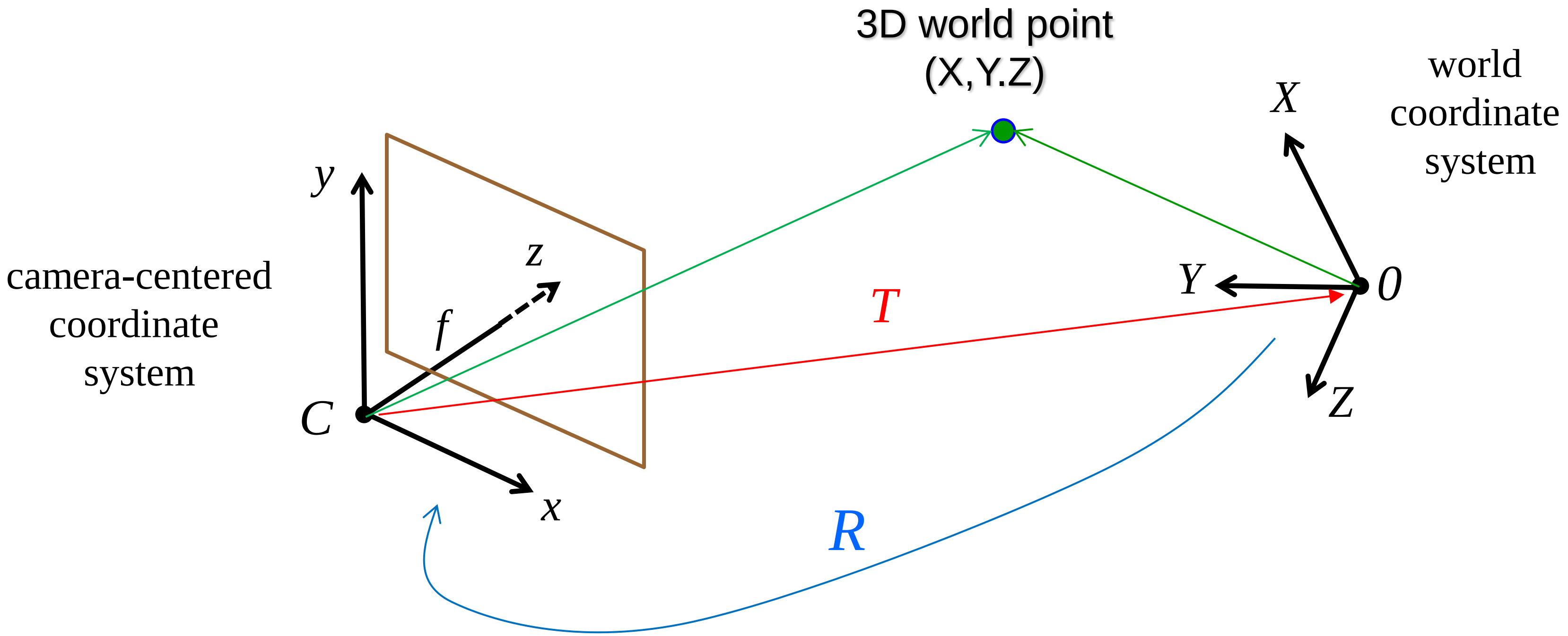
# Camera projection matrix



**In case of two or more cameras, 3D world coordinate system maybe different from a camera-based coordinate system:**

- $T$ is a (translation) vector defining relative position of camera's center
- orientation of $x,y,z$-axis of the camera-based coordinate system can be related to the axis of the world coordinate system via rotation matrix $R$

# Camera projection matrix



Converting world coordinates of a point into camera-based 3D coordinate system

$$\begin{bmatrix}x\\y\\z\end{bmatrix} = R \cdot \begin{bmatrix}X\\Y\\Z\end{bmatrix} + T$$

camera-based 3D coordinates

world 3D coordinates

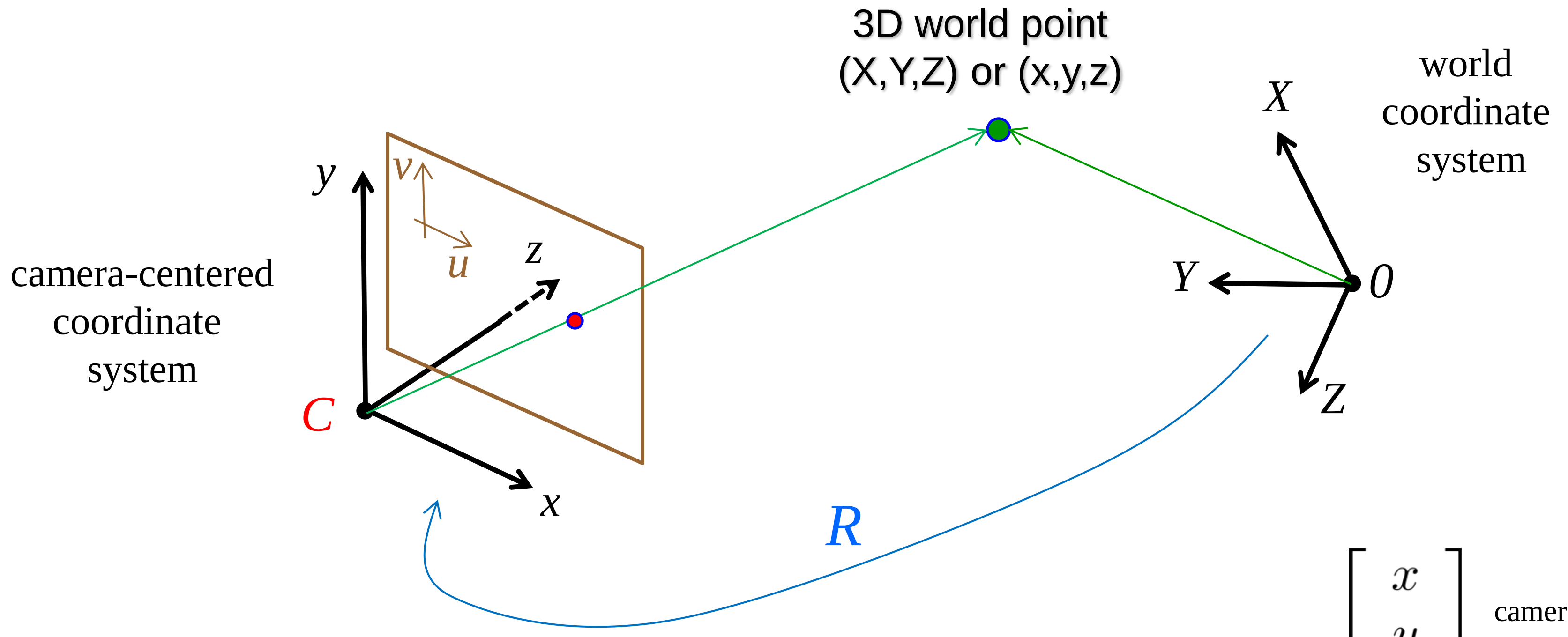
(here vector $T$ is world's center in camera's coordinates)

using **homogeneous representation** for 3D points in world coordinate system

$$\underset{3x1}{\begin{bmatrix}x\\y\\z\end{bmatrix}} = \underbrace{\left[\begin{array}{c|c} R & T \end{array}\right]}_{3x4} \cdot \underset{4x1}{\begin{bmatrix}X\\Y\\Z\\1\end{bmatrix}}$$

we get **linear transformation** (matrix multiplication)

# Camera projection matrix



Remember, projecting to 2D image coordinates…

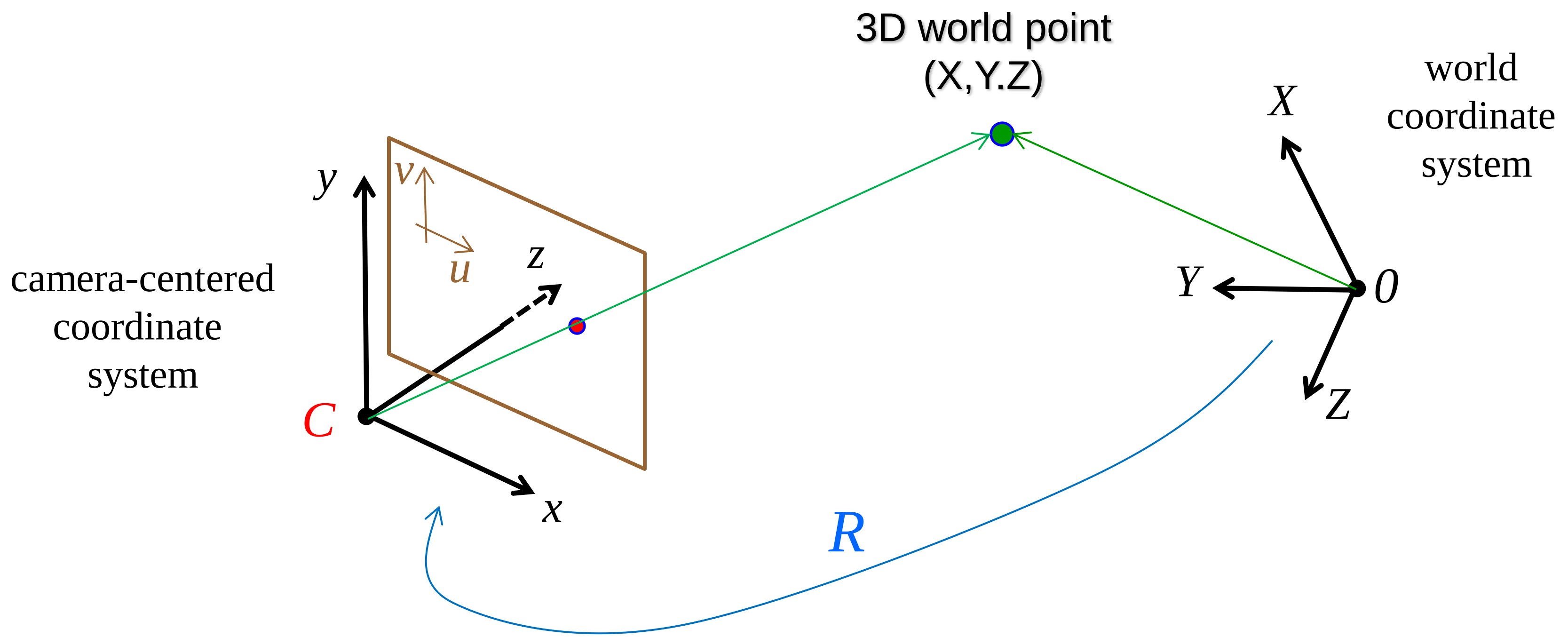
$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = K \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \Rightarrow \quad \underset{\text{3x3}}{\begin{bmatrix} wu \\ wv \\ w \end{bmatrix}} = \underset{\text{3x3}}{\overset{\text{5 d.o.f}}{K}} \cdot \overbrace{\underbrace{\left[\begin{array}{c|c} \overset{\text{3 d.o.f}}{R} & \overset{\text{3 d.o.f}}{T} \end{array}\right]}_{\text{3x4}} \cdot \underset{\text{4x1}}{\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}}^{\begin{bmatrix} x \\ y \\ z \end{bmatrix} \text{ camera-based 3D coordinates}}$$

homogeneous image coordinates

camera-based 3D coordinates

**project** **rotate** **translate**

# Camera projection matrix



3D world point
(X,Y.Z)

world
coordinate
system

camera-centered
coordinate
system

projection matrix $P$

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = K \cdot \left[\begin{array}{c|c} R & T \end{array}\right] \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \Longleftrightarrow \quad \tilde{p} = P \cdot \tilde{X}$$

3x1 3x4 4x1

homogeneous
2D image
coordinates
$\tilde{p}$

**intrinsic**
camera
parameters

**extrinsic**
camera
parameters

homogeneous
3D world
coordinates
$\tilde{X}$

# Homogeneous coordinates in 2D and 3D

Trick of adding one more coordinate

- translation becomes matrix multiplication
- 2D points become 3D rays

$$\text{in } \mathbb{R}^2 \quad (u,v) \Rightarrow \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} wu \\ wv \\ w \end{bmatrix} \quad \text{in } \mathbb{P}^2$$

homogeneous 2D image coordinates

$$\overset{\text{in } \mathbb{R}^3}{(X,Y,Z)} \Rightarrow \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \sim \begin{bmatrix} wX \\ wY \\ wZ \\ w \end{bmatrix} \quad \text{in } \mathbb{P}^3$$

homogeneous 3D scene coordinates

## Converting *from* homogeneous coordinates

$$\underset{\text{in } \mathbb{P}^2}{\begin{bmatrix} x \\ y \\ w \end{bmatrix}} \Rightarrow \underset{\text{in } \mathbb{R}^2}{(x/w, y/w)}$$

$$\underset{\text{in } \mathbb{P}^3}{\begin{bmatrix} X \\ Y \\ Z \\ w \end{bmatrix}} \Rightarrow \underset{\text{in } \mathbb{R}^3}{(X/w, Y/w, Z/w)}$$

# Camera calibration

**Goal**: estimate intrinsic camera parameters

- focal length $f$, image center $(u_c, v_c)$, other elements of **matrix $K$**

- if needed, corrections for lens distortions (*radial distortion* in fish eye lenses) not represented by $K$

## **Motivation**:

- if $K$ is known, only 6 *d.o.f* remains in projection matrix $P = K \cdot (R|T)$ (3 *d.o.f*. for each rotation $R$ and translation $T$ )

  => it becomes **easier to estimate projection matrices** corresponding to different viewpoints as camera(s) move around

- using *calibrated* camera(s) is a way to **remove projective ambiguity** in *structure from motion* 3D reconstruction (*more later*)

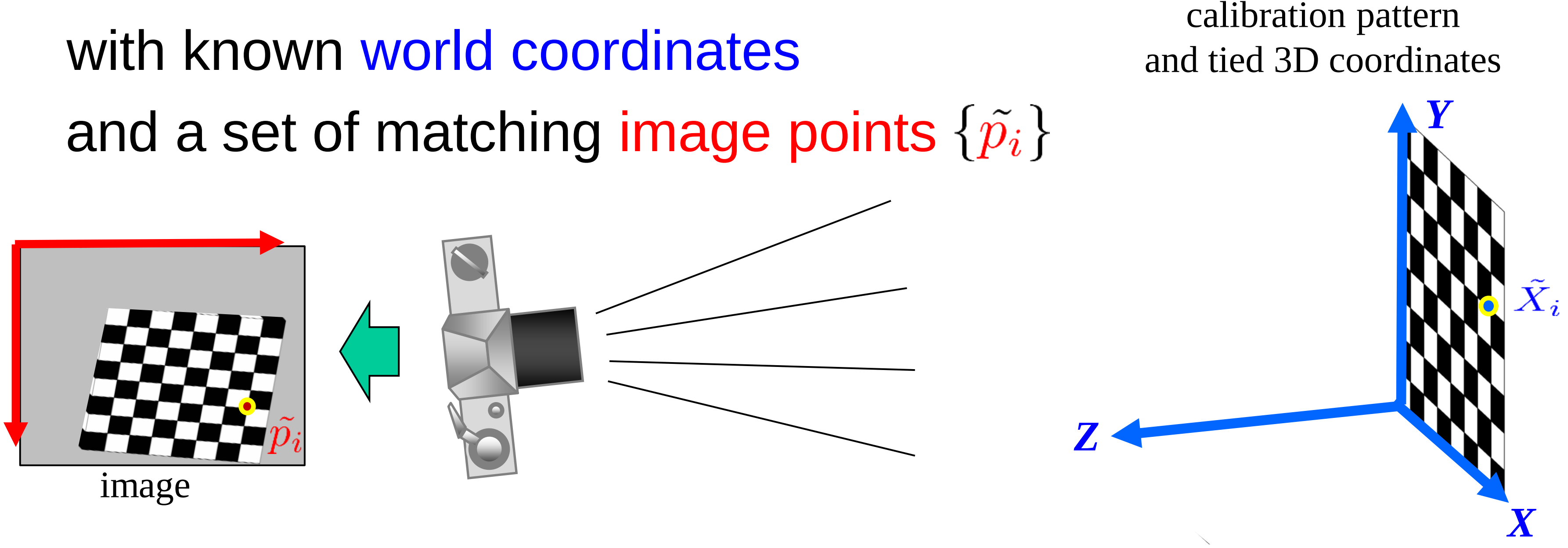
# Camera calibration

## Basic calibration technique:

assume a set of 3D points $\{\tilde{X}_i\}$

with known world coordinates

and a set of matching image points $\{\tilde{p}_i\}$



$$\tilde{X}_i \leftrightarrow \tilde{p}_i$$

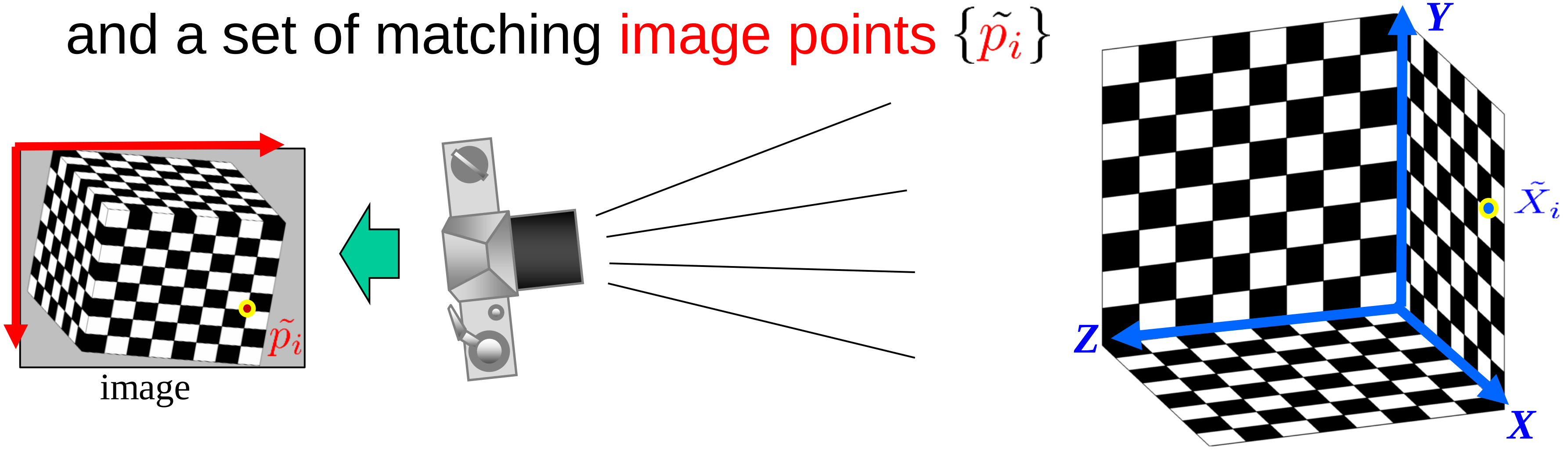
- find camera matrix *P* from known matches (**resection problem**)
- then, find intrinsic and extrinsic parameters (use **matrix factorization**)

# Camera calibration

**NOTE: should not use 3D points $\{\tilde{X}_i\}$ on a single plane**
("degenerate configurations", see H&Z Sec 7.1)

## Basic calibration technique:

assume a set of 3D points $\{\tilde{X}_i\}$
with known world coordinates
and a set of matching image points $\{\tilde{p}_i\}$

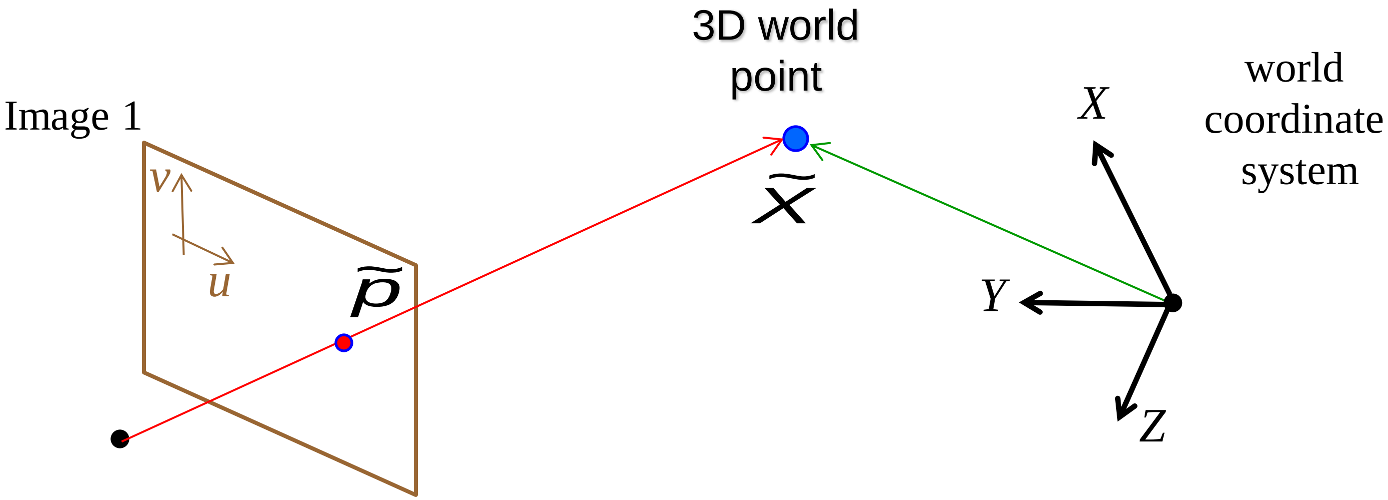


calibration rig
(*Tsai grid*)

$\tilde{X}_i \leftrightarrow \tilde{p}_i$

- find camera matrix *P* from known matches (**resection problem**)
- then, find intrinsic and extrinsic parameters (use **matrix factorization**)

# Camera projection matrix (estimating from $\tilde{X}_i \leftrightarrow \tilde{p}_i$)



$P$ has 12 entries, 11 d.o.f.

**Q:** How many matched pairs $\tilde{X}_i \leftrightarrow \tilde{p}_i$ are needed ? **A:** 5.5 ☺

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

estimate unknown projection matrix $P$

**(resection problem)**

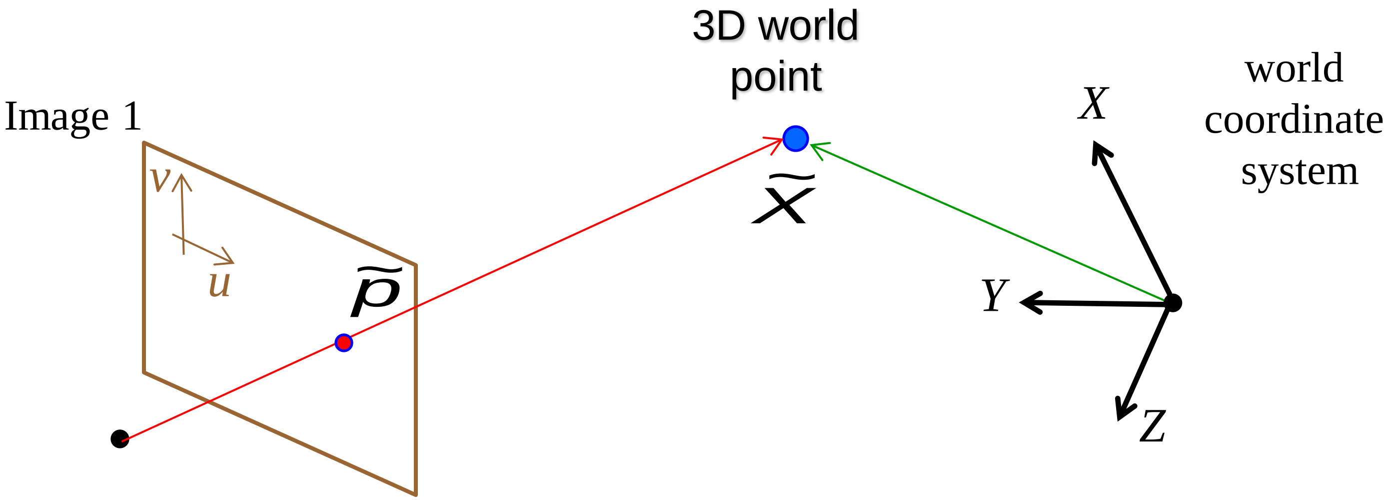
**Q:** Solving for $a,b,\ldots,k,l$ ?

**A:** similar to estimating homographies

(see Topic 3, or H&Z p.179)

*Exercise*: prove that coplanar $\{\tilde{X}_i\}$ give undetermined system of equations for $P$.

# Camera projection matrix (estimating from $\tilde{X}_i \leftrightarrow \tilde{p}_i$)



$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

estimate unknown projection matrix $P$

**(resection problem)**

- Use more than 6 matched pairs

$$\tilde{X}_i \leftrightarrow \tilde{p}_i$$

to compensate for errors
(*homogeneous least squares*)

# Extracting intrinsic parameters from *P*

Now, assume that 3x4 projection matrix *P* is already estimated

$$P = \underset{\text{known}}{\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix}} = \underset{\text{unknown}}{\boxed{\overset{\text{3x3}}{K} \cdot \overbrace{\left[\begin{array}{c|c} \quad R \quad & T \end{array}\right]}^{\text{3x4}}}}$$

How can we get *K* (as well as *R*,*T*) from *P* ?

# Extracting intrinsic parameters from $P$

$$P = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} \overset{?}{=} K \cdot \left[ \begin{array}{c|c} R & T \end{array} \right]$$

## matrix factorization:

H&Z Sec 6.2.4 (p. 163)

**Theorem** [$\mathfrak{Q}\mathfrak{R}$ or $\mathfrak{R}\mathfrak{Q}$ factorization]: for any *n*×*n* matrix *A* there is an orthogonal matrix $\mathfrak{Q}$ and an upper (or ***r***ight) triangular matrix $\mathfrak{R}$ such that $A = \mathfrak{R}\mathfrak{Q}$.

(If *A* is invertible and the diagonal elements in $\mathfrak{R}$ are chosen positive than the factorization is **unique.**)

$$P = \Big[ \underbrace{\begin{matrix} a & b & c \\ e & f & g \\ i & j & k \end{matrix}}_{A} \Big| \underbrace{\begin{matrix} d \\ h \\ l \end{matrix}}_{a} \Big] \overset{A = \mathcal{R}\mathcal{Q}}{=} \underbrace{\mathcal{R}}_{K} \cdot \Big[ \underbrace{\mathcal{Q}}_{R} \Big| \underbrace{\mathcal{R}^{-1} a}_{T} \Big]$$

scale $\mathfrak{R}$ to make bottom right element equal 1

# Calibrated Camera (*camera normalization*)

## Once intrinsic parameters $K$ are known

- can “**normalize**” the camera:

  switch to a new image coordinate system $(\tilde{u},\tilde{v})$ defined as

$$\begin{bmatrix} w\tilde{u} \\ w\tilde{v} \\ w \end{bmatrix} = K^{-1}\cdot\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

**Q**: what kind of transform is this for camera’s image?

- then, camera’s **new projection matrix** $\tilde{P}$ becomes

$$\tilde{P} = K^{-1}P = \cancel{K^{-1}\cdot K}\cdot\left[\begin{array}{c|c} R & T \end{array}\right] = \boxed{\left[\begin{array}{c|c} R & T \end{array}\right]}$$

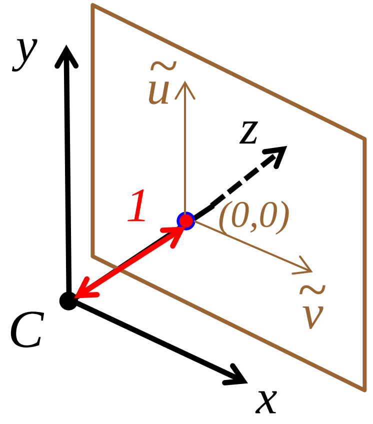
**rotation** and **translation** only

# Calibrated (Normalized) Camera

After normalization, “effective” intrinsic parameters form an **identity matrix**

$$\left[\begin{array}{c|c} R & T \end{array}\right] = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\tilde{K} = I} \cdot \underbrace{\left[\begin{array}{c|c} R & T \end{array}\right]}_{\text{extrinsic parameters}}$$

camera-centered coordinate system



normalized image embedded in $\mathbb{R}^3$

**Geometric interpretation**:

focal length $f = 1$

point (0,0) = intersection of image plane with optical axis

# Calibrated (Normalized) Camera

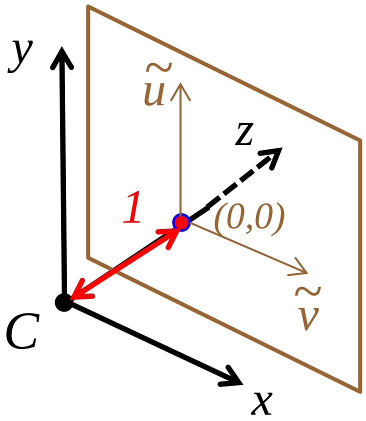
To project onto a calibrated camera (a.k.a. *normalized camera*) one needs only its position (**translation+rotation**) in world coordinates

calibrated/normalized camera's projection matrix

$$P = \left[ \begin{array}{c|c} R & T \end{array} \right]$$

still 3x4 matrix
but only 6 d.o.f

camera-centered coordinate system



normalized image embedded in $\mathbb{R}^3$

**Property** for normalized camera: (homogeneous) image coordinates for any pixel coincide with this pixel's camera-centered world coordinates **(can treat "normalized" pixels as points in $\mathbb{R}^3$)**

$$\tilde{p} \propto \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ 1 \end{bmatrix} \equiv \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix}$$

# Calibrated (Normalized) Camera

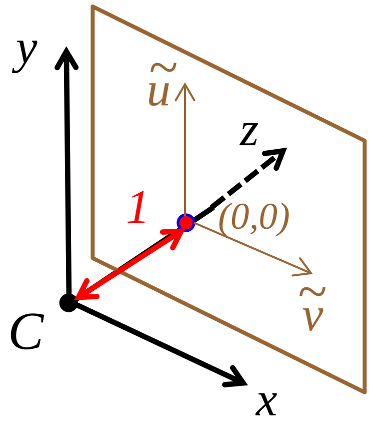
To project onto a calibrated camera (a.k.a. *normalized camera*) one needs only its position (**translation+rotation**) in world coordinates

calibrated/normalized camera's projection matrix

$$P = \left[\begin{array}{c|c} \quad R \quad & T \end{array}\right]$$

still 3x4 matrix
but only 6 d.o.f

camera-centered
coordinate
system



normalized image
embedded in $\mathbb{R}^3$

**from normalized back to original camera:**

use $K$ as a *warp* $p = K\tilde{p}$ $(\mathbb{P}^2 \rightarrow \mathbb{P}^2)$

$\Rightarrow$ $K$ can be interpreted as
a homography mapping normalized image
embedded in $\mathbb{R}^3$ to the "digital space"
(i.e. pixels in the original image)

**Q**: why restrict $K$ to upper triangular ?

*hint*: $K=\mathfrak{R}$ in $\mathfrak{R}\mathfrak{Q}$ decomposition

# Calibrated (Normalized) Camera

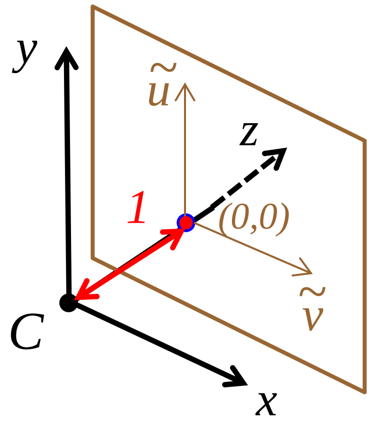
To project onto a calibrated camera (a.k.a. *normalized camera*) one needs only its position (**translation+rotation**) in world coordinates

calibrated/normalized camera's projection matrix

$$P = \left[ \begin{array}{c|c} R & T \end{array} \right]$$

still 3x4 matrix
but only 6 d.o.f

camera-centered coordinate system



normalized image embedded in $\mathbb{R}^3$

**The main point of calibration/normalization:** converts any camera to a "standardized" pin hole camera model shown on the left. After calibration, images are independent of how the camera is made and depend only on camera's location/orientation.

NOTE: in general, "calibration" process also correct for lens distortions (barrel, etc.)

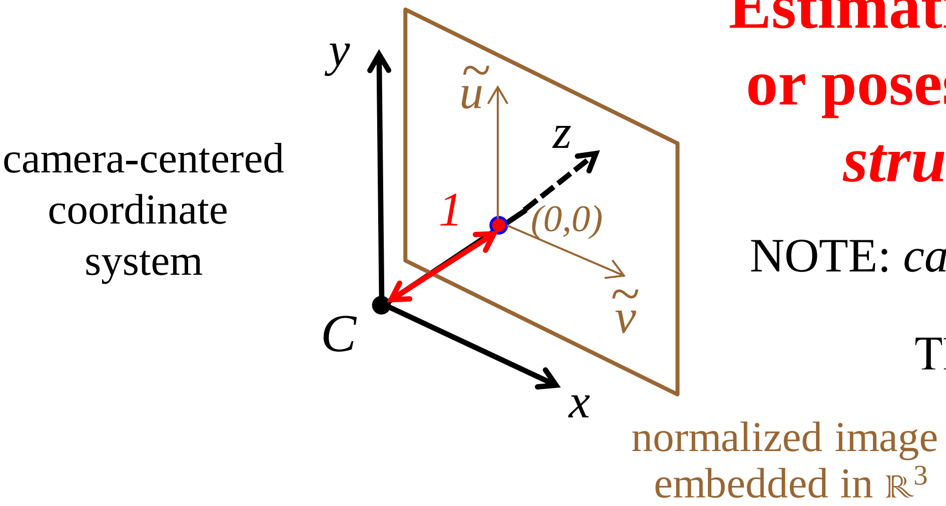
# Calibrated (Normalized) Camera

To project onto a calibrated camera (a.k.a. *normalized camera*) one needs only its position (**translation+rotation**) in world coordinates

calibrated/normalized camera's projection matrix

$$P = \left[ \begin{array}{c|c} R & T \end{array} \right]$$

still 3x4 matrix
but only 6 d.o.f



**Estimating multiple camera viewpoints or poses $P_n$ is the "motion" part of the *structure-from-motion* problem**

NOTE: *camera calibration* uses known 3D points $\{\tilde{X}_i\}$.

The "structure" part of *SfM* problem estimates unknown 3D scene points $\{\tilde{X}_i\}$.

**(later in this topic)**

# Calibrated (Normalized) Camera

**For simplicity, the rest of this topic assumes that all images are normalized (calibrated cameras)**

**unless explicitly stated otherwise**

# Two cameras geometry

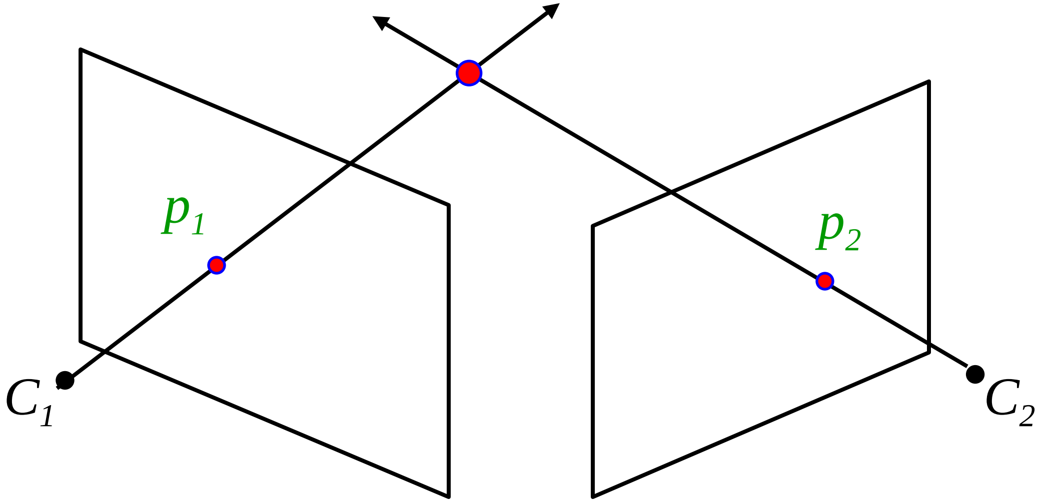
## Epipolar geometry

essential & fundamental matrices

Motivation: helps reconstruction

# Stereo reconstruction

From 2D images back to 3D scene



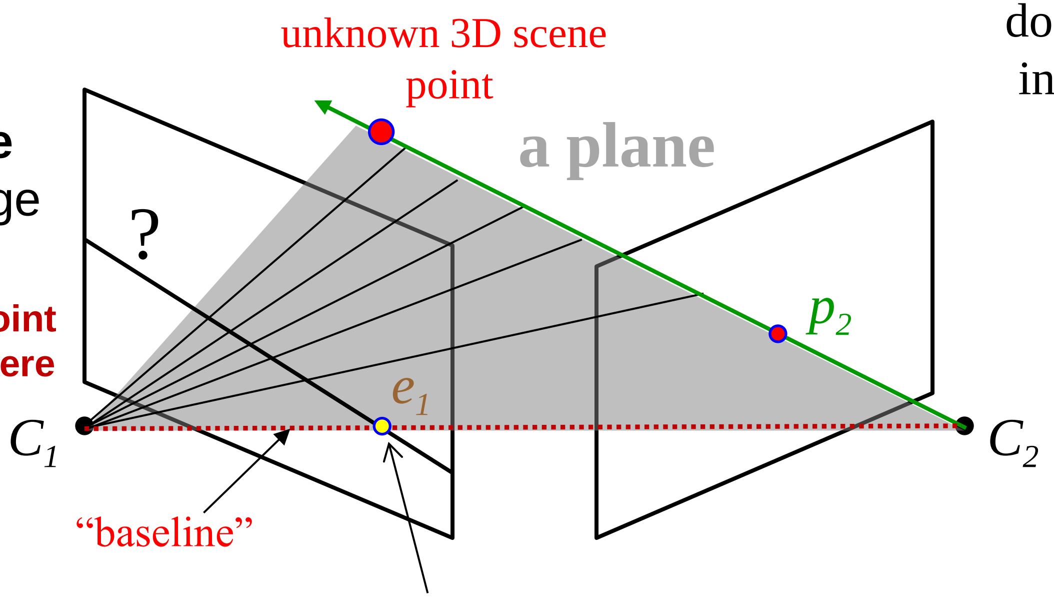
**Triangulation:** can reconstruct a point as an intersection of two rays, **assuming**…

- known projection matrix (camera position)
- known ***point correspondence***

# Epipolar lines

- Find pairs of corresponding pixels (that come from the same 3D scene point)
  - not trivial (remember mosaicing)

**A rhetorical question**:
does any ray from $C_1$
intersect ray $C_2 p_2$ ?



***epipolar* line**
in the left image
for point $p_2$

**corresponding point must be somewhere on this line**

**left image epipole**
(definition: point where optical center $C_2$ projects onto left image plane)

Any right image point $p_2$ corresponds to a line passing though **epipole $e_1$**.

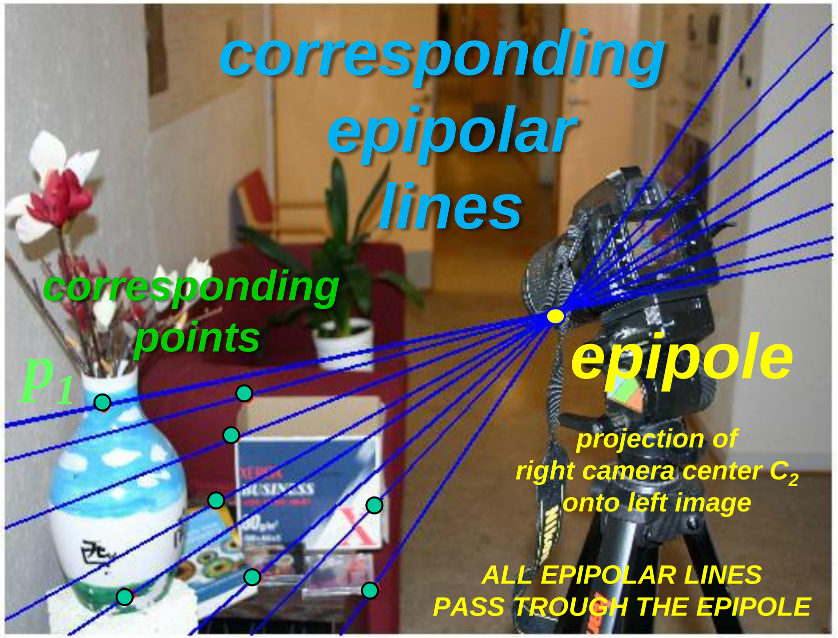
**It is a projection of ray** $C_2 \rightarrow p_2$ (ray $C_2 \rightarrow$ unknown 3D scene point).

# Epipolar lines

**Example** [from Carl Olsson]

**(two stationary cameras)**

consider some features
in the right image
(projections of some 3D points)



left camera image
(contains the right camera)

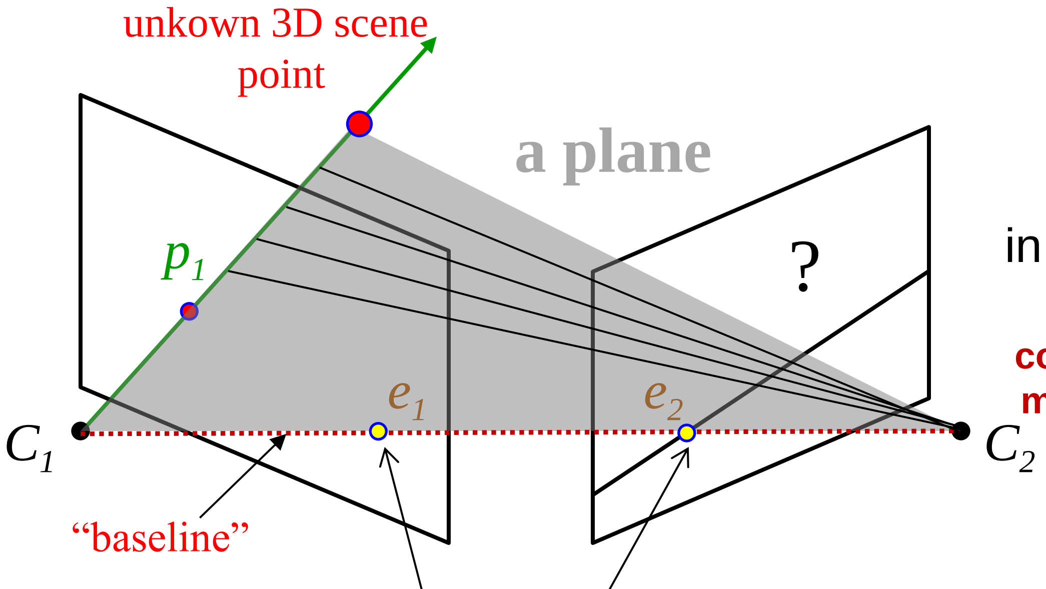
right camera image

Any right image point $p_2$ corresponds to some left image **epipolar line**.

**It is a projection of ray** $C_2 \rightarrow p_2$ (ray $C_2 \rightarrow$ unknown 3D scene point).

# Epipolar lines

**Similarly**, for any given point $p_1$ in the left image…



***epipolar* line**
in the right image
for point $p_1$

**corresponding point must be on this line**

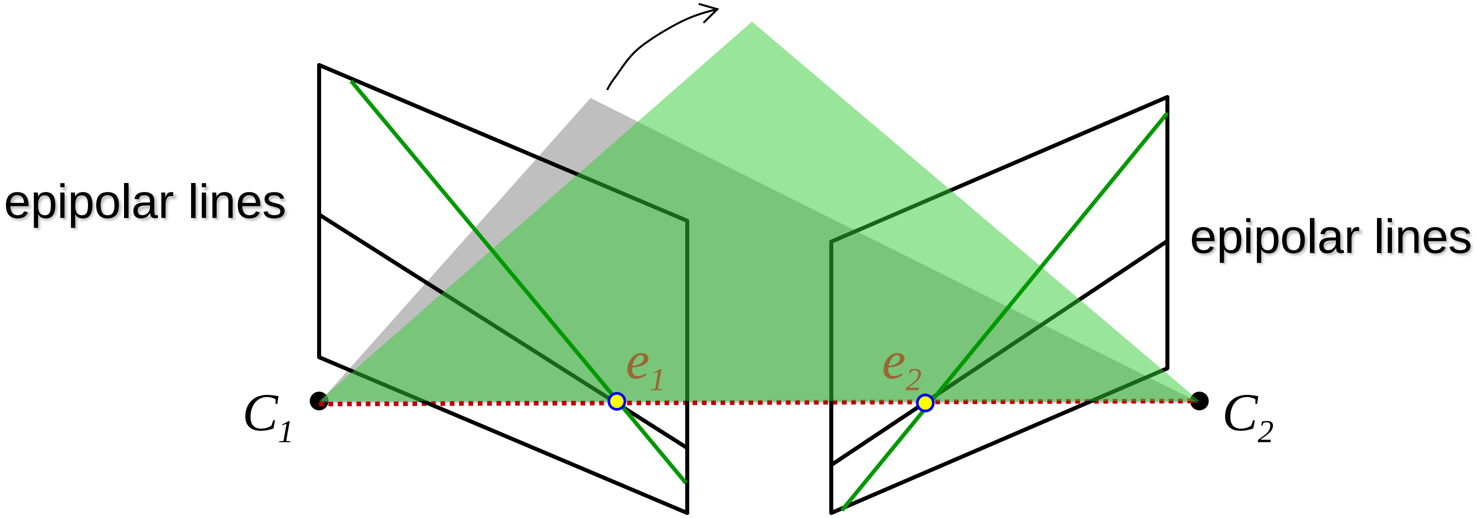
**epipoles**
(points where *base line* $C_1 C_2$ intersects two image planes)

**epipolar constraint for the right image**: for any point $p_1$ in the left image, the corresponding point in the right image must be on the line where plane $p_1 C_1 C_2$ intersects the right image (right image *epipolar line*)

- reduces correspondence problem to 1D search along conjugate *epipolar lines*

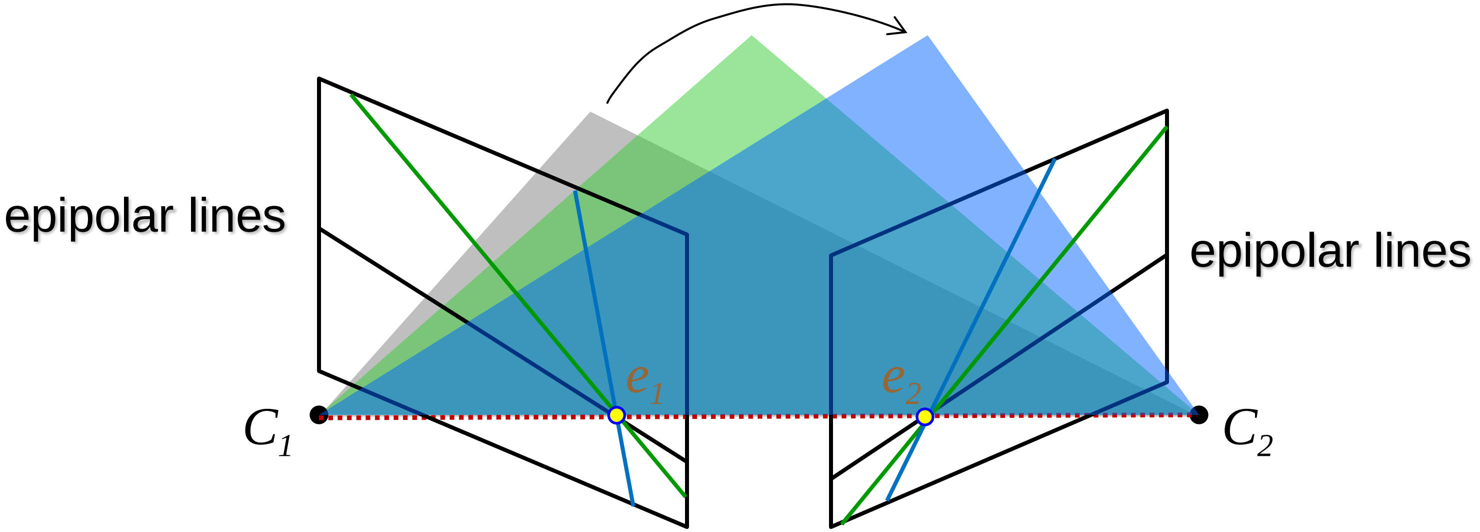
# Epipolar lines

**System of corresponding epipolar lines depends only on camera set up and it does not depend on 3D scene.**



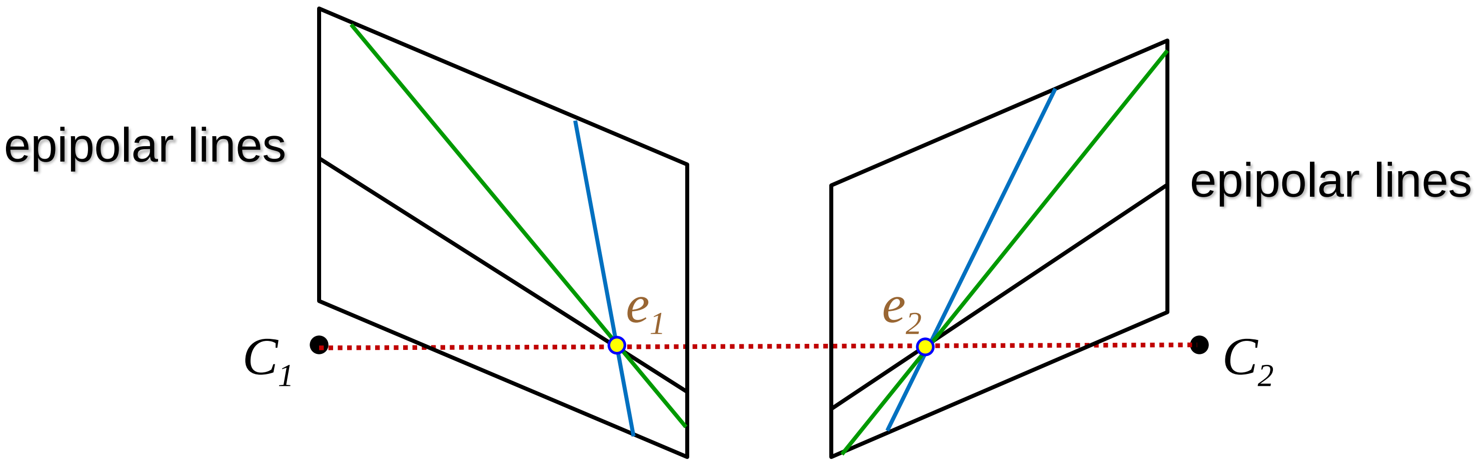
# Epipolar lines

**System of corresponding epipolar lines depends only on camera set up and it does not depend on 3D scene.**



- Intersection of **epipolar planes** (planes containing baseline $C_1C_2$ ) with image planes define a system of corresponding *epipolar lines*
- Corresponding points can be only on corresponding epipolar lines
  - important to know such lines when searching for corresponding pairs of points

# Epipolar lines



- **How can we compute epipolar lines for a given pair of images?**

- if known, camera projection matrices $P_1$ and $P_2$ contain all information

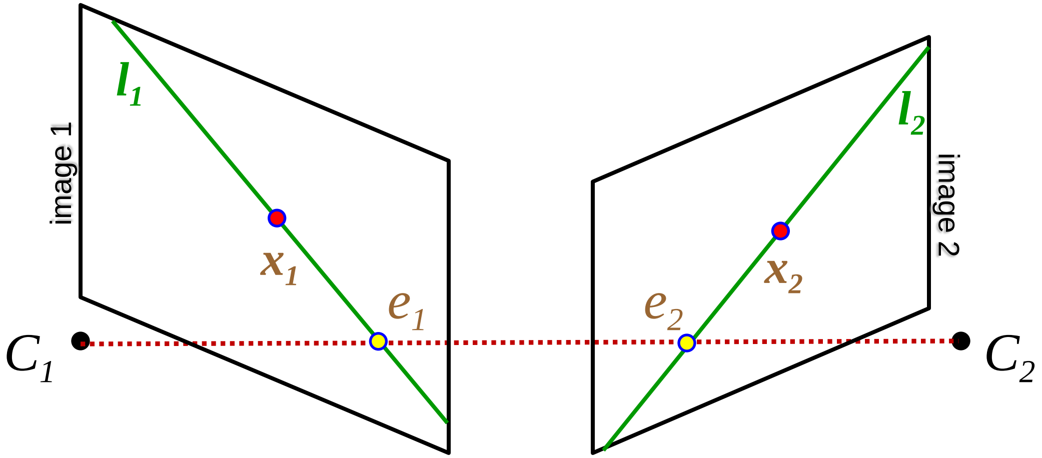
$$e_1 = P_1 C_2 \qquad e_2 = P_2 C_1 \qquad x_1 = P_1 X \qquad x_2 = P_2 X \qquad (X \text{ – any 3D point})$$

- but only relative position of two cameras really matters:

can estimate a single 3x3 ***essential matrix*** rather than two 3x4 matrices $P = (R|T)$ …

# Essential matrix $E$ (definition)

The system of corresponding epipolar lines
is fully described by a 3x3 matrix $E$ in equation below



$$\rlap{$\underbrace{\phantom{x_2^T E}}_{(l_1)^T}$}x_2^T\,\rlap{$\underbrace{\phantom{E x_1}}_{l_2}$}\overset{\text{3x3 matrix}}{E} x_1 = 0$$

for any pair of pixels/points $\mathbf{x}_1$ and $\mathbf{x}_2$
on the corresponding epipolar lines
(assuming calibrated cameras)

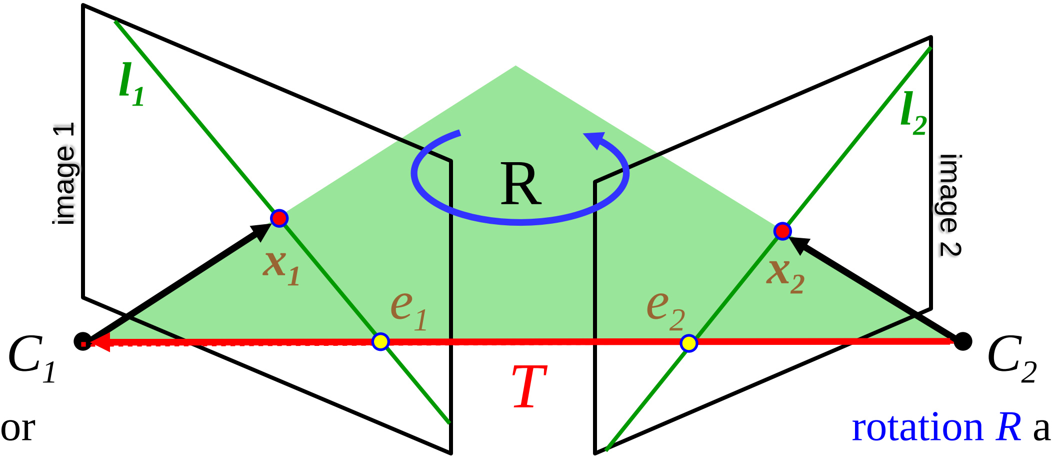
NOTE: given $\mathbf{x}_1$ in image 1 vector $\mathbf{l}_2 = E\mathbf{x}_1$ gives equation $\mathbf{x}_2 \cdot \mathbf{l}_2 = 0$ (a line in image 2)

given $\mathbf{x}_2$ in image 2 vector $\mathbf{l}_1 = E^T\mathbf{x}_2$ gives equation $\mathbf{x}_1 \cdot \mathbf{l}_1 = 0$ (a line in image 1)

# Essential matrix $E$ (proof of existence)

**Recall:** assuming calibrated cameras, pixels $x_1$ and $x_2$ in (homogeneous) image coordinates can be treated as **3D points (vectors)** in the corresponding camera-centered coordinates of 3D space

$$P_1 = [\, I \mid 0 \,]$$



$$P_2 = [\, R \mid T \,]$$

use camera 1 for “**world coordinates**”

rotation $R$ and translation $T$ convert camera 1 (world) coordinates to camera 2

dot product cross product

$$\boxed{x_2 \cdot [T \times (Rx_1)] = 0}$$

for any pair of pixels/points $x_1$ and $x_2$ on the corresponding epipolar lines (assuming calibrated cameras)

**co-planarity constraint for $x_1$ and $x_2$**

treating $x_1$ and $x_2$ as vectors in $\mathbb{R}^3$

NOTE: $Rx_1$ is vector $x_1$ in camera 2 coordinates and $T\times Rx_1$ is the green plane’s normal (camera 2 coordinates)

# Essential matrix $E$ (proof of existence)

NOTE: cross product $a \times b$ can be represented as matrix multiplication

$$a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad \Rightarrow \quad a \times b = \boxed{\underbrace{\begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}}_{\text{notation: } [a]_\times}} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$\boxed{a \times b \equiv [a]_\times b}$$

3x3 *skew-symmetric* matrix, **rank 2**
(a.k.a. *antisymmetric* matrix $M = -M^T$ )

**Q:** example of a null vector for $[a]_x$ ?

dot product cross product

$$\boxed{x_2 \cdot [T \times (Rx_1)] = 0} \quad \Leftrightarrow \quad \boxed{x_2^T [T]_\times R x_1 = 0}$$

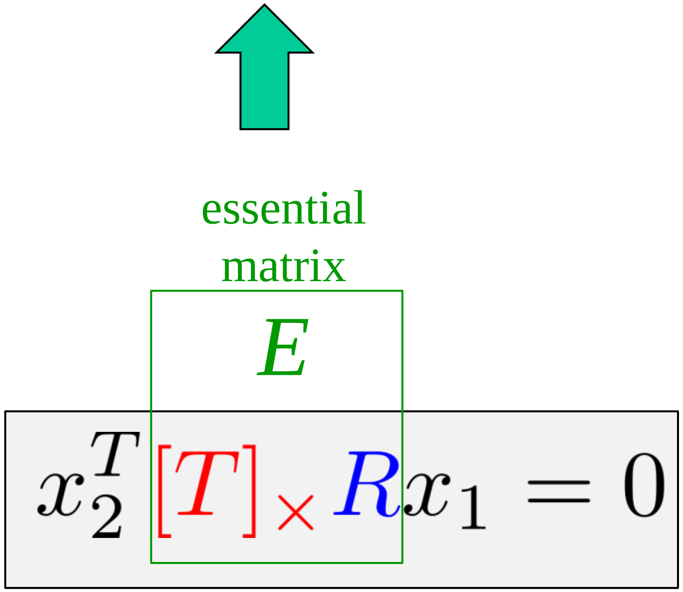
**co-planarity constraint for $x_1$ and $x_2$**
treating $x_1$ and $x_2$ as vectors in $\mathbb{R}^3$

matrix expression

# Essential matrix $E$ (proof of existence)

NOTE: due to homogeneous coordinates, scale of $E$ is arbitrary

$$\boxed{x_2^T E x_1 = 0}$$



essential matrix

$$x_2^T \overset{E}{\boxed{[T]_\times R}} x_1 = 0$$

matrix expression

Th Oct 19

# Essential matrix *E*

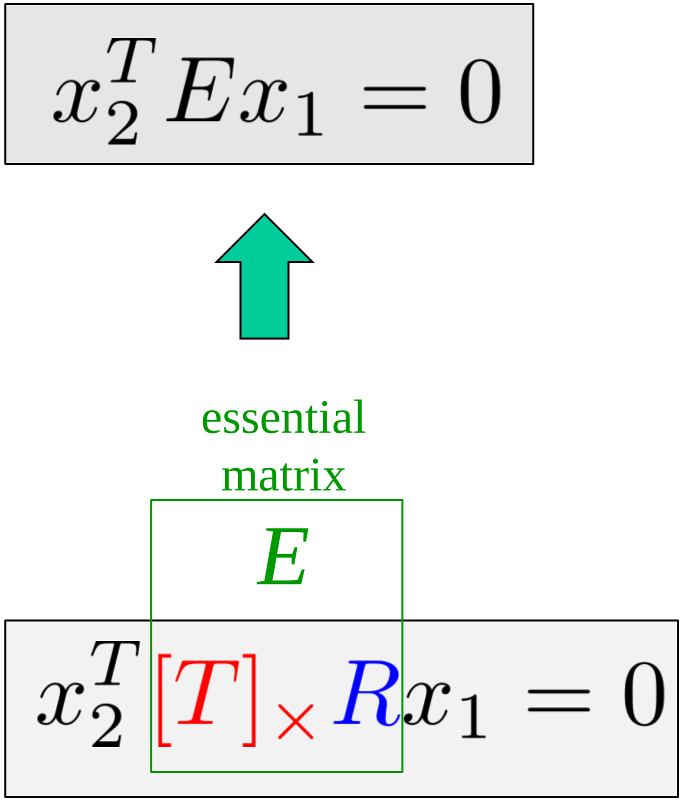
**Theorem** [***existence*** and ***uniqueness*** of essential matrix]: Assume two calibrated cameras with non-zero baseline. There exists (unique up to scale) 3x3 matrix E such that for any $X \in \mathcal{P}^3$

$$x_1^T E x_2 \;=\; 0$$

where $x_1, x_2 \in \mathcal{P}^2$ are projections of $X$ on two cameras, *i.e.* $x_i = P_i X$ for cameras' projection matrices $P_1$ and $P_2$.

**nontrivial exercise**: prove up-to-scale uniqueness of *E*

NOTE: due to homogeneous coordinates, scale of *E* is arbitrary



$$x_2^T E x_1 = 0$$

essential matrix

$$x_2^T \underbrace{[T]_\times R}_{E} x_1 = 0$$

matrix expression

COMMENT: In practice (as discussed later) *E* is estimated from observed projections of a given (finite) cloud of 3D points $\{X\}$ onto two cameras (that is, a given set of matched pairs of pixels in two images $\{(x_1,x_2)\}$. One can construct certain ***critical configurations*** of point cloud $\{X\}$ and camera positions allowing **multiple essential matrices** *E* such that $x_1^T E x_2 \;\approx\; 0$ for all given matched pairs. Examples: entire point cloud {X} and camera centers lie on one plane or on a cone, see [Kahl & Hartley, "Critical Curves and Surfaces for Euclidean Reconstruction", ECCV'02,]

For general 3D scenes and camera positions, critical configurations are **unlikely** to happen in practice, particularly when sets of matched pairs of pixels are sufficiently large.

# Essential matrix $E$

**Theorem** [*existence* and *uniqueness* of essential matrix]: Assume two calibrated cameras with non-zero baseline. There exists (unique up to scale) 3x3 matrix E such that for any $X \in \mathcal{P}^3$

$$x_1^T E x_2 \;=\; 0$$

where $x_1, x_2 \in \mathcal{P}^2$ are projections of $X$ on two cameras, *i.e.* $x_i = P_i X$ for cameras' projection matrices $P_1$ and $P_2$.

**nontrivial exercise**: prove up-to-scale uniqueness of $E$

NOTE: due to homogeneous coordinates, scale of $E$ is arbitrary

$$\boxed{x_2^T E x_1 = 0}$$

essential matrix

$$\boxed{x_2^T \overbrace{[T]_\times R}^{E} x_1 = 0}$$

matrix expression

$E$ is defined by a relative position of two cameras ($R$ and $T$), as expected

$$E = [T]_\times R$$

**Q**: How many *d.o.f* in $E$ ?

**A**: 5 = 3 (rotation) + 3-1 (**scale of *T* is arbitrary**)

# Essential matrix $E$

**Theorem** [*existence* and *uniqueness* of essential matrix]: Assume two calibrated cameras with non-zero baseline. There exists (unique up to scale) 3x3 matrix E such that for any $X \in \mathcal{P}^3$

$$x_1^T E x_2 \;=\; 0$$

where $x_1, x_2 \in \mathcal{P}^2$ are projections of $X$ on two cameras, *i.e.* $x_i = P_i X$ for cameras' projection matrices $P_1$ and $P_2$.

**nontrivial exercise**: prove up-to-scale uniqueness of $E$

NOTE: due to homogeneous coordinates, scale of $E$ is arbitrary

$$\boxed{x_2^T E x_1 = 0}$$

essential matrix

$$x_2^T \overbrace{[T]_\times R}^{E} x_1 = 0$$

matrix expression

$E$ is defined by a relative position of two cameras ($R$ and $T$), as expected

$$E = [T]_\times R$$

**Q**: What is the rank of $E$ ?

# Fundamental matrix *F*

**Question:** are there epipolar lines in **uncalibrated cameras**?

**Answer:** baseline, epipoles, epipolar planes & lines exist due to 3D geometry.

(camera normalization or specific image coordinate system can only change their representation)

$$x_2^T E x_1 = 0$$

assuming calibrated camera coordinates

**Remember**: $$\tilde{x} = K^{-1} x$$

calibrated (normalized) coordinates

original image coordinates

=> $$\overbrace{x_2^T K^{-T}}^{\tilde{x}_2^T} E \overbrace{K^{-1} x_1}^{\tilde{x}_1} = 0$$

$$x_2^T \underbrace{K^{-T} E K^{-1}}_{F} x_1 = 0$$

$F$- fundamental matrix

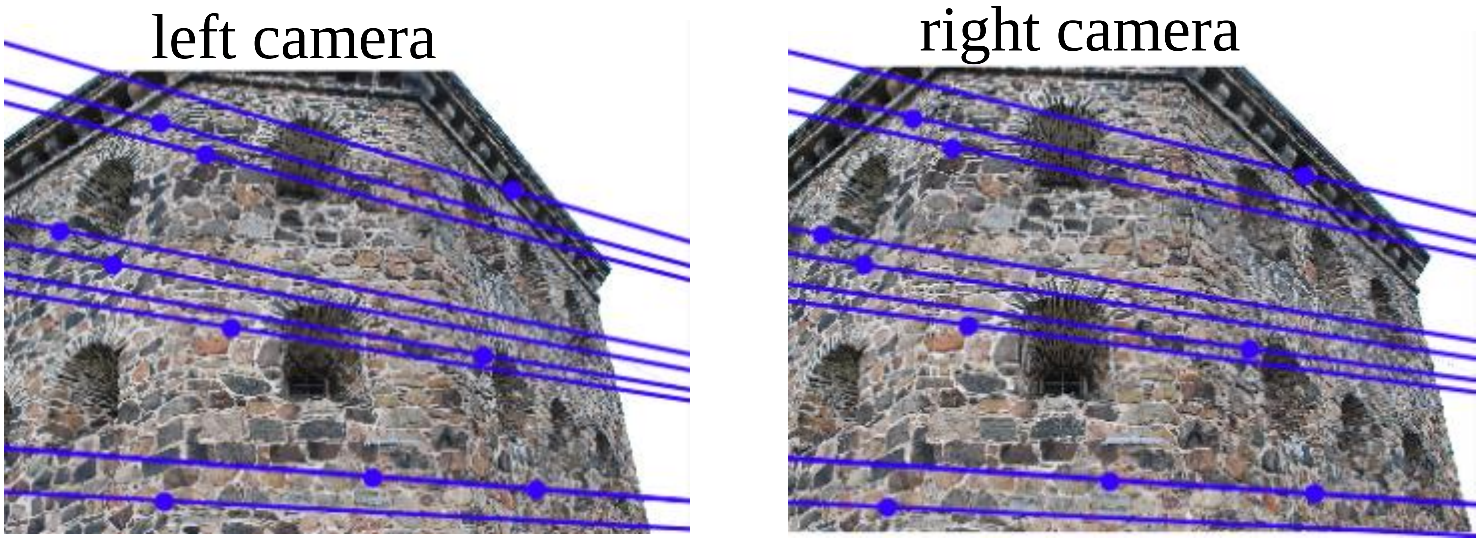
$$x_2^T F x_1 = 0$$

**defines epipolar lines for uncalibrated cameras**

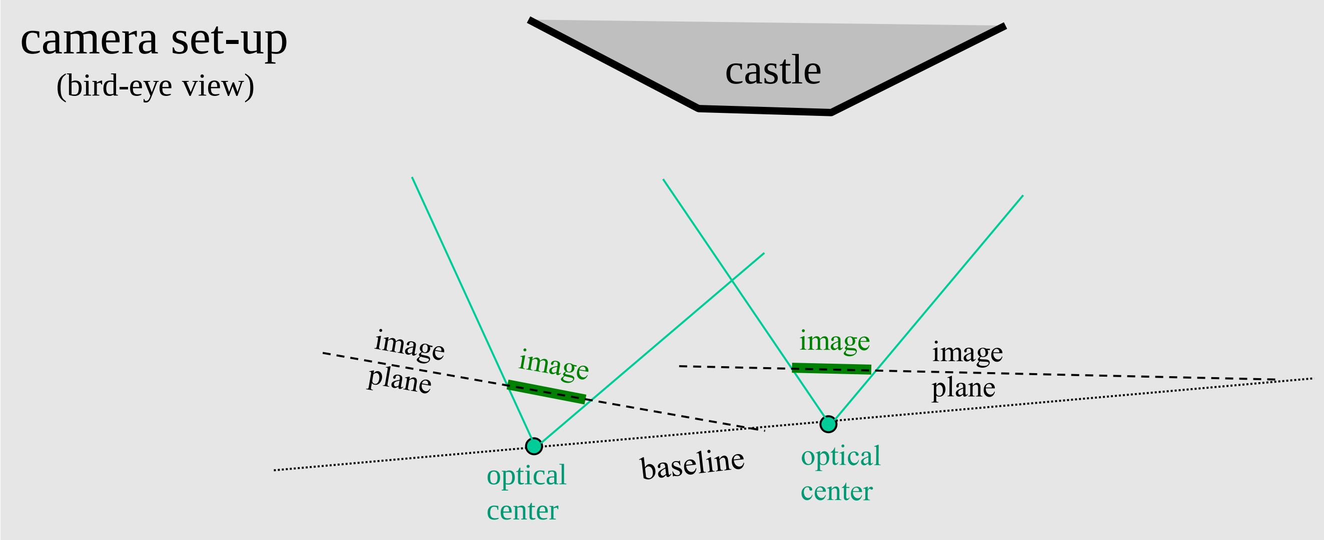
# Example

NOTE: epipoles do not have to be within image box.

In fact, they can be points at infinity (parallel epipolar lines, see later)



the blue dots in two images are representative pairs of matched features



# Estimating $F$ or $E$ from $N \geq 8$ matches

## 8-point method

Assume corresponding points $\mathbf{x}_i \leftrightarrow \bar{\mathbf{x}}_i$ in two images

(matched pair corresponding to a projection of unknown 3D point $X_i$ )

They must lie on the corresponding epipolar lines, thus

$$\bar{\mathbf{x}}_i^T F \mathbf{x}_i = 0$$

(use $E$ for calibrated images)

If $\mathbf{x}_i = (x_i, y_i, z_i)$ and $\bar{\mathbf{x}}_i = (\bar{x}_i, \bar{y}_i, \bar{z}_i)$ then

$$\begin{aligned} \bar{\mathbf{x}}_i^T F \mathbf{x}_i \quad = \quad & F_{11}\bar{x}_i x_i + F_{12}\bar{x}_i y_i + F_{13}\bar{x}_i z_i \\ + \; & F_{21}\bar{y}_i x_i + F_{22}\bar{y}_i y_i + F_{23}\bar{y}_i z_i \\ + \; & F_{31}\bar{z}_i x_i + F_{32}\bar{z}_i y_i + F_{33}\bar{z}_i z_i \quad = \quad 0 \end{aligned}$$

One matching pair $\mathbf{x}_i \leftrightarrow \bar{\mathbf{x}}_i$ gives **only one linear equation**.

**Eight** is enough to determine elements of 3x3 matrix $F$ (as scale is arbitrary)

Note: enforcing known properties (e.g. rank=2) allows to use fewer points.

# Estimating $F$ or $E$ from $N \geq 8$ matches

**In matrix form**: one row for each of $N \geq 8$ correspondences

$$\underbrace{\overset{\text{Nx9}}{\begin{bmatrix} \bar{x}_1 x_1 & \bar{x}_1 y_1 & \bar{x}_1 z_1 & \cdots & \bar{z}_1 z_1 \\ \bar{x}_2 x_2 & \bar{x}_2 y_2 & \bar{x}_2 z_2 & \cdots & \bar{z}_2 z_2 \\ \bar{x}_3 x_3 & \bar{x}_3 y_3 & \bar{x}_3 z_3 & \cdots & \bar{z}_3 z_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{x}_N x_N & \bar{x}_N y_N & \bar{x}_N z_N & \cdots & \bar{z}_N z_N \end{bmatrix}}}_{\mathbf{A}} \underset{\mathbf{f}}{\overset{\text{9x1}}{\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ \vdots \\ F_{33} \end{bmatrix}}} = \underset{\mathbf{0}}{\overset{\text{Nx1}}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}}$$

➡ $$\boxed{\mathbf{A}\,\mathbf{f} = \mathbf{0}}$$

If matched points
have some errors
(not exact locations) ?

# Estimating $F$ or $E$ from $N \geq 8$ matches

solve *homogeneous least squares*

$$\boxed{\min_{\|\mathbf{f}\|=1} \|\mathbf{A}\mathbf{f}\|}$$

as in homography estimation,
constraint $\|\mathbf{f}\|=1$ fixes the scale of **f** (i.e. $F$)

$$\begin{bmatrix} E_{11} \\ E_{12} \\ E_{13} \\ \vdots \\ E_{33} \end{bmatrix}$$

for $E$ use **e** instead of **f**

Use eigen vector for the smallest eigen value of 9x9 matrix $\mathbf{A}^T\mathbf{A}$

Need $N \geq 8$ to get a unique minimizer **f** (up to sign, –**f** also works).

If $N = 8$ then perfect (zero) least squares loss is achieved at a unique solution (up to sign).
- *dim*(**A**)=8x9, *rank*(**A**)=8, and **f** is (unit norm) right null vector of **A**

If $N \leq 7$ then the problem is under-constrained. The (right) null space of **A** has dimension $\geq 2$ and there are many unit norm solutions **f** achieving zero loss.

# What's left to cover

- **More on Estimation of *E* and *F***
  - properties of E and F
  - limitations of **8-point method** (no enforcement of rank or other constraints for *E* or *F*)
  - more advanced **5-point method** (see H&Z book, we do not cover this in class)
  - similarly to homography estimation in previous topics, we cover only least squares for *algebraic* errors (*reprojection* errors use more advanced optimization)
- Extraction of cameras pose (projection matrices) from *E*
- Structure from Motion
  - **triangulate** (estimate structure)
  - bundle adjustment
  - reconstruction ambiguities

# Properties of $E$ and $F$

Note that *rank* (*E*) =2 since $E = [T]_\times R$ and $rank([T]_\times) = 2$

**Theorem**: 3x3 matrix *E* is *essential* ($\exists R, T : E = [T]_\times R$) if and only if two of its singular values are equal, and the third is zero.

[H&Z. Sec 9.6 p.257]

Then, SVD for essential matrix is

(scale ambiguity allows to use 1 for singular values)

$$\boxed{E = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T}$$

For the fundamental matrix $F$ we only have *rank* (*F*) =2 constraint, while two non-zero singular values of *F* can be different.
Of course, the third singular value is still zero.

# Summary of properties

| essential matrix $E$ | fundamental matrix $F$ |
|---|---|
| • epipolar lines $x_2^T E x_1 = 0$<br>(for two calibrated cameras) | • epipolar lines $x_2^T F x_1 = 0$<br>(for two arbitrary cameras) |
| • rank 2 $E = [T]_\times R$ | • rank 2 $F = K^{-T} E K^{-1}$ |
| • epipoles $e_1$ and $e_2$ are right and left null vectors for $E$<br>$Ee_1 = \mathbf{0}$ $e_2^T E = \mathbf{0}^T$ | • epipoles $e_1$ and $e_2$ are right and left null vectors for $F$<br>$Fe_1 = \mathbf{0}$ $e_2^T F = \mathbf{0}^T$ |
| • 5 d.o.f (6 from *R&T*, - scale of T) | • 7 d.o.f (9 par., - scale & det *F*=0) |
| • two equal non-zero singular values | • two non-zero singular values |

# Limitations of 8-point method

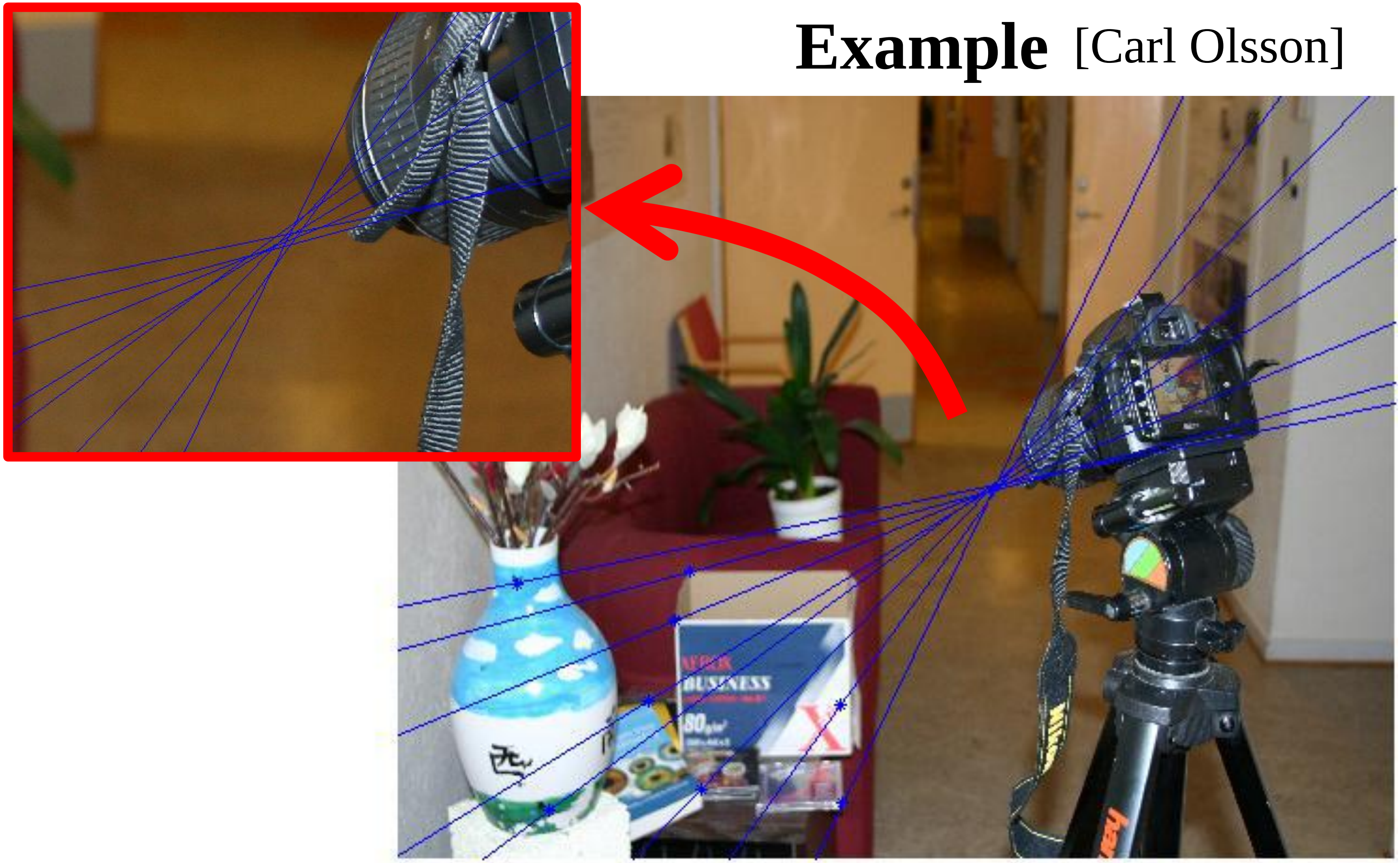
*homogeneous least squares*

$$\boxed{\min_{\|\mathbf{f}\|=1} \|\mathbf{A}\mathbf{f}\|}$$

**Issue**: optimal $F$ may not satisfy $\det(F)=0$ and $\mathrm{rank}(F)=2$.

# Limitations of 8-point method

**Example** [Carl Olsson]



**Epipole is not well defined if rank is not constrained to 2**

# Naïve solution

*homogeneous least squares*

$$\boxed{\min_{\|\mathbf{f}\|=1} \|\mathbf{A}\mathbf{f}\|}$$

**Issue**: optimal $F$ may not satisfy det($F$)=0 and rank($F$)=2.

**One "solution"**: find the "closest" rank 2 matrix $\tilde{F}$ s.t.

$$\boxed{\min_{rank(\tilde{F})=2} \|\tilde{F} - F\|}$$

where $\|\tilde{F} - F\| := \sqrt{\sum_{ij} (\tilde{F}_{ij} - F_{ij})^2}$

*Frobenius norm*

# Naïve solution

**Theorem** (*low rank approximation*) [Eckart-Young-Mirsky]:

Assuming SVD for mxn matrix $A = U\ diag(s_1, s_2, ..., s_n)\ V^T$

$\min\limits_{rank(\tilde{A})=k} \|\tilde{A} - A\|$ is solved by $\tilde{A} = U\ diag(\underbrace{s_1, .., s_k,}_{k \text{ largest singular values of } A} 0, .., 0)\ V^T$

the minimizer is unique iff $s_{k+1} \neq s_k$

**Issue**: optimal $F$ may not satisfy det($F$)=0 and rank($F$)=2.

**One "solution"**: find the "closest" rank 2 matrix $\tilde{F}$ s.t.

$$\boxed{\min\limits_{rank(\tilde{F})=2} \|\tilde{F} - F\|}$$

where $\underbrace{\|\tilde{F} - F\|}_{\text{Frobenius norm}} := \sqrt{\sum_{ij}(\tilde{F}_{ij} - F_{ij})^2}$

# Naïve solution

**Theorem** (*low rank approximation*) [Eckart-Young-Mirsky]:

Assuming SVD for mxn matrix $A = U\ diag(s_1, s_2, ..., s_n)\ V^T$

$$\min_{rank(\tilde{A})=k} \|\tilde{A} - A\| \text{ is solved by } \tilde{A} = U\ diag(\underbrace{s_1, .., s_k,}_{k \text{ largest singular values of } A} 0, .., 0)\ V^T$$

the minimizer is unique iff $s_{k+1} \neq s_k$

**Issue**: optimal $F$ may <u>not</u> satisfy det($F$)=0 and rank($F$)=2.

**One "solution"**: find the "closest" rank 2 matrix $\tilde{F}$ s.t.

$$\boxed{\min_{rank(\tilde{F})=2} \|\tilde{F} - F\|} \Longrightarrow \boxed{\tilde{F} = U \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T}$$

# Naïve solution

If point matches $\mathbf{x}_i \leftrightarrow \bar{\mathbf{x}}_i$ are in normalized camera images, solve *homogeneous least squares*

$$\boxed{\min_{\|\mathbf{e}\|=1} \|\mathbf{A e}\|}$$

$$\begin{bmatrix} E_{11} \\ E_{12} \\ E_{13} \\ \vdots \\ E_{33} \end{bmatrix}$$
$$\mathbf{e}$$

**Issue**: optimal $E$ may not have SVD $E = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$.

**One "solution"**: find the "closest" essential matrix $\tilde{E}$

[H&Z, Sec.11.7.3, p.294]

$$\min_{s_1=s_2,\ s_3=0} \|\tilde{E} - E\| \quad \Rightarrow \quad \boxed{\tilde{E} = U \begin{bmatrix} \frac{s_1+s_2}{2} & 0 & 0 \\ 0 & \frac{s_1+s_2}{2} & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T}$$

# Much better solution

use **5-point method** (see H&Z book)

- a more principled approach directly enforcing the necessary constraints on matrix *E* during estimation, rather than trying to fix the issue by “post-processing” the problematic matrix.
- not covered in CS484/684

# The Fundamental Matrix Song

Go here if video does not play automatically:
https://www.youtube.com/watch?v=DgGV3l82NTk&feature=emb_logo

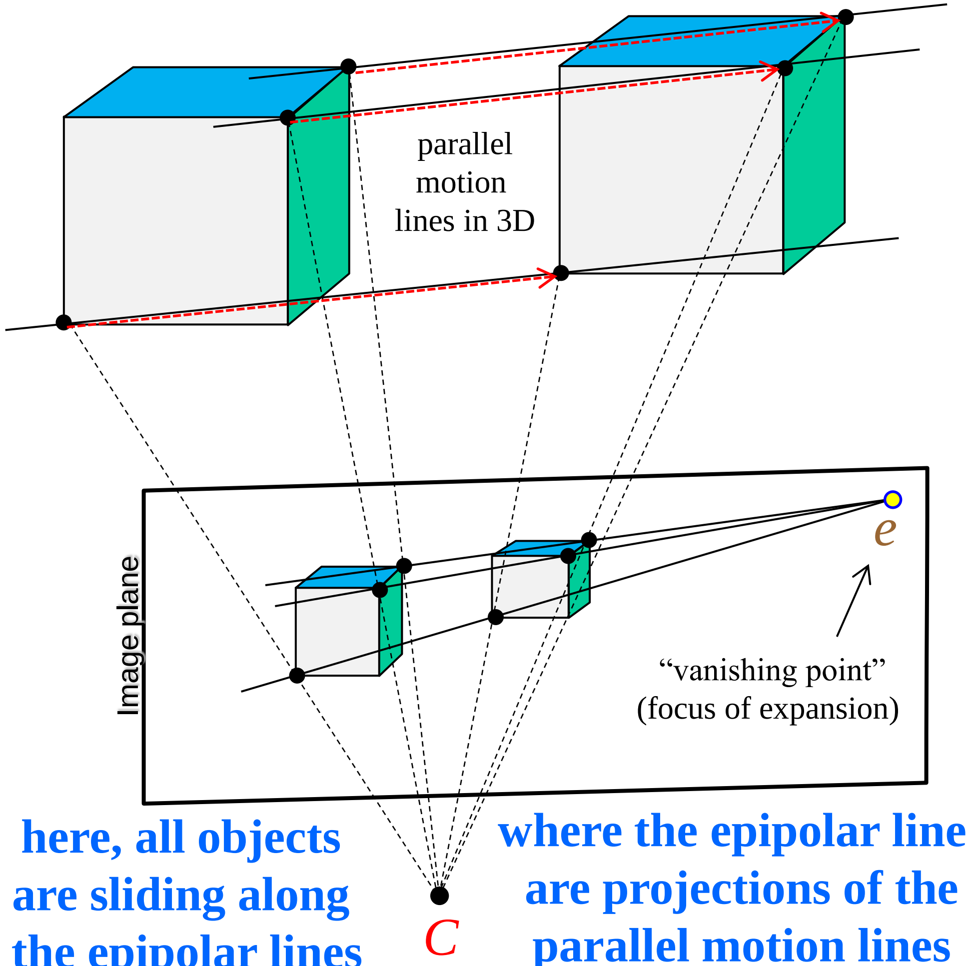
# What's left to cover

- **More on Estimation of *E* and *F***
  - properties of E and F
  - limitations of **8-point method** (no enforcement of rank or other constraints for *E* or *F*)
  - more advanced **5-point method** (see H&Z book, we do not cover this in class)
  - similarly to homography estimation in previous topics, we cover only least squares for *algebraic* errors (*reprojection* errors use more advanced optimization)
- **Extraction of camera pose (projection matrices) from *E***
- **Structure from Motion**
  - **triangulate** (estimate structure)
  - bundle adjustment
  - reconstruction ambiguities

# Motion and Epipolar lines

## *example*: camera translation

**equivalently,** assume a stationary camera, but all 3D scene points translate by vector $C_1C_2$

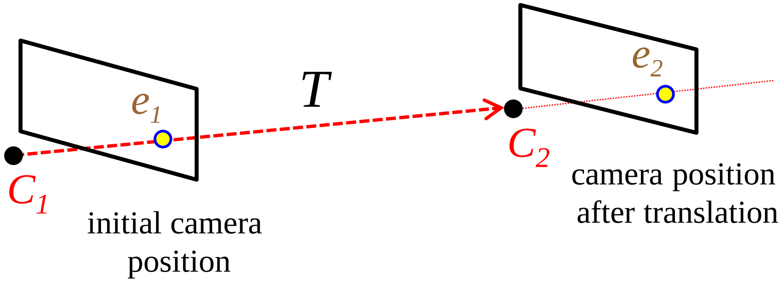


**here, all objects are sliding along the epipolar lines**

**where the epipolar lines are projections of the parallel motion lines**

The epipole is the projection of the point at infinity for all parallel motion lines in 3D

- assume **no camera rotation** $E = [T]_\times$
- vector of camera translation is the same as the base line $T = C_1C_2$



- **epipoles and epipolar lines are identical in both images**

*hint*: $E = [T]_\times$ is antisymmetric

*e.g.* equal left/right null vectors $e_1 = e_2 \sim T$

*note*: easy to estimate $T$ from $E$ (up to scale)

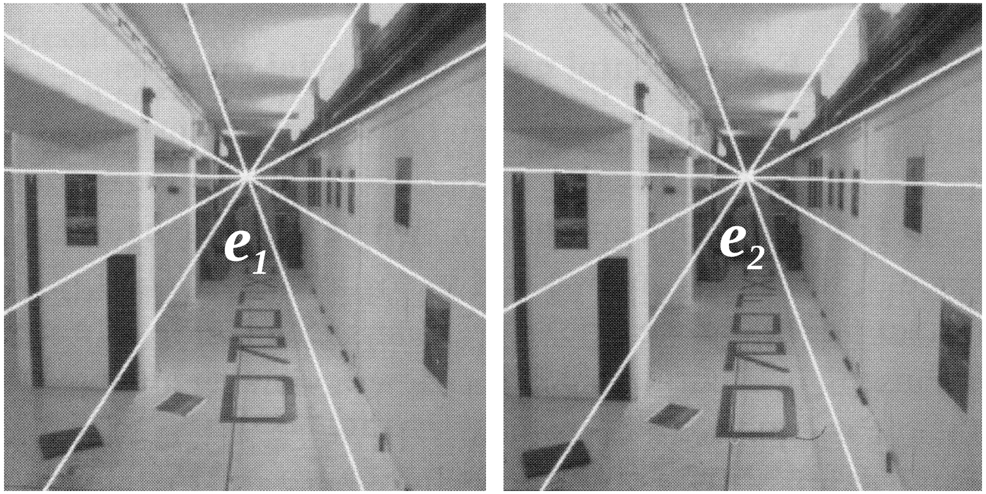
$$e \sim T = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \Leftrightarrow E = [T]_\times = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix}$$

# Motion and Epipolar lines

## *example*: camera translation

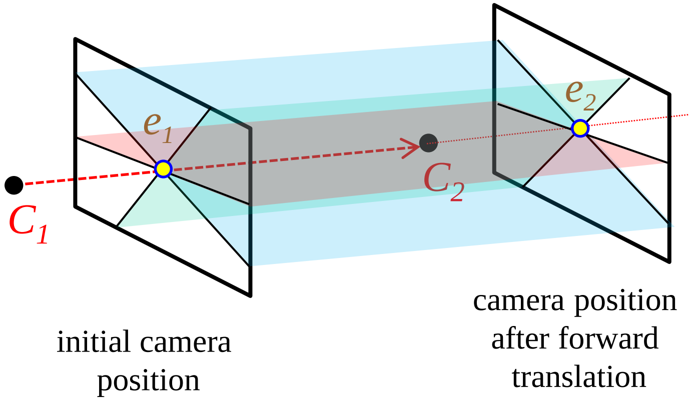
**Example**: **forward** camera motion

(i.e. along the optical axis)



(images from H&Z p.248)

note how objects slide along the epipolar lines



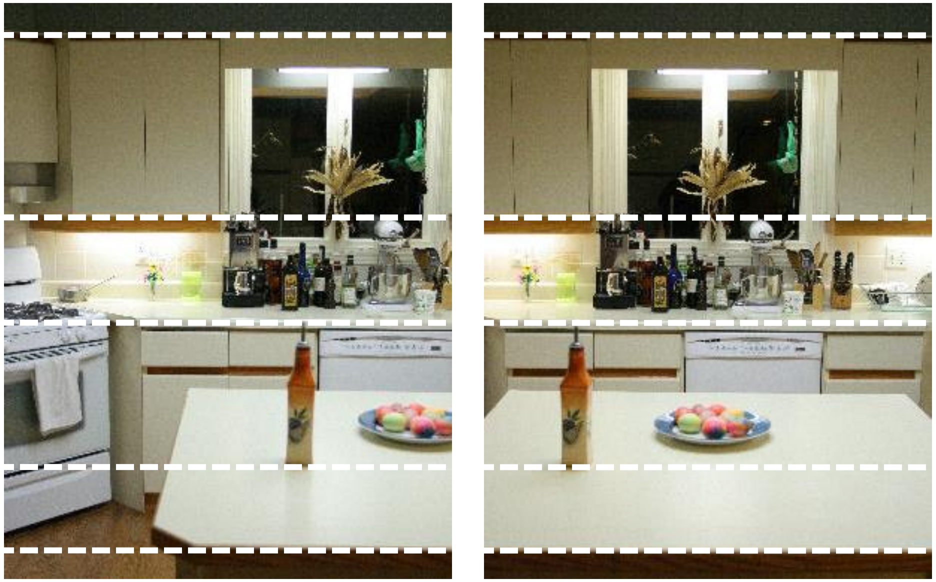
- epipole $e_1 = e_2 = e \sim T$ is at the image center where the camera's optical axis intersects the image plane

$$e = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \Rightarrow \quad E = [e]_\times = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

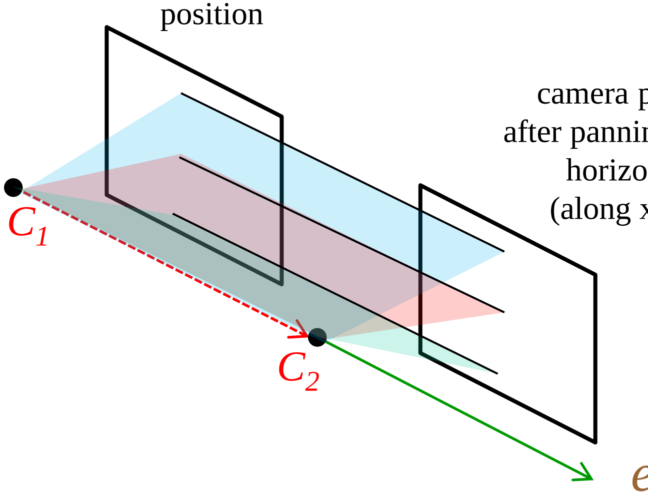
# Motion and Epipolar lines

## *example*: camera translation

**Example**: panning camera motion



note how objects slide along the epipolar lines



- epipole $e_1 = e_2 = e \sim T$ is a point at infinity for the image pane
- epipolar lines are parallel lines $\boxed{y_1 = y_2}$

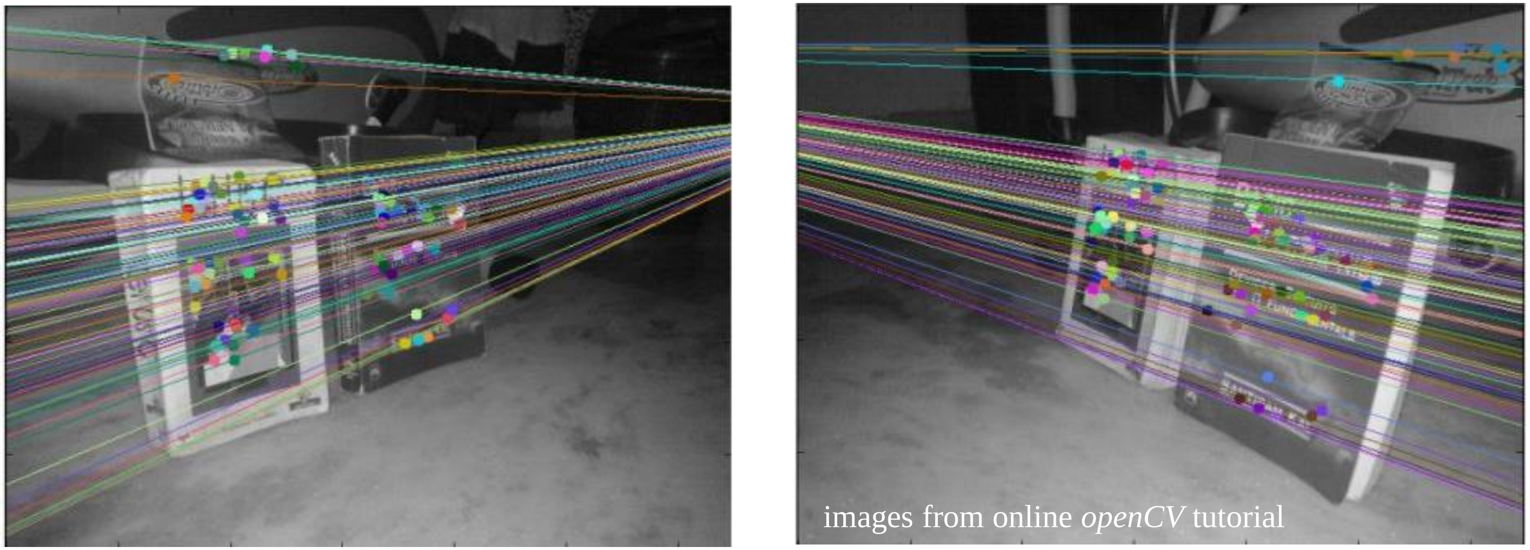
$$e = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \Rightarrow \quad E = [e]_\times = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

# Motion and Epipolar lines

*example*: camera translation and **rotation $R \neq I$**

Epipolar lines depend on both **translation** and **rotation** as $E = [T]_\times R$.

**Camera rotation is responsible for different positions of the epipoles and epipolar lines in two images** (e.g. for $e_2 \neq e_1$).



images from online *openCV* tutorial

**Question**: Is it possible to estimate motion ($R$, $T$) from $E$ ?

NOTES:

$T$ is still the left null vector ($e_2$) of $E = [T]_\times R$, e.g. since $T^\top [T]_\times = \mathbf{0}^\top$. Thus, **it is easy to find $T$**, e.g. using SVD of $E$.

The right null vector (e1) for $E$ is now $R^T$ $T$. Both T and $R^T$ $T$ represent translation, but in different cameras' coordinates.

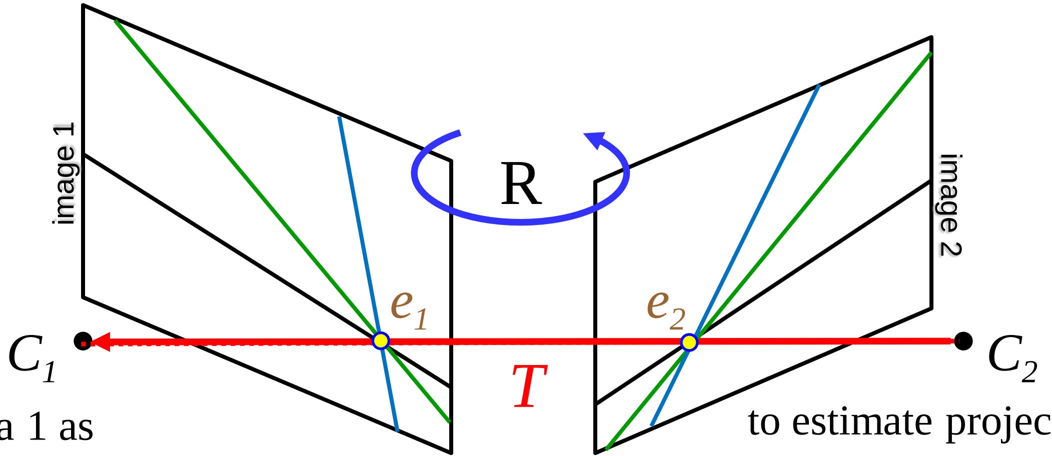
Thus, the main remaining problem is **how to recover rotation $R$** ?

Outline: treat unknown orthogonal matrix $\boldsymbol{R}$ as a (3 d.o.f.) homography aligning the epipolar lines. It is determined by $E$.

# Extracting cameras from essential matrix $E$

**Now assume essential matrix $E$ is given, need to find $P_1$ and $P_2$**

$$P_1 = [\, I \mid 0 \,]$$



$$P_2 = [\, R \mid T \,]$$

can choose camera 1 as
"**world coordinates**"

to estimate projection matrix for camera 2 need rotation $R$ and translation $T$ converting camera 1 (world) coordinates to camera 2

Given essential matrix $E = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$

find rotation $R$ and translation $T$ such that $E = [T]_\times R$

mathematical formulation of the problem

# Extracting cameras from essential matrix $E$

## **Four distinct $R$,$T$ solutions**

(up to scale)

Assume SVD decomposition $E = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$

such that $det(UV^T) = 1$ (if $det(UV^T)$ = -1 switch the sign of the last column in $V$).

Then, using special matrix $W := \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ we have

$E = [T]_\times R$ for any combination of $R = UWV^T$ or $UW^TV^T$

and $T = \pm U_3$ (scale is arbitrary)

see [H&Z:sec 9.6.2, p.258] for proof

the last column of $U$

corresponding to zero singular value

(*the left null-vector for $E$*)

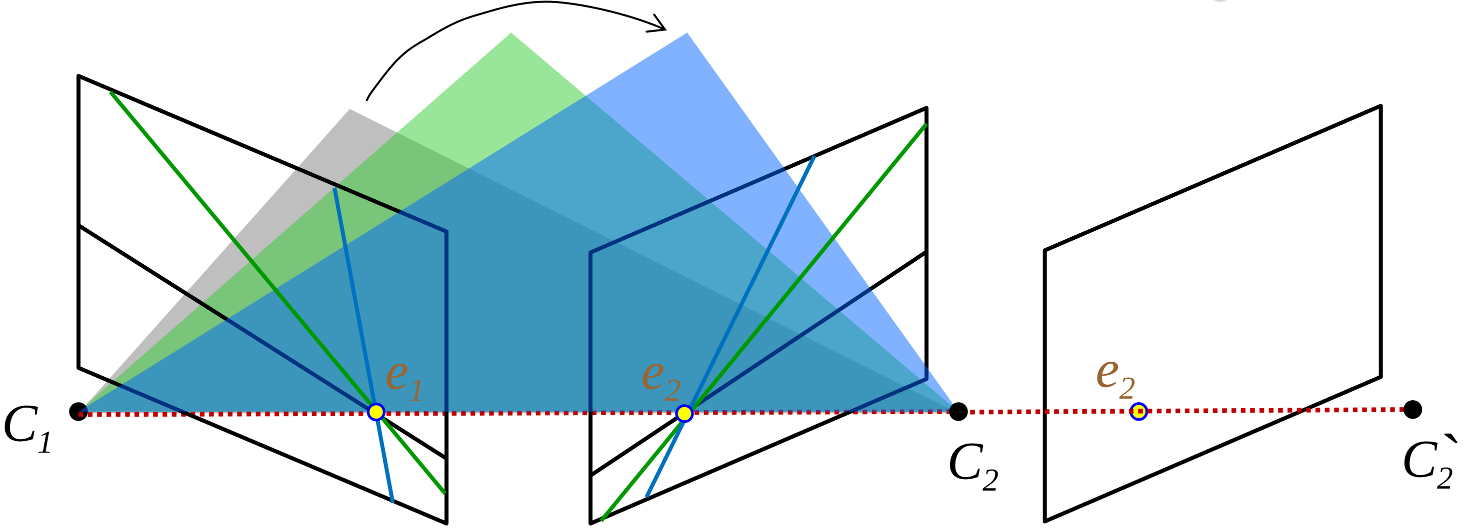
**Q: Why?**

# Extracting cameras from essential matrix $E$

## Four distinct $R$,$T$ solutions

(up to scale of T)

translate the second camera along the baseline...



$$E = [T]_\times R \text{ for any combination of } \quad R = UWV^T \text{ or } UW^TV^T$$

$$\text{and} \quad T = \pm U_3 \quad \text{(scale is arbitrary)}$$

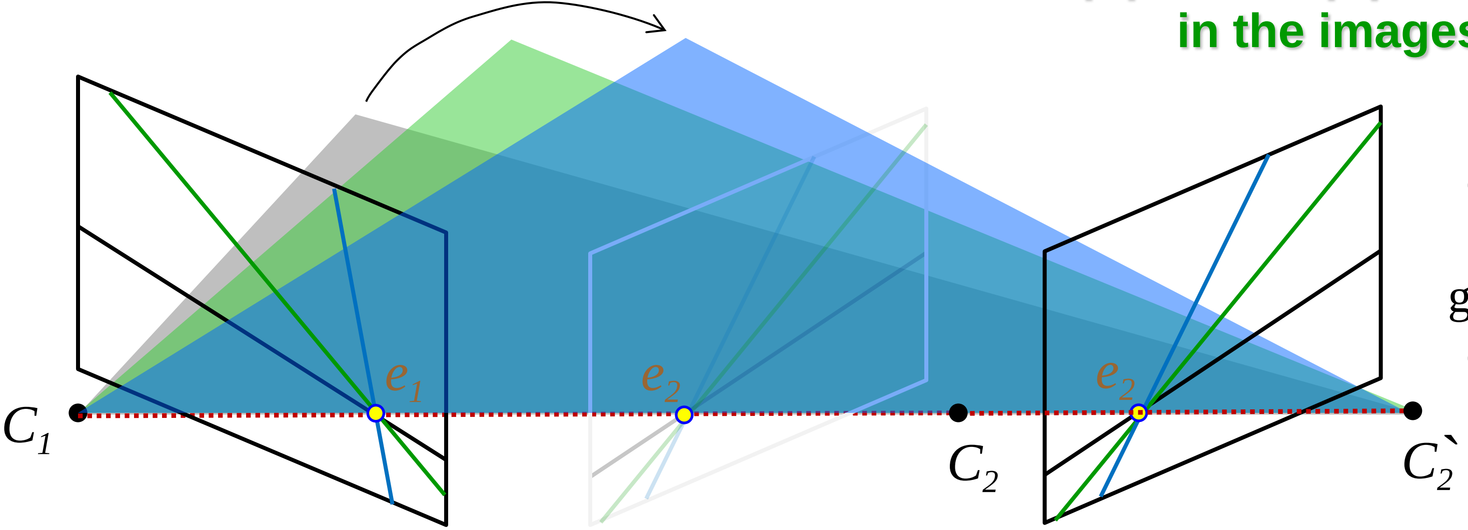
**Q: Why?**

# Extracting cameras from essential matrix $E$

## Four distinct $R$,$T$ solutions

(up to scale of T)

**baseline length |T| does not change epipole or epipolar lines in the images**



same epipolar planes give same epipolar lines

$$E = [T]_\times R \text{ for any combination of } \quad R = UWV^T \text{ or } UW^TV^T$$

$$\text{and} \quad T = \pm U_3 \quad \text{(scale is arbitrary)}$$

Moreover, **epipolar lines cannot distinguish**

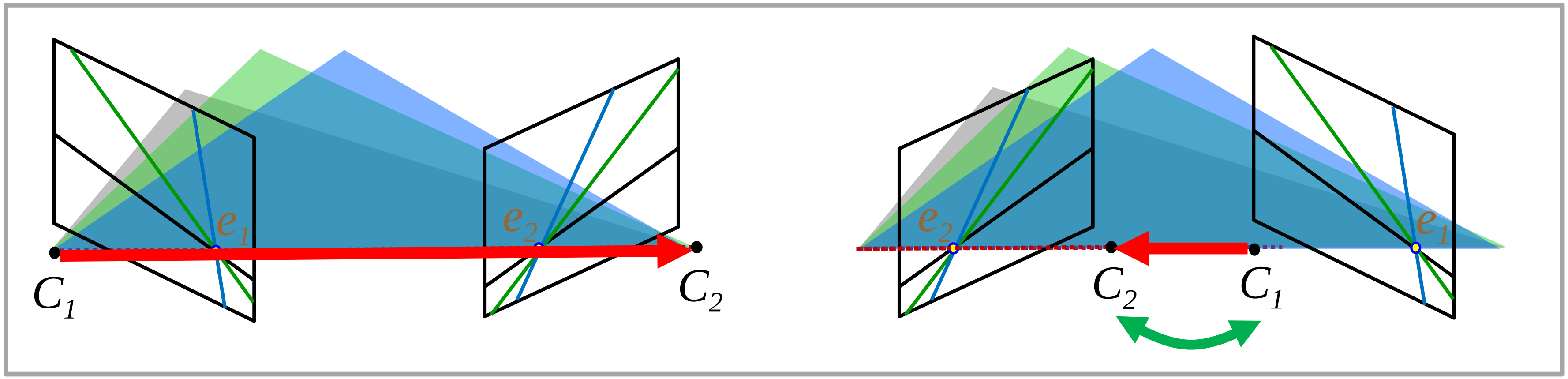
"left" vs "right" relative position of camera 2 on baseline

"front" vs "back" orientation of camera 2

**Q: Why?**

# Extracting cameras from essential matrix $E$

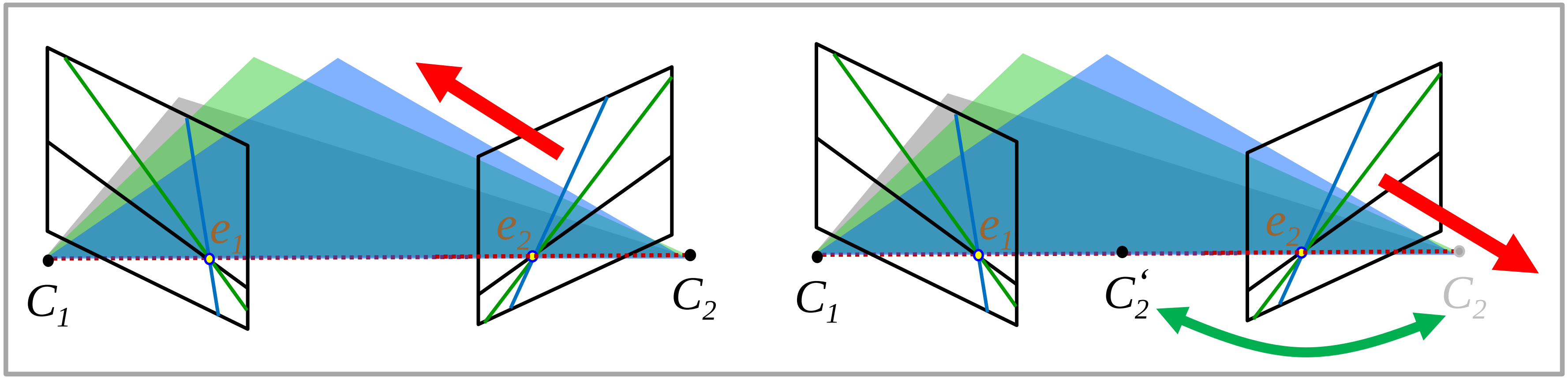
Two modifications below change viewpoints (i.e. optical centers) but **all epipolar lines** (i.e. essential matrix) **remain the same**!!!



$$T = \pm U_3$$

**order reversal**

("left-right" flip of two cameras)



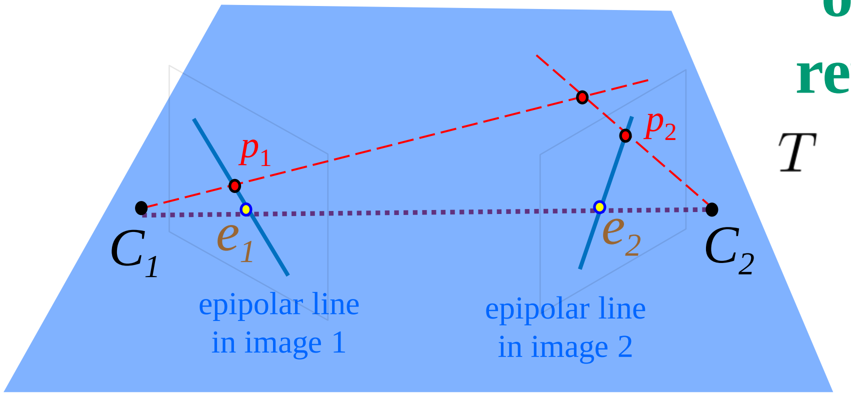
$$R = UWV^T \text{ or } UW^TV^T$$

**orientation reversal**

("front-back" flip of camera 2)

# Extracting cameras from essential matrix $E$

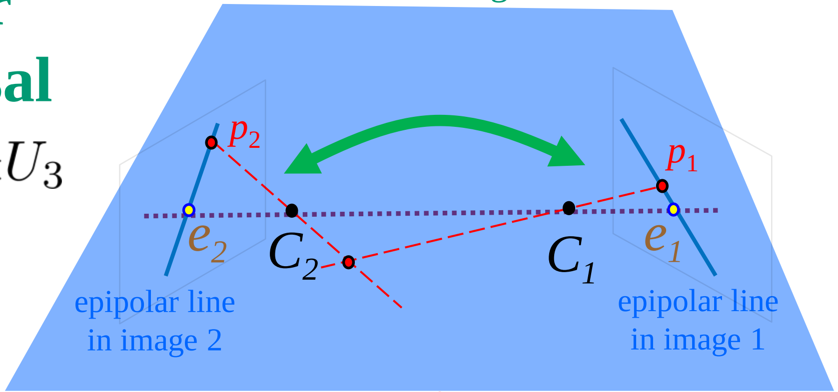
Let’s focus on some fixed baseline plane
and the corresponding epipolar lines...



**order reversal**

$$T = \pm U_3$$

“left-right” flip
of two cameras along the baseline

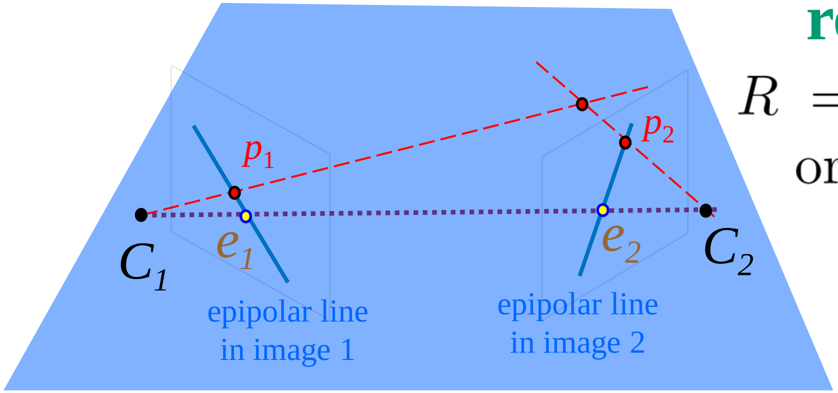


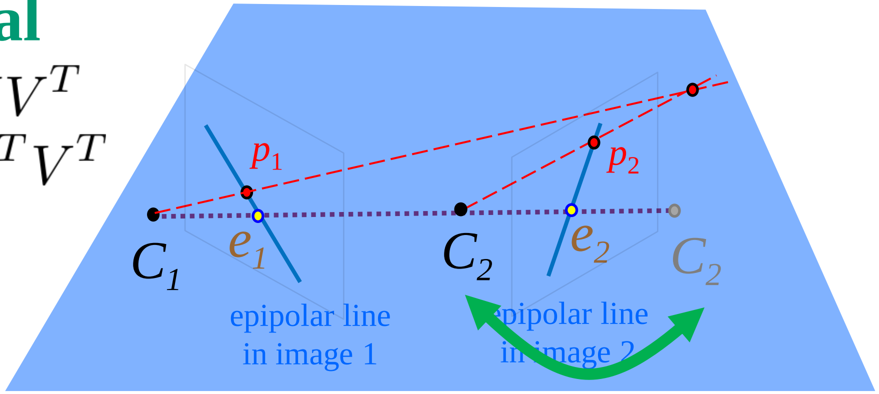
3D reconstruction flips symmetrically

**orientation reversal**

$$R = UWV^T \text{ or } UW^TV^T$$

“front-back” flip
of camera 2 viewpoint w.r.t. its image plane





3D reconstruction is fundamentally different

Two types of “viewpoint flip” can be combined => **four distinct motions** and **different reconstructions**

# Extracting cameras from essential matrix $E$

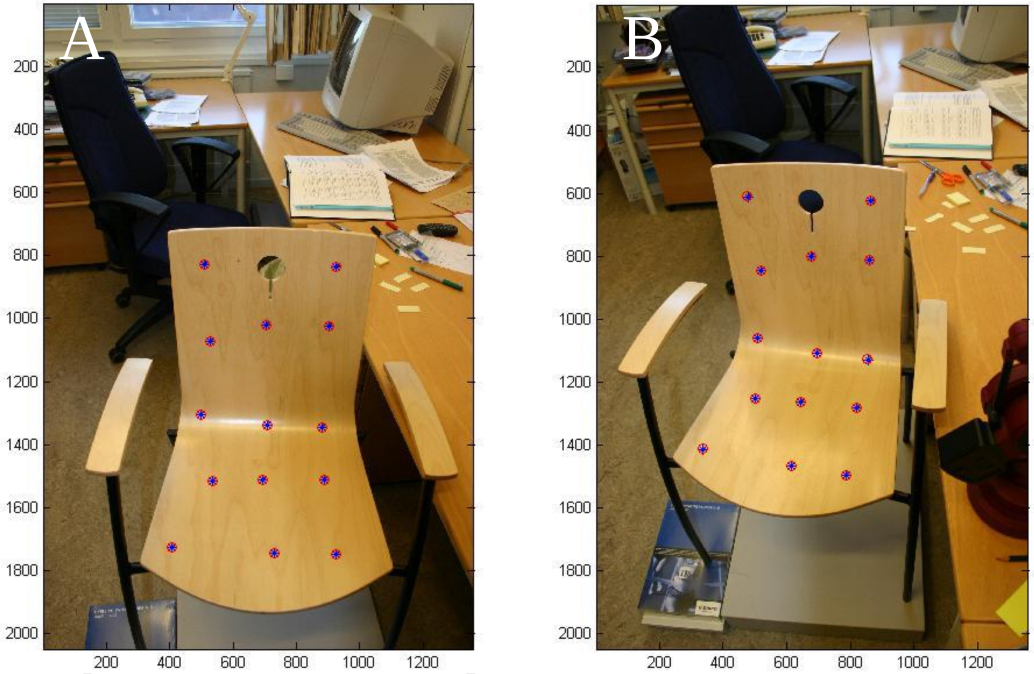
## Four distinct $R$,$T$ solutions

(up to scale of T)

**Example:**

[from Carl Olsson]

Two given views of a chair



14 known correspondences (for 14 **non-coplanar** 3D points)
allow to estimate essential matrix ***E***
assuming $K$ is known
(e.g. 8 point method)

# Extracting cameras from essential matrix $E$

## Four distinct $R$,$T$ solutions

(up to scale of T)

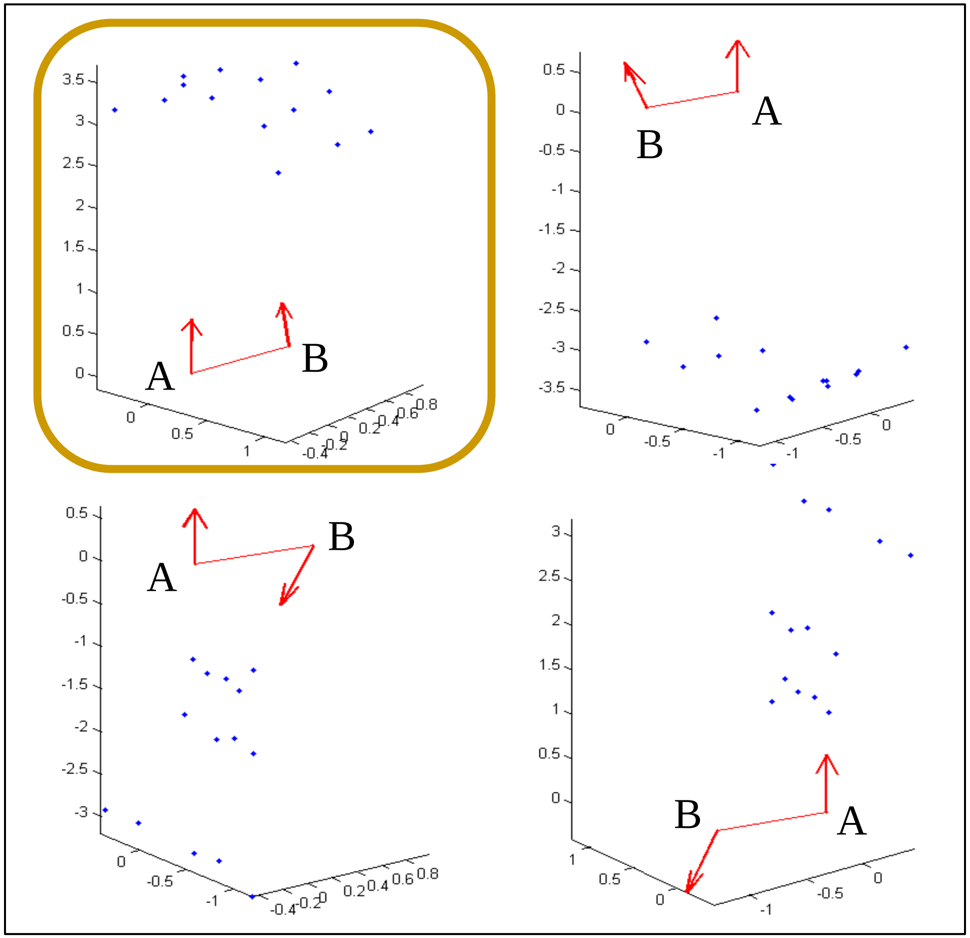
## Example:

[from Carl Olsson]

- four distinct **relative camera positions** (motion $R$, $T$) computed from $E$ (up to scale)

- 3D structure $\{X_i\}$ computed from correspondences $\mathbf{x}_i \leftrightarrow \bar{\mathbf{x}}_i$ by *triangulation* (more soon...) up to a *similarity transformation* (i.e. scale+position+orientation)

baseline reversal ($T=\pm U_3$)

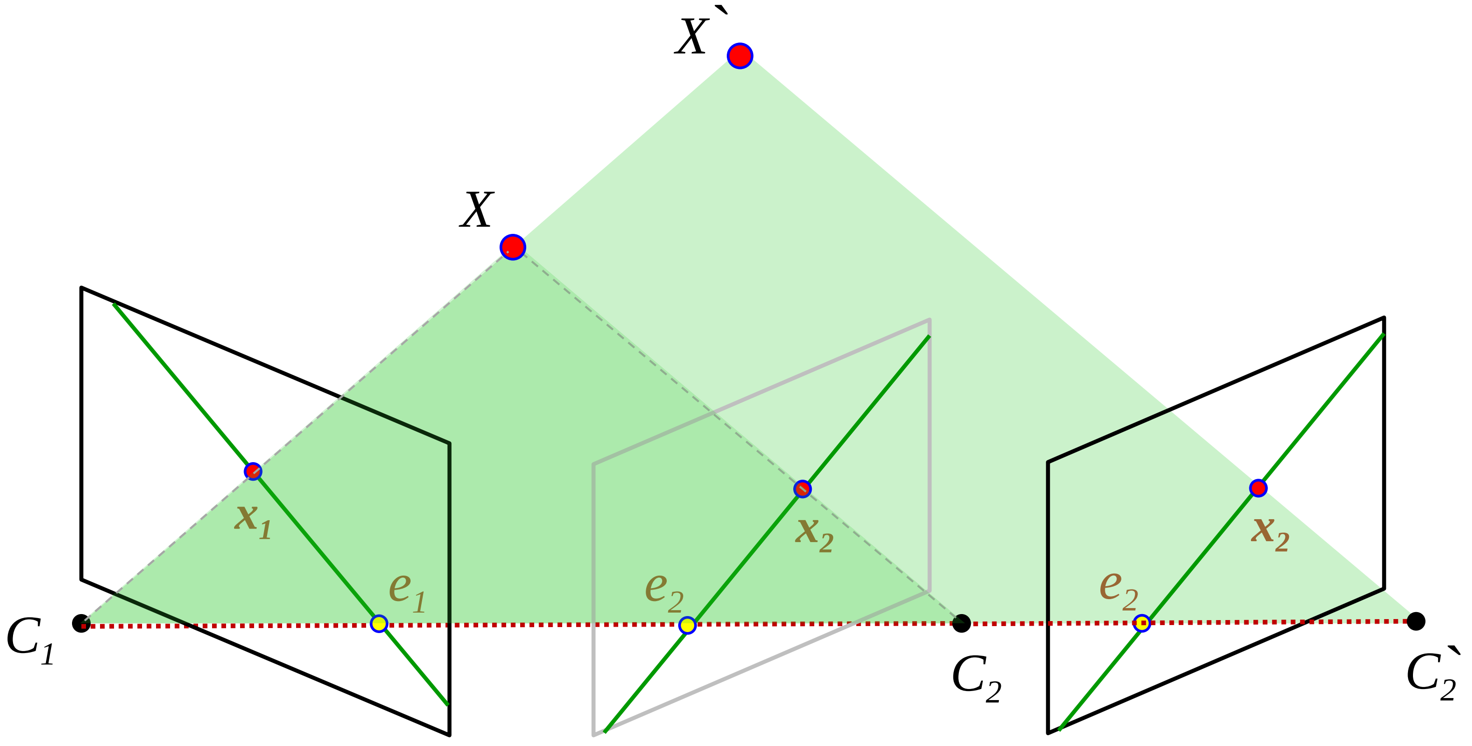


camera $B$ orientation flips ($R=UWV^T$ or $UW^TV^T$)

**Note: only one solution has positive "depths" for both cameras**

# Causes for 3D reconstruction ambiguity:

- **scale** remember: epipolar geometry can not help to estimate baseline length |*T*|



baseline $T = C_1 C_2$

larger baseline
$T' = C_1 C_2'$

# Causes for 3D reconstruction ambiguity:

- **scale**

- **position**+**orientation** ?

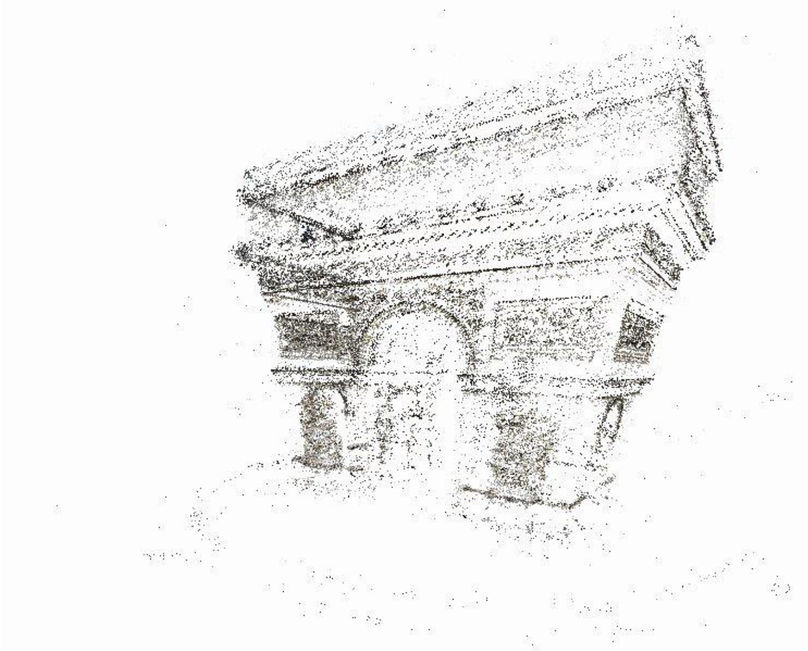
epipolar geometry determines only **relative** camera positions

# Extracting cameras from fundamental matrix $F$

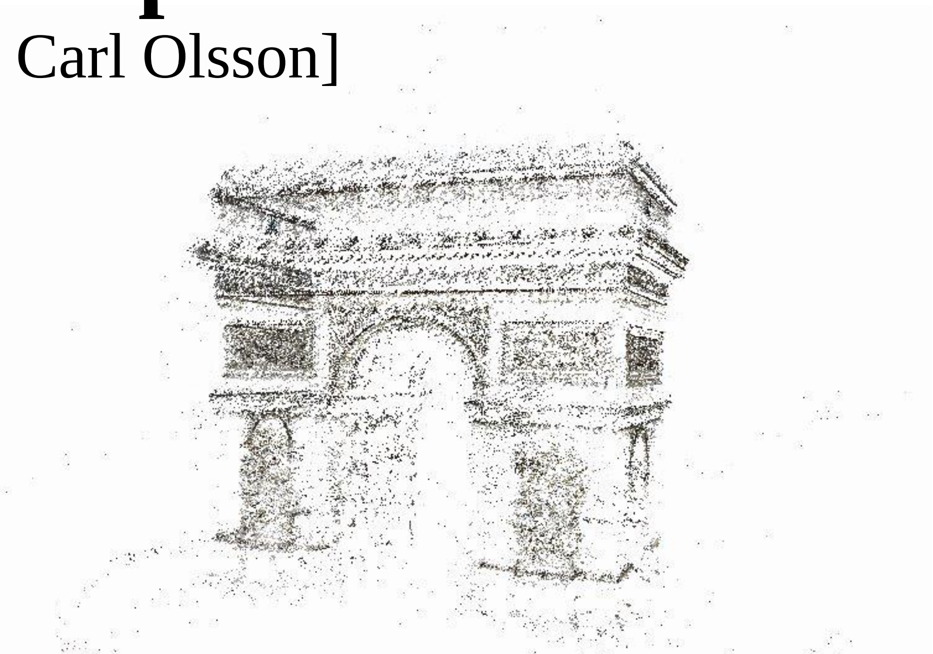
One can also estimate camera projection matrices from **fundamental matrix**, but there are more ambiguities [see H&Z]

## Examples

[from Carl Olsson]



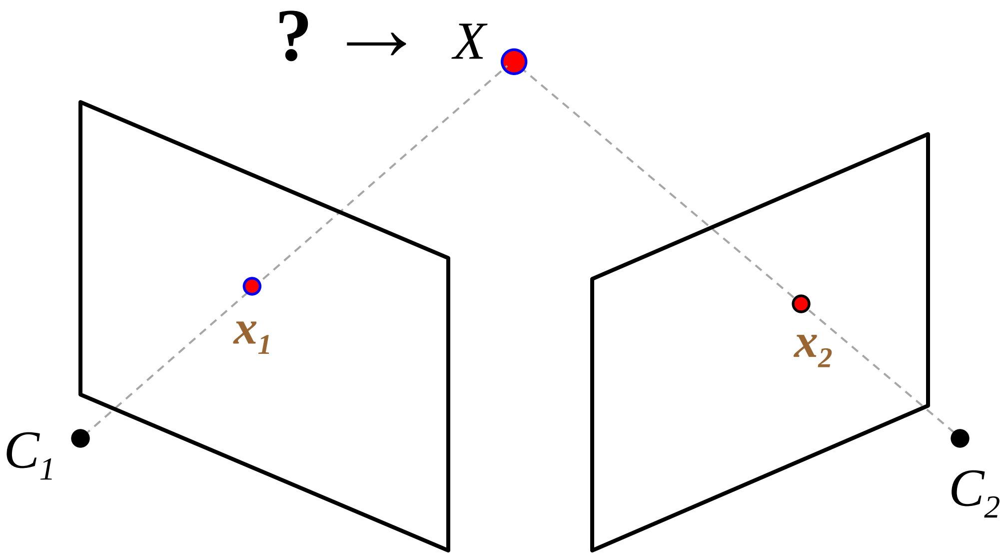
“projective” ambiguity
(cameras estimated from $F$)



3D reconstruction with
similarity transform ambiguity
(cameras estimated from $E$)

# Triangulation

Now, assume known projection matrices $P_1$ , $P_2$ and a match $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$



$$P_1 = [\ I \mid 0\ ]$$

$$P_2 = [\ R \mid T\ ]$$

projection constraints

$$\begin{bmatrix} w_1 u_1 \\ w_1 v_1 \\ w_1 \end{bmatrix} = P_1 \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} w_2 u_2 \\ w_2 v_2 \\ w_2 \end{bmatrix} = P_2 \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

6 equations with 5 unknown $(X, Y, Z, w_1, w_2)$

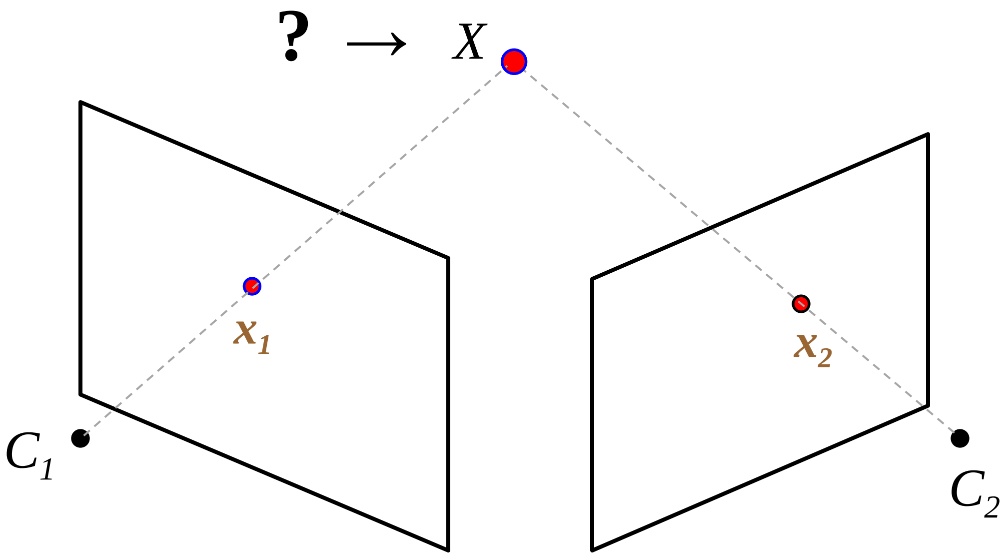
But, we do not care about $w_1$ & $w_2$ – **eliminate** them (*à la* slide 15 topic 6)

=> 4 equations with 3 unknown $(X, Y, Z)$

# Triangulation

Now, assume known projection matrices $P_1$ , $P_2$ and a match $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$

$$P_1 = [\ I \mid 0\ ]$$



$$P_2 = [\ R \mid T\ ]$$

projection constraints

$$\begin{bmatrix} w_1 u_1 \\ w_1 v_1 \\ w_1 \end{bmatrix} = P_1 \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} w_2 u_2 \\ w_2 v_2 \\ w_2 \end{bmatrix} = P_2 \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

One equation is redundant only if points $x_1$, $x_2$ are exactly on the corresponding epipolar lines (the corresponding rays intersect in 3D).

**Due to errors, use least squares.**

# Structure-from-Motion workflow

## Basic sequential reconstruction

- For the first two images, use 8-point algorithm to estimate essential matrix $E$, cameras, and triangulate some points $\{X_i\}$.
- Each new view should see some previously reconstructed scene points $\{X_i\}$ ("feature matches" with previous cameras). Use such points to estimate new camera position (*resection problem*).
- Add new scene points using triangulation, e.g. for new "matches" with previously non-matched (and non-triangulated) features in earlier views.
- If there are more cameras, iterate previous two steps.

- **Issues**
  - errors can accumulate
  - new views are used only to add new 3D points, but they can help to improve accuracy for previously reconstructed scene

# Structure-from-Motion workflow

## **“Bundle adjustment”**

*i*-th “feature track”

$$tr_i := \{\boxed{x_{ik}} | k \in \boxed{V(i)}\}$$

feature *i*
location
in image k

set of images
where feature *i*
is visible

$$\min_{\{P_k\},\{X_i\}} \sum_{i} \sum_{k \in V(i)} \|x_{ik} - P_k X_i\|$$

re-projection error

# Structure-from-Motion workflow

https://www.youtube.com/watch?v=i7ierVkXYa8

from Carl Olsson

# Applications of multi-view geometry:

Pose estimation

Rigid motion segmentation

Augmented reality

Special effects in video

Volumetric 3D reconstruction

Depth reconstruction (stereo-next topic)

# Examples:

We were fitting a single essential/fundamental matrix to
a pair of images corresponding to two different view points

**Q**: Can matched features in two images support **more than one fundamental matrix**?

# Examples:

We were fitting a single essential/fundamental matrix to
a pair of images corresponding to two different view points

**Q**: Can matched features in two images support **more than one fundamental matrix**?

Hint: we assumed that the scene is stationary and only camera moved or, equivalently, that the camera is stationary but the whole 3D scene moved (R,T).

# Examples:

We were fitting a <u>single</u> essential/fundamental matrix to
a pair of images corresponding to two different view points

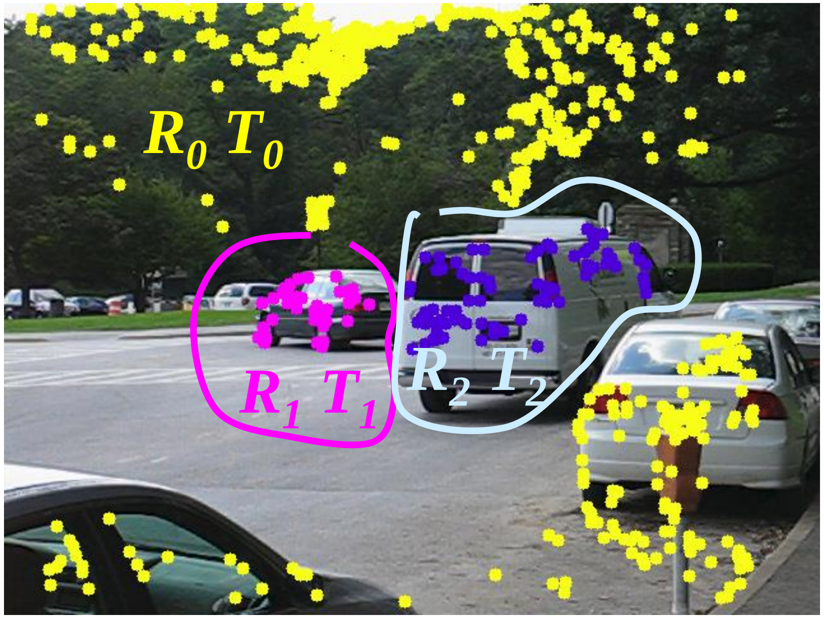
**Q**: Can matched features in two images support **more than one fundamental matrix**?

multi-model fitting with fundamental matrices (UFL, previous topic)

# Examples: Rigid Motion Estimation

We were fitting a single essential/fundamental matrix to a pair of images corresponding to two different view points

**Q**: Can matched features in two images support **more than one fundamental matrix**?



multi-model fitting with fundamental matrices (UFL, previous topic)

# Examples: Augmented Reality

- if camera position *C* and orientation *R* are known (in addition to *K*) then can insert “new” objects into the 3D scene
- particularly useful for movies: camera path can be computed

  - can generate correct views of new objects

https://www.youtube.com/watch?v=aPl1aBw_x4M

https://www.youtube.com/watch?v=Te2ZJzbuw5I

Cinema 4D Camera Motion Tracking
(“boujou”)

# Examples: Dense 3D Reconstruction

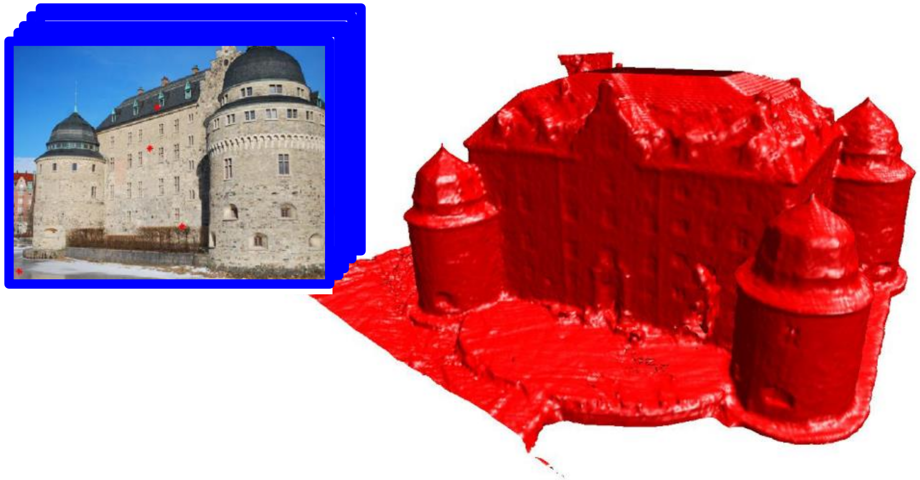
**Sparse** reconstruction (cloud of points in 3D) is done by triangulating point correspondences (part of Structure-from-Motion problem)

How about **dense** (surface in 3D) reconstruction from n views?

# Examples: Dense 3D Reconstruction

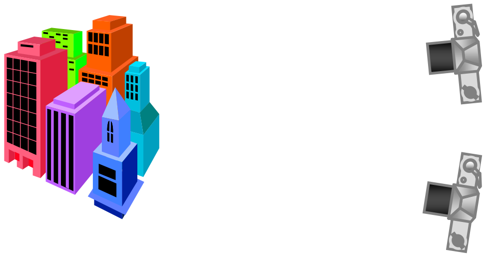
## A. computing *Surfaces in 3D Volumes*
(volumetric reconstruction)

multiple wide baseline views

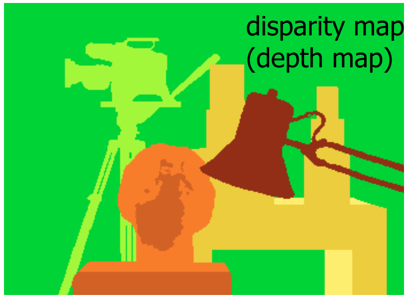


Relates to volumetric **segmentation (topic 9)**

## B. computing dense *Depth Maps*
(stereo)



two narrow baseline views



will discuss **in topic 8: stereo**

# From sparse features to dense reconstructions

- Assume known relative position of cameras (epipolar lines)

- Now, we can move towards **denser** reconstruction

  - find many more matches (correspondences) using known epipolar lines: constrained search space significantly reduces ambiguity for feature matching

  - use **"regularization"** to estimate surfaces or *depth maps*