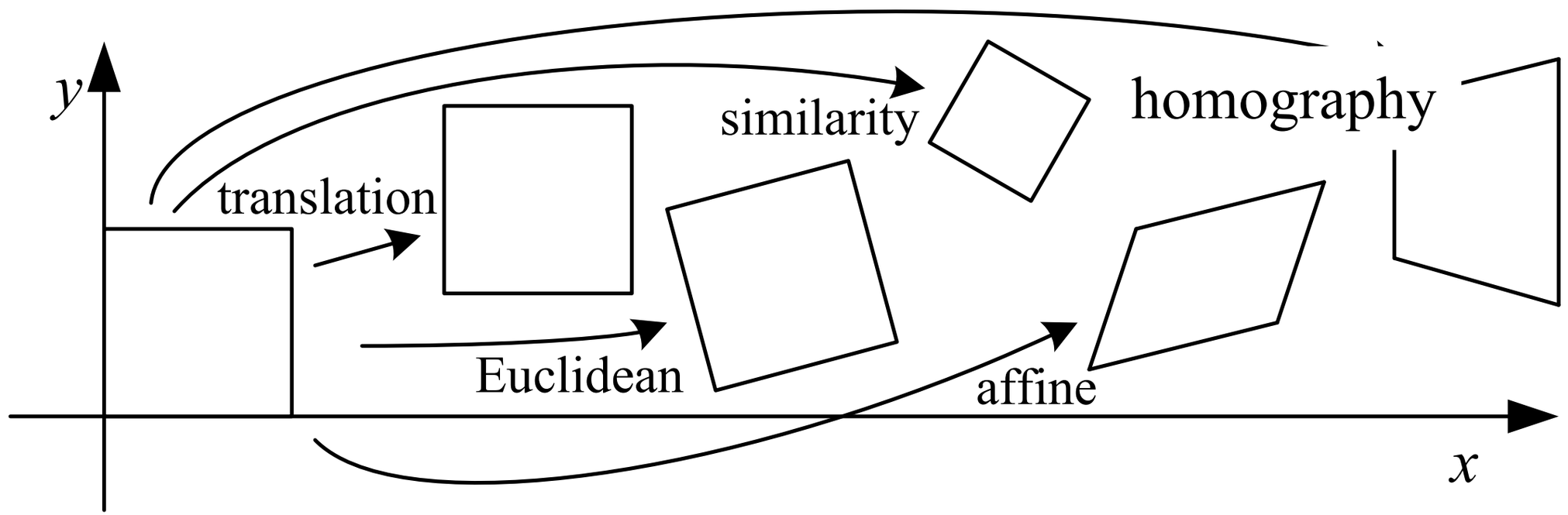
Transformations



Slide credits: Deva Ramanan

Agenda

- Camera models
- Camera calibration
- Homographies

[Reference material: Forsyth Ponce 1.1,1.2 and CVPR03 Tutorial (in lecture directory)]

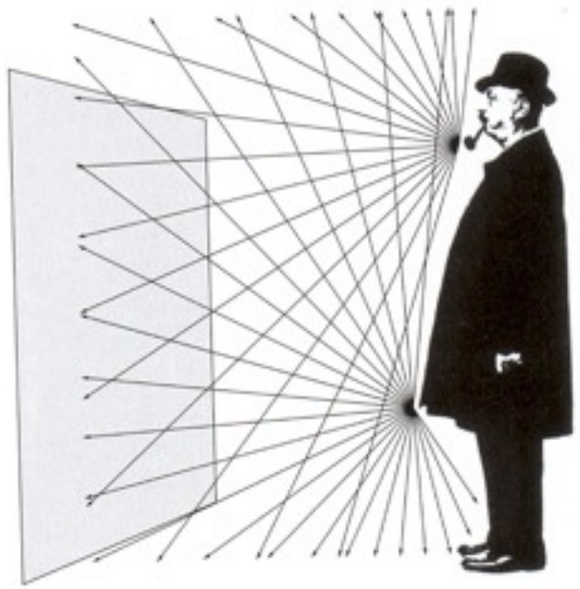
Agenda

- **Camera models**
- Camera calibration
- Homographies

[Reference material: Forsyth Ponce 1.1,1.2 and CVPR03 Tutorial (in lecture directory)]

Pinhole optics

A bad camera



Light-sensitive
Image plane
(pixels)

A better camera

Image is
upside down

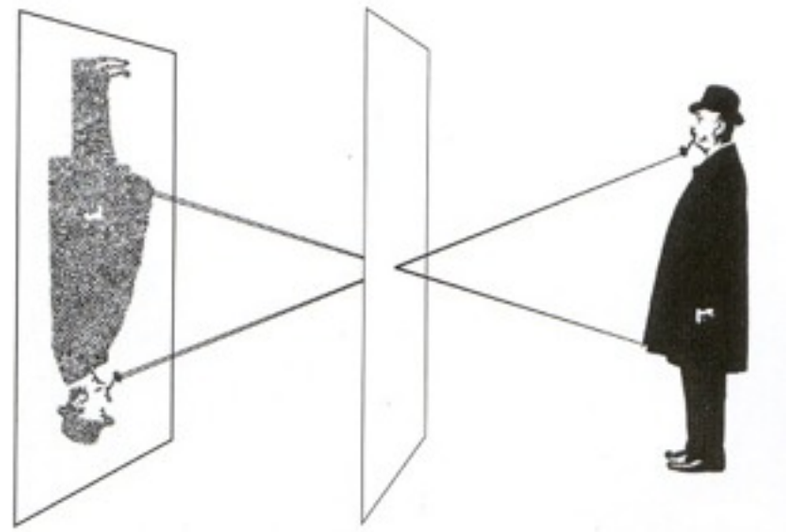
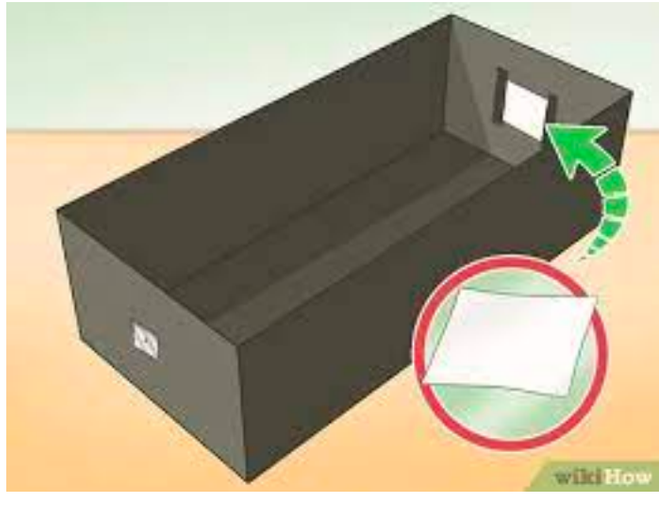


Image
plane
(pixels)

pinhole

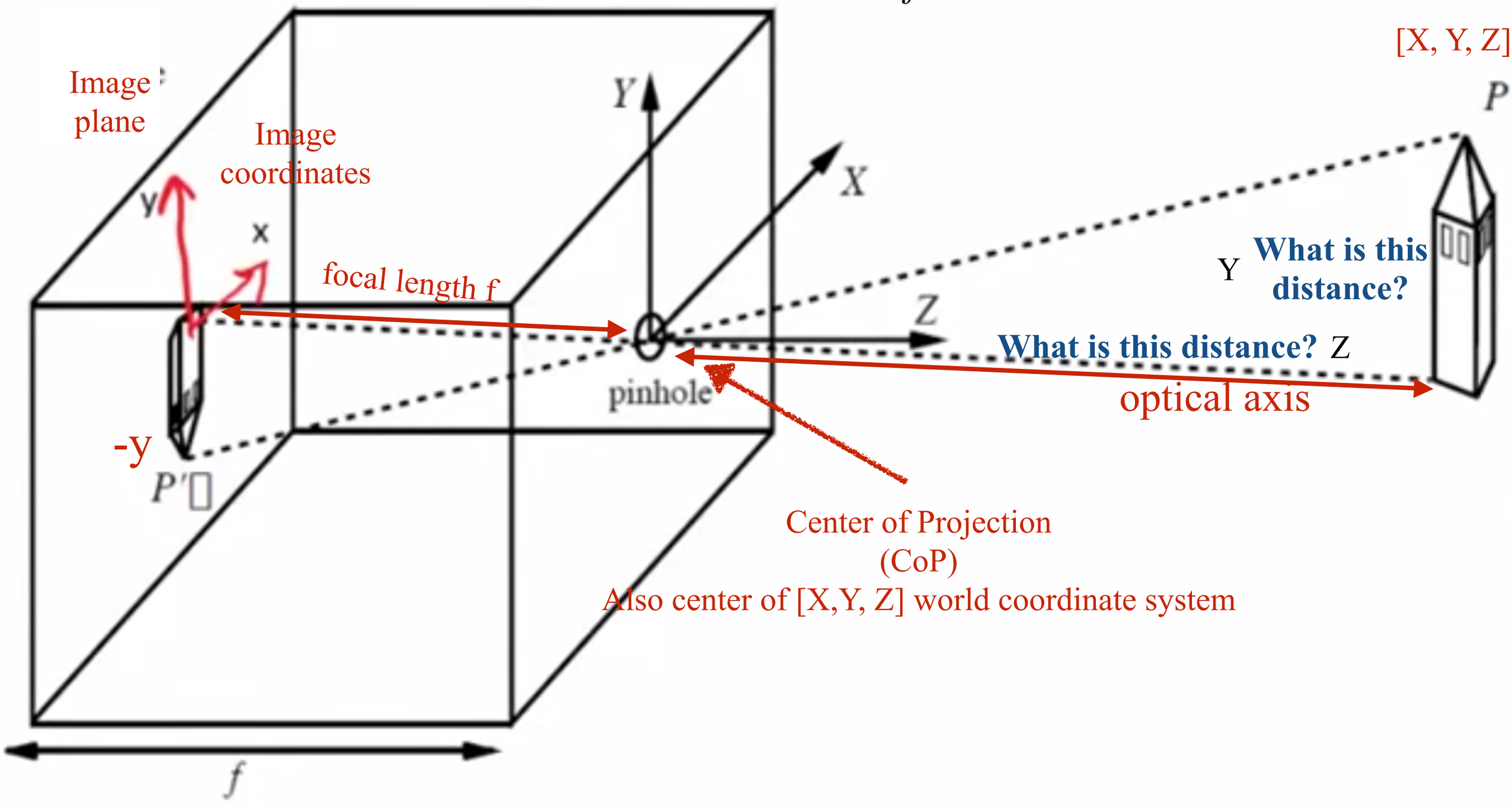


You can try this at home with a shoebox!

Pinhole Camera

Similar triangles:

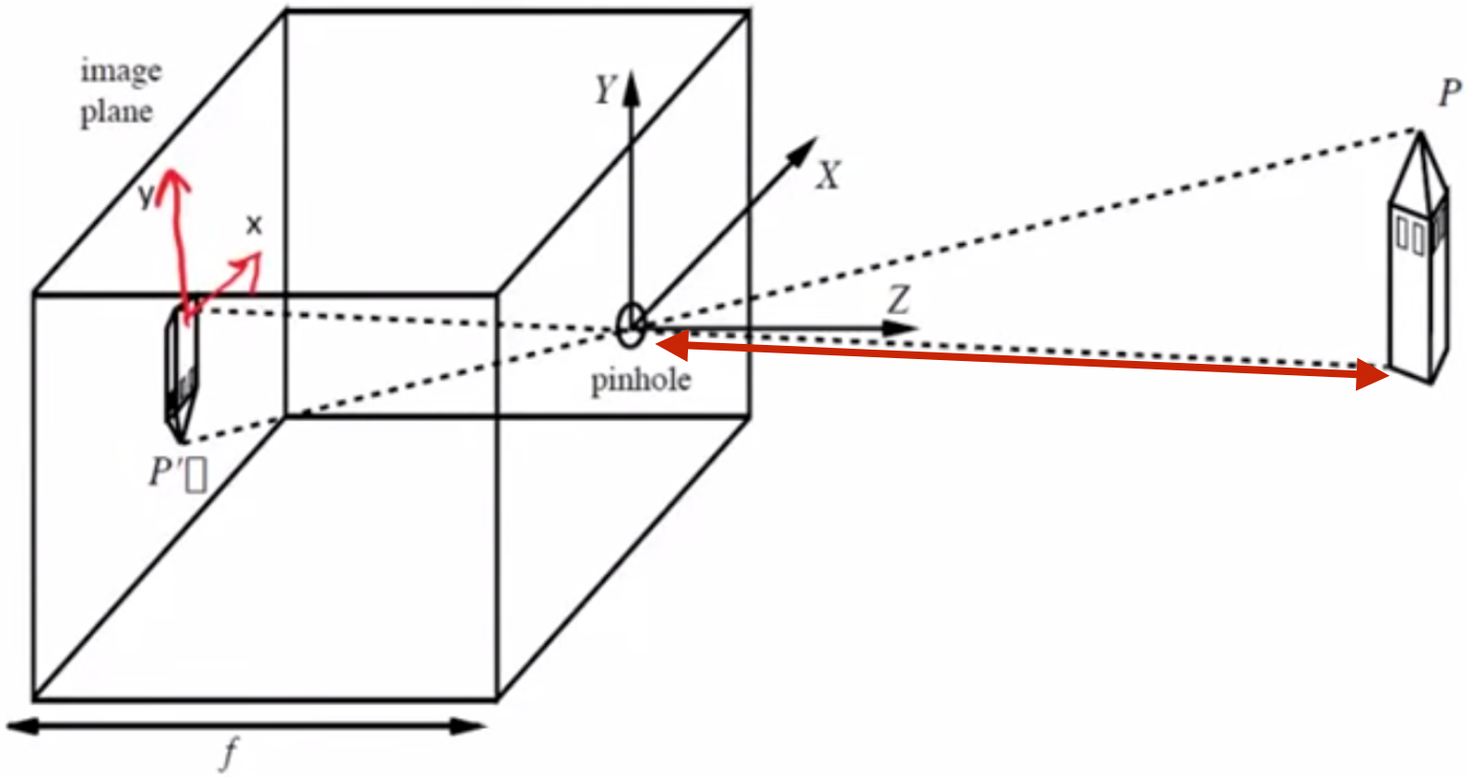
$$\frac{-y}{f} = \frac{Y}{Z} \qquad y = -f\frac{Y}{Z} \qquad x = -f\frac{X}{Z}$$

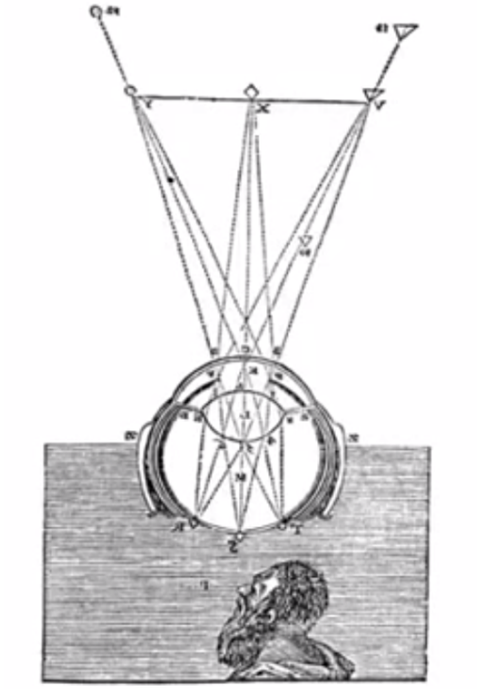


For some point P, given the world coordinates [X, Y, Z] for this point we want to compute the image coordinates for this point [x, y]

Annoying detail: image inversion

Why don't we see an upside-down world?

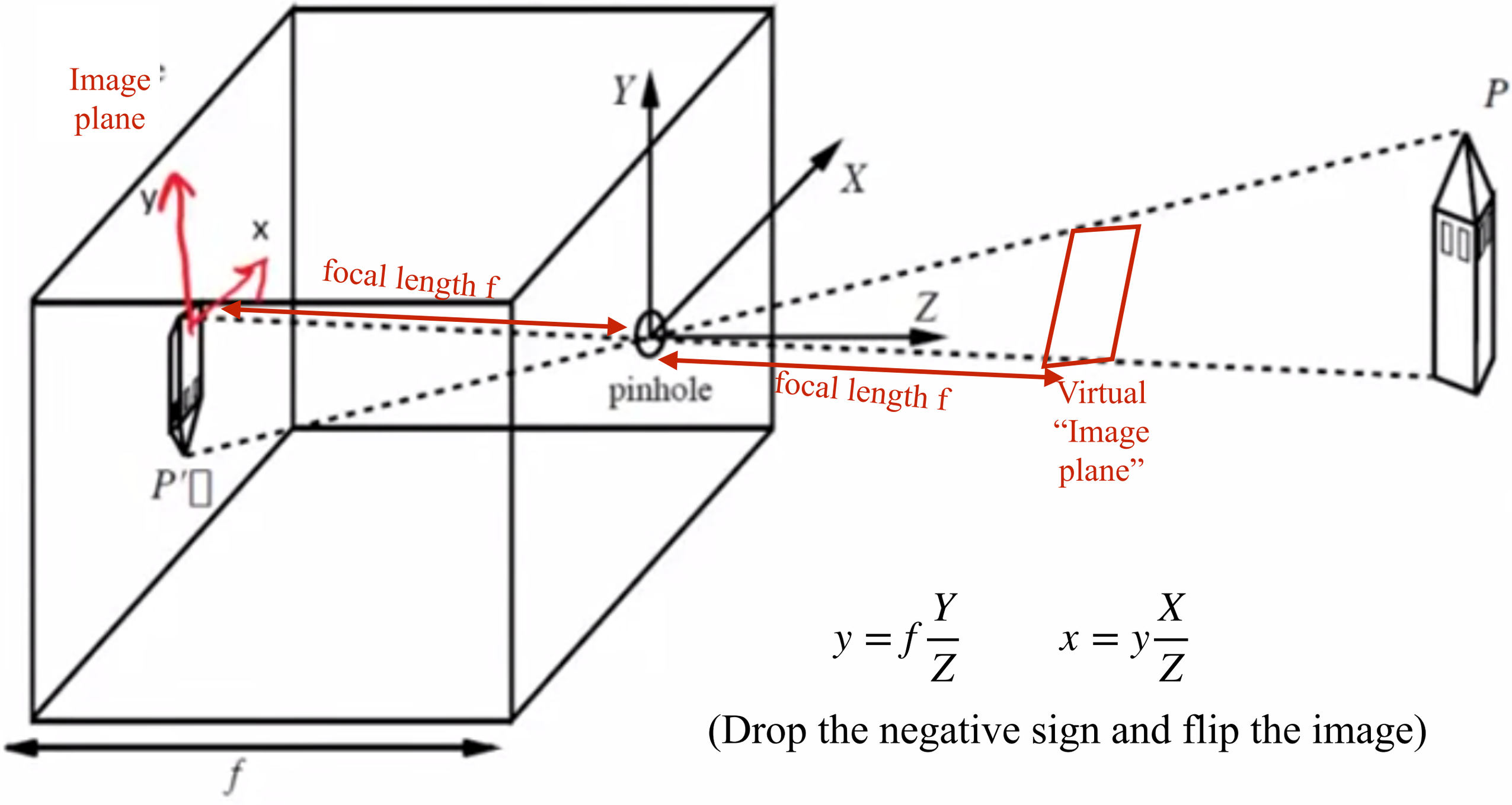




From Descartes(1937), La Dioptrique

This question perplexed folks for a while.
But software / your brain can simply invert the image.

Trick to avoid inversion:



$$y = f\frac{Y}{Z} \qquad x = y\frac{X}{Z}$$

(Drop the negative sign and flip the image)

We will use these equations from now on
(pretend that we flip the inverted image)

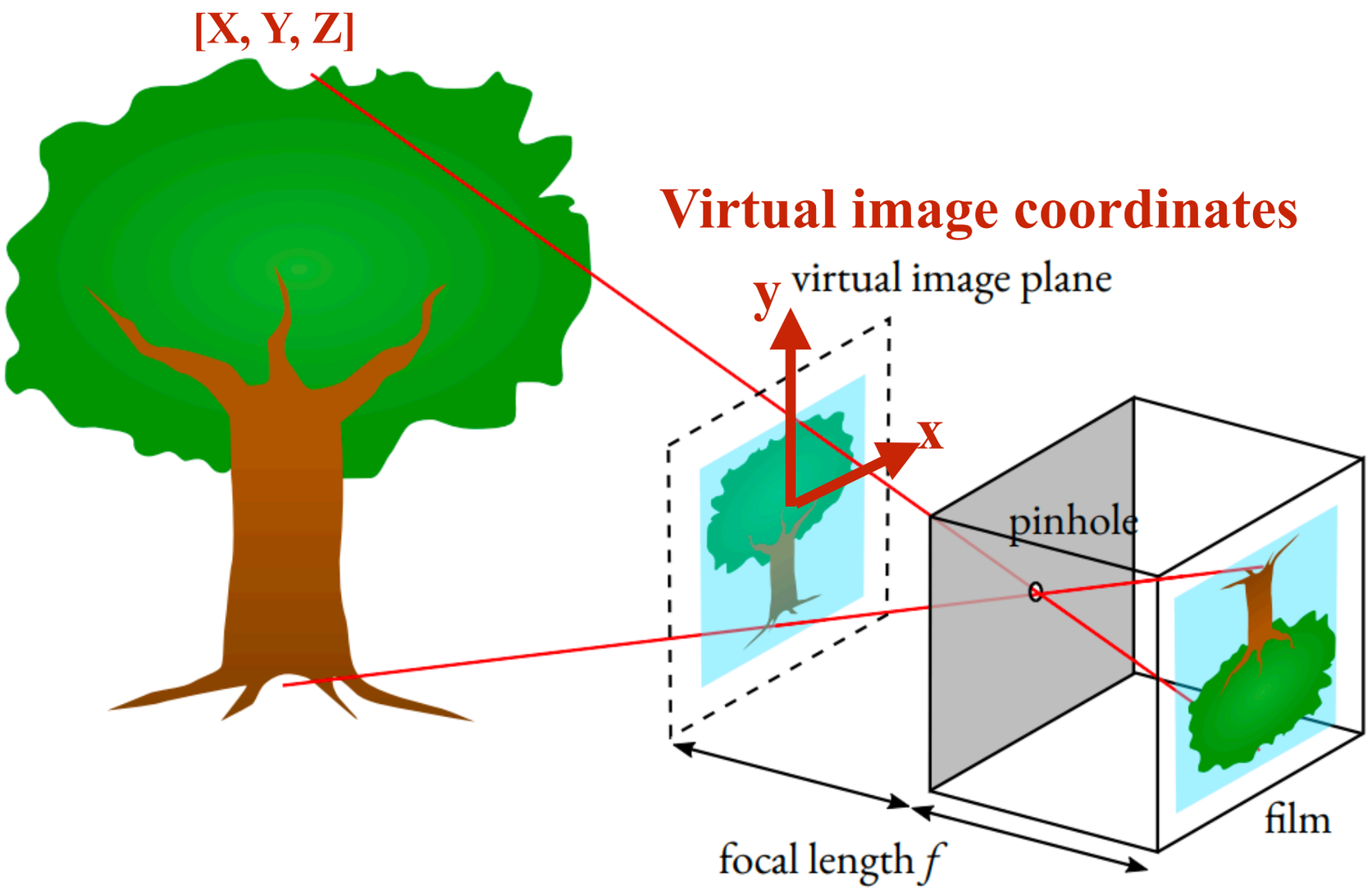

[X, Y, Z]
Virtual image coordinates
y
virtual image plane
x
pinhole
film
focal length f

Image projection revisited

Camera intrinsics matrix K

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

image coordinates (x, y) ("homogenous notation")

World coordinates: (X, Y, Z)

Expand matrix equations:

$$\lambda x = fX \qquad \lambda y = fY$$

$$\lambda = Z$$

Solve for (x,y):

$$x = \frac{fX}{\lambda} = \frac{fX}{Z} \qquad y = \frac{fY}{\lambda} = \frac{fY}{Z}$$

(Same as equations on previous slide)

Image projection revisited

Camera intrinsics matrix K

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

image coordinates (x, y) ("homogenous notation")

World coordinates: (X, Y, Z)

Homogenous image coordinates: (x_1, x_2, x_3)

Regular image coordinates: $\left(\frac{x_1}{x_3}, \frac{x_2}{x_3}\right)$

Special case: f = 1

Camera intrinsics matrix K

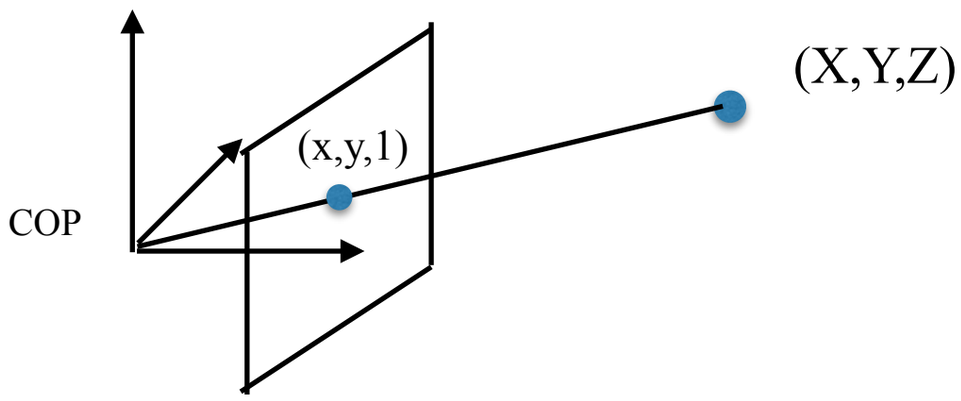
$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

image coordinates (x, y) ("homogenous notation")

World coordinates: (X, Y, Z)

$\lambda = Z$

$$Z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$



view $(x, y, 1)$ as a 3D point on the image plane

3D point in the real world is obtained by scaling ray pointed at image coordinate (x, y, 1) by the depth Z

3D rigid-body transformations

3D translations

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \begin{bmatrix} X + t_x \\ Y + t_y \\ Z + t_z \end{bmatrix}$$

3D rotations

$$R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

How to write this as a single matrix multiplication?

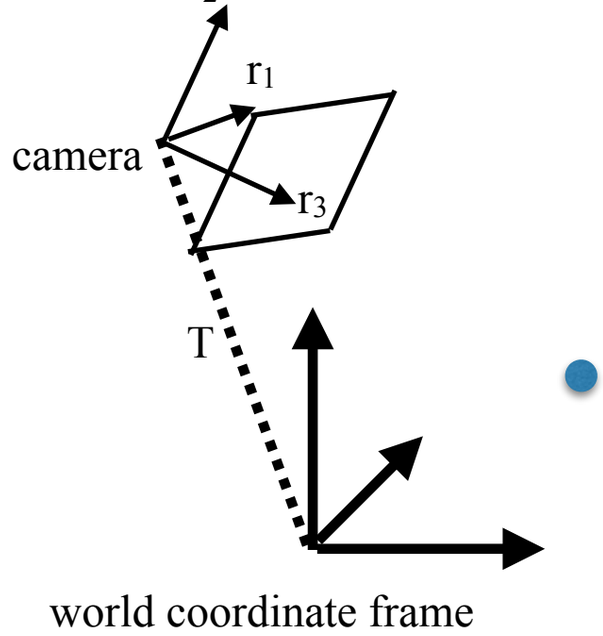
$$R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

“homogenous” world coordinates

Rotation matrix: $R^\top R = I$

Let's model a camera moving through world coordinate frame:

$$R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



Next we have to project the point onto the image

Camera projection

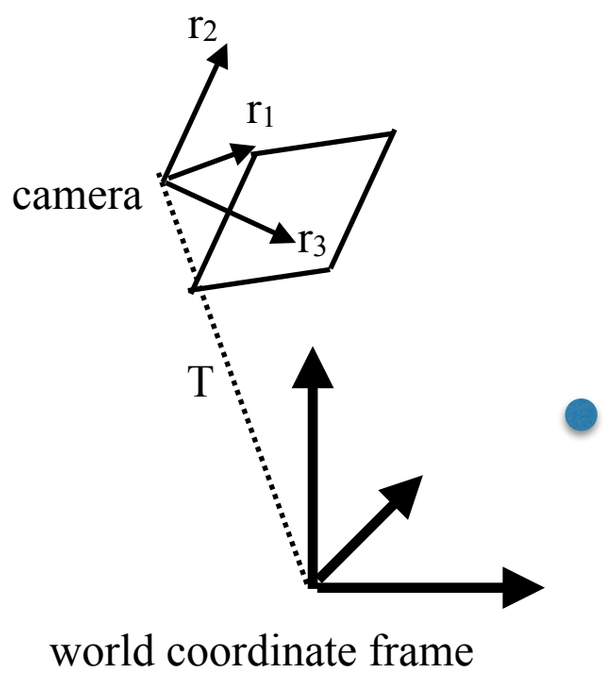
$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

2D point in image coordinates (homogenous)

Camera intrinsics matrix K
Project the point onto the camera image plane

Camera extrinsics (rotation and translation)
Convert the point from world coordinates to camera coordinates

3D point in world coordinates (homogenous)



Fancier intrinsics

$$\begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Camera intrinsics matrix K

Now let's capture more details about how cameras work:

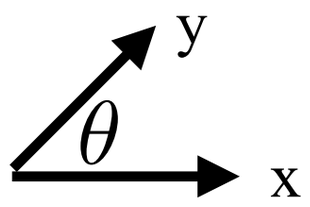
$$x_s = s_x x$$
$$y_s = s_y y$$

} non-square pixels

$$x' = x_s + o_x$$
$$y' = y_s + o_y$$

} shifted origin

$$x'' = x' + s_\theta y'$$



skewed image axes

$$K = \begin{bmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

To "calibrate" a camera, I need to compute this matrix K for a given camera (tells me how to project from world to image coordinates)

(obtain simpler intrinsics by setting s_x, s_y = f and s_θ, o_x, o_y = 0)

Camera Projection

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

2D point in image coordinates (homogenous)

Camera instrinsic matrix K

Camera extrinsics (rotation and translation)

3D point in world coordinates (homogenous)

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

Camera projection matrix M

Question: how many degrees of freedom in M?

Agenda

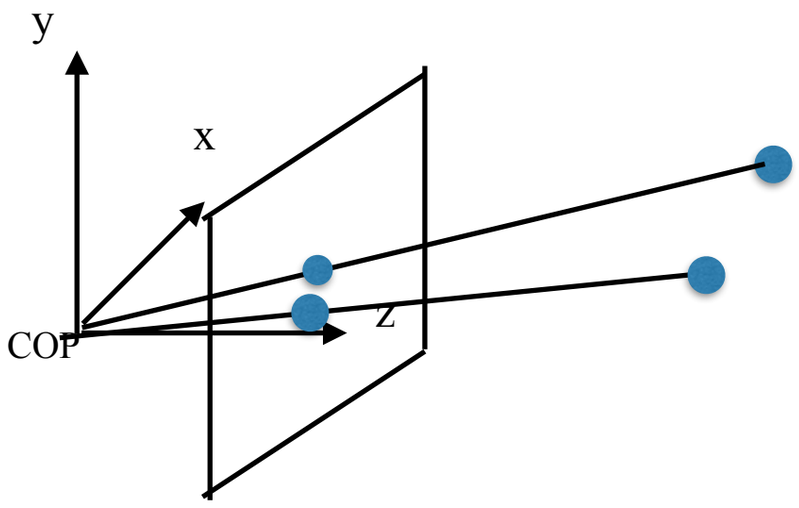
- Camera models
- Camera matrices (intrinsics + extrinsics, affine)
- **Camera calibration**
- Homographies

Camera Calibration

For a given camera, how can we find the the camera matrix M_{3X4}?
We need a set of scene points $P_1,..,P_N$
and the corresponding projections onto the image plane $p_1,..,p_N$

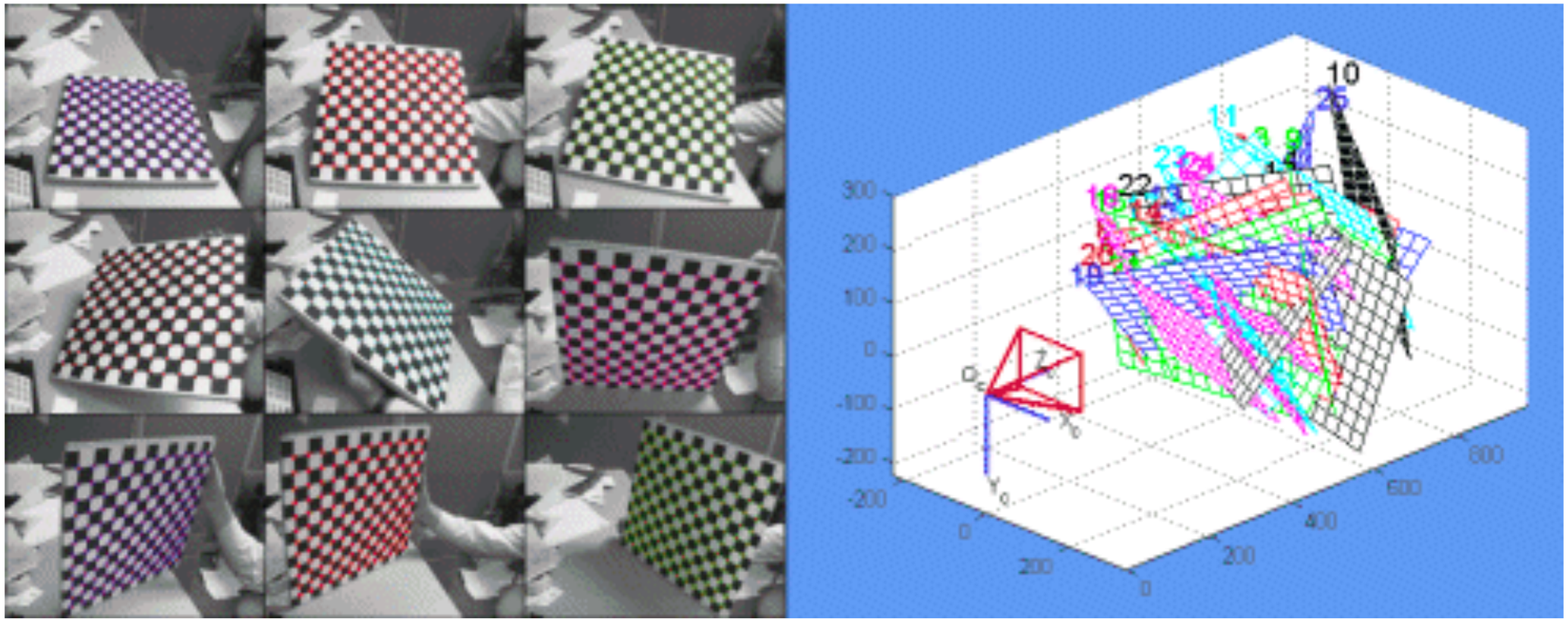
2D point in image coordinates

$$\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$



$$\mathbf{P}_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

3D point in world coordinates

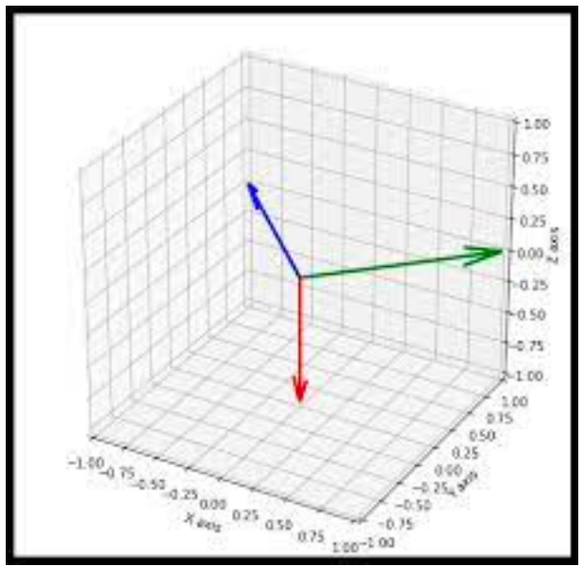


We need a tool: SVD

Any matrix $A \in \mathbb{R}^{mxn}$ can be decomposed into a product of 3 matrices: $A = U\Sigma V^{\top}$

where U and V are orthonormal and Σ is diagonal

Orthonormal



$$x_i \cdot x_j = 0 \quad \text{if } i \neq j$$

$$\|x_i\| = 1 \quad \forall i$$

All rotation matrices are orthonormal: $X^T X = I$

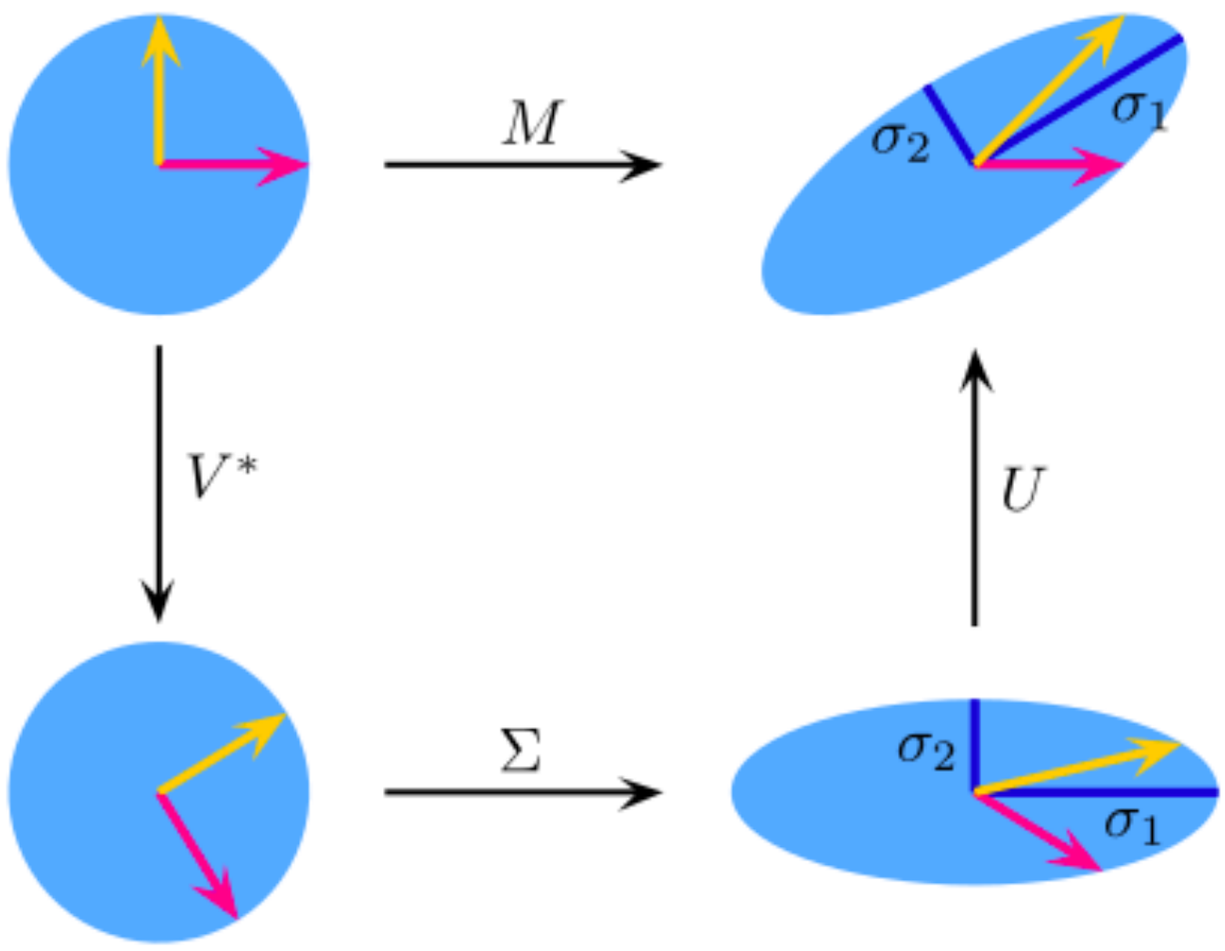
$$X^T = X^{-1}$$

We need a tool: SVD

Any matrix $A \in \mathbb{R}^{mxn}$ can be decomposed into a product of 3 matrices: $A = U\Sigma V^\top$

where U and V are orthonormal and Σ is diagonal

Then $Ax = U\Sigma V^\top x$ We can break down this transformation into 3 steps:



$$M = U \cdot \Sigma \cdot V^*$$

Think of A as: un-rotate by V, scale by Σ, rotate by U

SVD

Any matrix $A \in \mathbb{R}^{mxn}$ can be decomposed into a product of 3 matrices:

where U and V are orthonormal and Σ is diagonal

Then $Ax = U\Sigma V^\top x$

$$V^T = \begin{bmatrix} v_1^\top \\ v_2^\top \\ \vdots \\ v_N^\top \end{bmatrix} \in \mathbb{R}^{nxn}$$

Right singular vectors

$\Sigma \in \mathbb{R}^{mxn}$ is a rectangular-diagonal matrix: $\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots \\ 0 & \sigma_2 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$ $\sigma_i \geq 0$

Singular values

$$U = [u_1, \dots, u_m] \in \mathbb{R}^{mxm}$$

Left singular vectors

$$A = U\Sigma V^T = \sum_i^{\min(m,n)} u_i \sigma_i v_i^T$$

We will define the order:
σ_1 is the largest
σ_N is the smallest

SVD

Any matrix $A \in \mathbb{R}^{mxn}$ can be decomposed into a product of 3 matrices:

where U and V are orthonormal and Σ is diagonal

$$A = U\Sigma V^T = \sum_{i}^{\min(m,n)} u_i \sigma_i v_i^T$$

The best k-rank approximation of A is given by constructing k largest singular vectors (largest based on the size of the singular values)

$$A = U\Sigma V^T \approx \sum_{i}^{k} u_i \sigma_i v_i^T$$

$$\min_{A':rank(A')<=k} ||A - A'||_F \approx \sum_{i}^{k} u_i \sigma_i v_i^T$$

k largest scaling factors

where $||A||_F = A(:)^T A(:)$

SVD

$$Ax = U\Sigma V^\top x$$

$$V^T = \begin{bmatrix} v_1^\top \\ v_2^\top \\ \vdots \\ v_N^\top \end{bmatrix} \in \mathbb{R}^{nxn}$$

Right singular vectors

Homogenous least squares problem:

Find h such that Ah is as small as possible:

$$\min_{h:h^T h=1} ||Ah||^2$$

Solution: v_N

The right singular vector associated with the smallest singular value

$$A = U\Sigma V^T = \sum_i^{\min(m,n)} u_i \sigma_i v_i^T$$

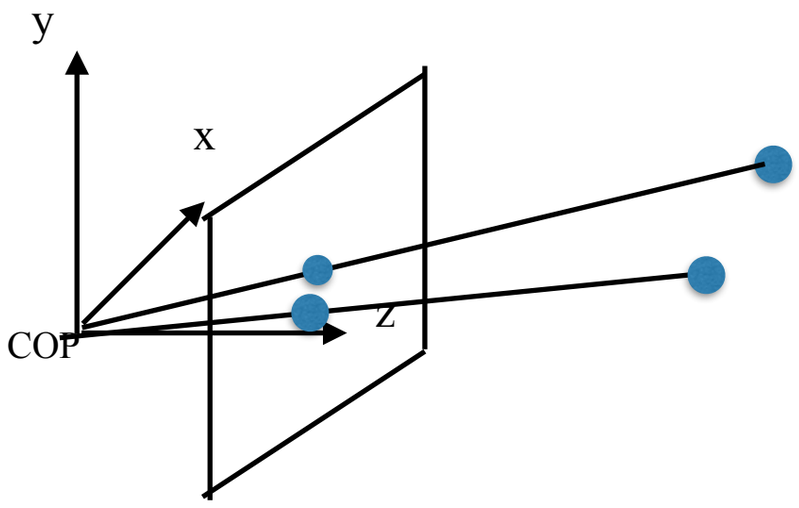
$$Av_N = \sum_i^{\min(m,n)} u_i \sigma_i v_i^T v_N = u_N \sigma_N$$

Camera Calibration

Given a set of scene points $P_1,..,P_N$
and the corresponding projections in the image plane $p_1,..,p_N$
Recover the camera matrix M_{3X4}

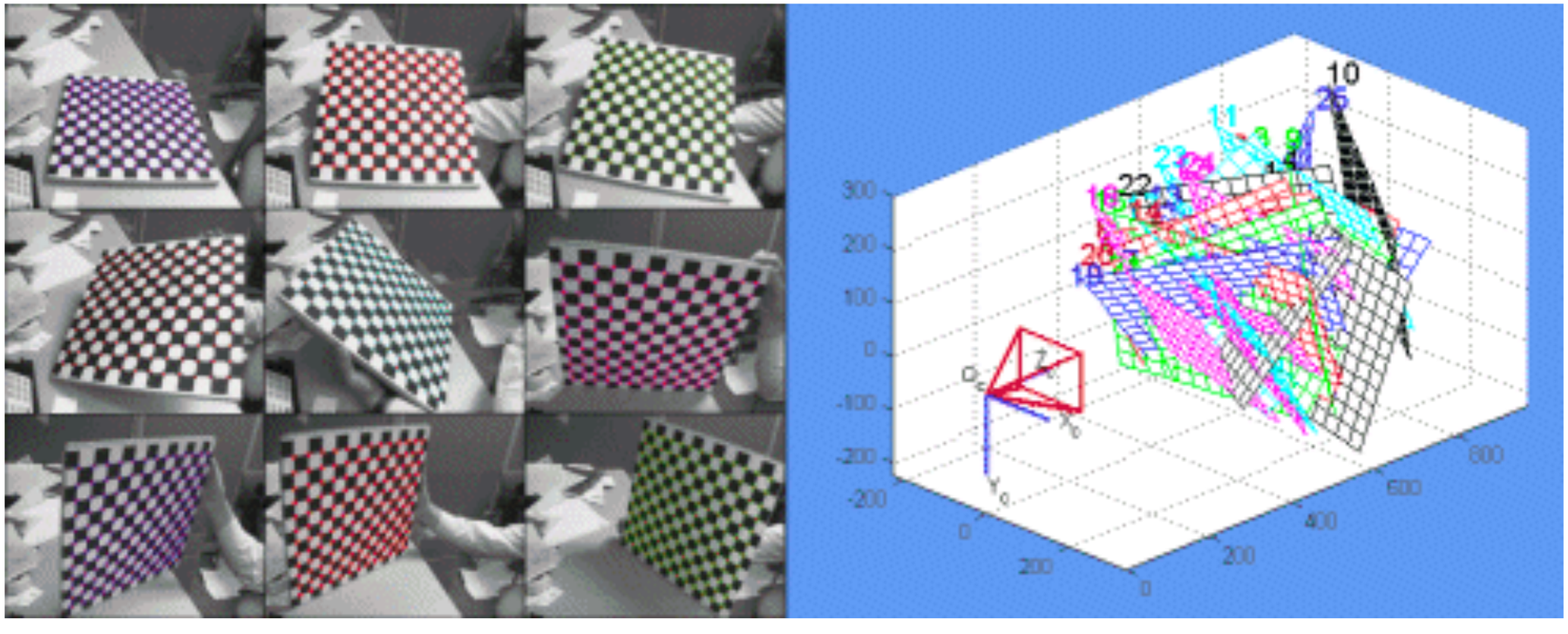
2D point in image coordinates

$$\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$



$$\mathbf{P}_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

3D point in world coordinates



The math for the calibration procedure follows a procedure that is used in many other problems involving camera geometry, so it's worth remembering:

Write relation between image point, camera projection matrix, and 3D:

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

What is known vs unknown here?

Write (non-linear) relations between coordinates:

$$x_i = \frac{\mathbf{m_1^T P_i}}{\mathbf{m_3^T P_i}}, \quad y_i = \frac{\mathbf{m_2^T P_i}}{\mathbf{m_3^T P_i}}$$

$$P_i = (X_i, Y_i, Z_i)$$

Known: x_i, y_i, P_i

Unknown: M

Want to solve for M **How can we do this?**

Estimating a camera matrix

$$x_i = \frac{\mathbf{m_1^T P_i}}{\mathbf{m_3^T P_i}}, \quad y_i = \frac{\mathbf{m_2^T P_i}}{\mathbf{m_3^T P_i}} \qquad P_i = (X_i, Y_i, Z_i)$$

Write constraints as linear unknowns in M

$$(m_{31}X_i + m_{32}Y_i + m_{33}Z_i + m_{34})x_i = m_{11}X_i + m_{12}Y_i + m_{13}Z_i + m_{14}$$
$$(m_{31}X_i + m_{32}Y_i + m_{33}Z_i + m_{34})y_i = m_{21}X_i + m_{22}Y_i + m_{23}Z_i + m_{24}$$

Known: x_i, y_i, P_i

$$LM(:) = \begin{bmatrix} 0 \\ 0 \\ \dots \end{bmatrix}$$

Homogenous linear system

M(:) is a vector version of M (12 x 1)

L is a matrix of size (2N x 12)
where N is the # of correspondences

How many degrees of freedom in M? 11

How many corresponding points needed? 6

In noise-free case, L will have an exact solution (non-zero null space)

What if the points are noisy?

$$LM(:) = \begin{bmatrix} 0 \\ 0 \\ \\ \dots \end{bmatrix}$$

Homogenous linear system

This will not have an exact solution,
so we need to find an approximate solution.
How can we do this?

$$\min_{||M(:)||^2=1} ||LM(:)||^2$$

How do we solve this?

Why do we need to constrain the norm of M(:)?

SVD!

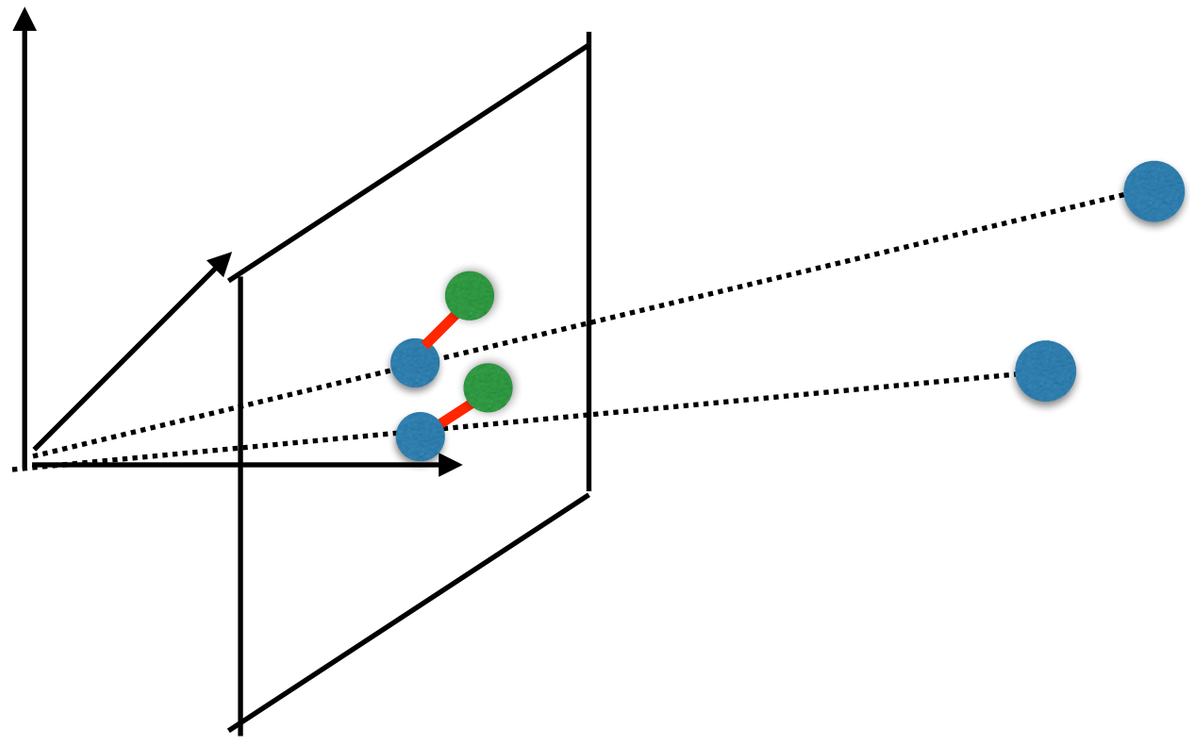
Min right singular vector of L = $U\Sigma V^T$ (right vector in V)

Is this the right error to minimize? If not, what is?

Aside: this technique of linearizing a nonlinear equation resulting from perspective projection is known as a *direct linear transform*

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

Ideal error: image reprojection error



$$Err(M) = \sum_i (x_i - \frac{m_1^T P_i}{m_3^T P_i})^2 + (y_i - \frac{m_2^T P_i}{m_3^T P_i})^2$$

2D point **projected 3D point**

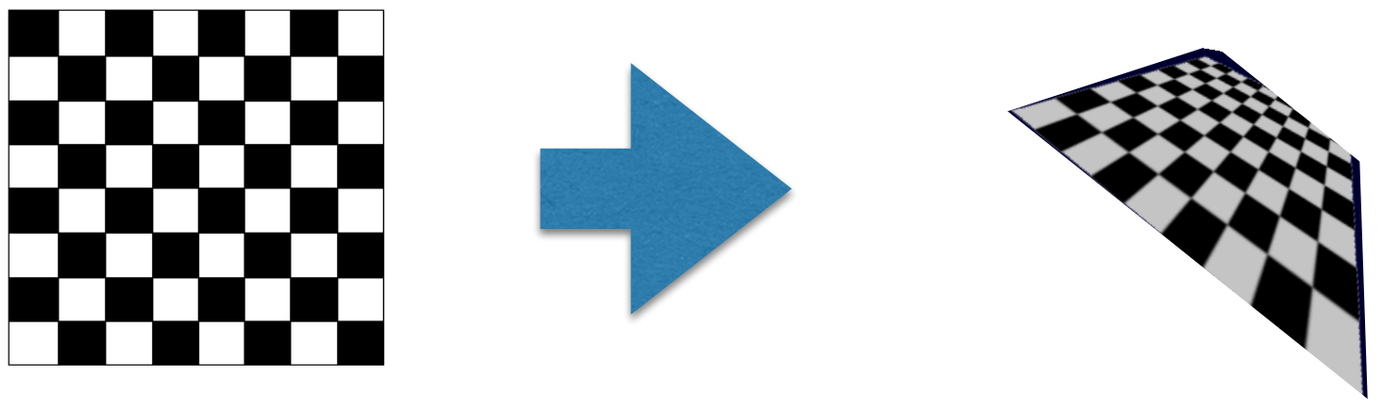
Initialize nonlinear optimization with the SVD solution from previous slide

Agenda

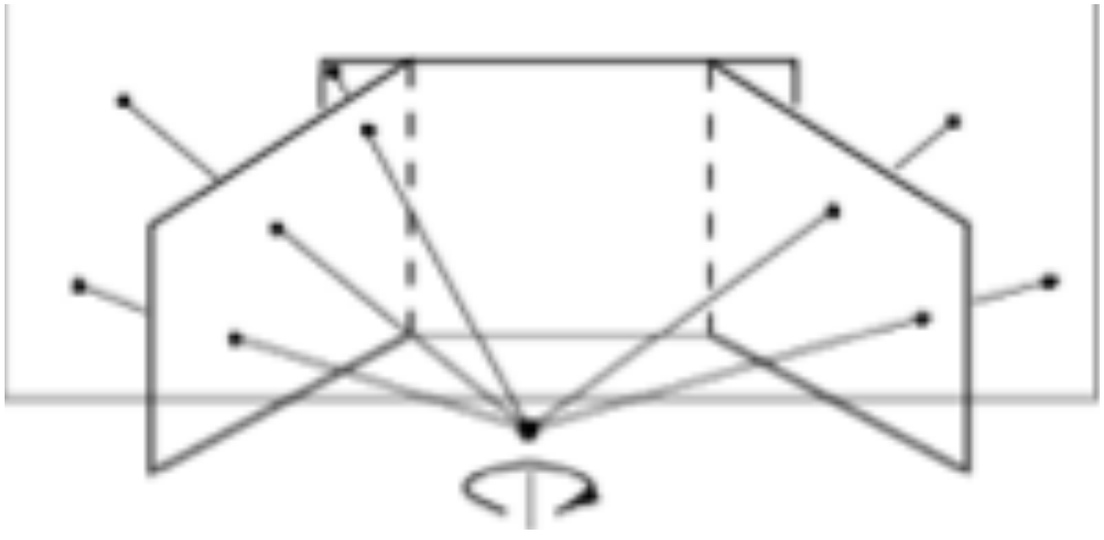
- Camera models
- Camera matrices (intrinsics + extrinsics, affine)
- Camera calibration
- **Homographies**

Homography transformations

1. Models image projection of an arbitrary planar scene



2. Models warping (perspective) effects from camera rotations



Projection of a 3D plane

Define a plane: Z = 0

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

This column doesn't matter

$$= \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} fr_{11} & fr_{12} & ft_x \\ fr_{21} & fr_{22} & ft_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

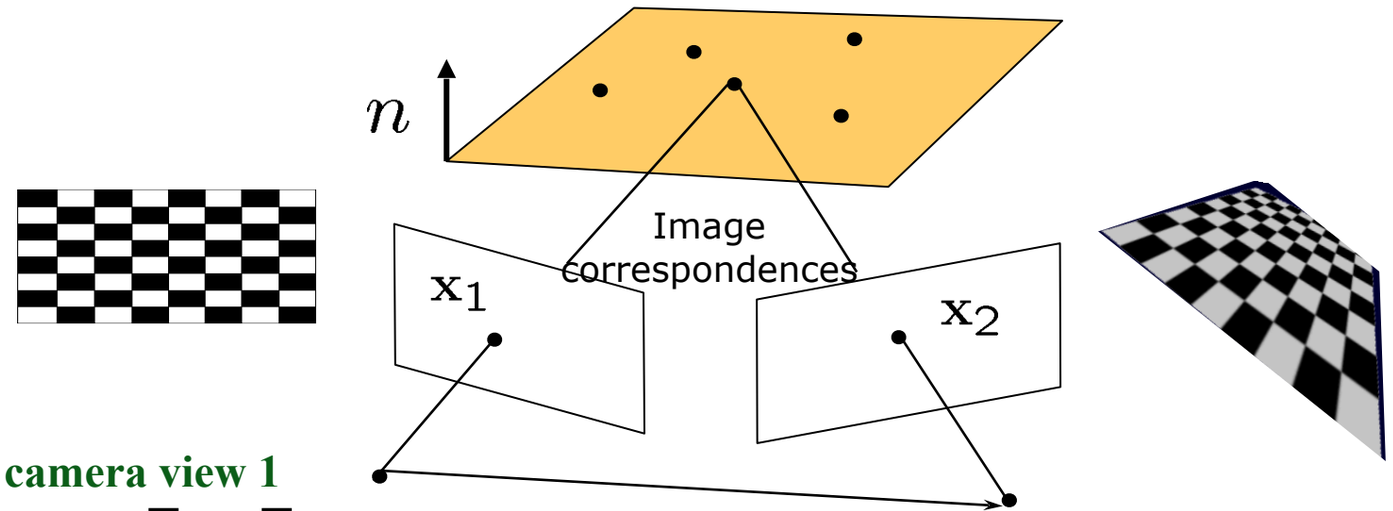
H is defined up to a scale factor

Homography matrix H

Convert between 2D location on object plane (Z = 0) and image coordinate with a 3X3 matrix H

(Above holds for any intrinsic matrix K, implying H can be any 3x3 matrix)

Relating 2 camera views of the same 3D plane



Project a point onto camera view 1

$$\lambda_1 \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = H_1 \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$\lambda_2 \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = H_2 \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Project the same point onto camera view 2

[X,Y, 1] are the same

solve for [X,Y,1] and set equal; rearrange

$$\lambda \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = H_2 H_1^{-1} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Relates the two image views of the plane

Can be used to warp from 1 camera view to another

[LHS and RHS are related by a scale factor]

Computing homography projections

$$\lambda \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Write out the equation for x_2:

$$x_2 = \frac{\lambda x_2}{\lambda} = \frac{ax_1 + by_1 + c}{gx_1 + hy_1 + i}$$

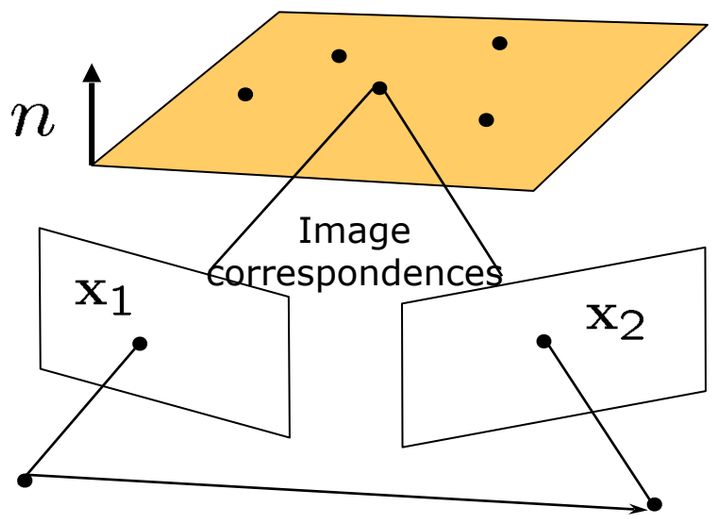
We can do the same for y_2

How do we solve for the homography?

We follow the same procedure as before for camera calibration!

Estimating homographies

Given corresponding 2D points in left and right image, estimate H



$$x_2 = \frac{\lambda x_2}{\lambda} = \frac{ax_1 + by_1 + c}{gx_1 + hy_1 + i}$$

Make into a linear equation: $x_2(gx_1 + hy_1 + i) = ax_1 + by_1 + c$

$$\vdots$$

$$AH(:) = \begin{bmatrix} 0 \\ 0 \\ \vdots \end{bmatrix}$$ Homogenous linear system

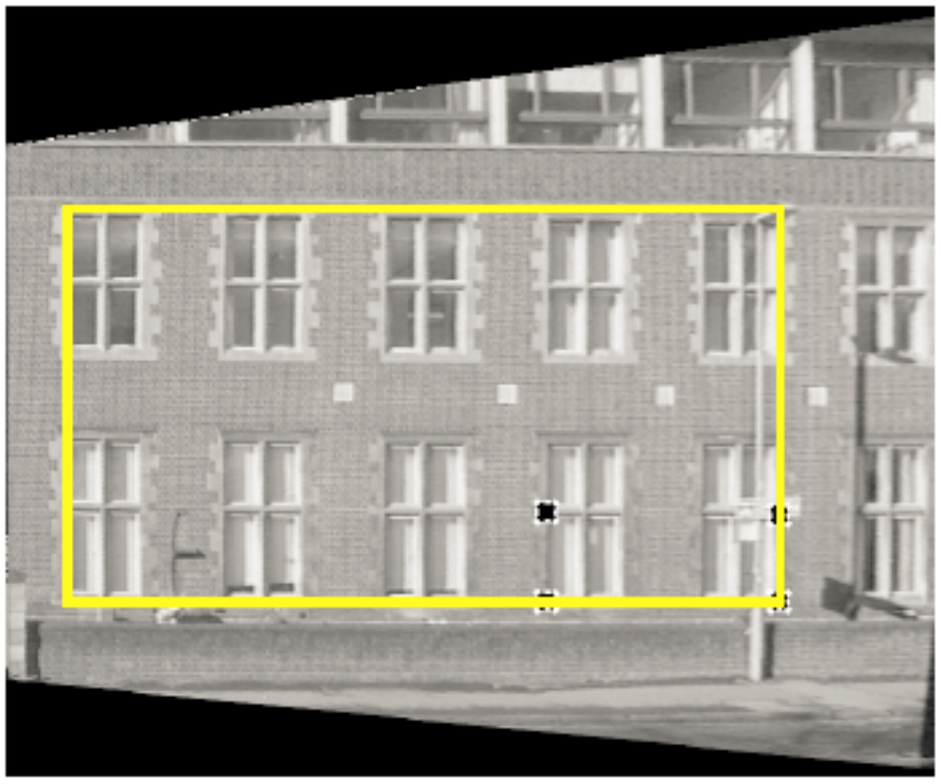
How many degrees of freedom in H? 8

How many corresponding points needed? 4

If we have more than 4 correspondences, and if they are noisy…

Solve with SVD!

“Frontalizing” planes using homographies

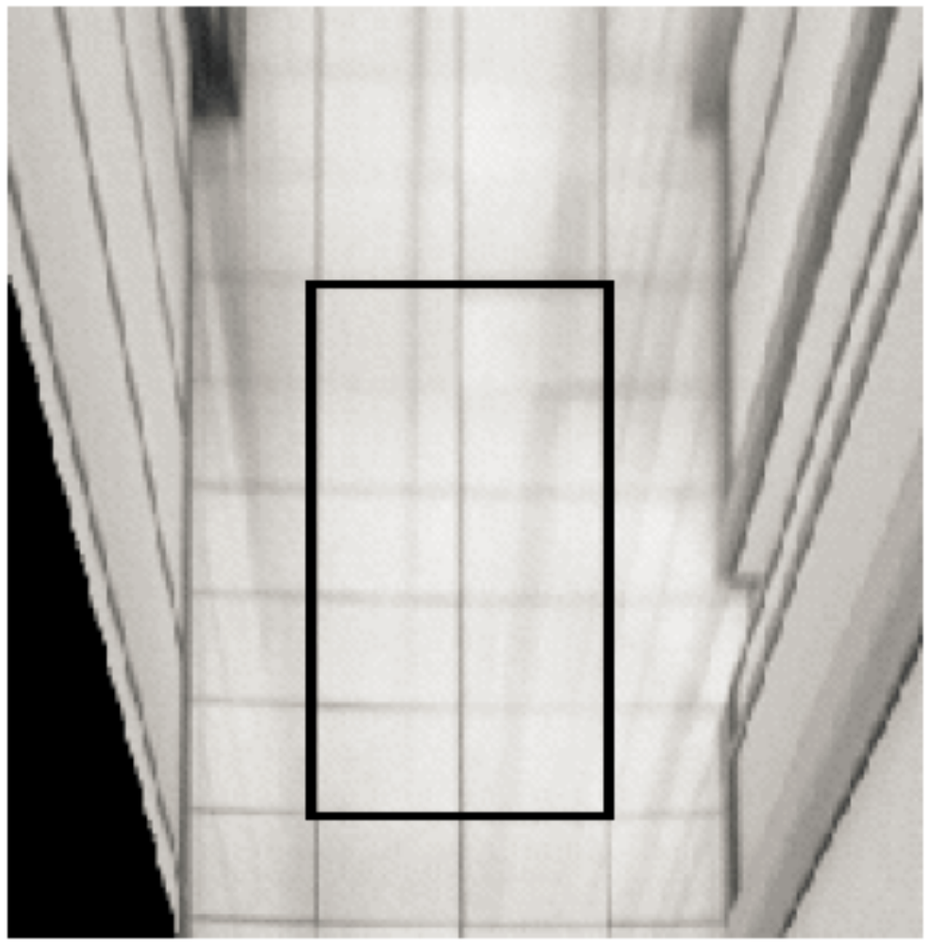


from Hartley & Zisserman

Estimate homography on (at least) 4 pairs of corresponding points (e.g., corners of quad/rect)

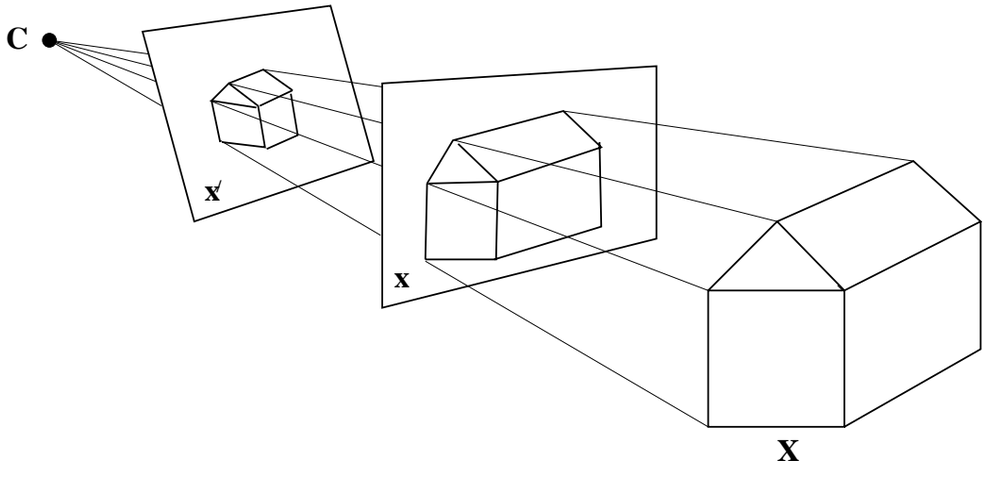
Apply homography on all (x,y) coordinates inside target rectangle to compute source pixel location

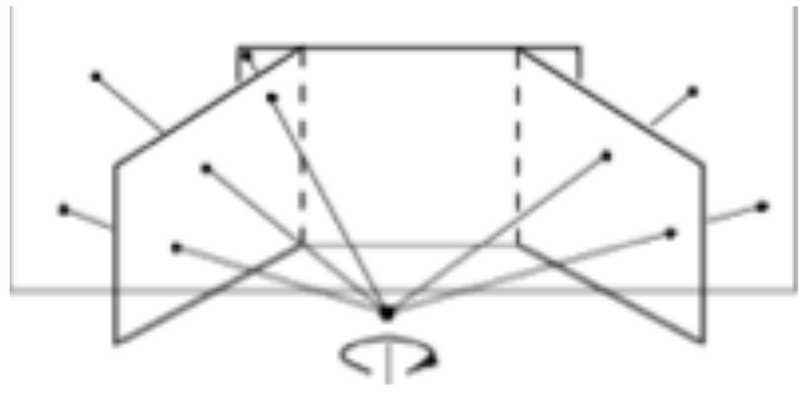
“Frontalizing” planes using homographies



from Hartley & Zisserman

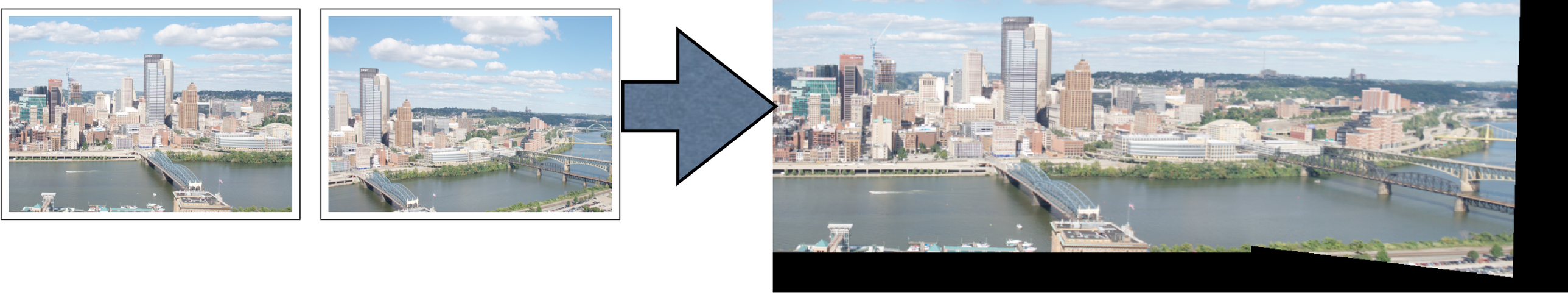
Special case of 2 views: rotations about camera center



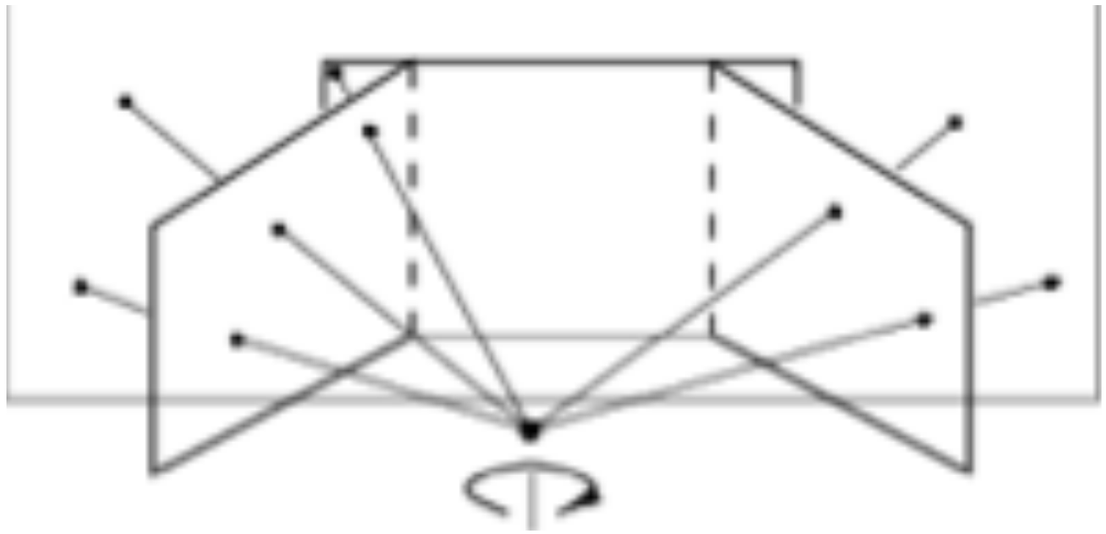


Can be modeled as planar transformations, *regardless of 3D scene geometry!*

Intuition: equivalent to looking at an image of a planar image



Rotations about camera center



Relation between 3D camera coordinates: $\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = R \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix}$

Rotation matrix

3D->2D projection: $\lambda_2 \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} f_2 & 0 & 0 \\ 0 & f_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix}$ (no translation)

Instrinsic camera matrix K

We can similarly project from (X1, Y1, Z1)

Combining both: $\lambda \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = K_2 R K_1^{-1} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$