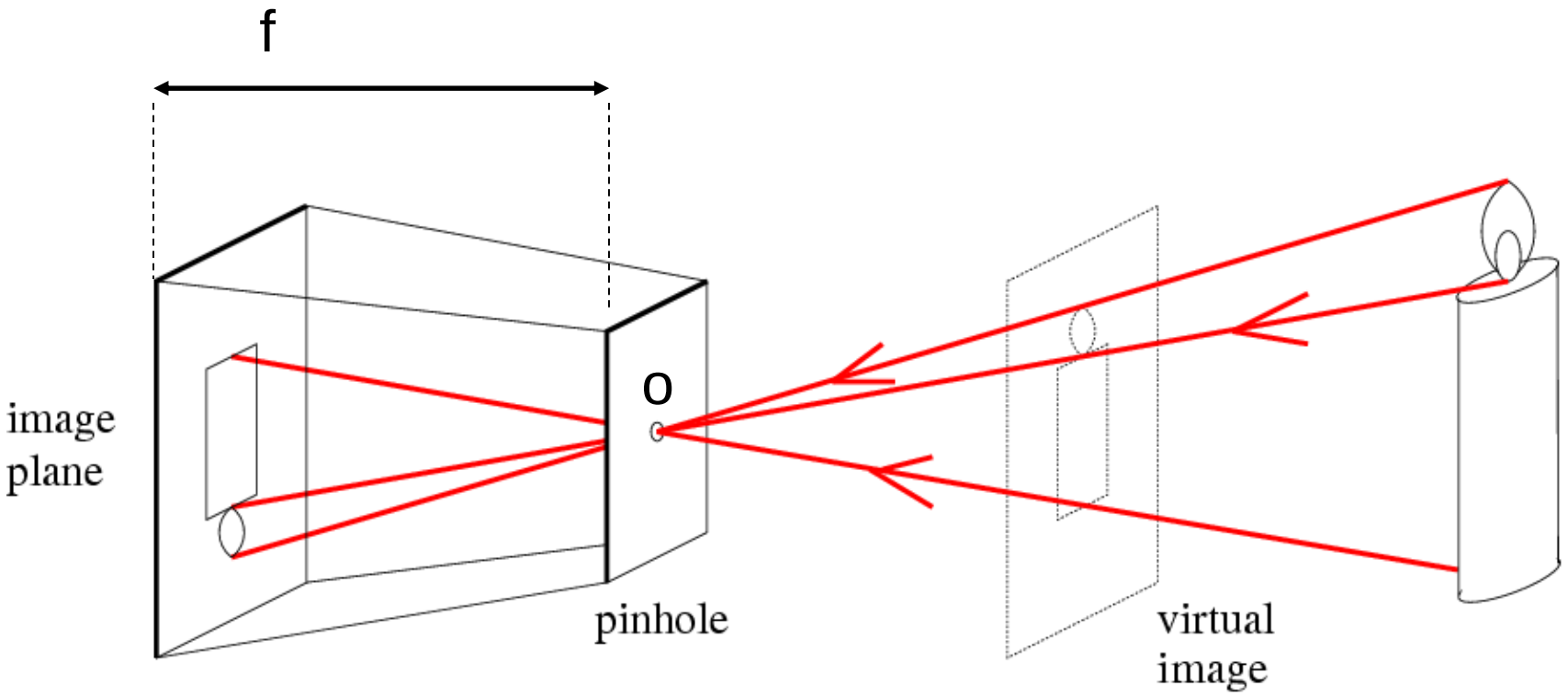
# CS231M · Mobile Computer Vision

## Structure from motion

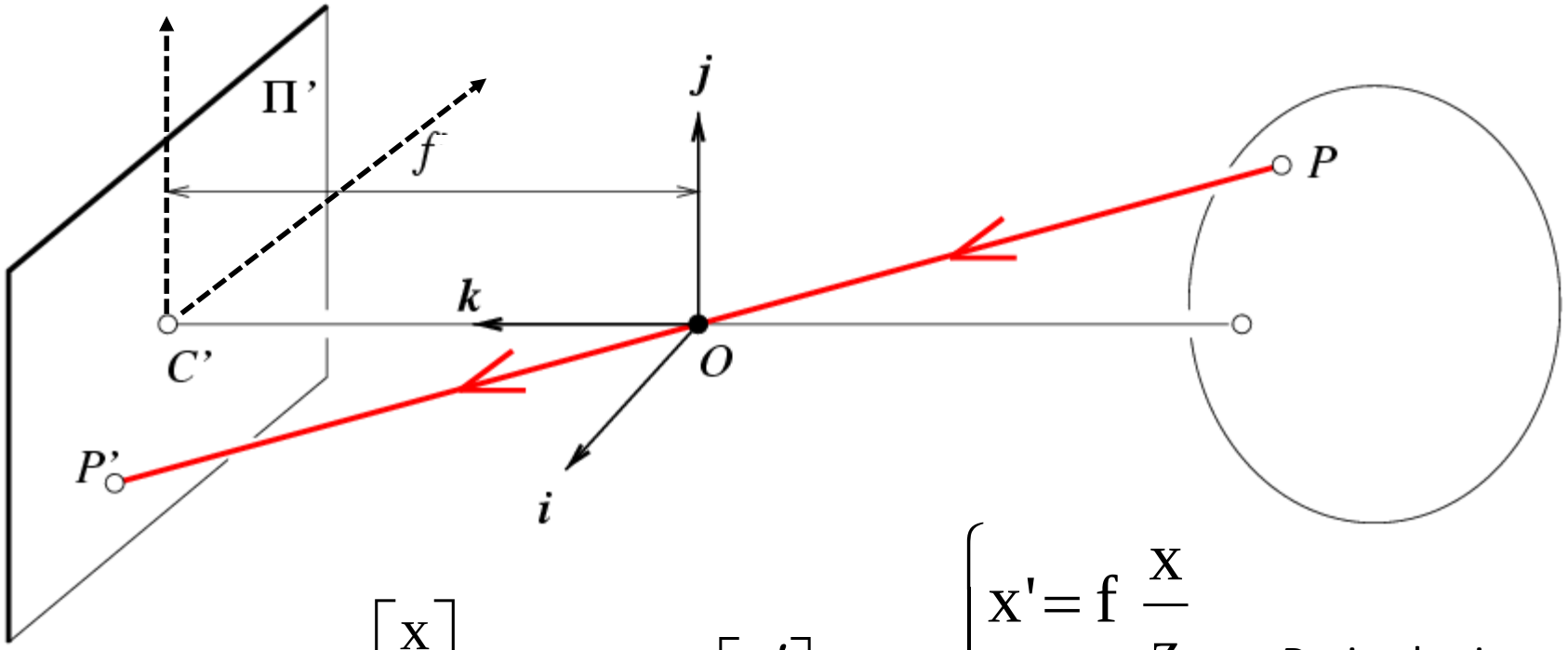

- Cameras
- Epipolar geometry
- Structure from motion

# Pinhole camera




f = focal length
o = center of the camera

# Pinhole camera

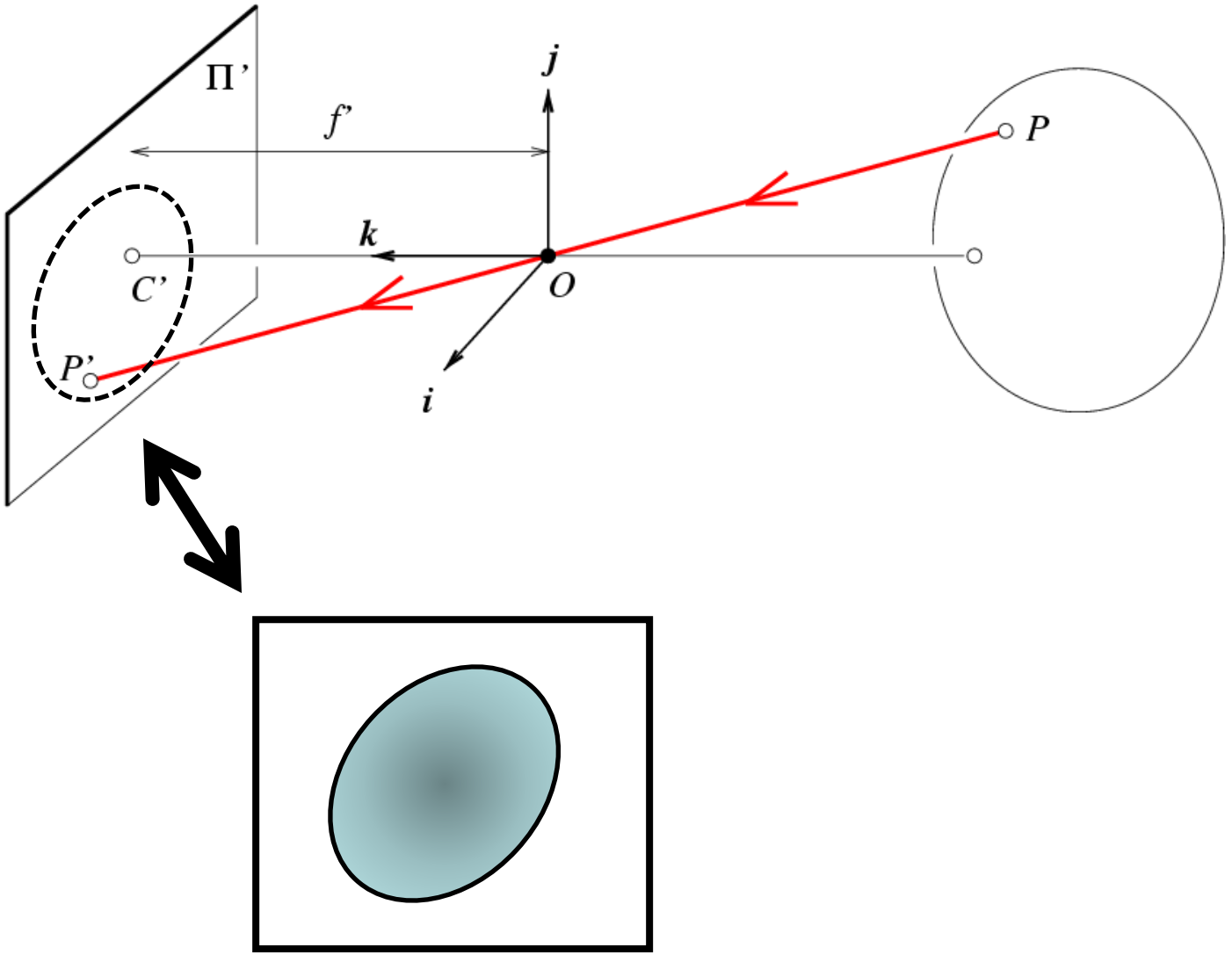



$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \rightarrow P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \qquad \begin{cases} x' = f\,\dfrac{x}{z} \\ y' = f\,\dfrac{y}{z} \end{cases}$$

Derived using similar triangles

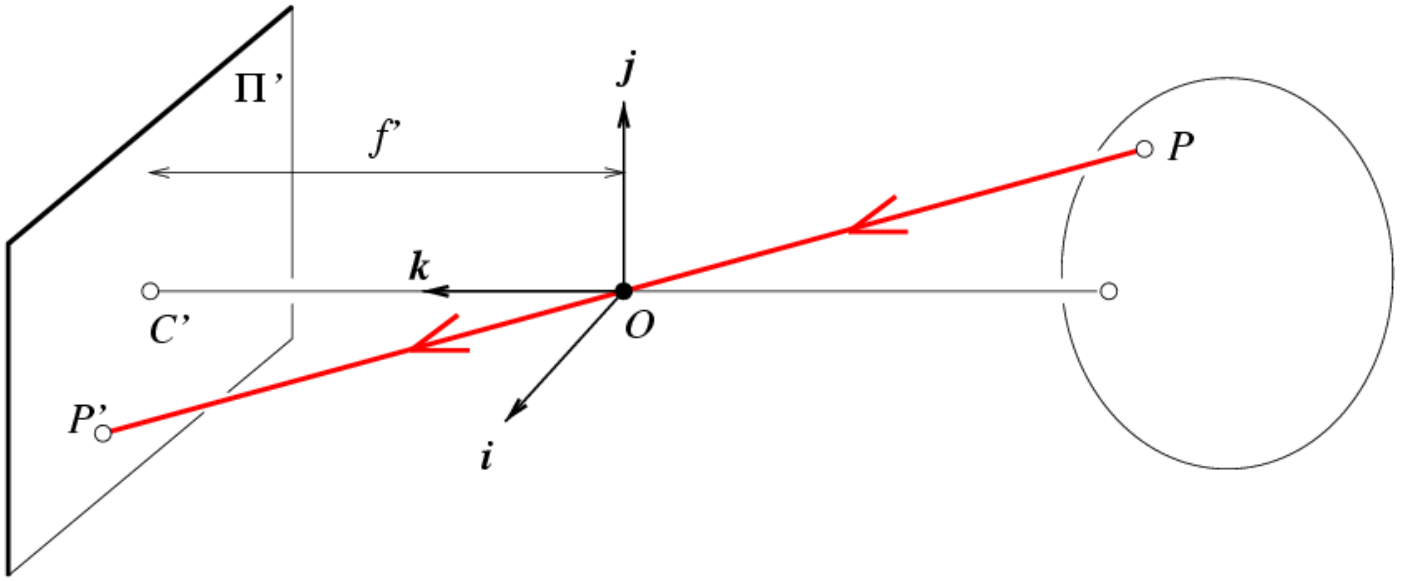
$$(x, y, z) \rightarrow \left(f\,\frac{x}{z}, f\,\frac{y}{z}\right)$$

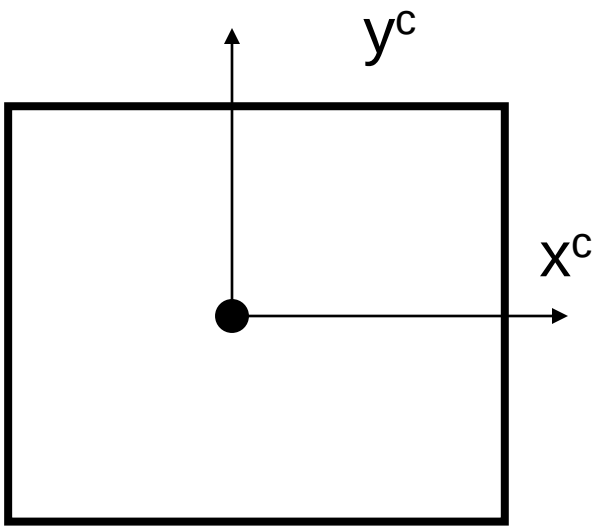
# From retina plane to images



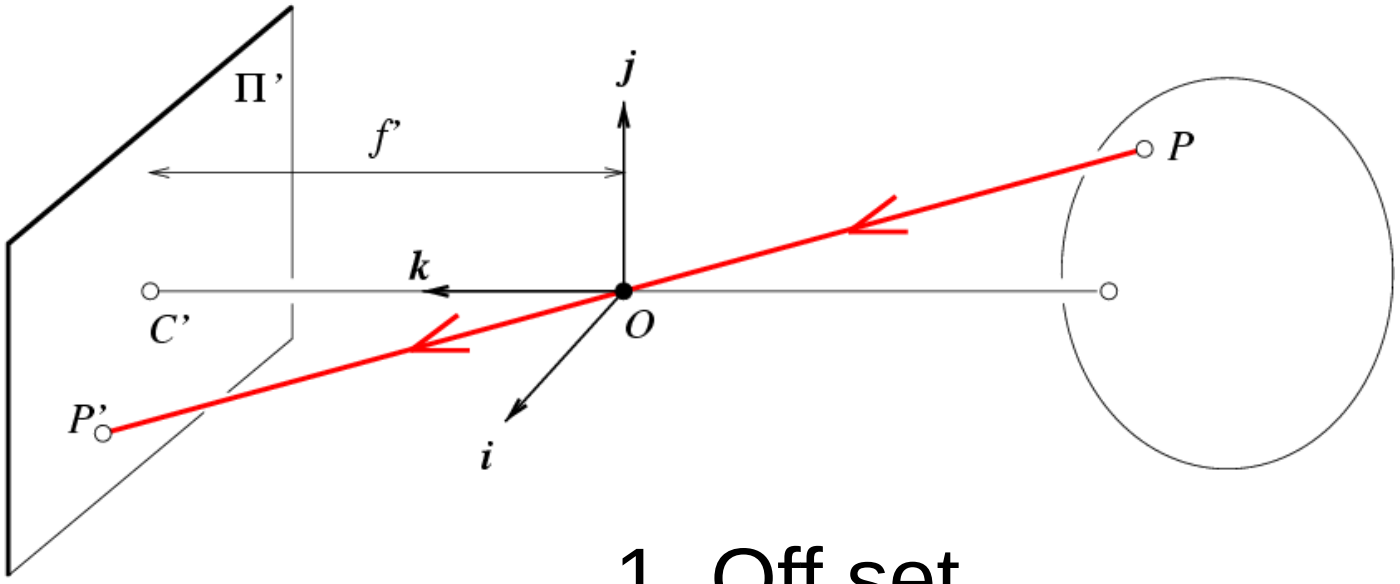

Pixels, bottom-left coordinate systems

# From retina plane to images

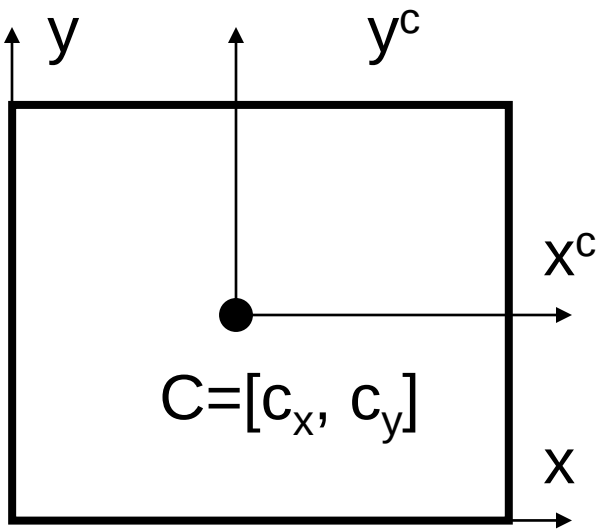


# Converting to pixels

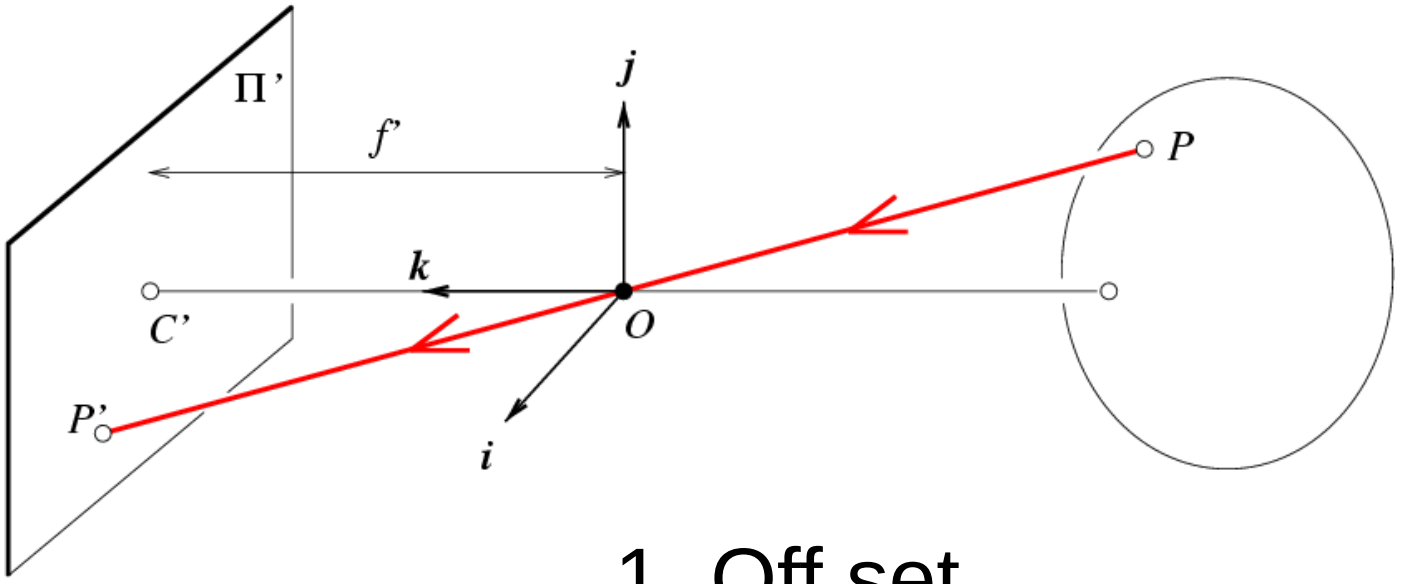



1. Off set

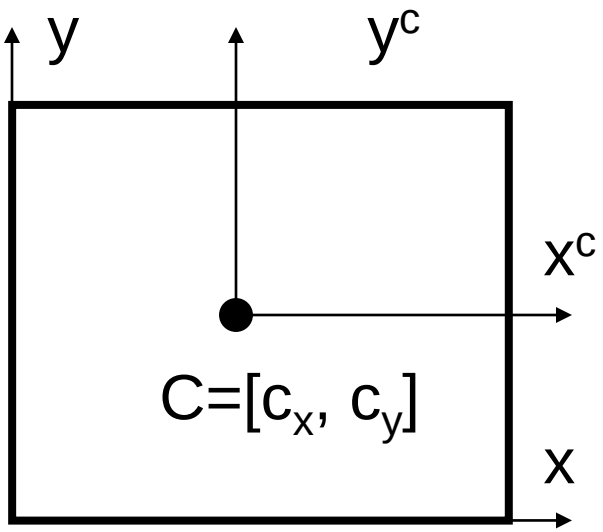



$$(x, y, z) \rightarrow (f\,\frac{x}{z} + c_x, f\,\frac{y}{z} + c_y)$$

# Converting to pixels




1. Off set
2. From metric to pixels




$$(x, y, z) \rightarrow (\underbrace{f\ k}_{\alpha}\frac{x}{z} + c_x, \underbrace{f\ l}_{\beta}\frac{y}{z} + c_y)$$

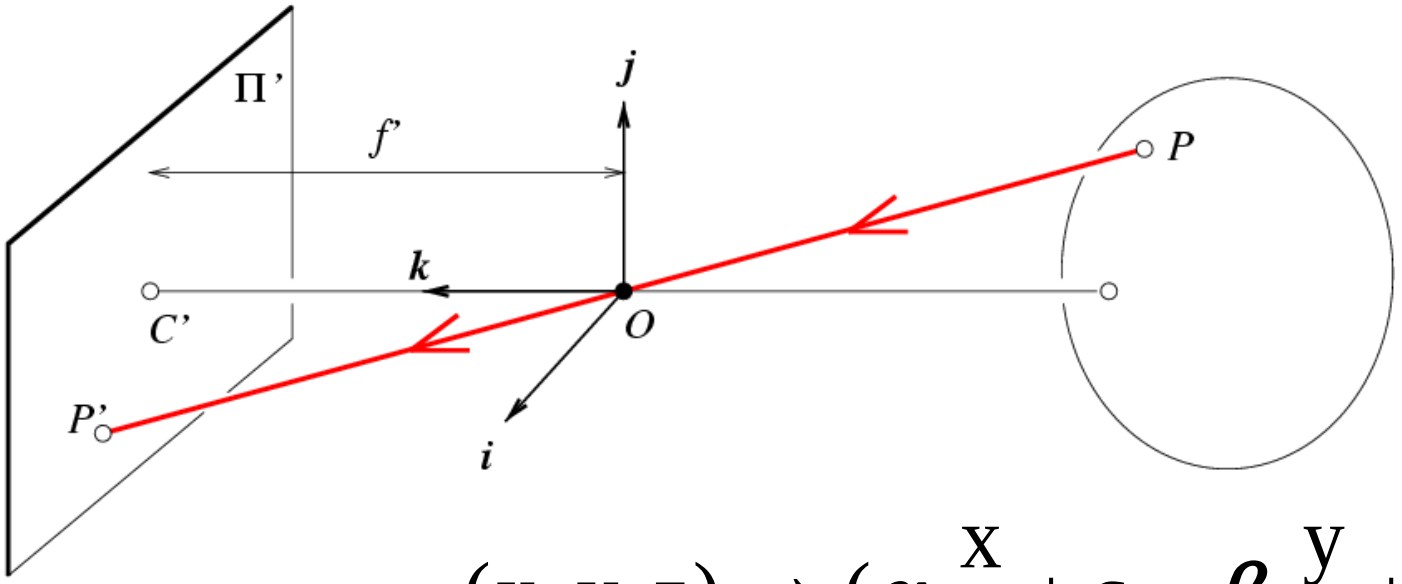
Units: k,l : pixel/m

f : m

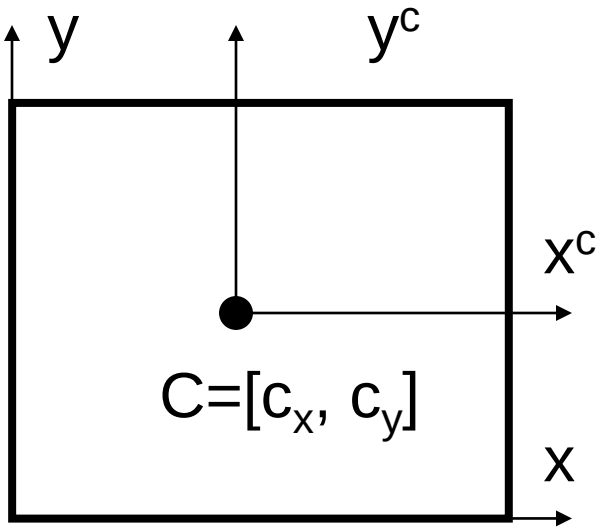
Non-square pixels

$\alpha$, $\beta$ : pixel

# Camera Matrix




$$(x, y, z) \rightarrow (\alpha \frac{x}{z} + c_x, \beta \frac{y}{z} + c_y)$$




- Matrix form?

# Homogeneous coordinates

For details see lecture on transformations in CS131A

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

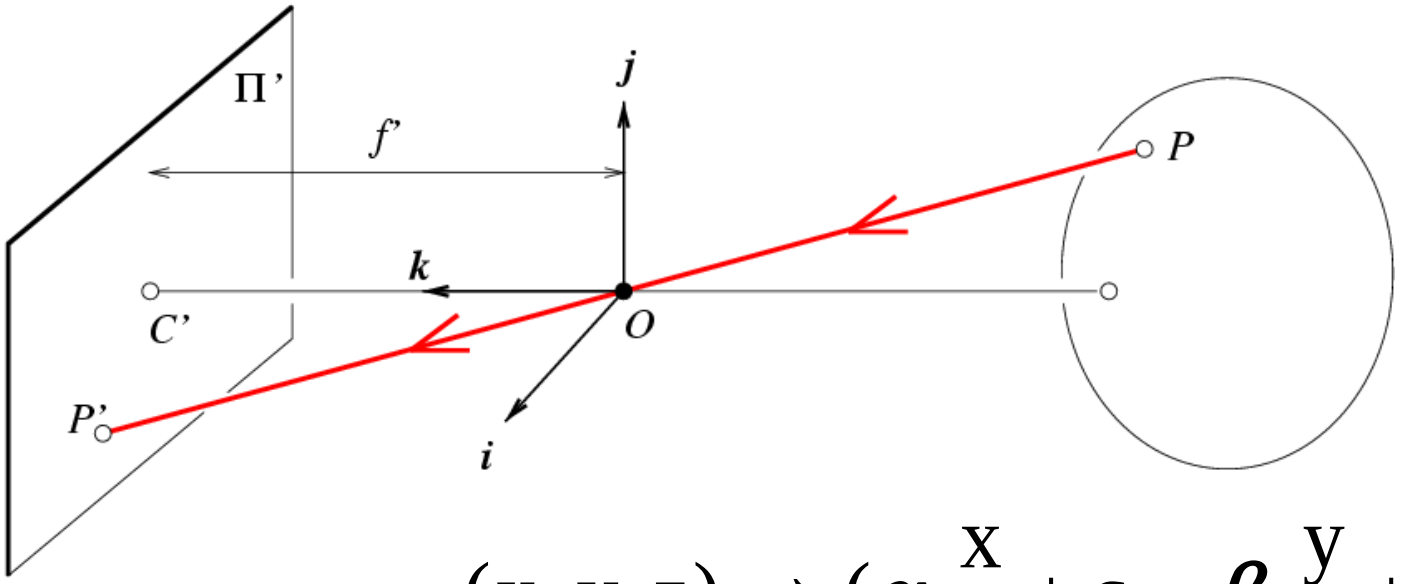
homogeneous scene coordinates

- Converting *from* homogeneous coordinates

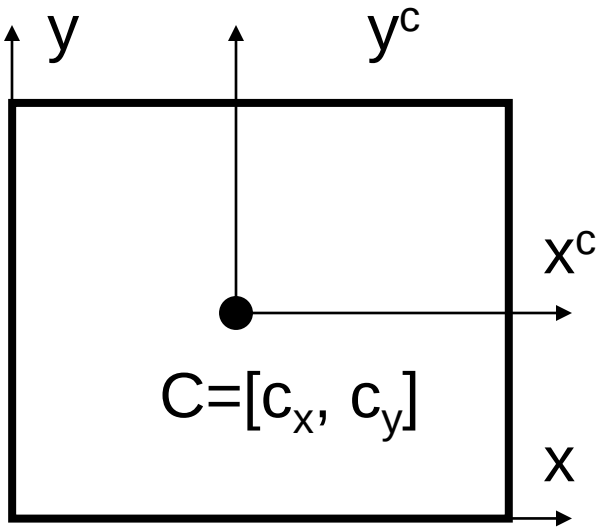
$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

# Camera Matrix



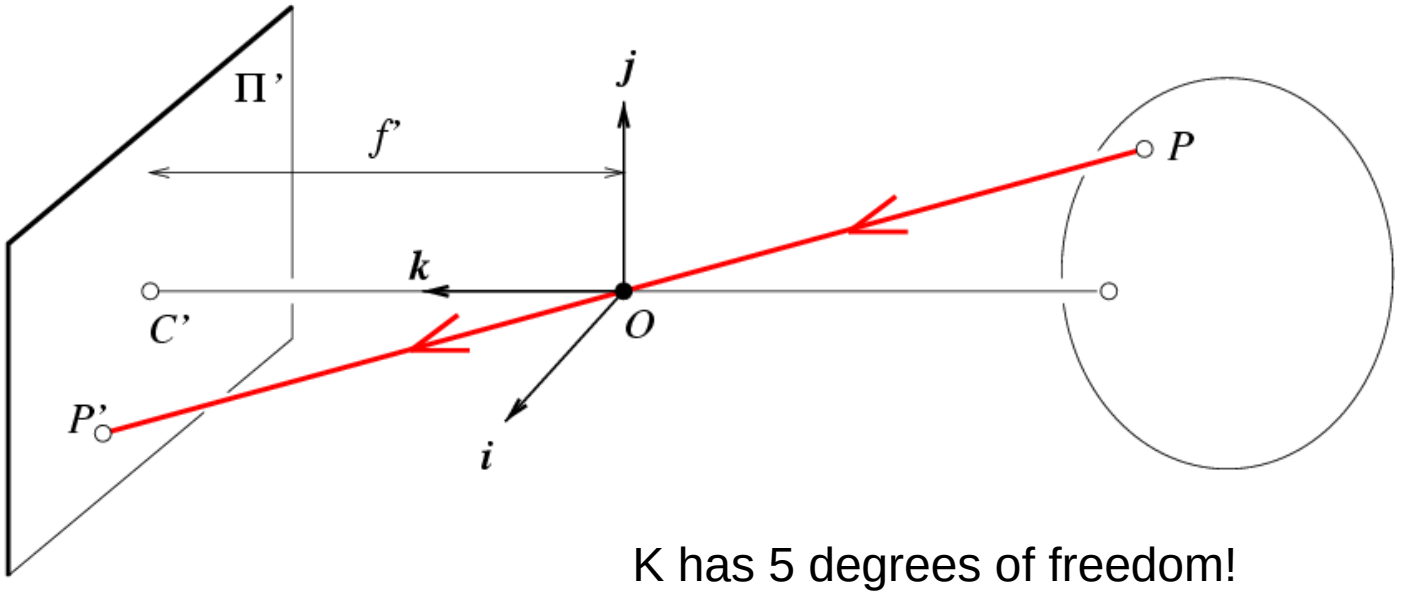

$$(x, y, z) \rightarrow (\alpha \frac{x}{z} + c_x, \beta \frac{y}{z} + c_y)$$



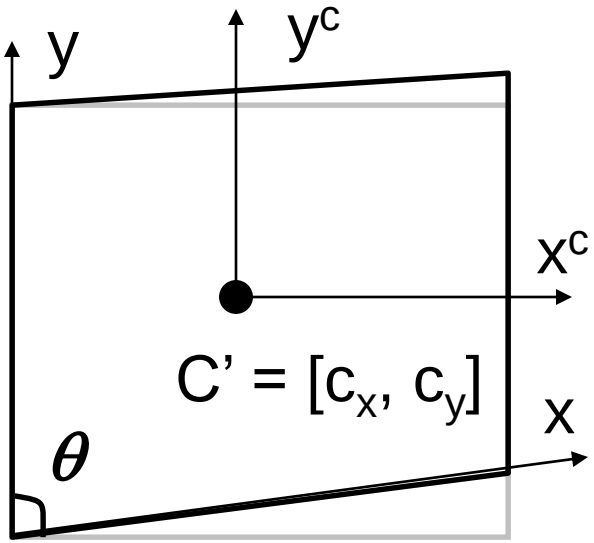

Camera matrix K

$$P' = \begin{bmatrix} \alpha x + c_x z \\ \beta y + c_y z \\ z \end{bmatrix} = \begin{bmatrix} \alpha & 0 & c_x & 0 \\ 0 & \beta & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

# Camera Skew



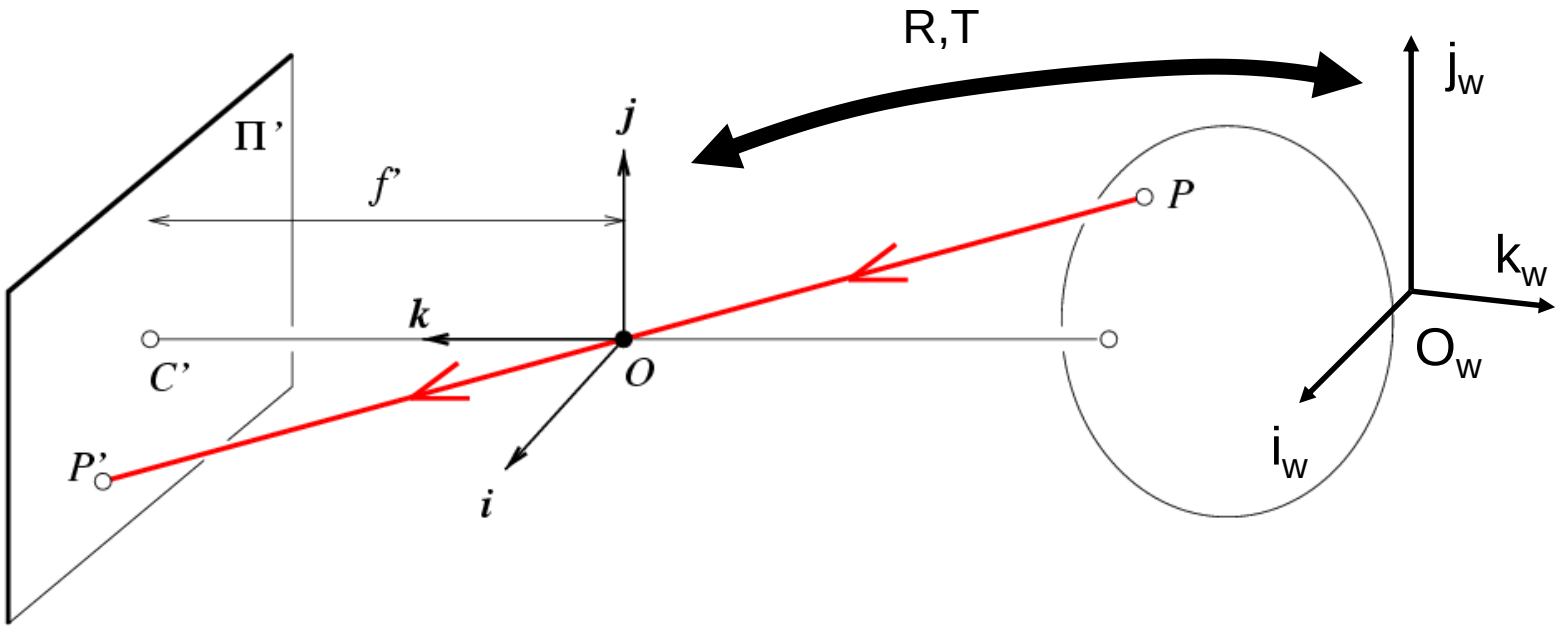

K has 5 degrees of freedom!




$$P' = \begin{bmatrix} \alpha & -\alpha\cot\theta & c_x & 0 \\ 0 & \frac{\beta}{\sin\theta} & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

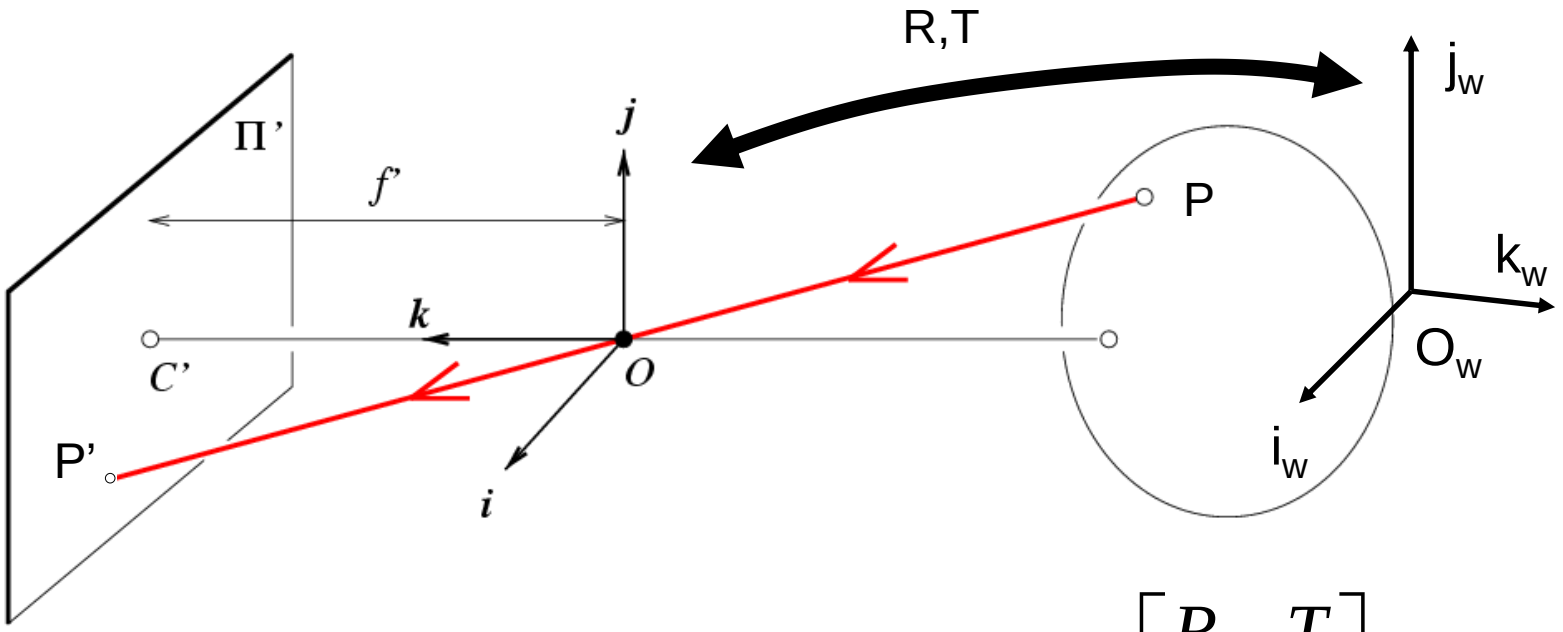
$$P' = M\ P = K\begin{bmatrix} I & 0 \end{bmatrix} P$$

# World reference system




- The mapping so far is defined within the camera reference system
- What if an object is represented in the world reference system

# World reference system




In 4D homogeneous coordinates: $P = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}_{4\times4} P_w$

Internal parameters

External parameters

$$P' = K\begin{bmatrix} I & 0 \end{bmatrix} P = K\begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}_{4\times4} P_w = K\begin{bmatrix} R & T \end{bmatrix} P_w$$

M

# Properties of Projection

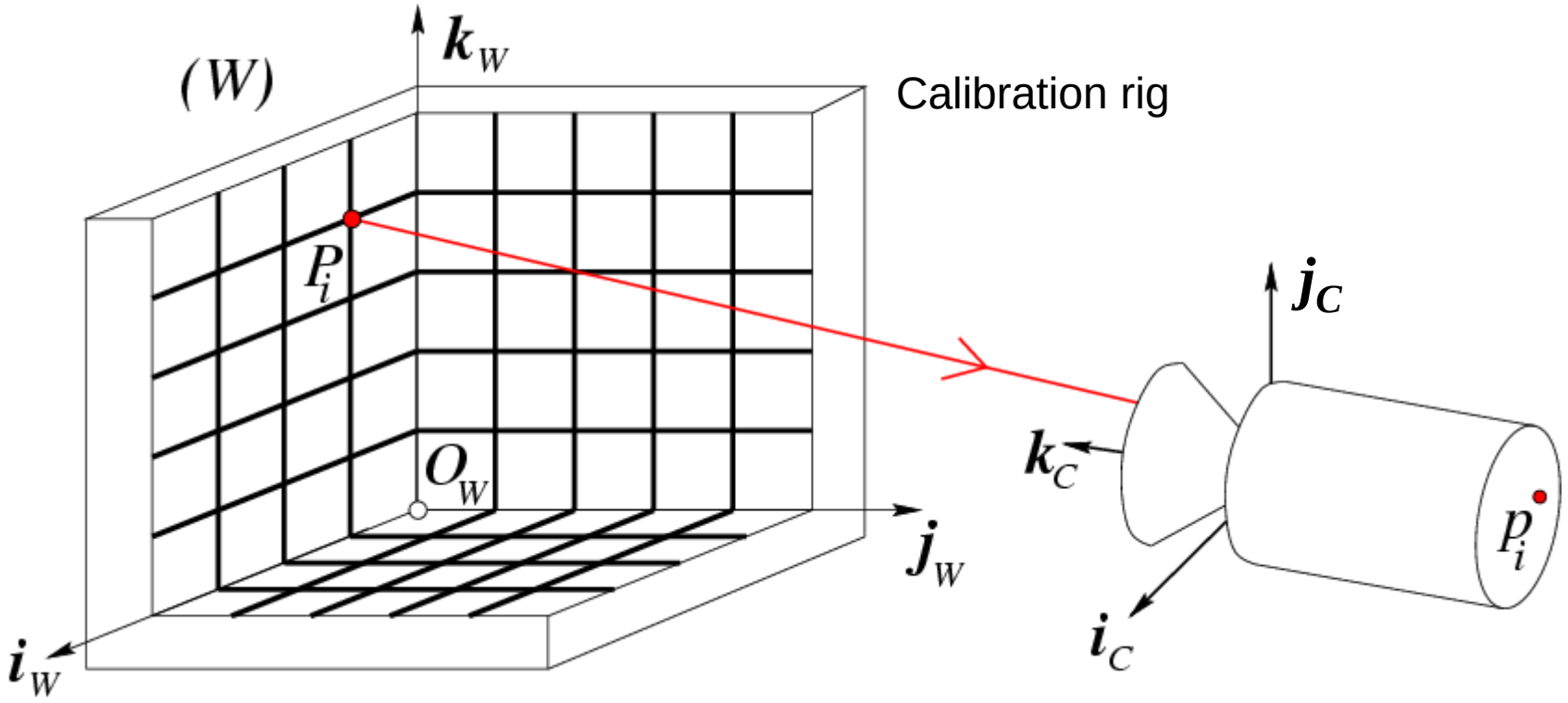
- Points project to points
- Lines project to lines
- Distant objects look smaller

# Properties of Projection

- Angles are not preserved
- Parallel lines meet!

Vanishing point

# Camera Calibration



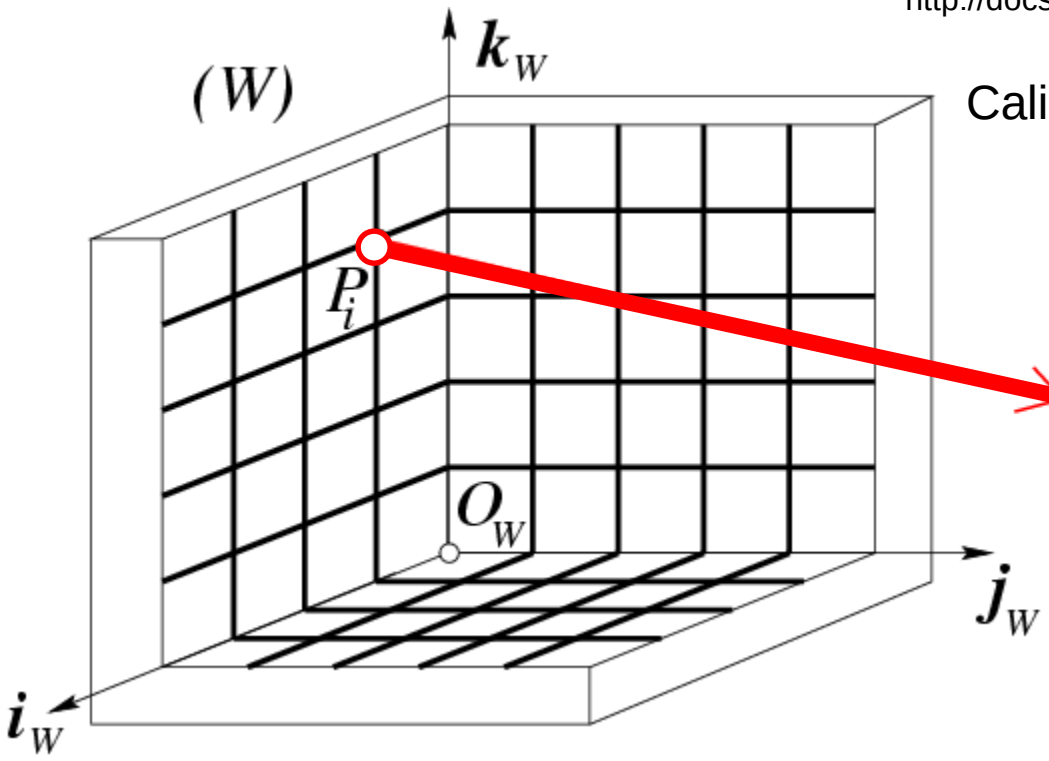

$$P' = K\begin{bmatrix} I & 0 \end{bmatrix} P = K\begin{bmatrix} R & T \end{bmatrix} P_w$$

- $P_1 \ldots P_n$ with known positions in $[O_w, i_w, j_w, k_w]$
- $p_1, \ldots p_n$ known positions in the image

Goal: compute intrinsic and extrinsic parameters

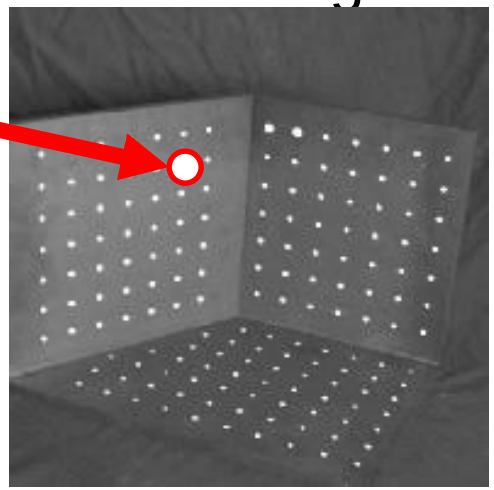
# Camera Calibration

http://docs.opencv.org/_downloads/camera_calibration.cpp




Calibration rig

image



$$P' = K\begin{bmatrix} I & 0 \end{bmatrix} P = K\begin{bmatrix} R & T \end{bmatrix} P_w$$

- $P_1 \ldots P_n$ with known positions in $[O_w, i_w, j_w, k_w]$
- $p_1, \ldots p_n$ known positions in the image

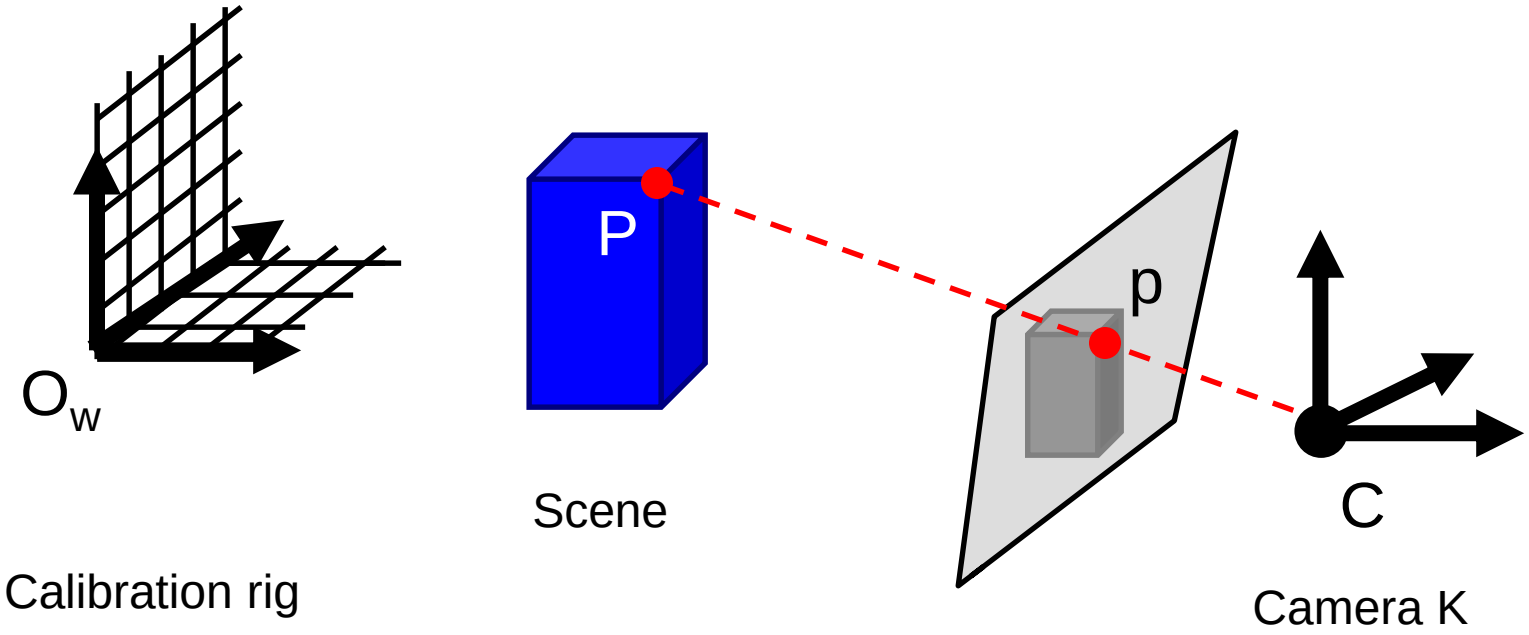
Goal: compute intrinsic and extrinsic parameters

# CS231M · Mobile Computer Vision

## Structure from motion


- Cameras
- Epipolar geometry
- Structure from motion

# Can we recover the structure from a single view?




## Why is it so difficult?

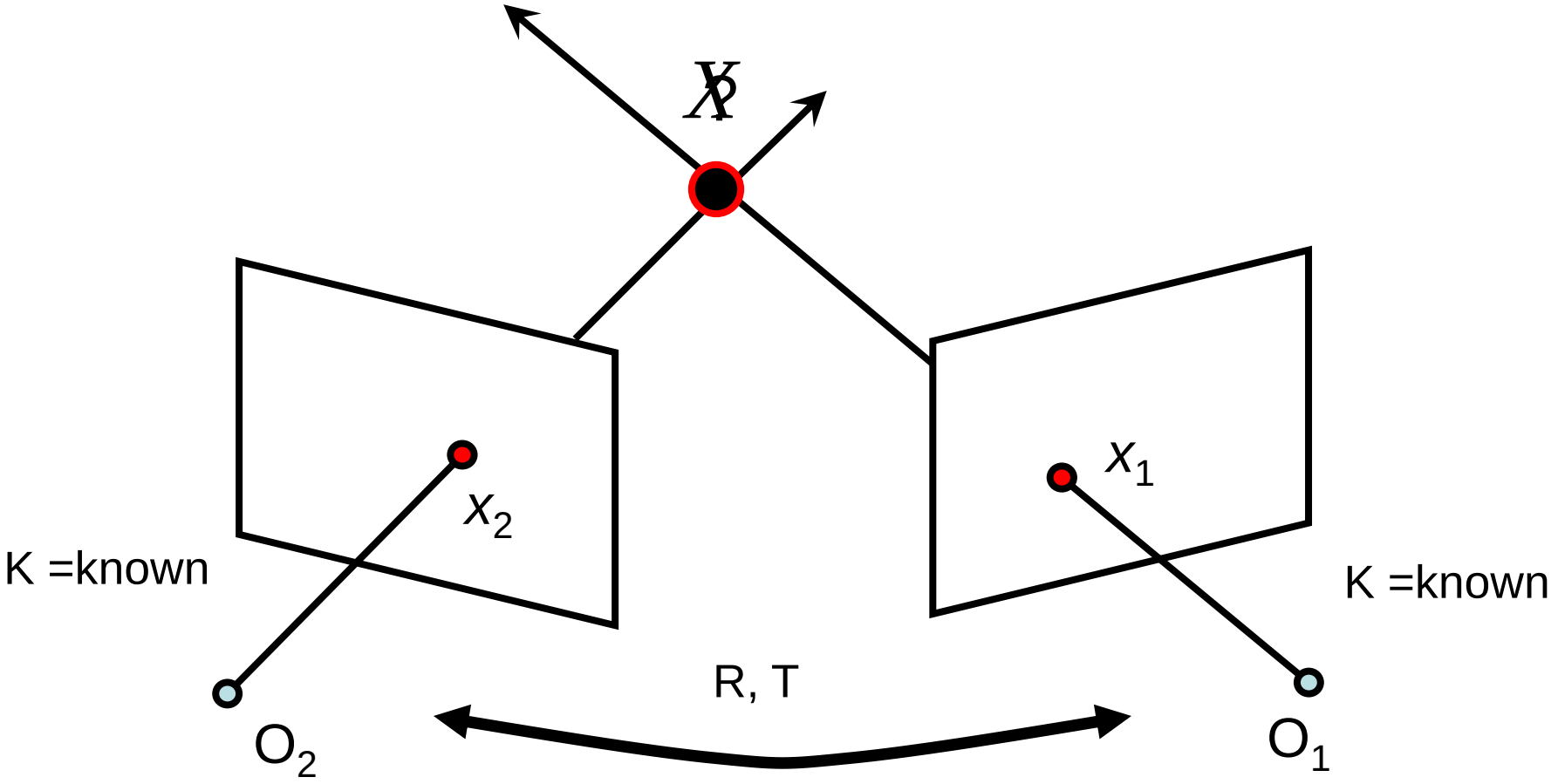
Intrinsic ambiguity of the mapping from 3D to image (2D)

# Can we recover the structure from a single view?

Intrinsic ambiguity of the mapping from 3D to image (2D)

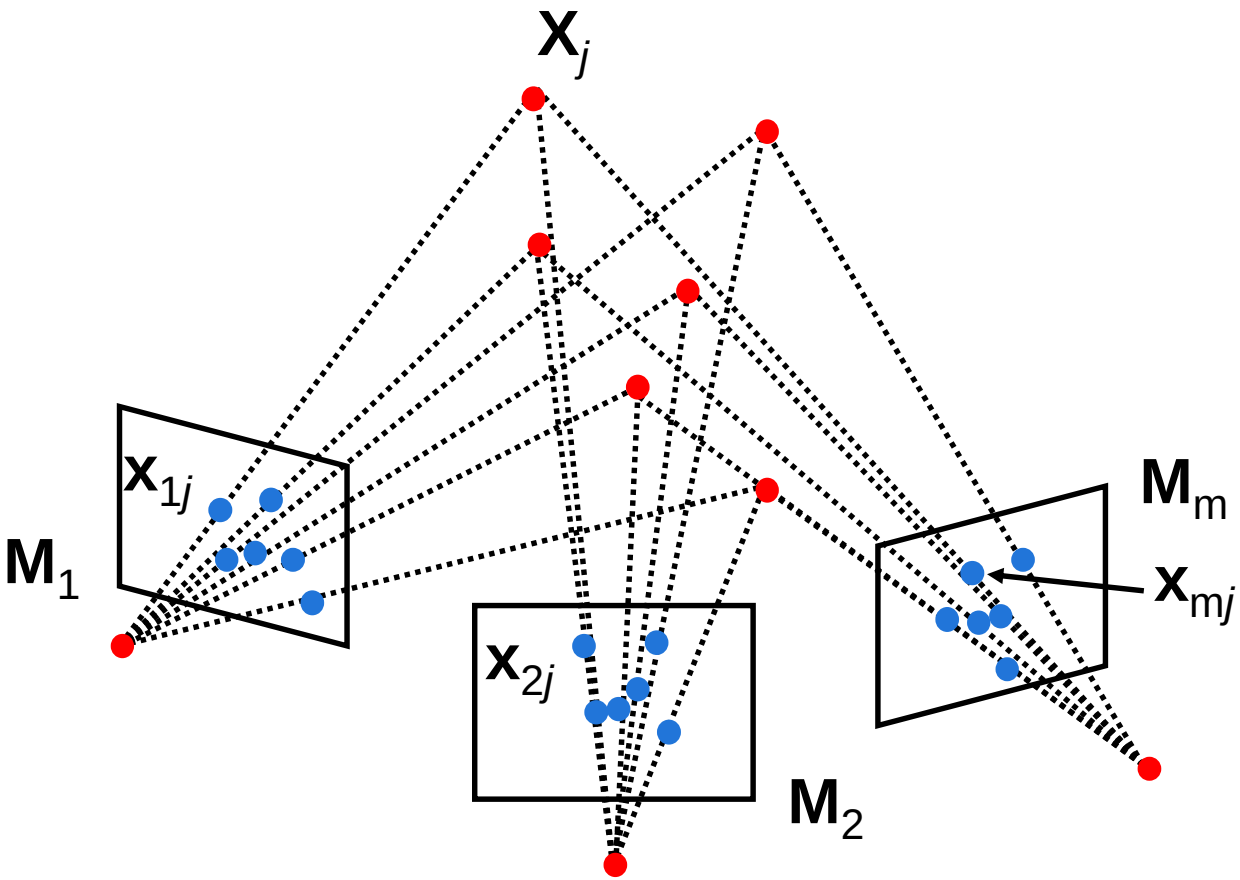
Courtesy slide S. Lazebnik

# Two eyes help!




This is called triangulation

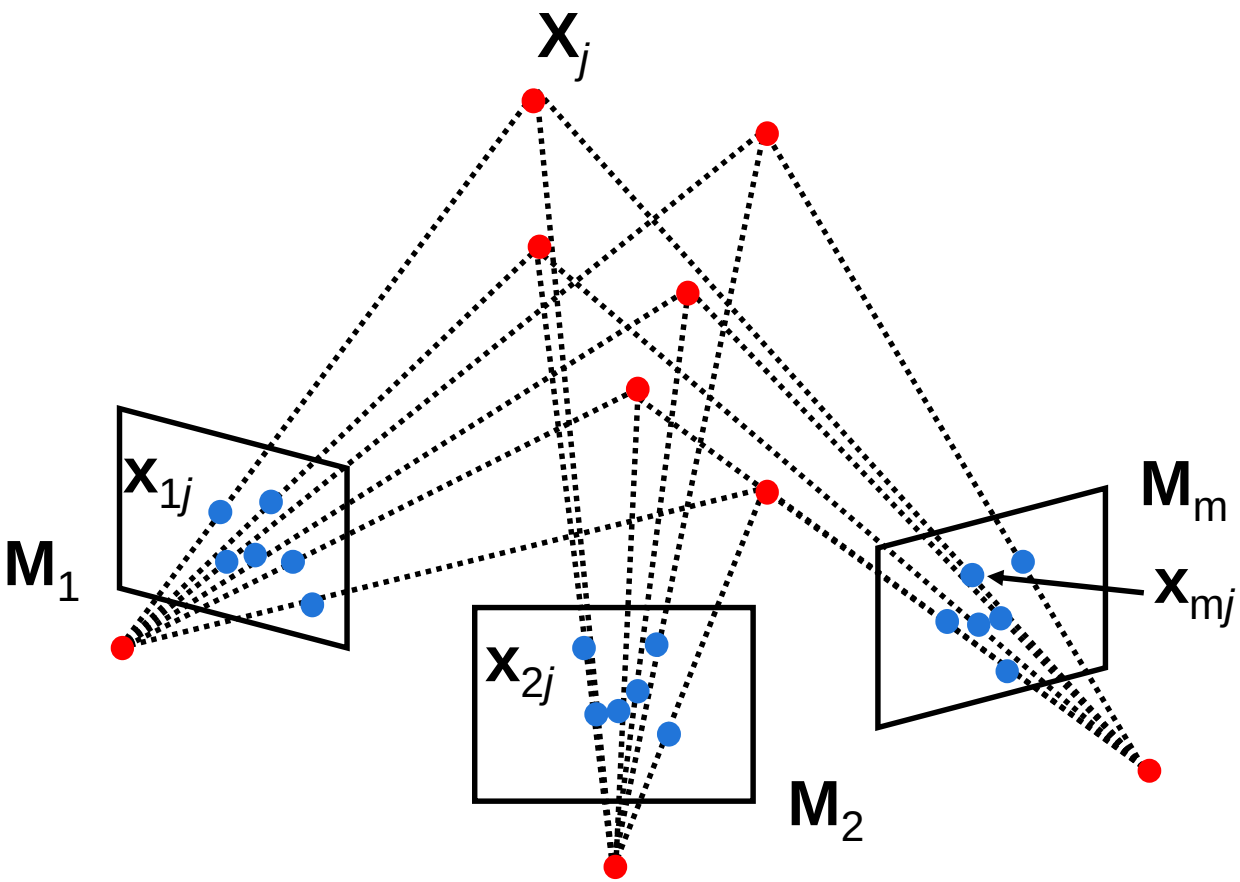
# Structure from motion problem




Given *m* images of *n* fixed 3D points

- $\mathbf{x}_{ij} = \mathbf{M}_i \mathbf{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$

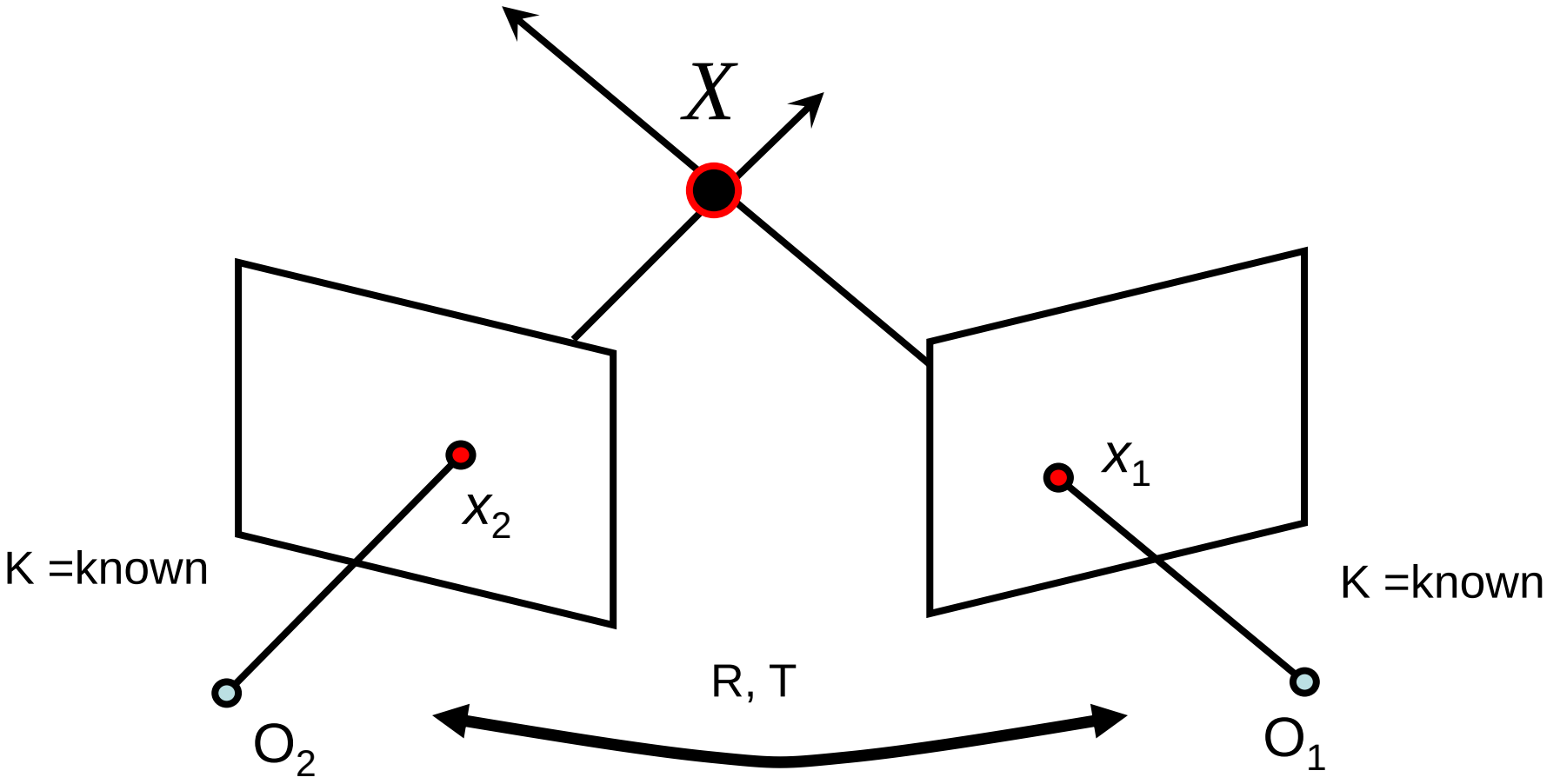
# Structure from motion problem




From the mxn correspondences $\mathbf{x}_{ij}$, can we estimate:

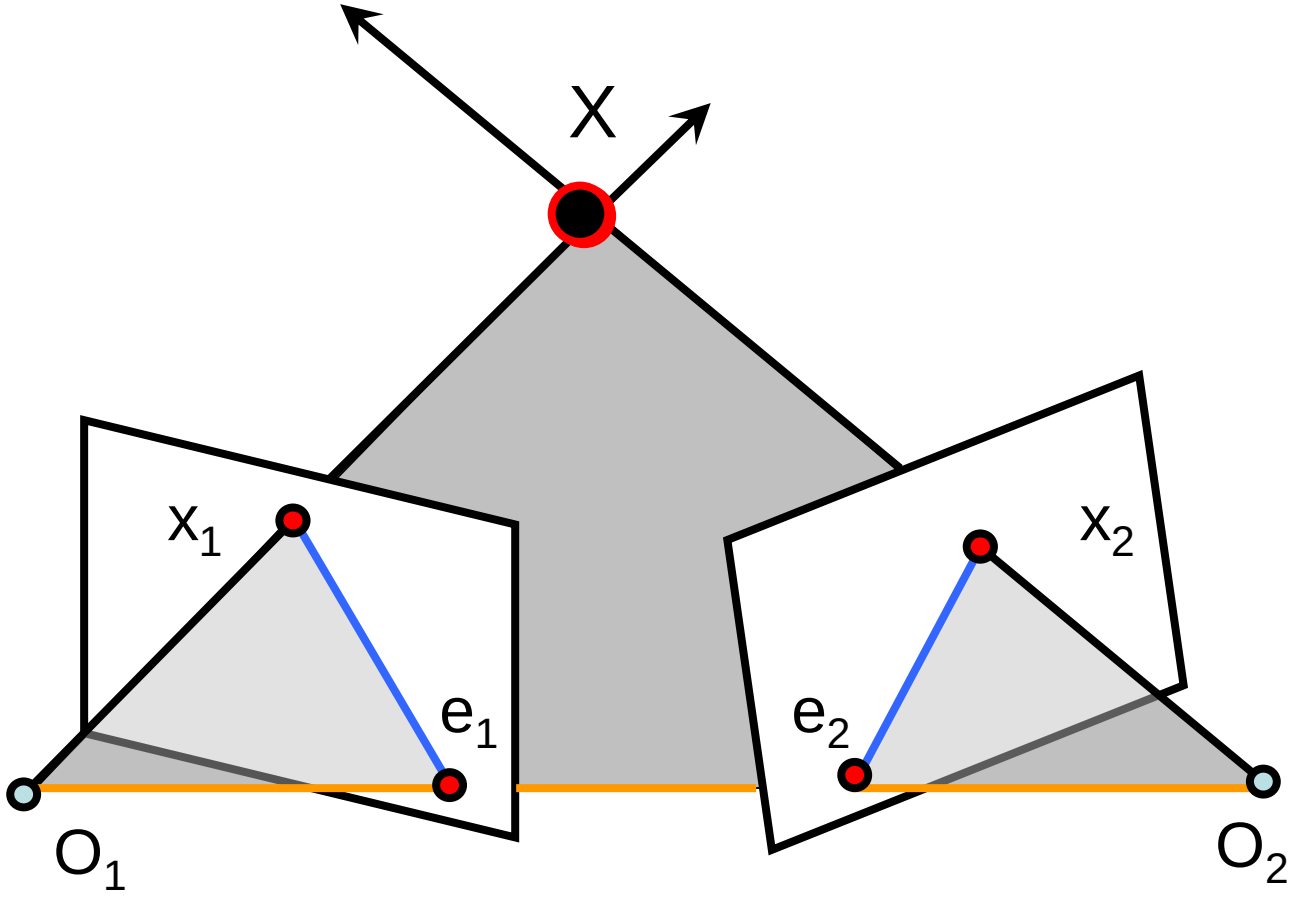
| | |
|---|---|
| •*m* projection matrices $\mathbf{M}_i$ | motion |
| •*n* 3D points $\mathbf{X}_j$ | structure |

# Study relationship between X, $x_1$ and $x_2$




Epipolar geometry!

# Epipolar geometry



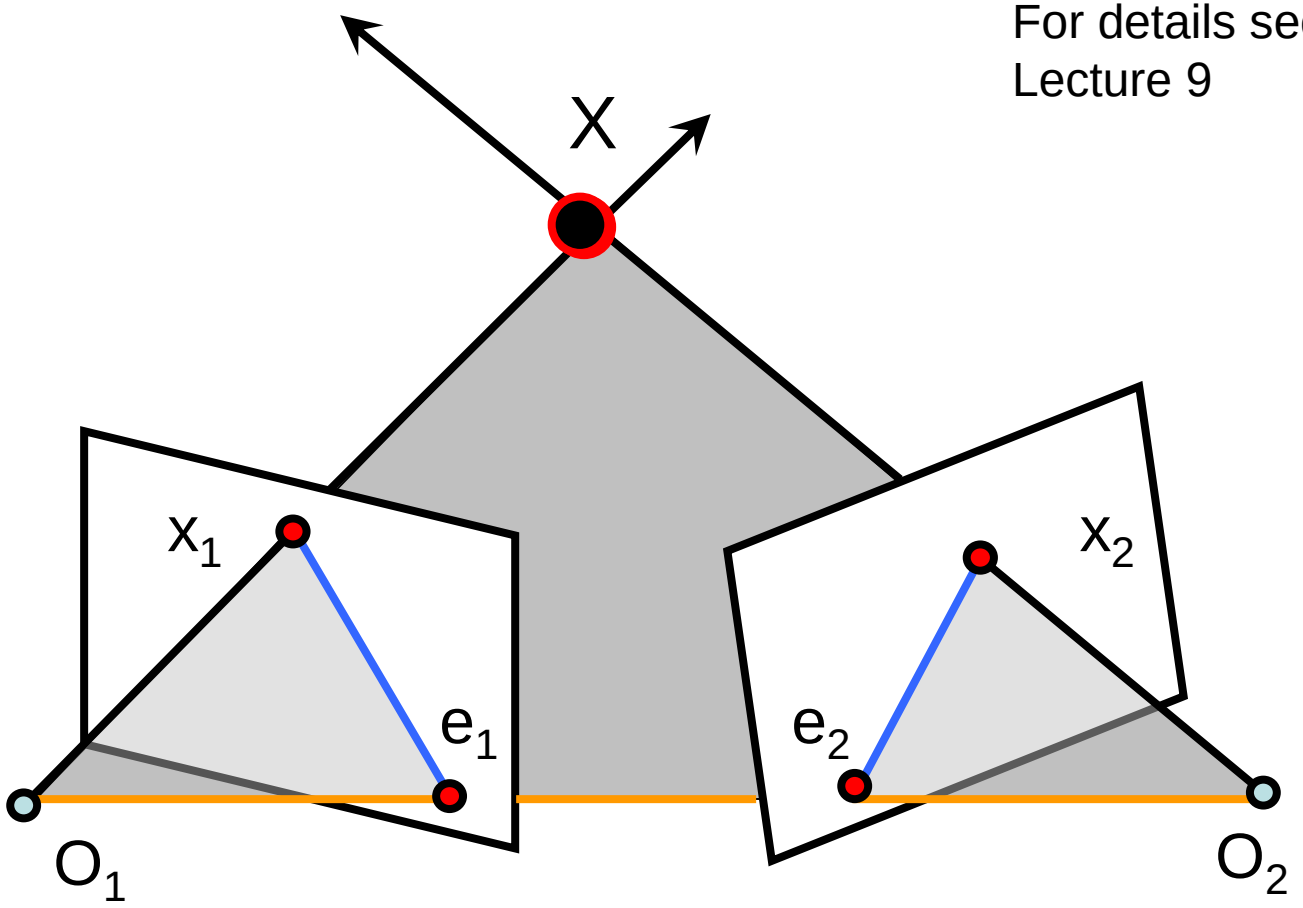

- Epipolar Plane
- Baseline
- Epipolar Lines

- Epipoles $e_1$, $e_2$

  = intersections of baseline with image planes
  = projections of the other camera center

# Epipolar geometry

For details see CS131A
Lecture 9

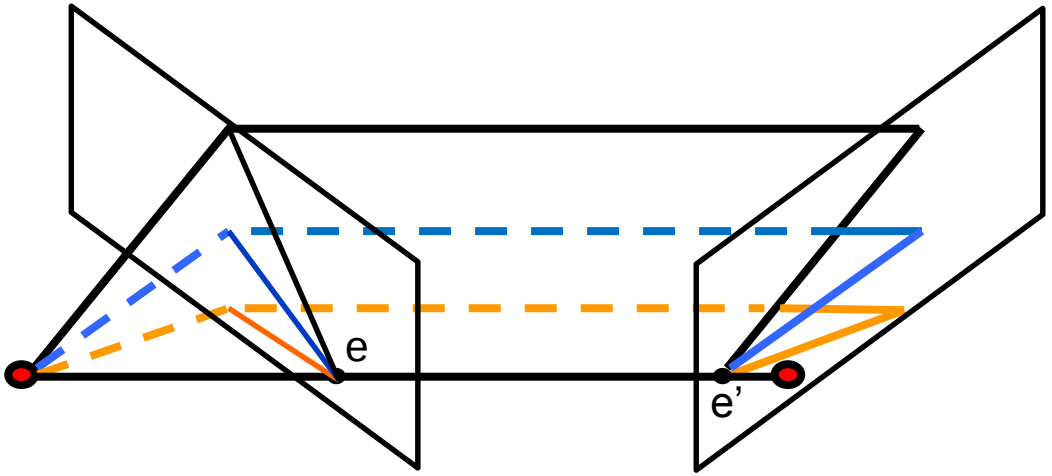


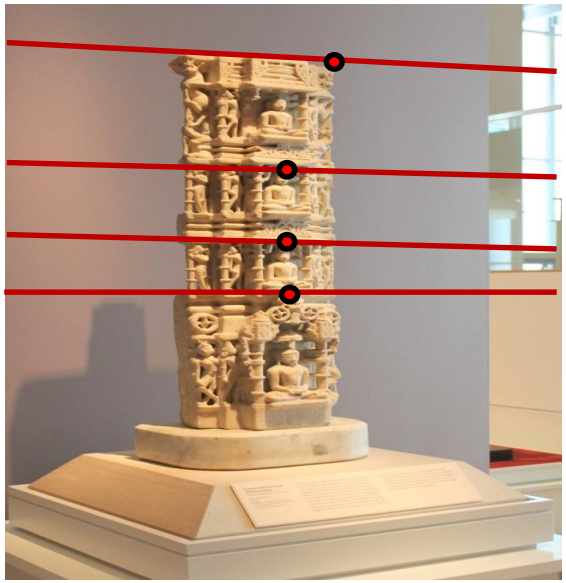

- Epipolar Plane
- Baseline
- Epipolar Lines

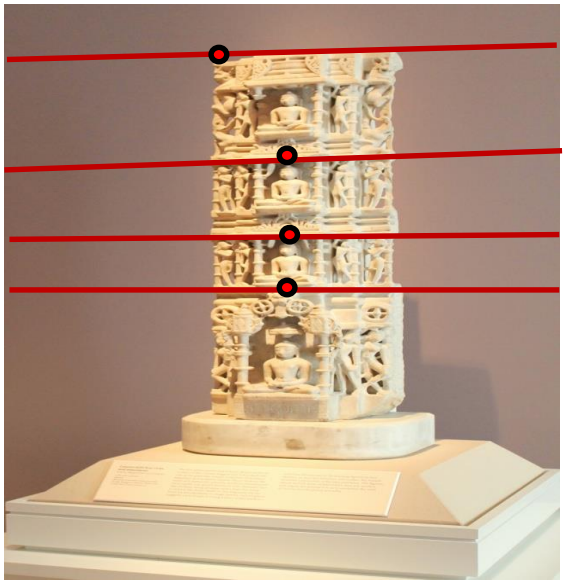
- Epipoles $e_1$, $e_2$

  = intersections of baseline with image planes
  = projections of the other camera center

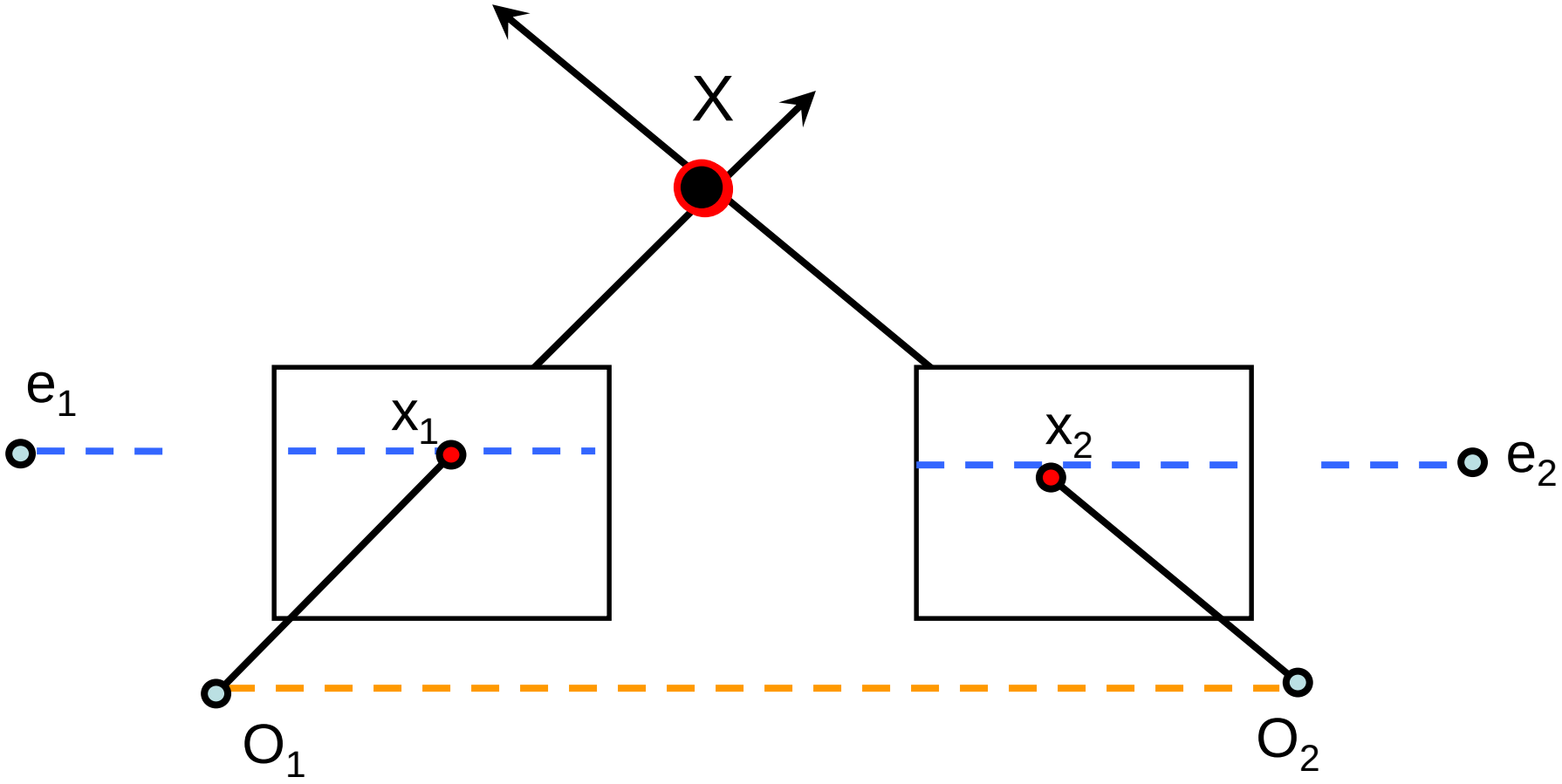
# Example: Converging image planes



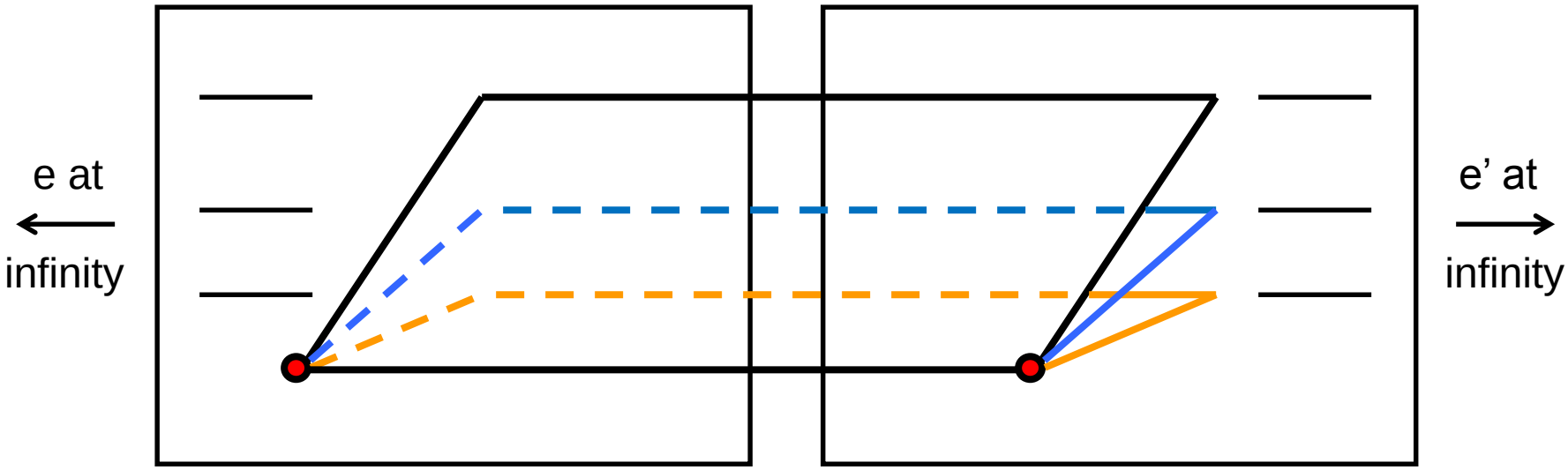


# Example: Parallel image planes

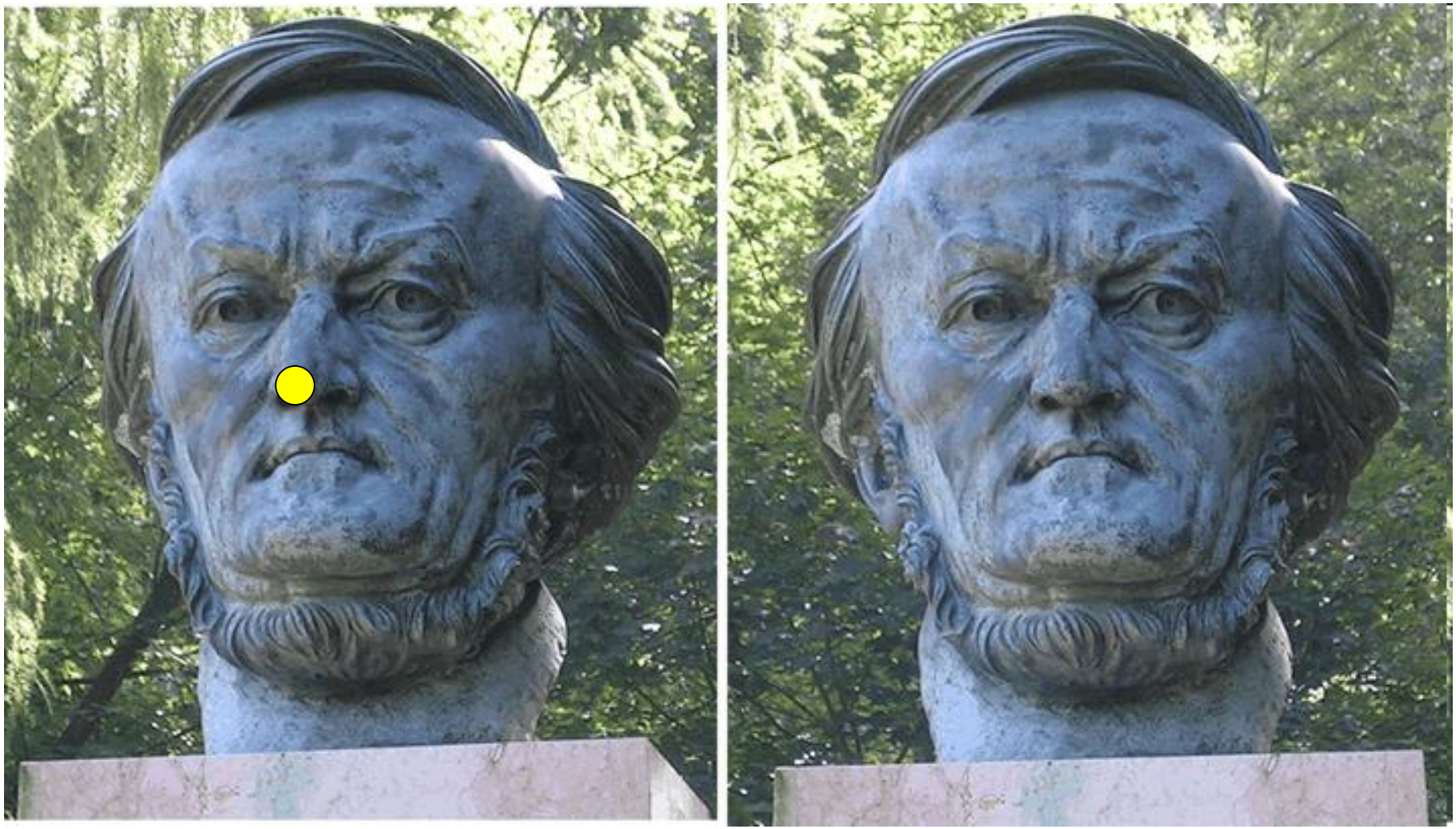



- Baseline intersects the image plane at infinity
- Epipoles are at infinity
- Epipolar lines are parallel to x axis

# Example: Parallel Image Planes

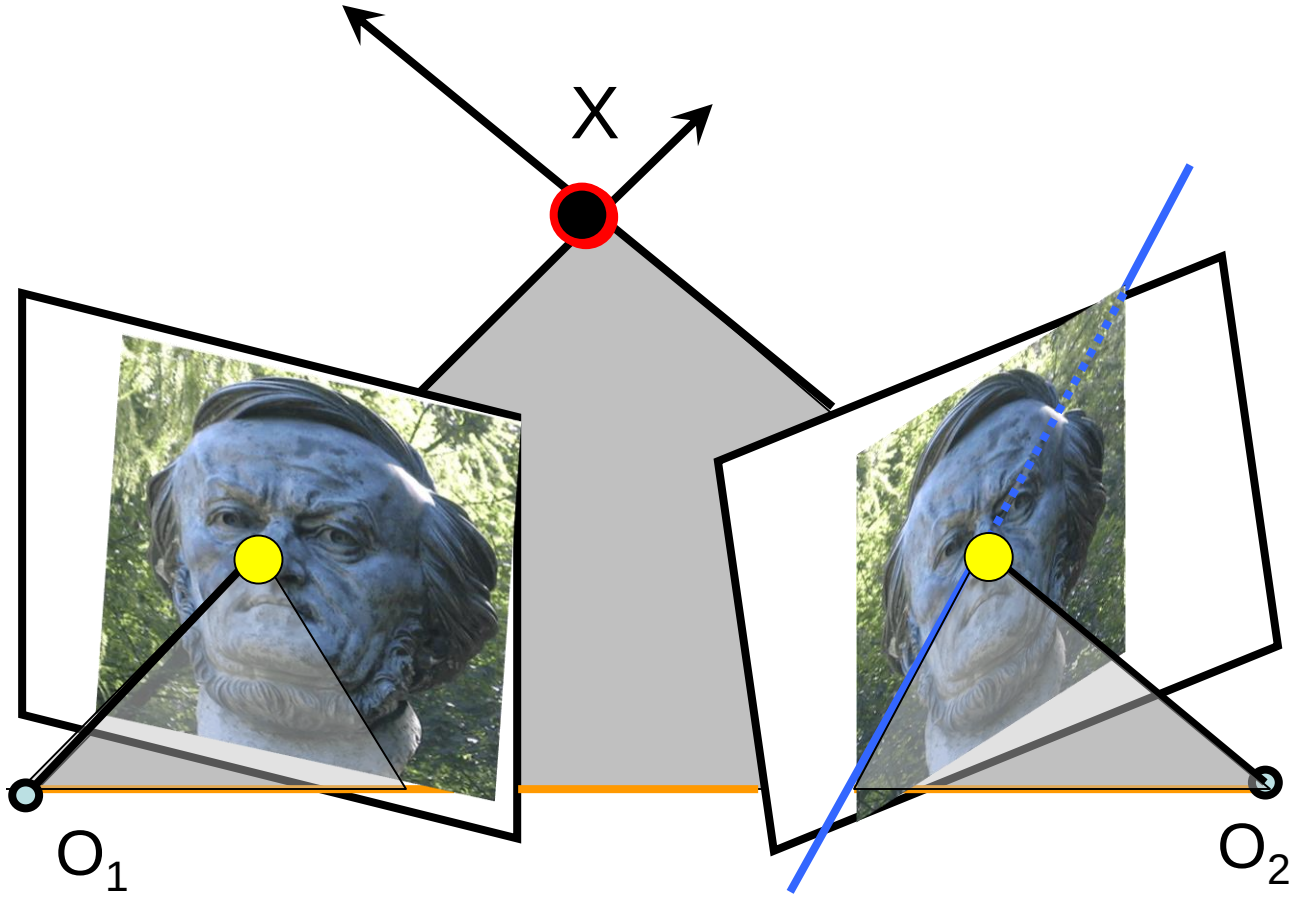



# Why are epipolar constraints useful?

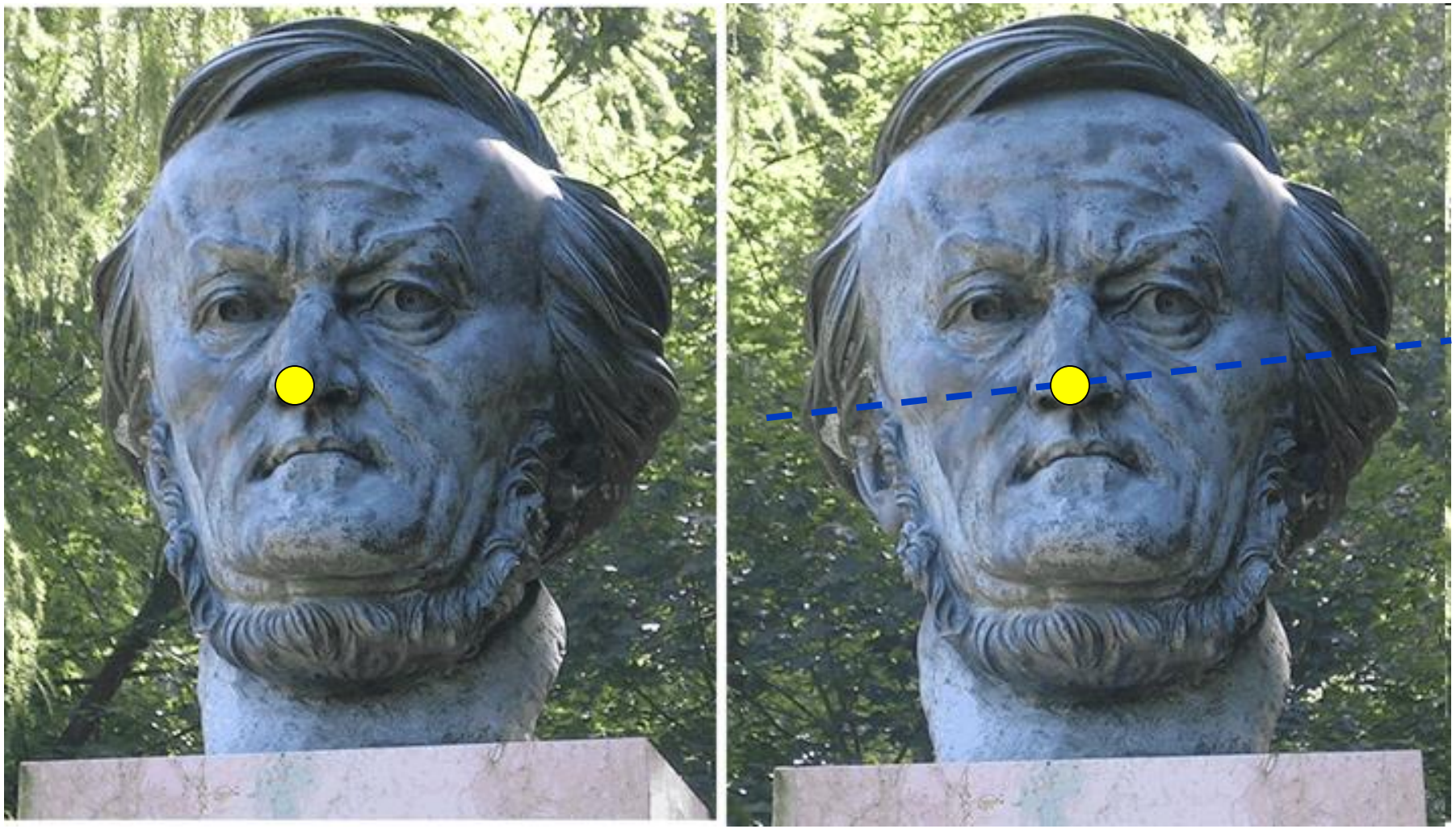


- Two views of the same object
- Suppose I know the camera positions and camera matrices
- Given a point on left image, how can I find the corresponding point on right image?

# Why are epipolar constraints useful?

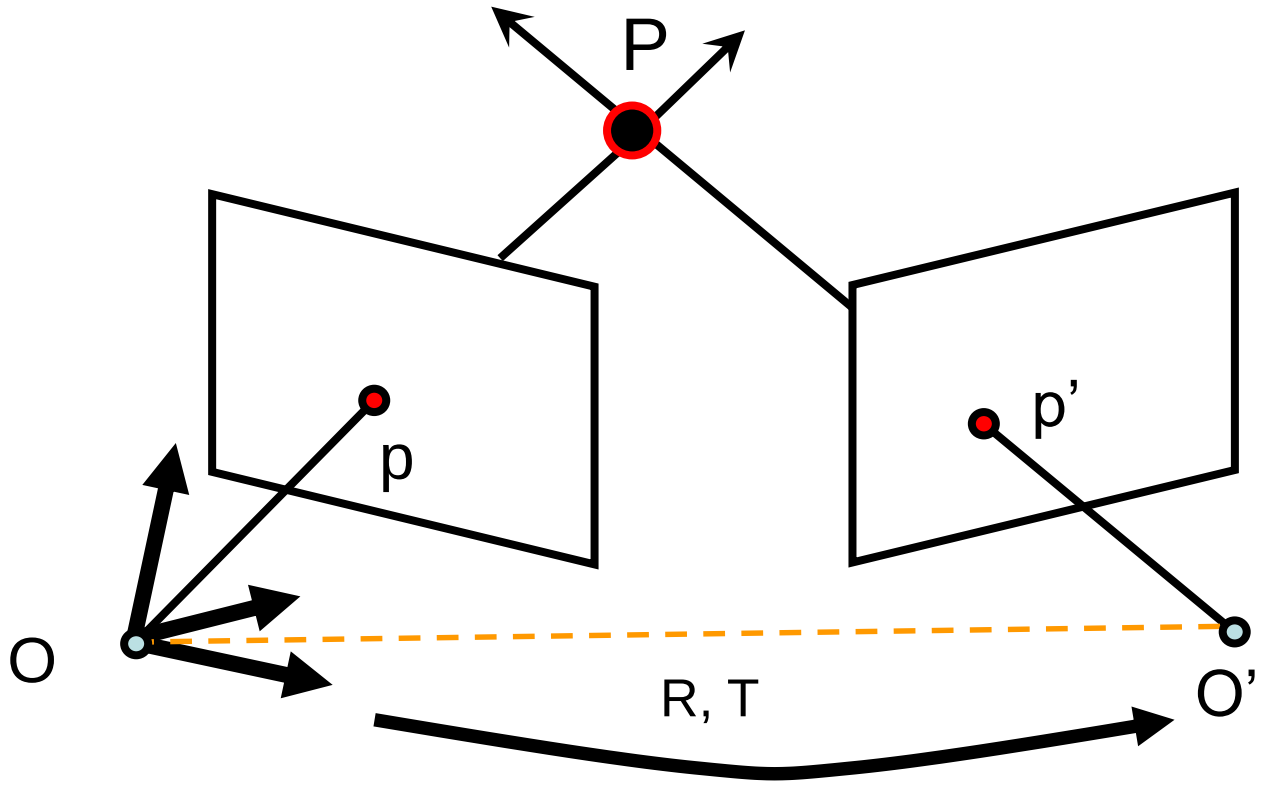


# Why are epipolar constraints useful?



- Two views of the same object
- Suppose I know the camera positions and camera matrices
- Given a point on left image, how can I find the corresponding point on right image?

# Essential matrix




Assume camera matrices are known

$$p^T \cdot E \; p' = 0$$

$$E = [T_\times] \cdot R$$

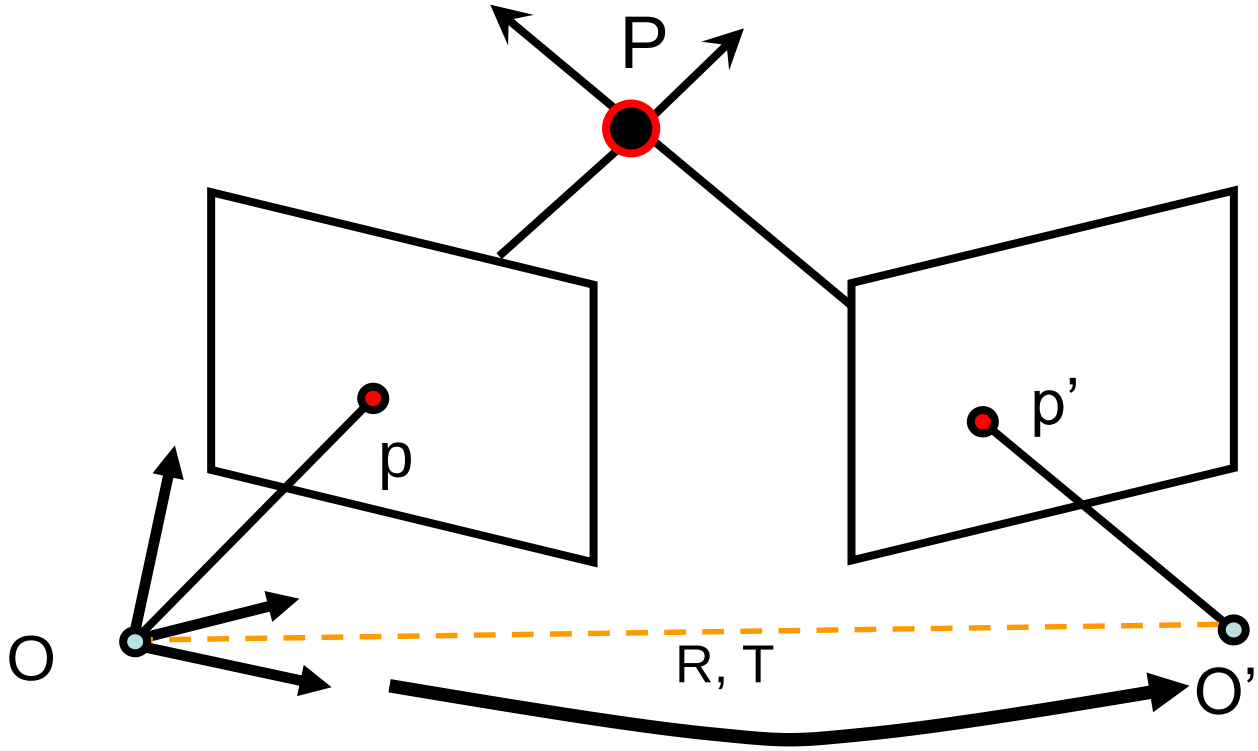
E = essential matrix

(Longuet-Higgins, 1981)

# Cross product as matrix multiplication

$$\mathbf{a}\times\mathbf{b}=\begin{bmatrix}0 & -a_z & a_y\\ a_z & 0 & -a_x\\ -a_y & a_x & 0\end{bmatrix}\begin{bmatrix}b_x\\ b_y\\ b_z\end{bmatrix}=[\mathbf{a}_\times]\mathbf{b}$$

# Epipolar Constraint



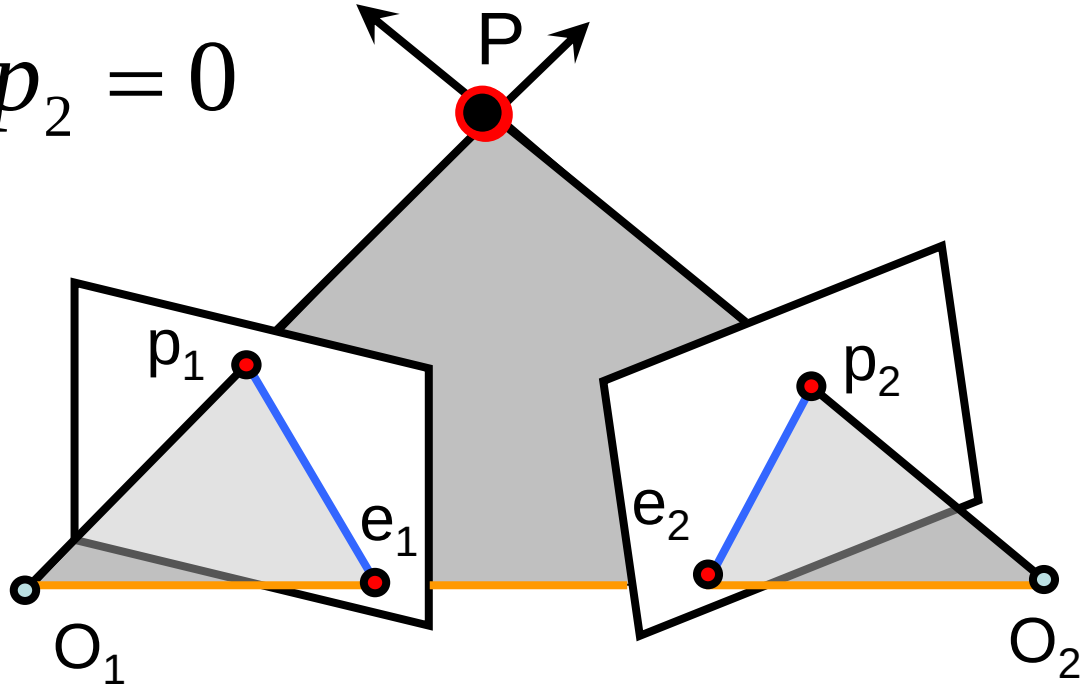

$$p^T F p' = 0$$

**F =** Fundamental Matrix
(Faugeras and Luong, 1992)

$$F = K^{-T} \cdot [T_\times] \cdot R\, K'^{-1}$$

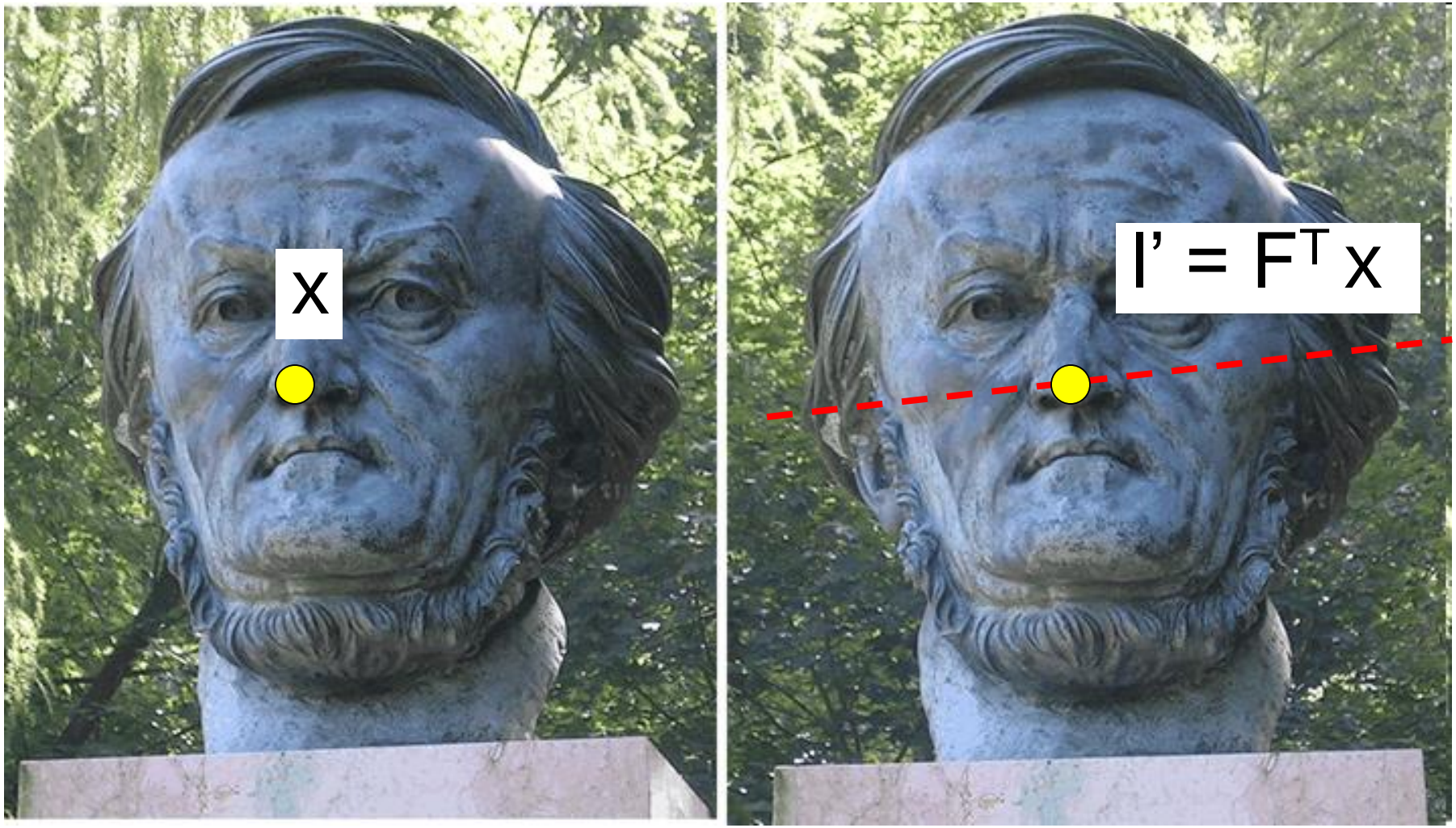
# Epipolar Constraint

$$p_1^{\ T} \cdot F \ p_2 = 0$$




- F $p_2$ is the epipolar line associated with $p_2$ ($l_1$ = F $p_2$)
- $F^T$ $p_1$ is the epipolar line associated with $x_1$ ($l_2$ = $F^T$ $p_1$)
- F $e_2$ = 0 and $F^T$ $e_1$ = 0
- F is 3x3 matrix; 7 DOF
- F is singular (rank two)

# Why F is useful?




- Suppose F is known
- No additional information about the scene and camera is given
- Given a point on left image, how can I find the corresponding point on right image?

# Why F is useful?

- F captures information about the epipolar geometry of 2 views + camera parameters

- MORE IMPORTANTLY: F gives constraints on how the scene changes under view point transformation (without reconstructing the scene!)

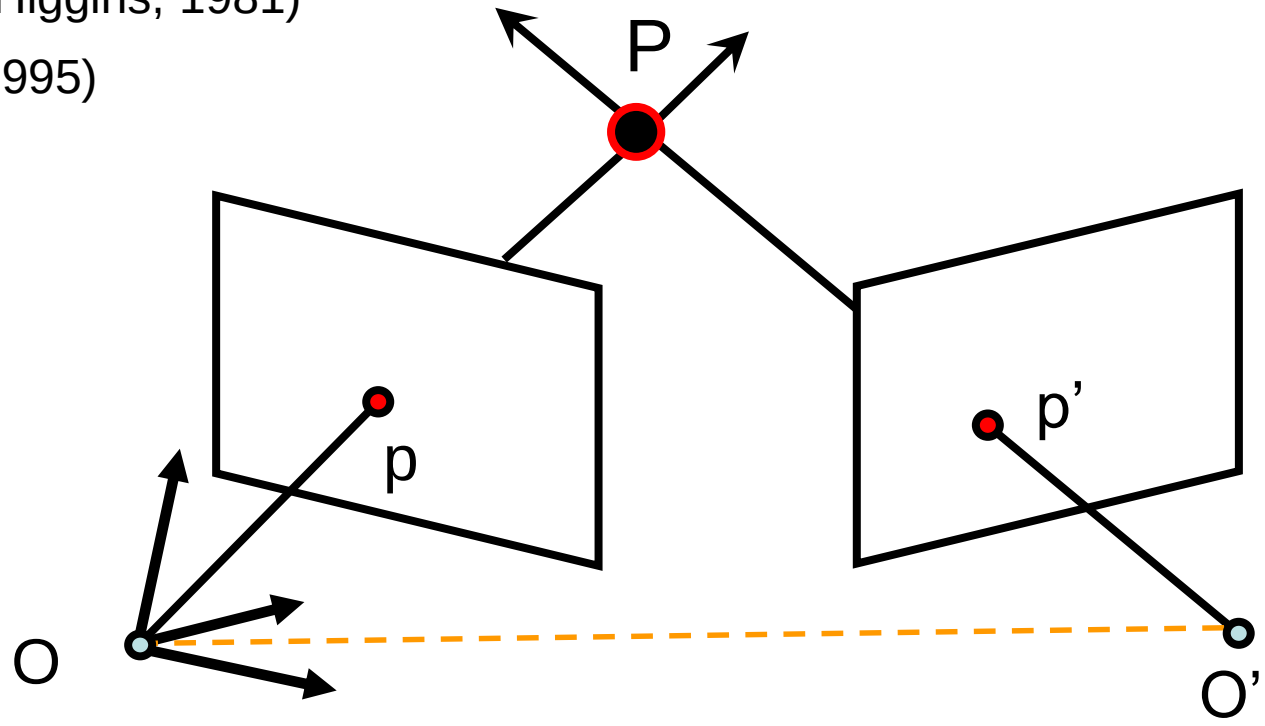
- Powerful tool in:
  - 3D reconstruction
  - Multi-view object/scene matching

# Estimating F

## The Eight-Point Algorithm

(Longuet-Higgins, 1981)

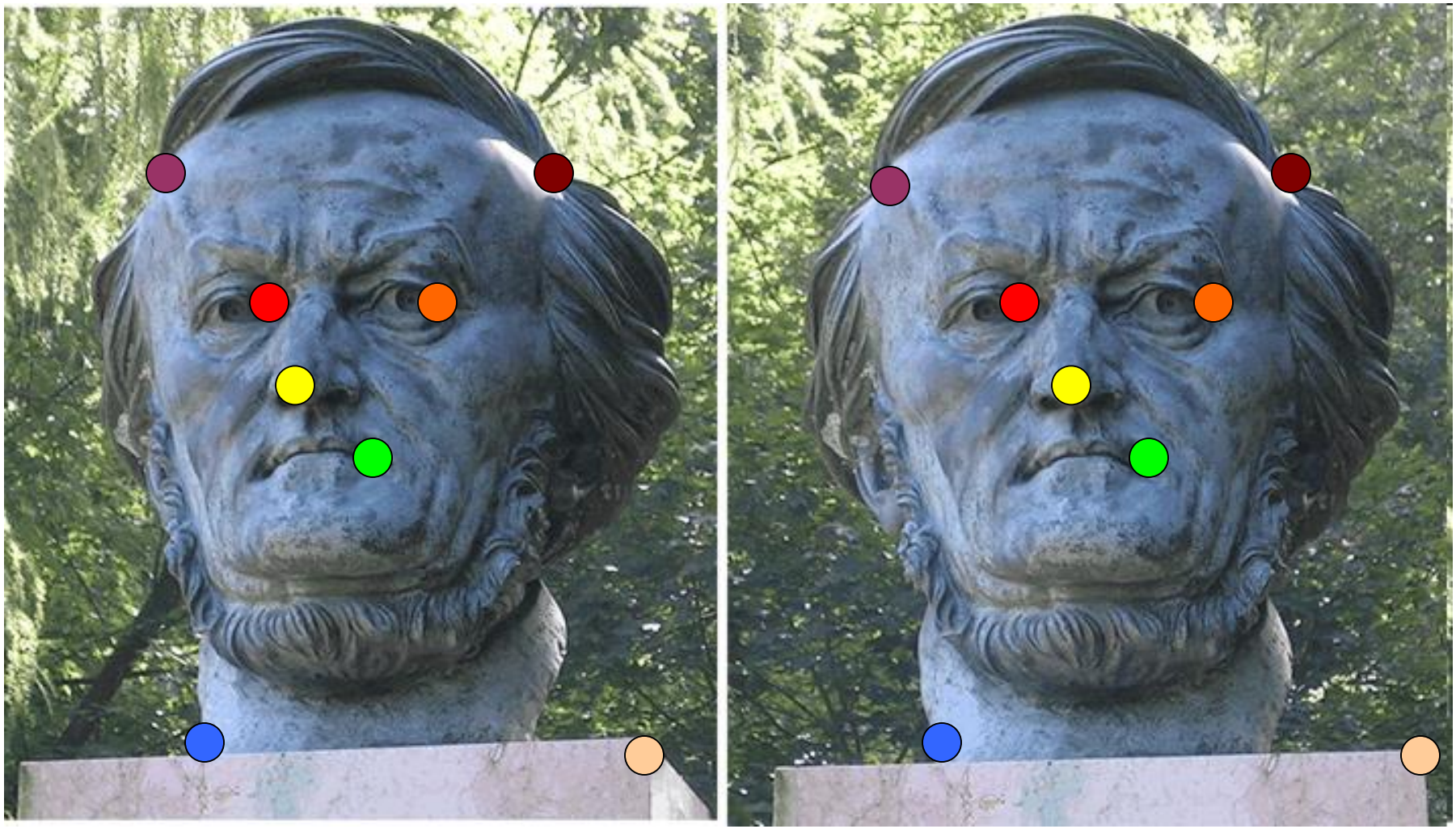
(Hartley, 1995)




$$P \rightarrow p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \qquad P \rightarrow p' = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}$$

$$p^T F p' = 0$$

# Estimating F



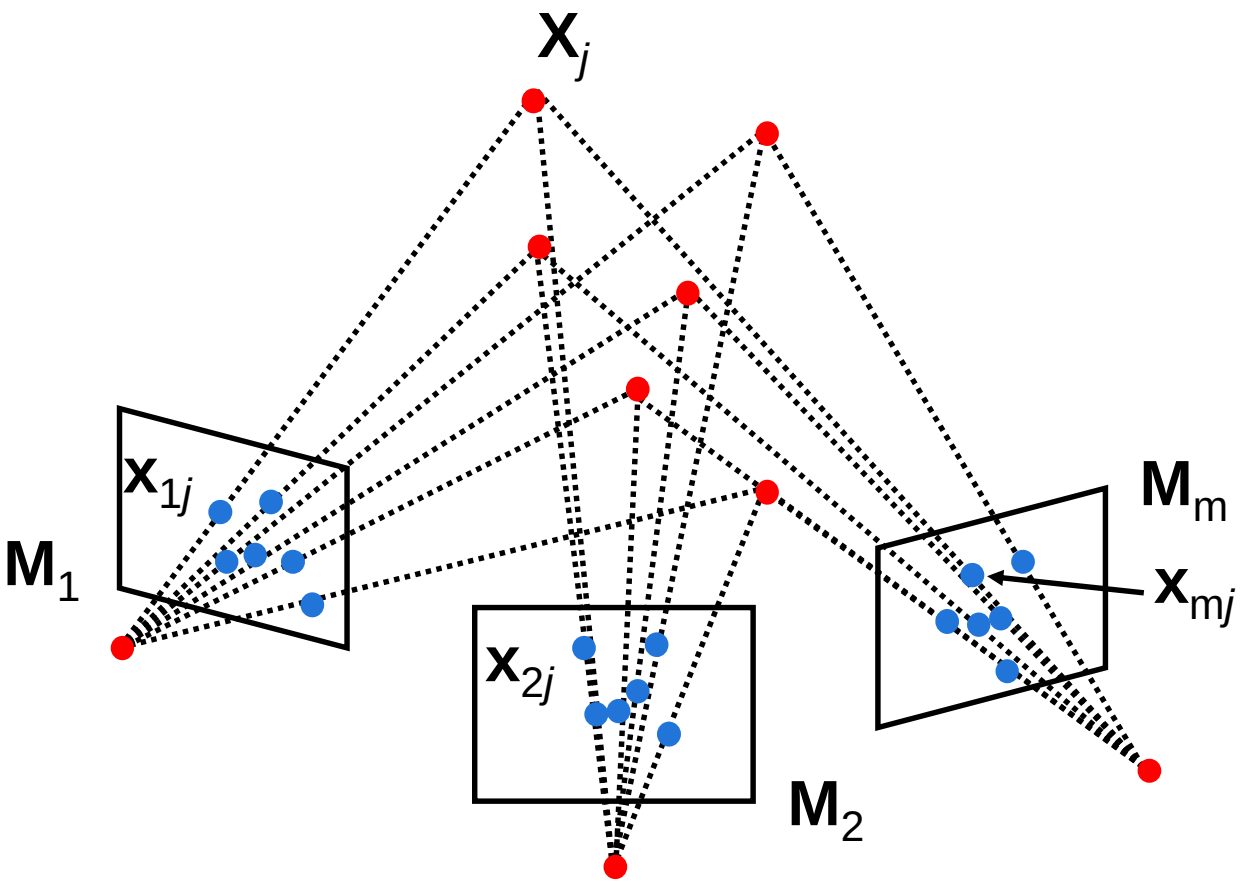
**OPENCV: findFundamentalMat**

# CS231M · Mobile Computer Vision

## Structure from motion


- Cameras
- Epipolar geometry
- Structure from motion

# Structure from motion problem



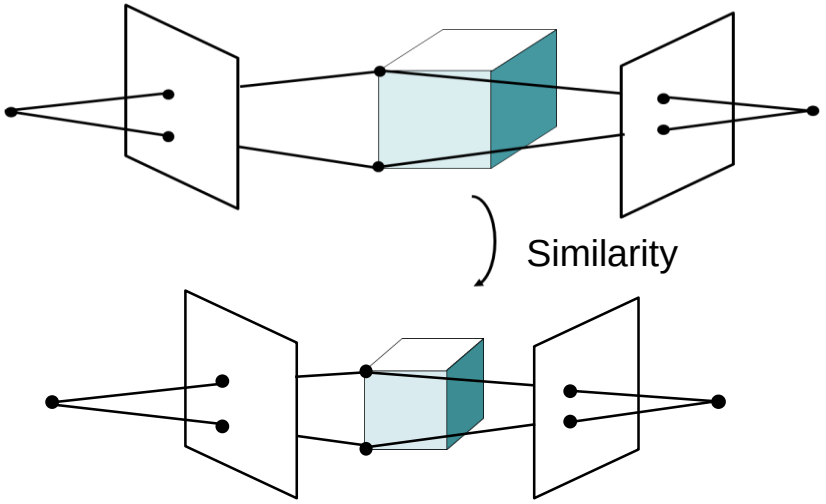

From the mxn correspondences $\mathbf{x}_{ij}$, can we estimate:

| | |
|---|---|
| •*m* projection matrices $\mathbf{M}_i$ | motion |
| •*n* 3D points $\mathbf{X}_j$ | structure |

# Similarity Ambiguity

- The scene is determined by the images only up a similarity transformation (rotation, translation and scaling)

- This is called **metric reconstruction**

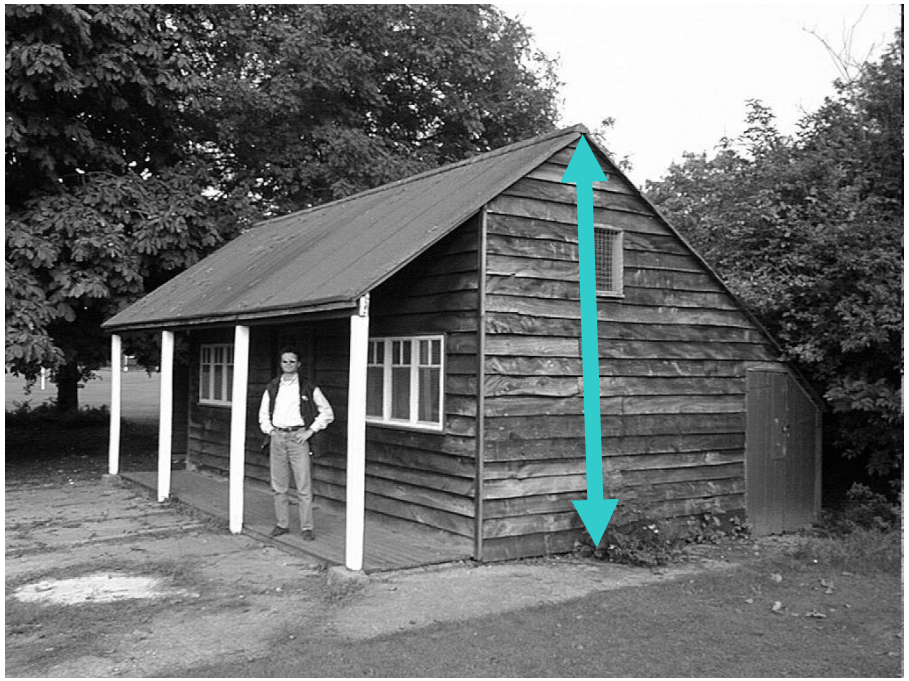



- The ambiguity exists even for (intrinsically) calibrated cameras
- For calibrated cameras, the similarity ambiguity is the only ambiguity

[Longuet-Higgins '81]

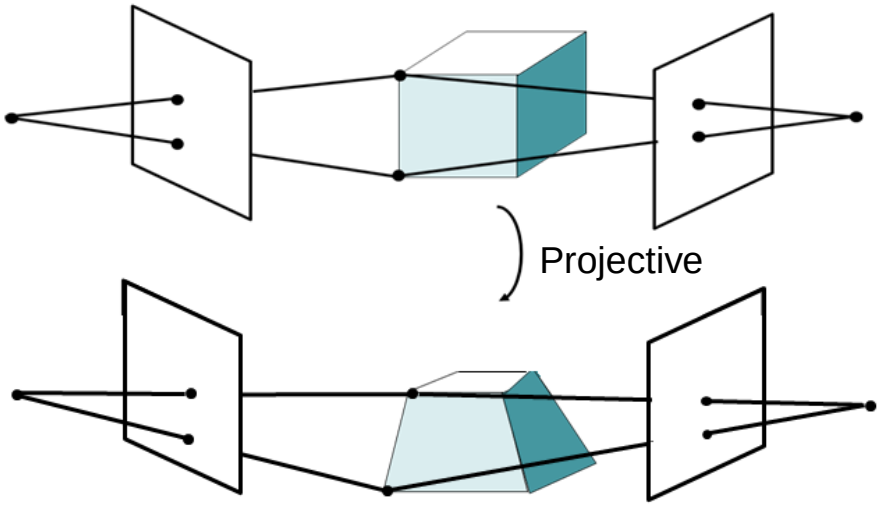
# Similarity Ambiguity

- It is impossible based on the images alone to estimate the absolute scale of the scene (i.e. house height)



http://www.robots.ox.ac.uk/~vgg/projects/SingleView/models/hut/hutme.wrl

# Structure from Motion Ambiguities



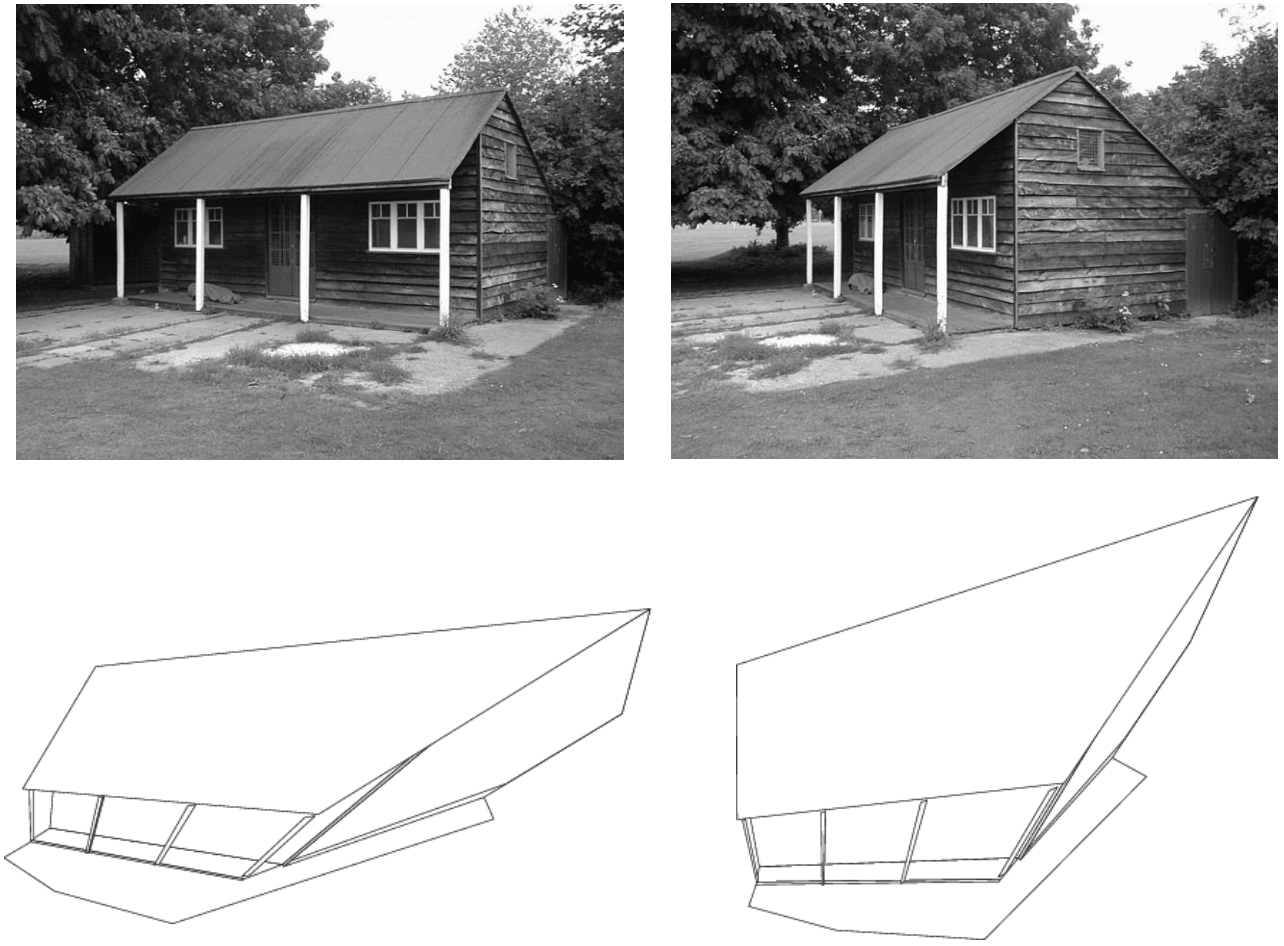

- In the general case (nothing is known) the ambiguity is expressed by an arbitrary affine or projective transformation

$$x_j = M_i X_j \qquad M_i = K_i \begin{bmatrix} R_i & T_i \end{bmatrix}$$

$$H X_j \qquad M_j H^{-1}$$

$$x_j = M_i X_j = \left(M_i H^{-1}\right)\left(H X_j\right)$$

# Projective Ambiguity

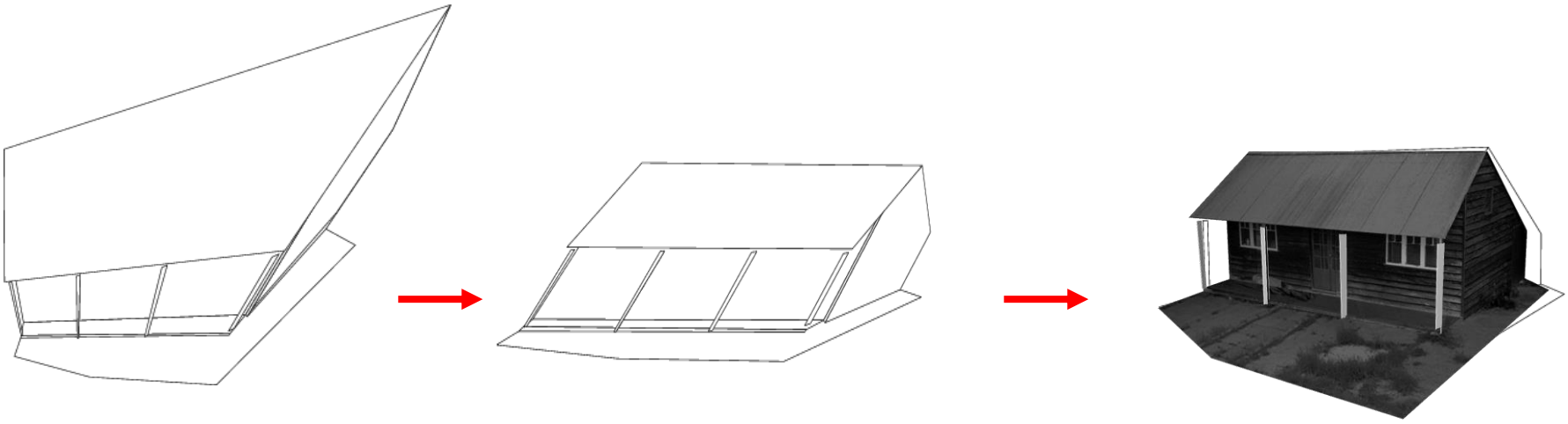


R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd edition, 2003

# Metric reconstruction (upgrade)

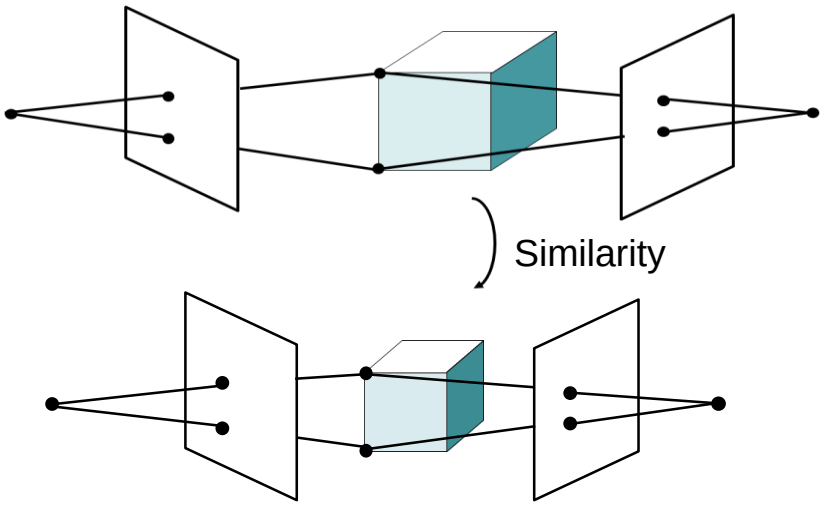
- The problem of recovering the metric reconstruction from the perspective one is called self-calibration

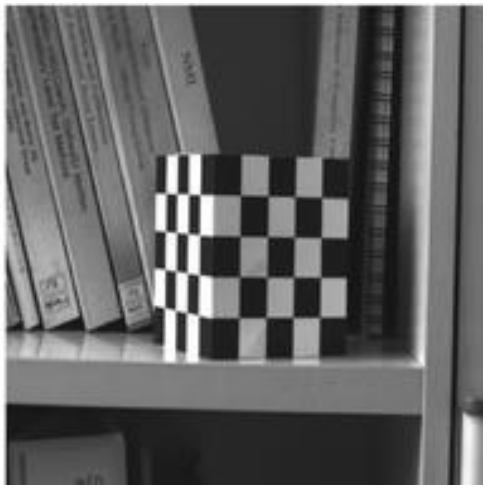
- Stratified reconstruction:
  - from perspective to affine
  - from affine to metric



# Mobile SFM

- Intrinsic camera parameters are known or can be calibrated.
- For calibrated cameras, the similarity ambiguity is the only ambiguity [Longuet-Higgins '81]
- No need for stratified solution or auto-calibration



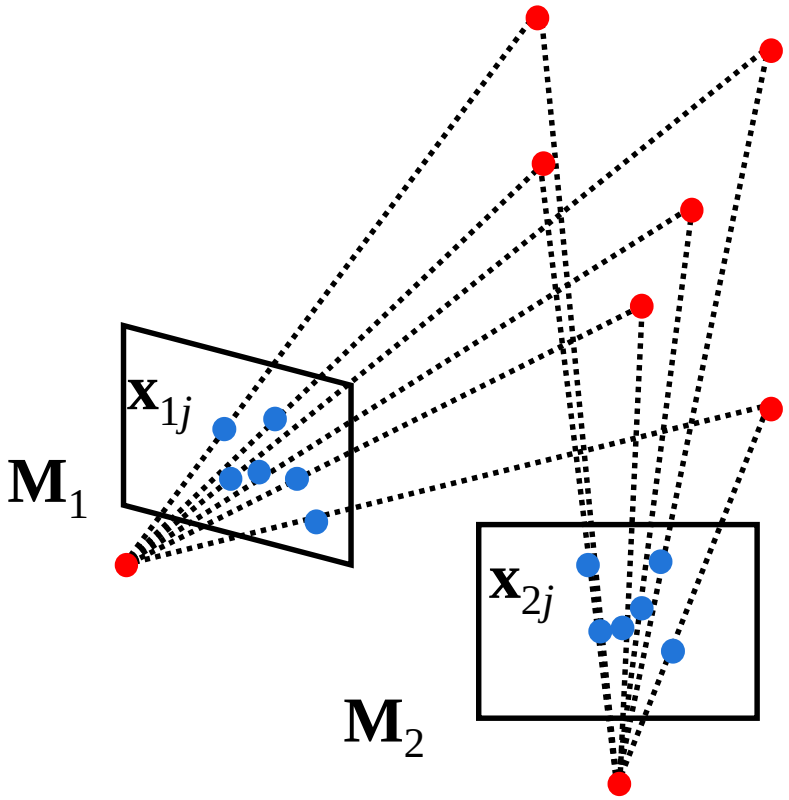



- Metric reconstruction can be determined if a calibration pattern is used or the absolute size of an known object is given.

# Structure-from-Motion Algorithms

- Algebraic approach (by fundamental matrix)
- Factorization method (by SVD)
- Bundle adjustment

# Algebraic approach (2-view case)




$$x_{ij} = M_i \, X_j$$

Apply a projective transformation H such that:

$$M_1 \, H^{-1} = \begin{bmatrix} I & 0 \end{bmatrix} \qquad M_2 \, H^{-1} = \begin{bmatrix} A & b \end{bmatrix}$$

Canonical perspective cameras

# Algebraic approach (2-view case)

1. Compute the fundamental matrix **F** from two views (eg. 8 point algorithm)
2. Compute **b** and **A** from **F**

Compute **b** as least sq. solution of $\mathbf{F}\,\mathbf{b} = 0$, with $|\mathbf{b}|=1$ using SVD; **b** is an epipole

$$\mathbf{A} = -[\mathbf{b}_\times]\,\mathbf{F}$$

3. Use **b** and **A** to estimate projective cameras

$$M_1 = \begin{bmatrix} I & 0 \end{bmatrix} \qquad M_2 = \begin{bmatrix} -[\mathbf{b}_x]\mathbf{F} & \mathbf{b} \end{bmatrix}$$

4. Use these cameras to triangulate and estimate points in 3D

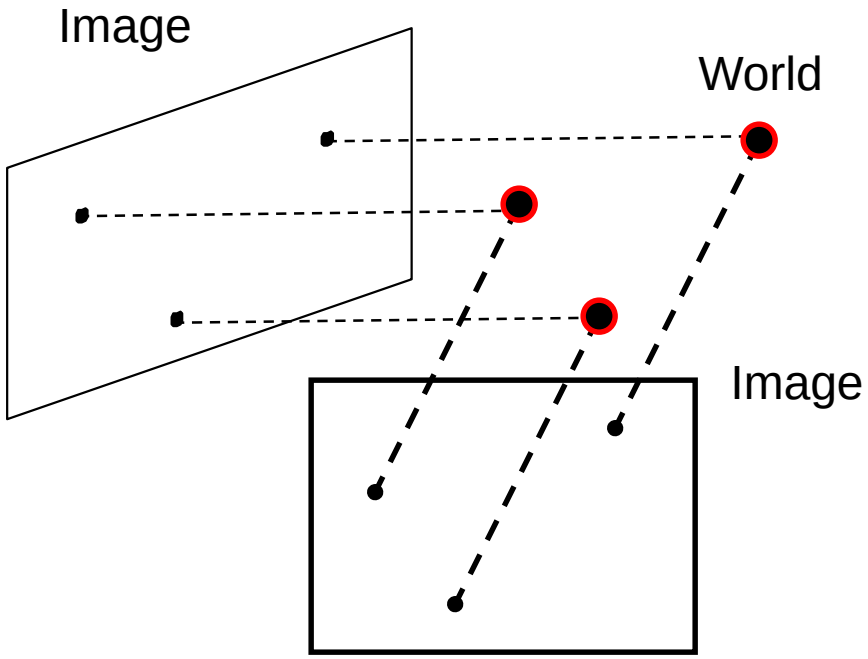
For details, see CS231A, lecture 7

# Structure-from-Motion Algorithms

- Algebraic approach (by fundamental matrix)
- Factorization method (by SVD)
- Bundle adjustment

# Affine structure from motion

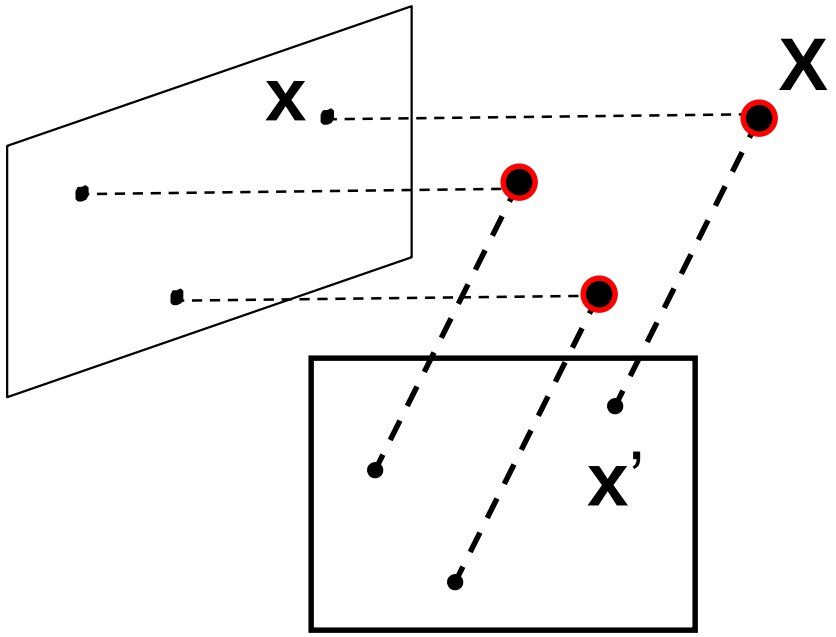
(simpler problem)




From the mxn correspondences $\mathbf{x}_{ij}$, estimate:

- *m* projection matrices $\mathbf{M}_i$ (affine cameras)
- *n* 3D points $\mathbf{X}_j$

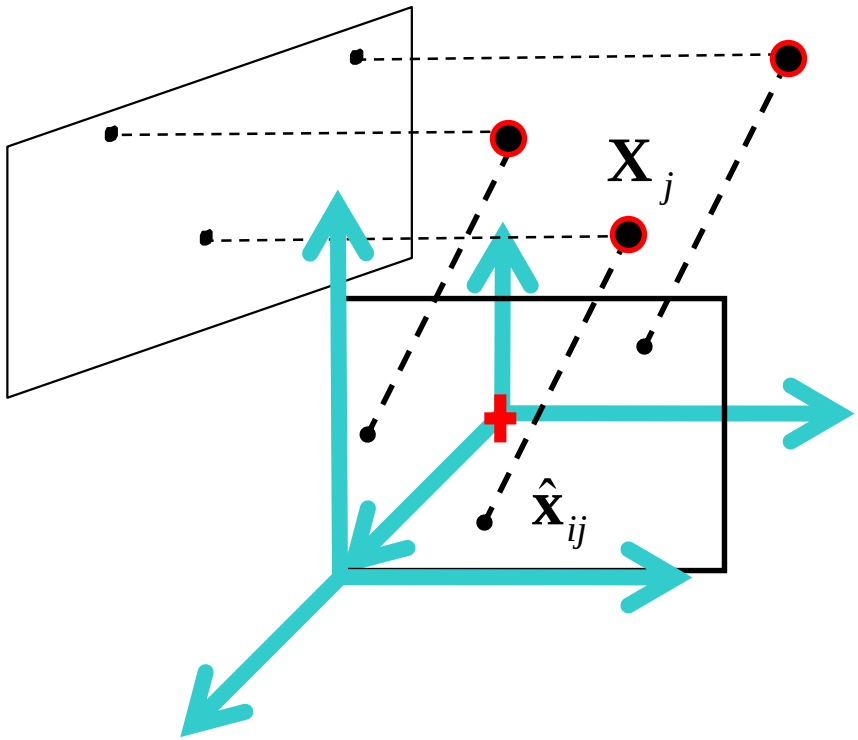
# Affine cameras




Camera matrix M for the affine case

$$\mathbf{x} = \begin{pmatrix} u \\ v \end{pmatrix} = M \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} = \mathbf{A}\mathbf{X} + \mathbf{b}; \qquad \mathrm{M} = \begin{bmatrix} \mathrm{A} & \mathrm{b} \end{bmatrix}$$

# Centering the data




Normalize points w.r.t. centroids of measurements from each image

$$\mathbf{x}_{ij} = \mathbf{A}\mathbf{X}_j + \mathbf{b} \rightarrow \hat{\mathbf{x}}_{ij} = \mathbf{A}_i\mathbf{X}_j$$

# A factorization method - factorization

Let's create a 2m × n data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix}$$

cameras ($2m$)

points ($n$)

# A factorization method - factorization

Let's create a 2m × n data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix} \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}$$

(2*m* × n)

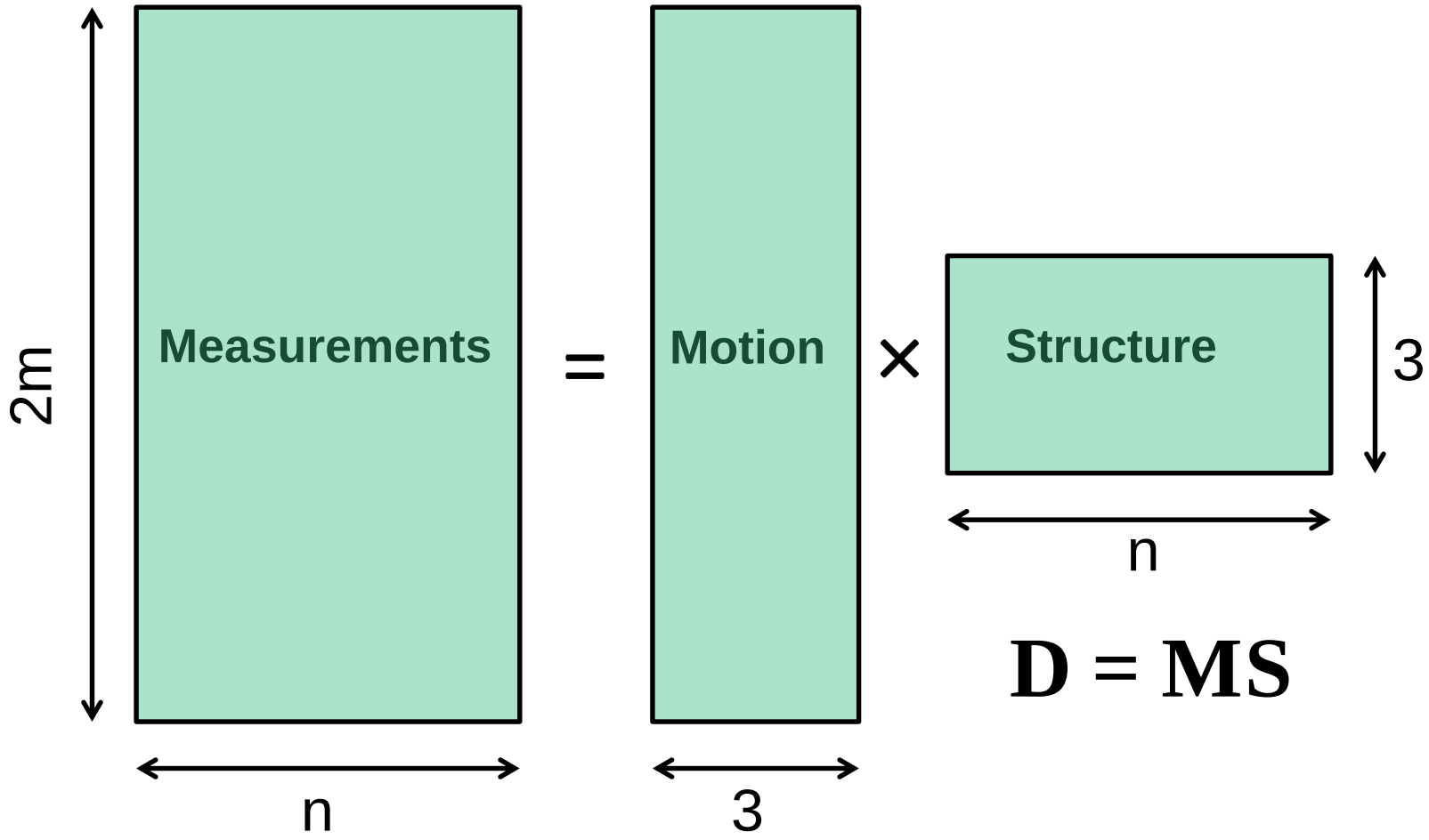
cameras (2*m* × 3) — M

points (3 × *n* ) — S

The measurement matrix **D = M S** has rank 3

(it's a product of a 2mx3 matrix and 3xn matrix)

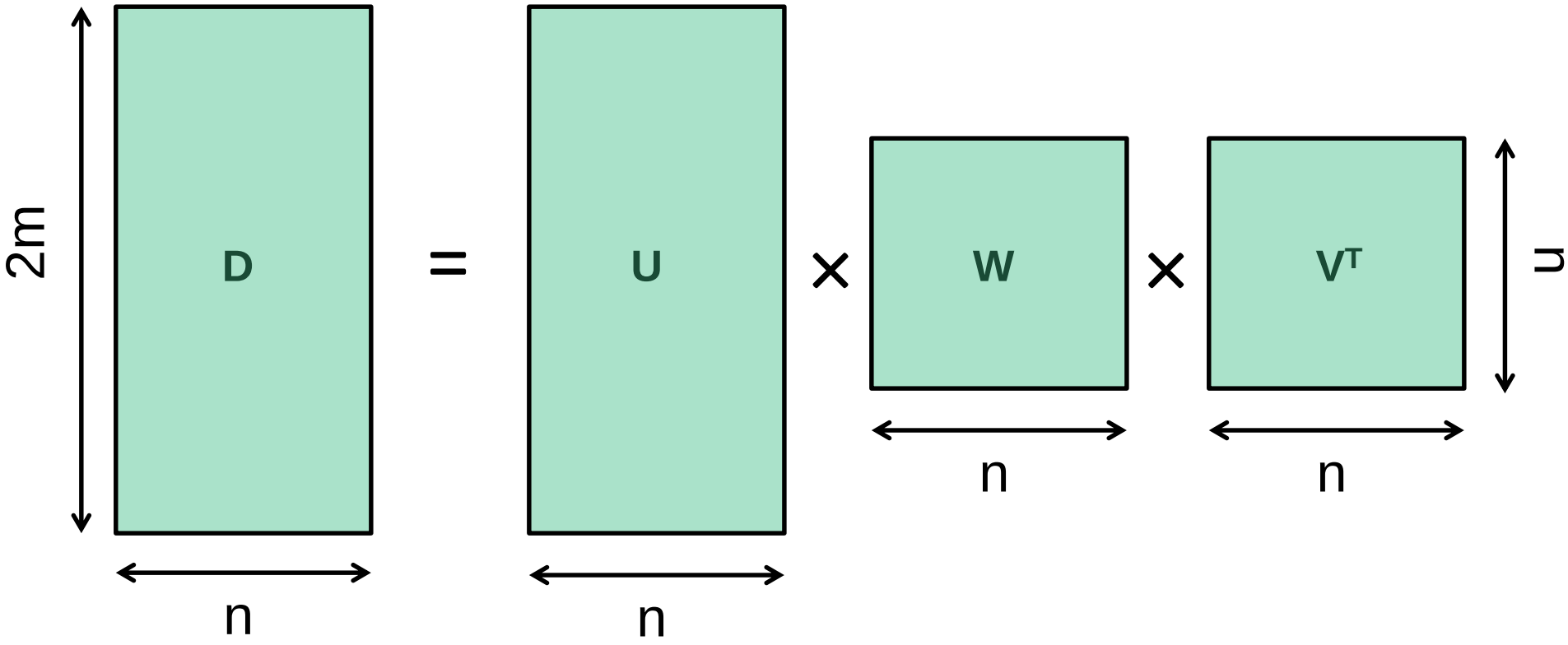
# Factorizing the Measurement Matrix




$$\mathbf{D = MS}$$

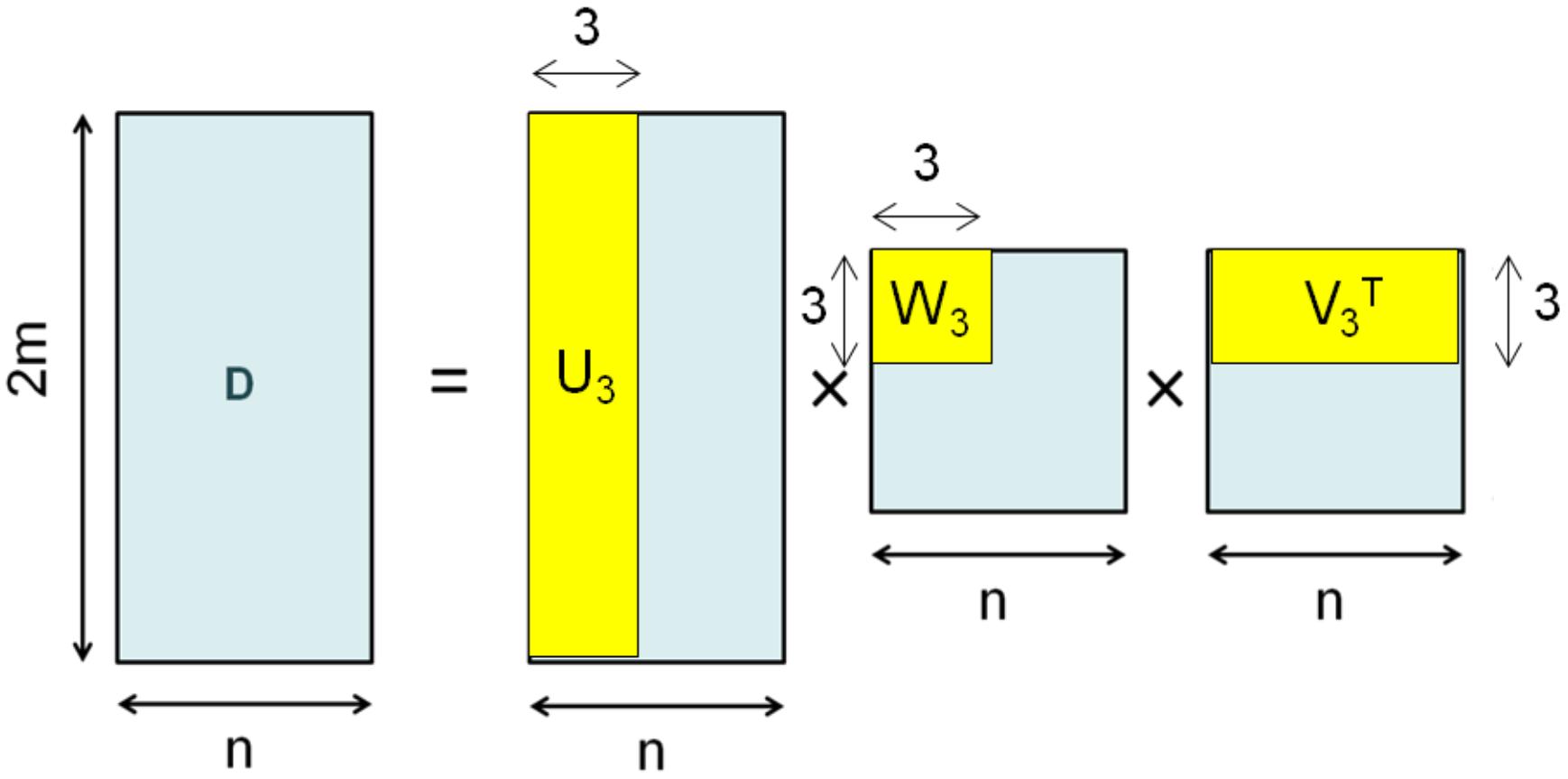
# Factorizing the Measurement Matrix

- Singular value decomposition of D:

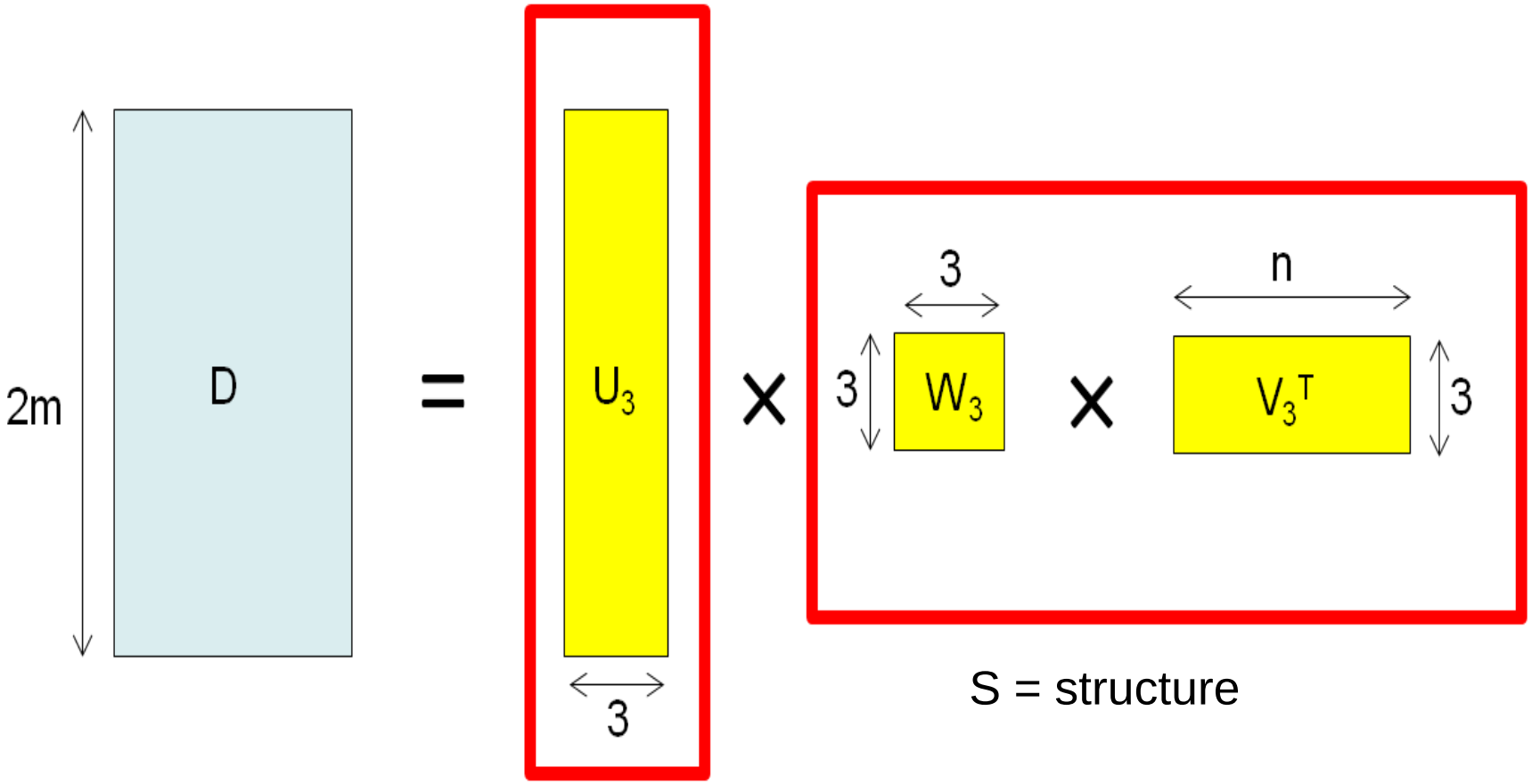


# Factorizing the Measurement Matrix

Since rank (D)=3, there are only 3 non-zero singular values



# Factorizing the Measurement Matrix



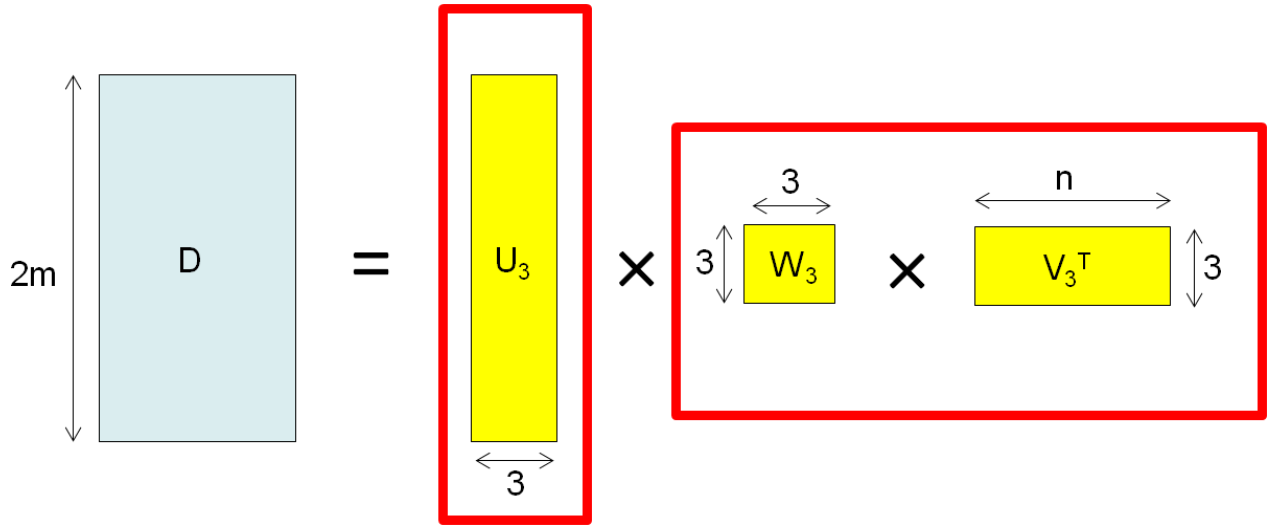

S = structure

M = Motion (cameras)

# Factorizing the Measurement Matrix

What is the issue here? D has rank>3 because of:

- measurement noise
- affine approximation

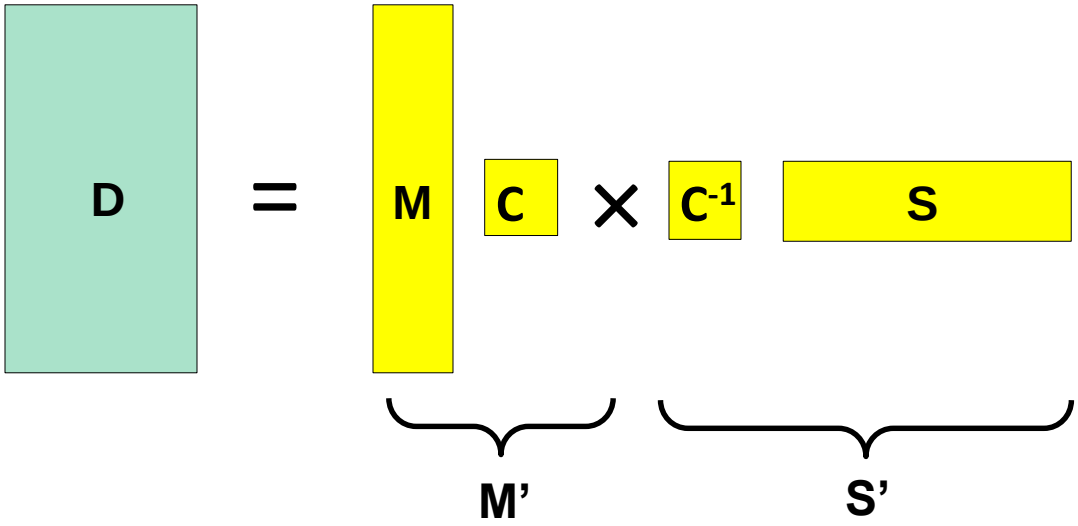



**Theorem:** When $\mathbf{D}$ has a rank greater than $p$, $\mathbf{U}_p\mathbf{W}_p\mathbf{V}_p^T$ is the best possible rank-$p$ approximation of $\mathbf{A}$ in the sense of the Frobenius norm.

$$\mathbf{D} = \mathbf{U}_3\mathbf{W}_3\mathbf{V}_3^T \qquad \begin{cases} \mathbf{A}_0 = \mathbf{U}_3 \\ \mathbf{P}_0 = \mathbf{W}_3\mathbf{V}_3^T \end{cases}$$

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} = \sqrt{\sum_{i=1}^{\min\{m,n\}}\sigma_i^2}$$

# Affine Ambiguity



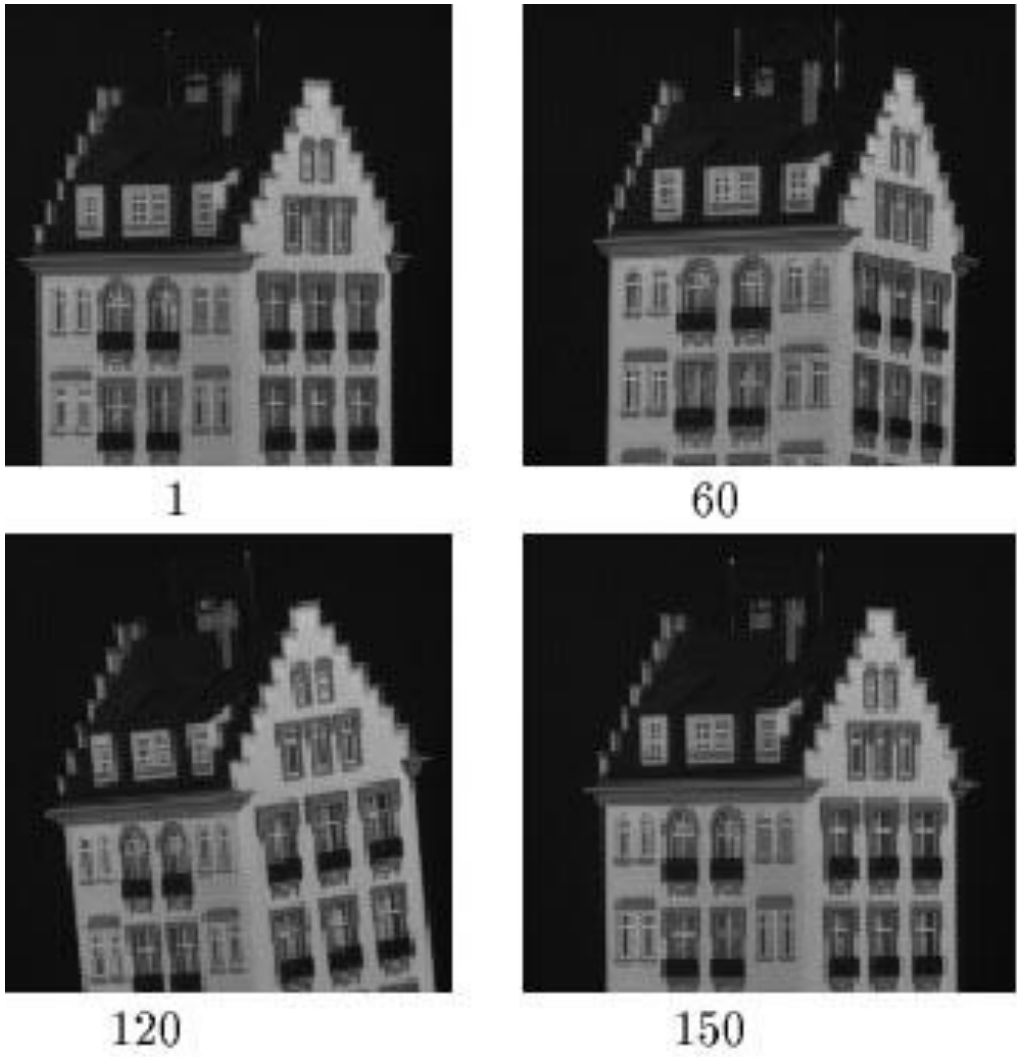

- The decomposition is not unique. We get the same **D** by using any 3×3 matrix **C** and applying the transformations:

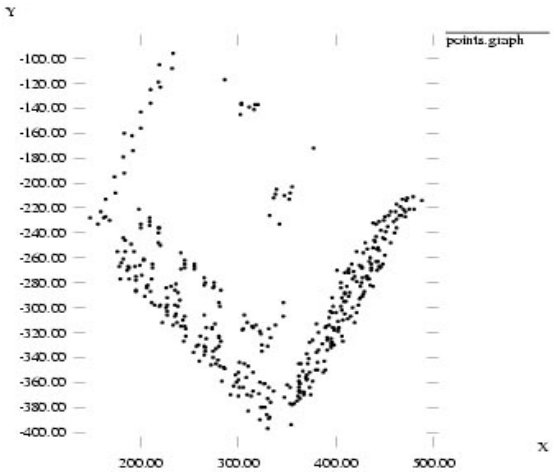
  $\mathbf{M} \rightarrow \mathbf{MC}$

  $\mathbf{S} \rightarrow \mathbf{C^{-1}S}$

- Additional constraints must be enforced to resolve this ambiguity

# Reconstruction results




C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

# Structure-from-Motion Algorithms

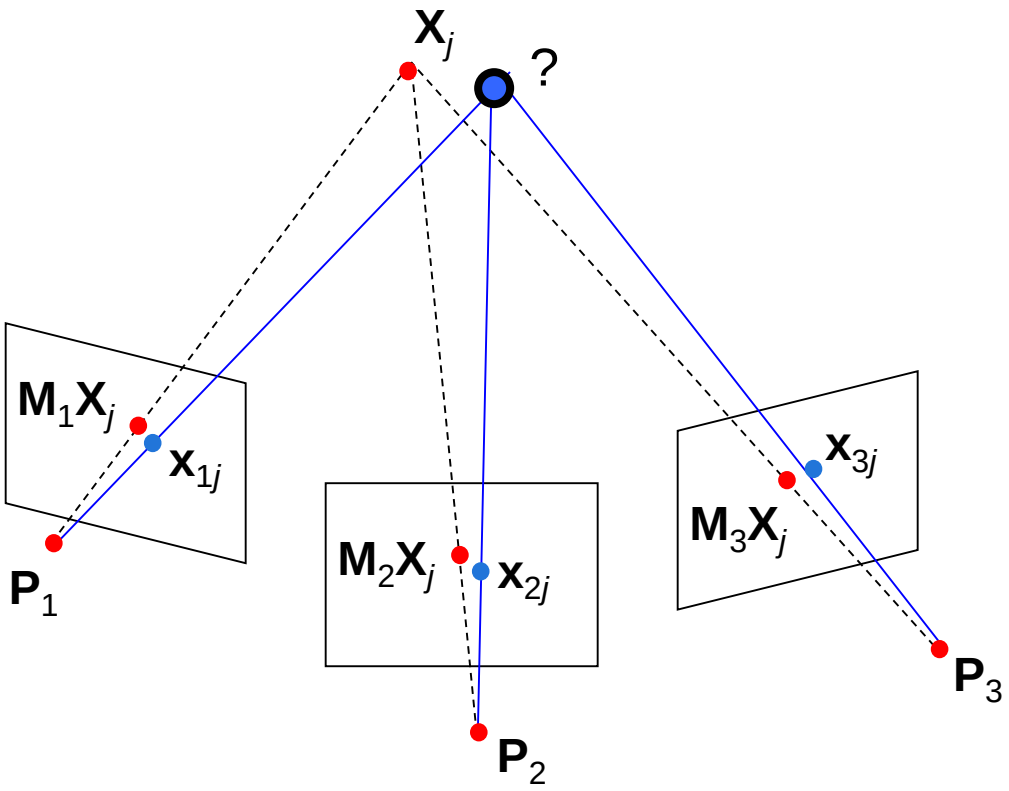
- Algebraic approach (by fundamental matrix)
- Factorization method (by SVD)
- Bundle adjustment

# Bundle adjustment

Non-linear method for refining structure and motion

Minimizing re-projection error

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



# Bundle adjustment

Non-linear method for refining structure and motion

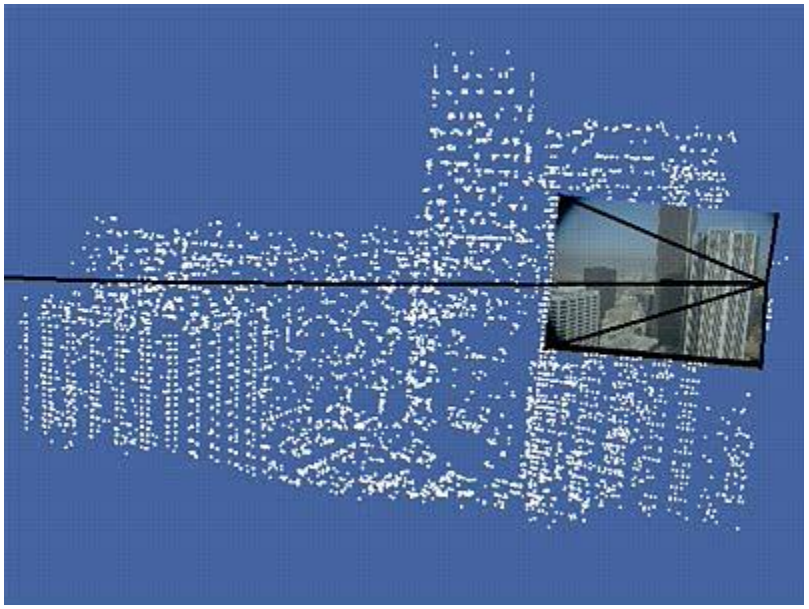
Minimizing re-projection error

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$

- Advantages
  - Handle large number of views
  - Handle missing data
  - Can leverage standard optimization packaged such as Levenberg-Marquardt
- Limitations
  - Large minimization problem (parameters grow with number of views)
  - Requires good initial condition

Used as the final step of SFM

# Results and applications

Courtesy of Oxford **Visual Geometry Group**

Lucas & Kanade, 81
Chen & Medioni, 92
Debevec et al., 96
Levoy & Hanrahan, 96
Fitzgibbon & Zisserman, 98
Triggs et al., 99
Pollefeys et al., 99
Kutulakos & Seitz, 99

Levoy et al., 00
Hartley & Zisserman, 00
Dellaert et al., 00
Rusinkiewic et al., 02
Nistér, 04
Brown & Lowe, 04
Schindler et al, 04
Lourakis & Argyros, 04
Colombo et al. 05

Golparvar-Fard, et al. JAEI 10
Pandey et al. IFAC , 2010
Pandey et al. ICRA 2011
Microsoft's PhotoSynth
Snavely et al., 06-08
Schindler et al., 08
Agarwal et al., 09
Frahm et al., 10

# CS231M · Mobile Computer Vision

## Next lecture:

Example of a SFM pipeline for mobile devices: **the VSLAM pipeline**