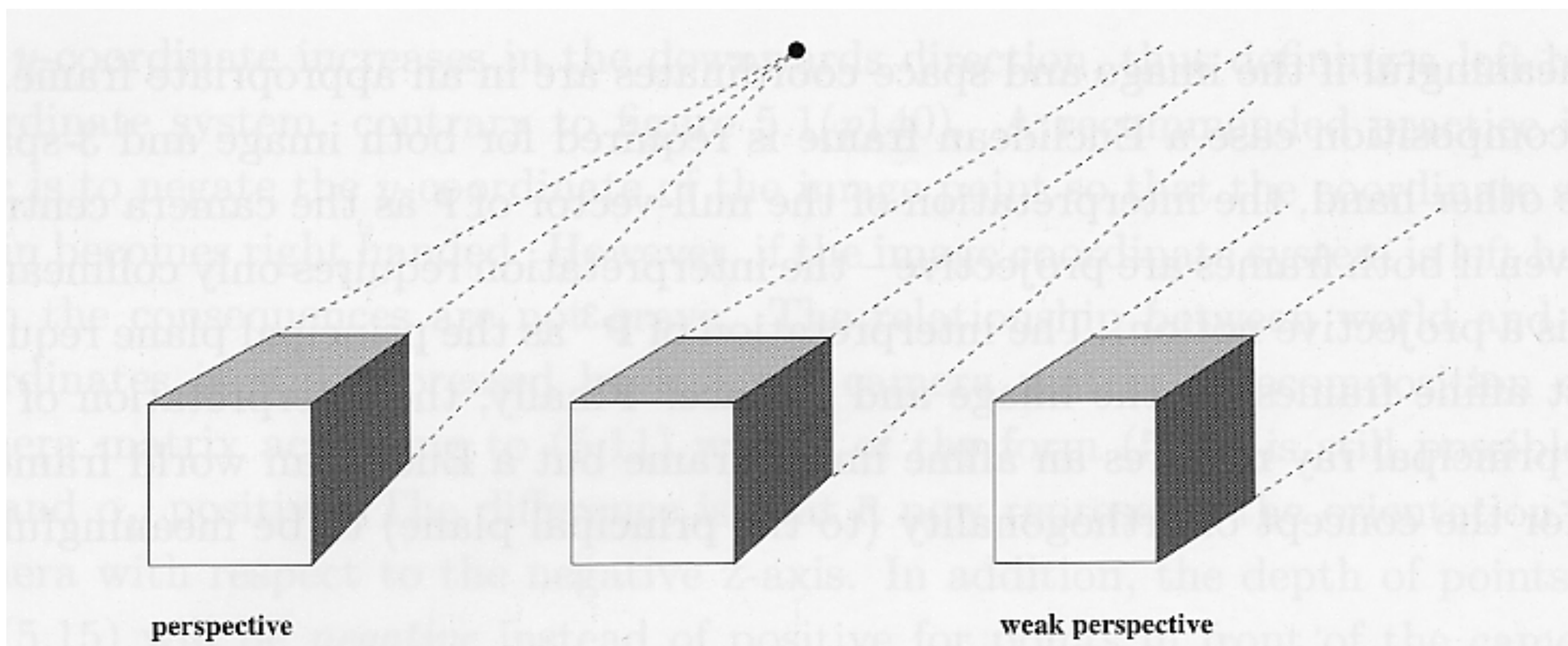
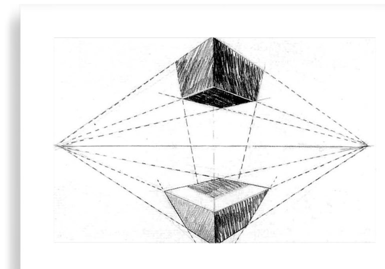
Cameras



Agenda

- Pinhole optics
  - Perspective projection (vanishing points, horizon, object height)
  - Camera matrices (intrinsics + extrinsics)
  - **Homographies (2 views of plane, rotation)**

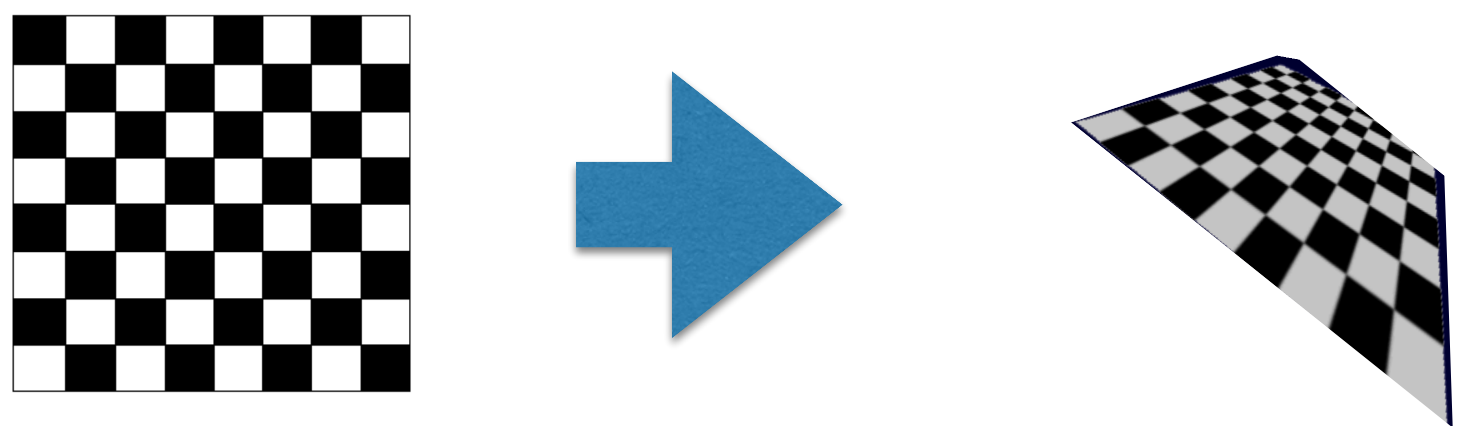


$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

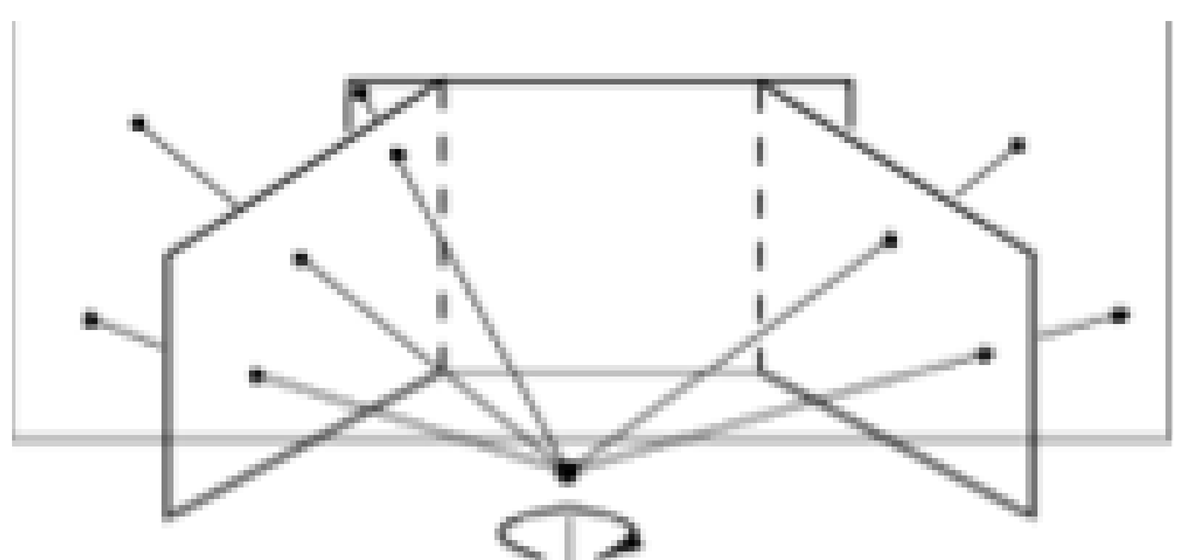
- Camera models
  - Properties of camera matrices (DOF, geometric intuition, pixel2rays)
  - Simplified cameras: orthographic, scaled orthographic, paraperspective, affine
  - Camera calibration (DLT v reprojection error)

Homography transformations

1. Models perspective effects for a planar scene



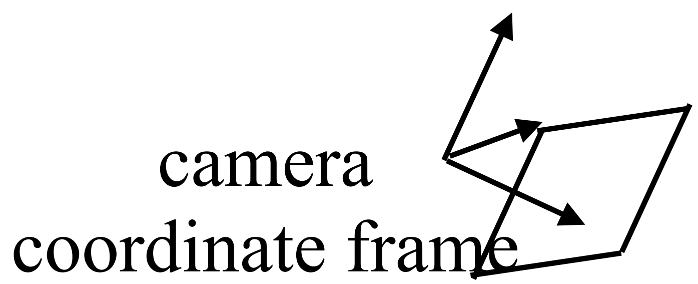
2. Models perspective effects from camera rotations

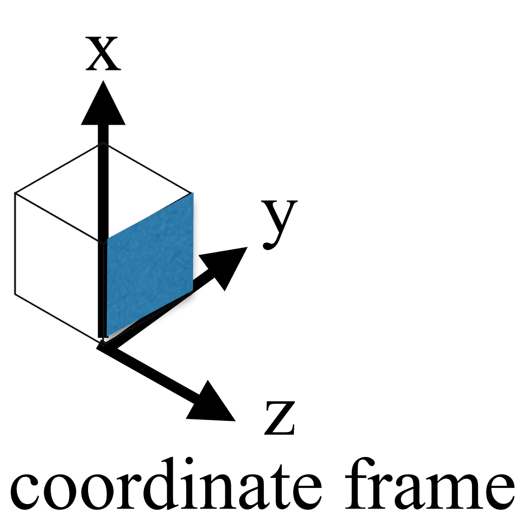


Let's analyze the projection of a 3D plane

Place world coordinate frame on blue object plane

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



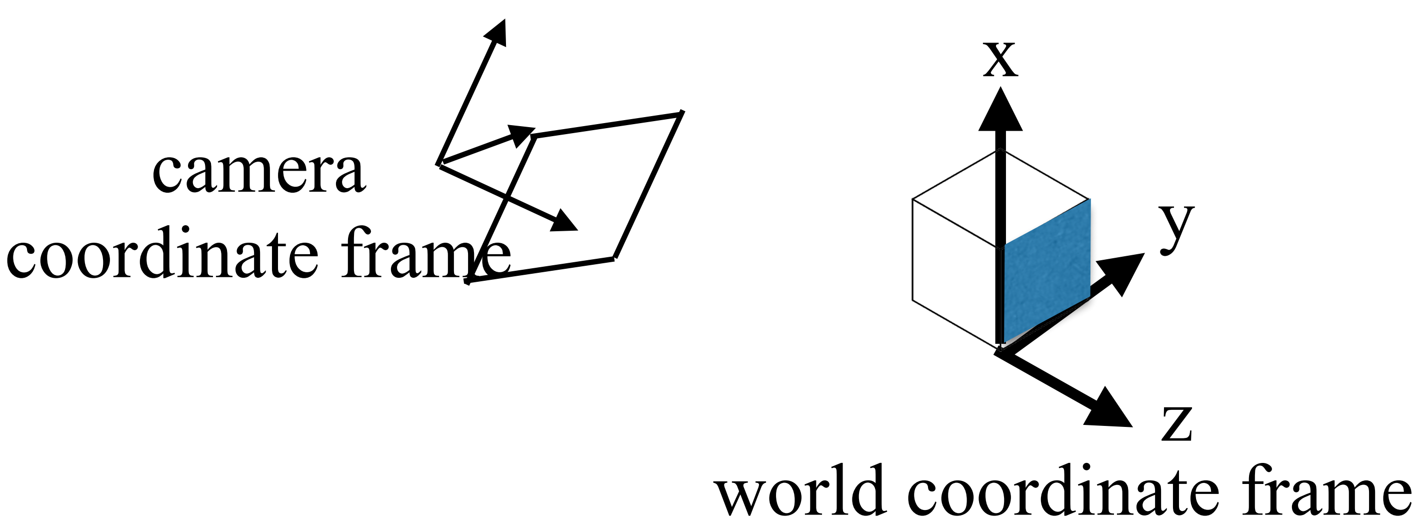


Let's analyze the projection of a 3D plane

Place world coordinate frame on blue object plane

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

Plug in Z=0



Let's analyze the projection of a 3D plane

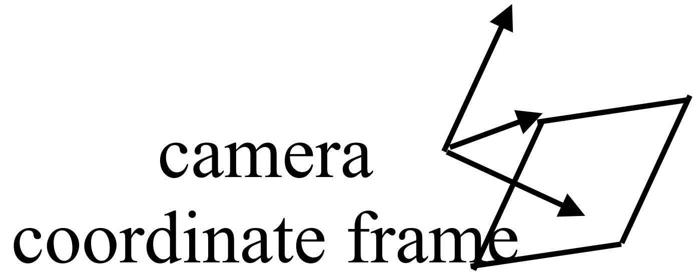
Place world coordinate frame on blue object plane

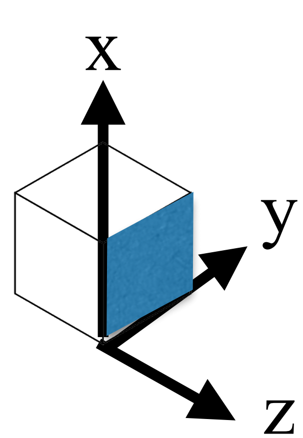
$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Plug in Z=0

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

Convert between 2D location on object plane and 2D image coordinate with a 3X3 matrix H





Let's analyze the projection of a 3D plane

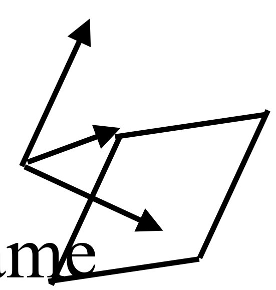
Place world coordinate frame on blue object plane

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

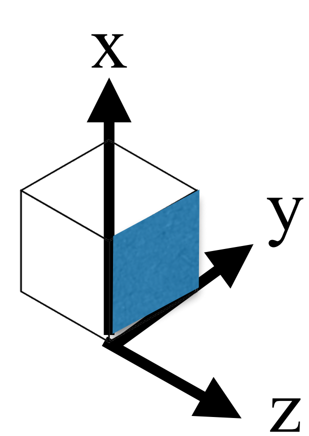
Plug in Z=0

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

Convert between 2D location on object plane and 2D image coordinate with a 3X3 matrix H



camera coordinate frame



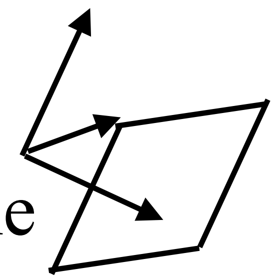
world coordinate frame

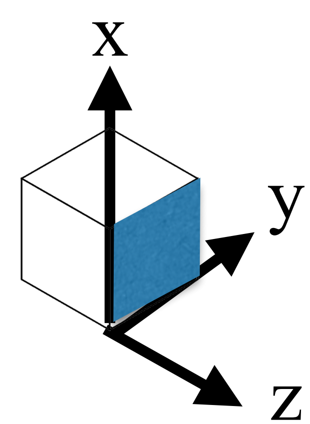
$$x = \frac{m_{11}X + m_{12}Y + m_{14}}{m_{31}X + m_{32}Y + m_{34}}$$

$$y = \frac{m_{21}X + m_{22}Y + m_{24}}{m_{31}X + m_{32}Y + m_{34}}$$

Inverse homographies

camera
coordinate frame





world coordinate frame

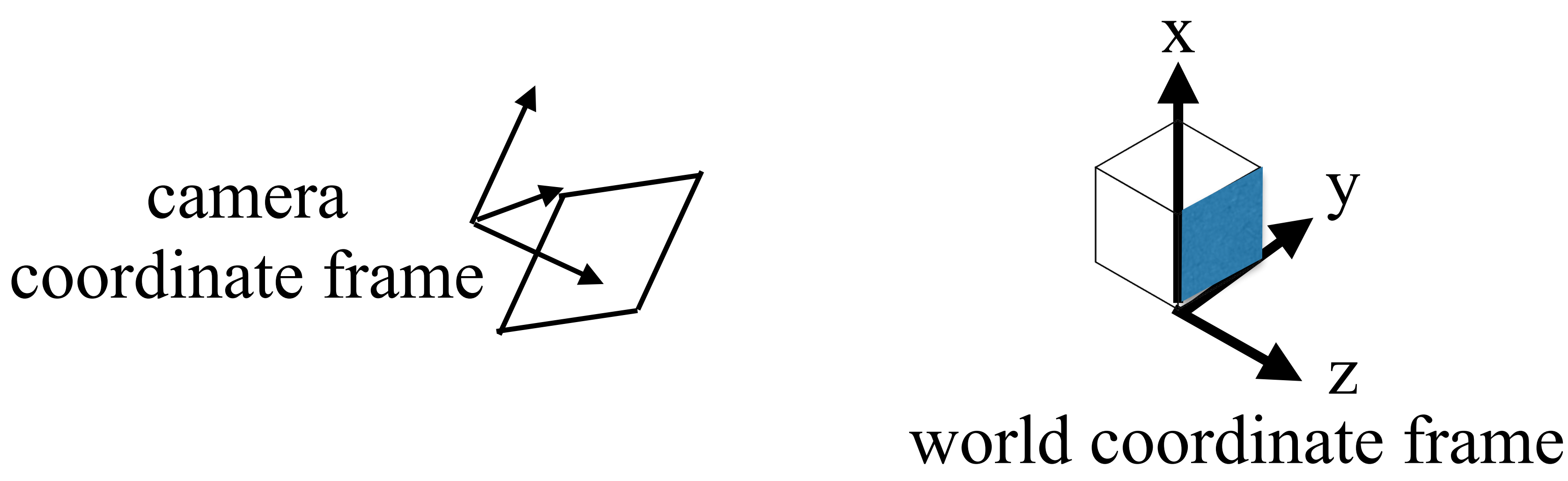
Given a point in world plane (X,Y), compute image position (x,y) as follows:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv H \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$x = \frac{m_{11}X + m_{12}Y + m_{14}}{m_{31}X + m_{32}Y + m_{34}}$$
$$y = \frac{m_{21}X + m_{22}Y + m_{24}}{m_{31}X + m_{32}Y + m_{34}}$$

Given a point in image (x,y), compute position on world plane (X,Y) as follows:

Inverse homographies



Given a point in world plane (X,Y), compute image position (x,y) as follows:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv H \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

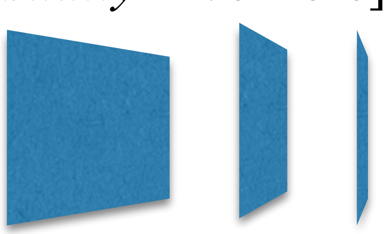
$$x = \frac{m_{11}X + m_{12}Y + m_{14}}{m_{31}X + m_{32}Y + m_{34}}$$
$$y = \frac{m_{21}X + m_{22}Y + m_{24}}{m_{31}X + m_{32}Y + m_{34}}$$

Given a point in image (x,y), compute position on world plane (X,Y) as follows:

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \equiv H^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

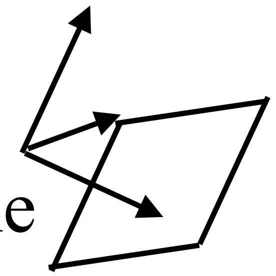
$$X = \frac{m'_{11}x + m'_{12}y + m'_{14}}{m'_{31}x + m'_{32}y + m'_{34}}$$
$$Y = \frac{m'_{21}x + m'_{22}y + m'_{24}}{m'_{31}x + m'_{32}y + m'_{34}}$$

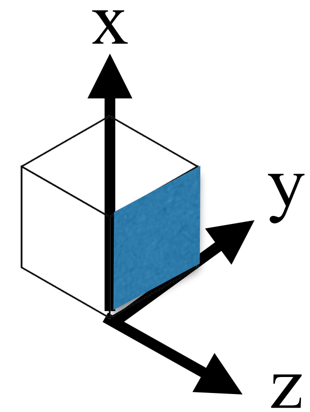
[Aside: H is *usually* invertible]



Inverse homographies

camera
coordinate frame





world coordinate frame

Given a point in world plane (X,Y), compute image position (x,y) as follows:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv H \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

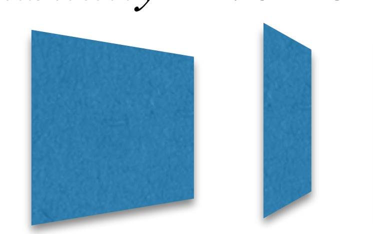
$$x = \frac{m_{11}X + m_{12}Y + m_{14}}{m_{31}X + m_{32}Y + m_{34}}$$
$$y = \frac{m_{21}X + m_{22}Y + m_{24}}{m_{31}X + m_{32}Y + m_{34}}$$

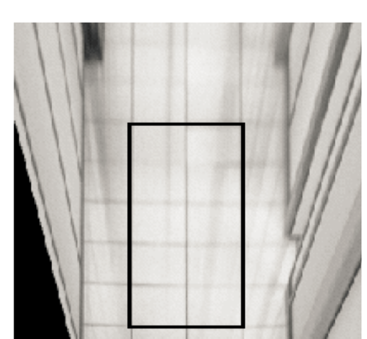
Given a point in image (x,y), compute position on world plane (X,Y) as follows:

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \equiv H^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$X = \frac{m'_{11}x + m'_{12}y + m'_{14}}{m'_{31}x + m'_{32}y + m'_{34}}$$
$$Y = \frac{m'_{21}x + m'_{22}y + m'_{24}}{m'_{31}x + m'_{32}y + m'_{34}}$$

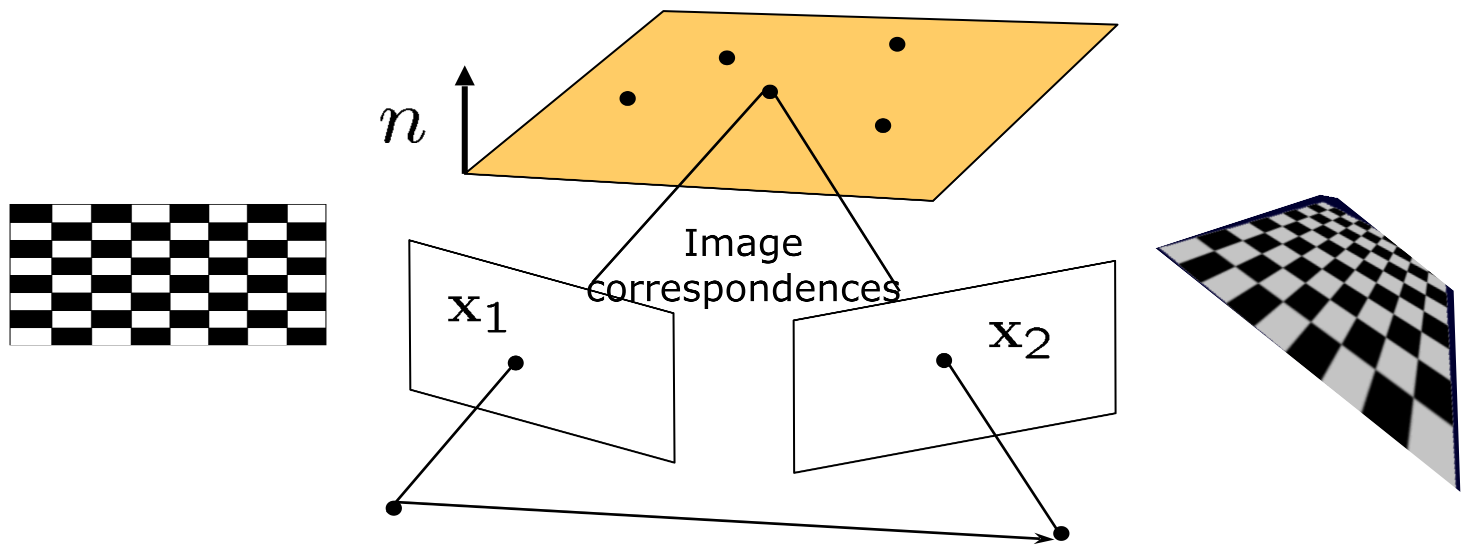
[Aside: H is *usually* invertible]





useful for frontalizing (and vice versa)

Relating 2 camera views of the same 3D plane



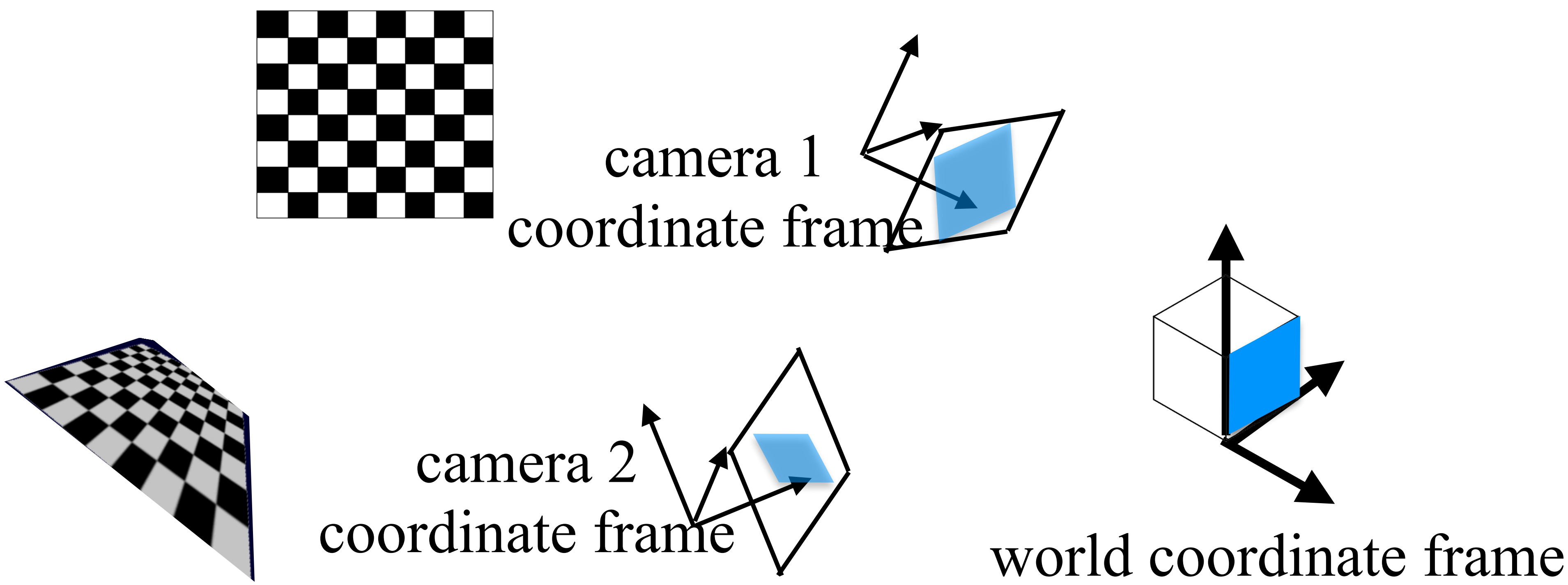
$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \equiv H_1 \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

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

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

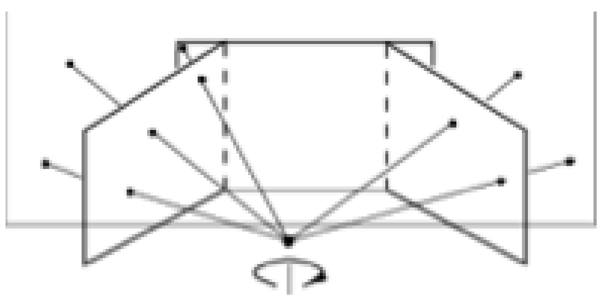
Relating 2 camera views of the same 3D plane: another perspective

Place world coordinate frame on the *normalized image* plane (f=1) of another camera!



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

Special case of 2 views: rotations about camera center



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

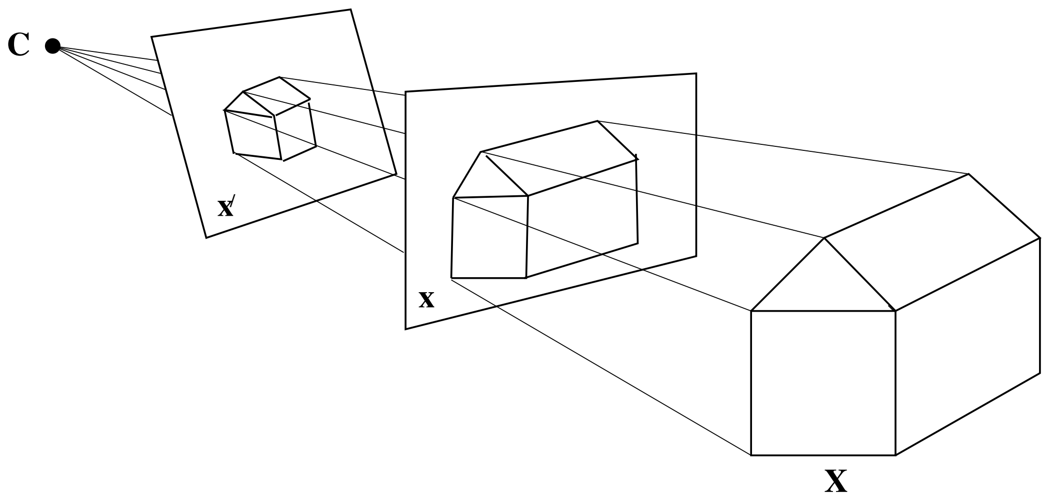
K_2

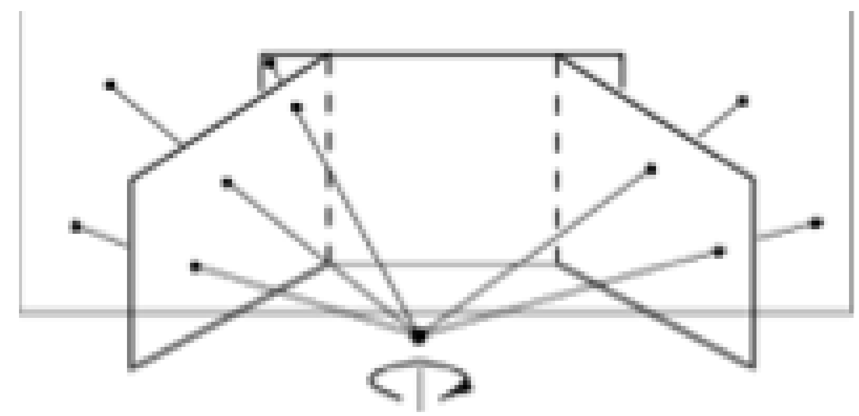
3D->2D projection: $\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} f_2 & 0 & 0 \\ 0 & f_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix}$

$\cdots$

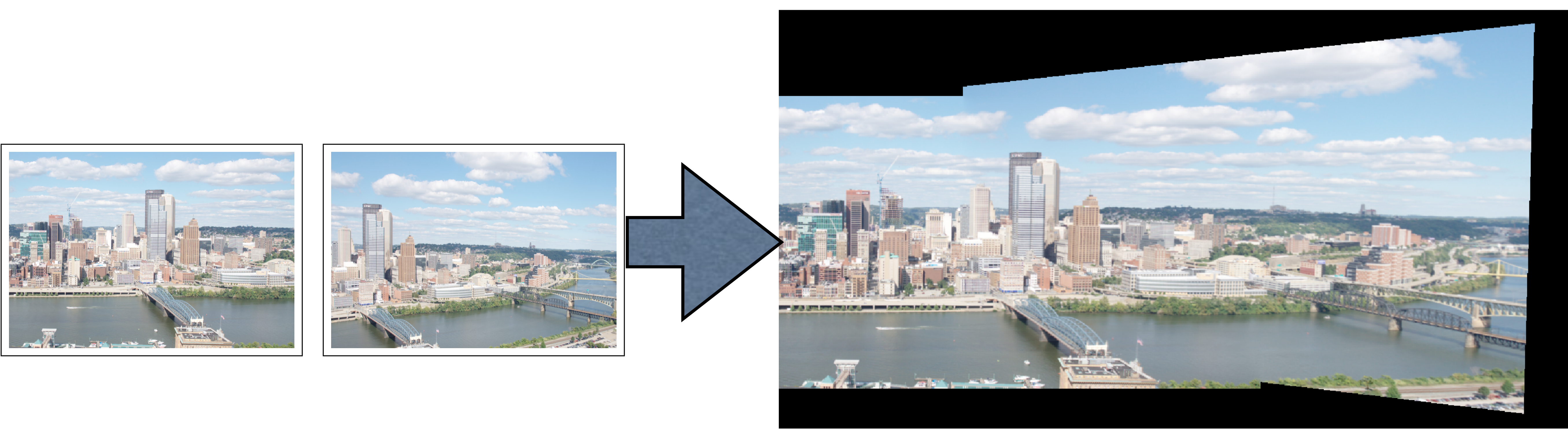
Combining both: $\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \equiv K_2 R K_1^{-1} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$

Geometric intuition: look at image of a planar image

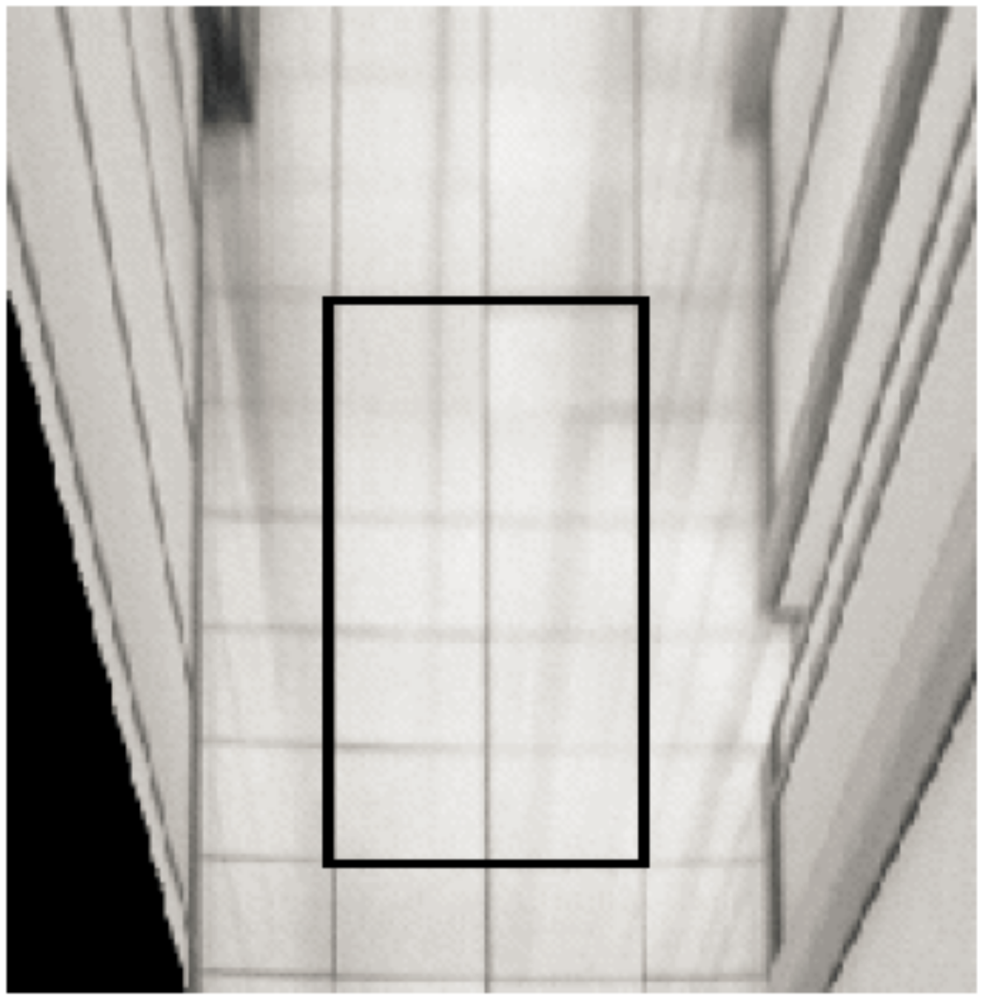




3D images are modelled with planar transformations,
regardless of 3D scene geometry!



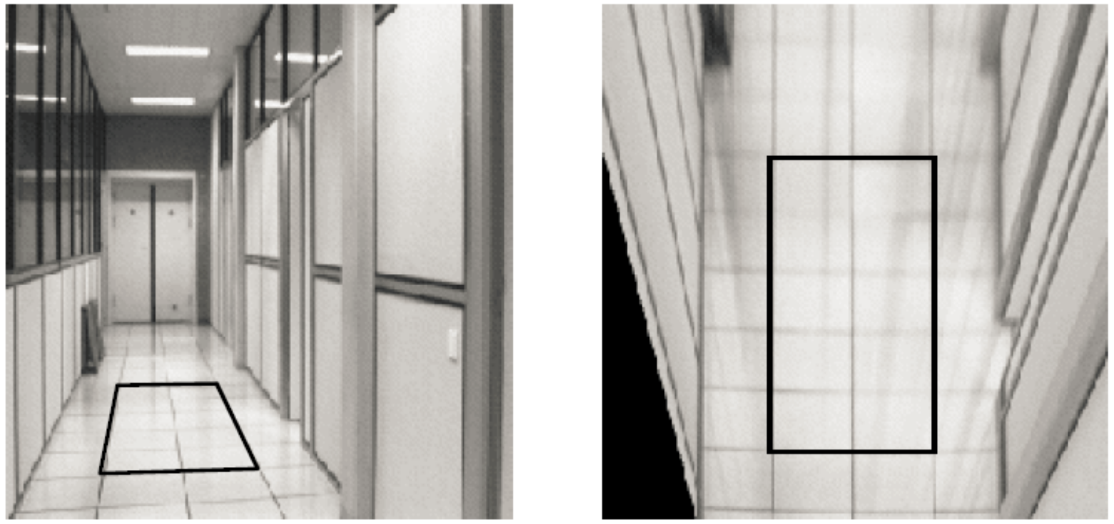
“Frontalizing” planes using homographies



from Hartley & Zisserman

Estimating homographies

Given sets of points in two views $\{(x_1,y_1)\}$ and $\{(x_2,y_2)\}$, how do we compute H?

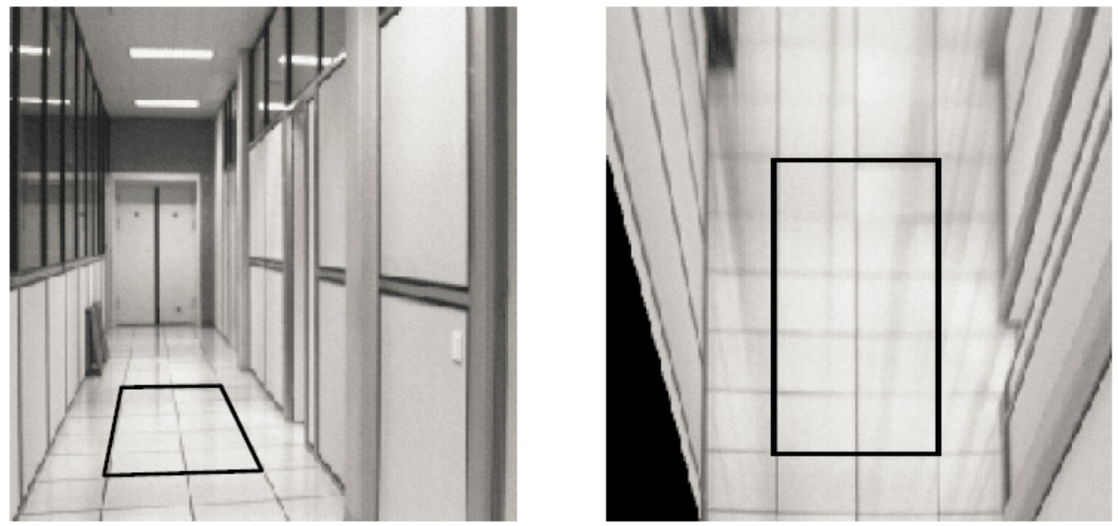


from Hartley & Zisserman

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

Estimating homographies

Given sets of points in two views $\{(x_1,y_1)\}$ and $\{(x_2,y_2)\}$, how do we compute H?



from Hartley & Zisserman

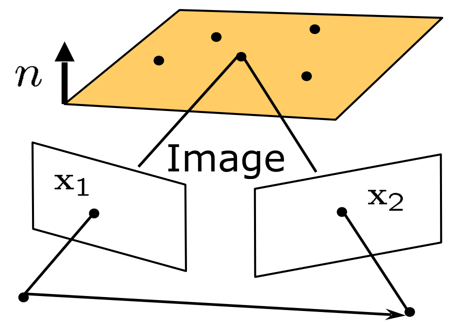
$$\lambda \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

$$x_2 = \frac{\lambda x_2}{\lambda} = \frac{a x_1 + b y_1 + c}{g x_1 + h y_1 + i}$$

Estimating Homographies (via the Direct Linear Transform)

[*kind of* similar to linear least squares for estimating warp]

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



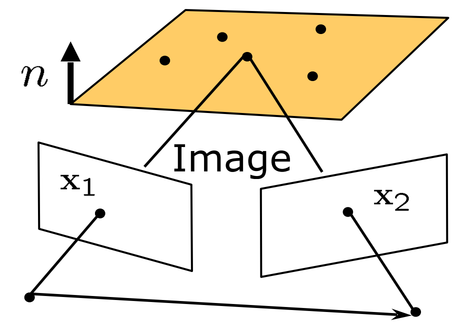
$$x_2(gx_1 + hy_1 + i) = ax_1 + by_1 + c$$

$$\vdots$$

Estimating Homographies (via the Direct Linear Transform)

[*kind of* similar to linear least squares for estimating warp]

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



$$x_2(gx_1 + hy_1 + i) = ax_1 + by_1 + c$$

$$\vdots$$

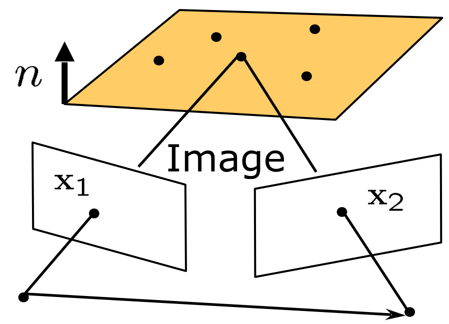
$$AH(:) = \begin{bmatrix} 0 \\ 0 \\ \vdots \end{bmatrix}$$ Homogenous linear system

How many degrees of freedom in H?
How many corresponding points needed?

Estimating Homographies (via the Direct Linear Transform)

[*kind of* similar to linear least squares for estimating warp]

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



$$x_2(gx_1 + hy_1 + i) = ax_1 + by_1 + c$$

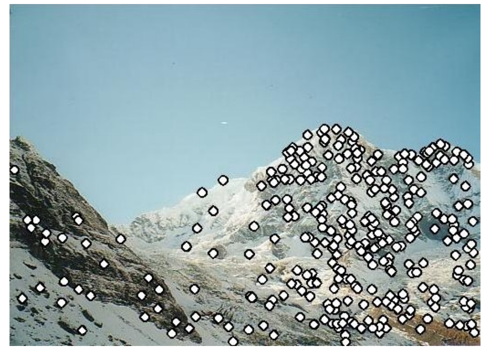
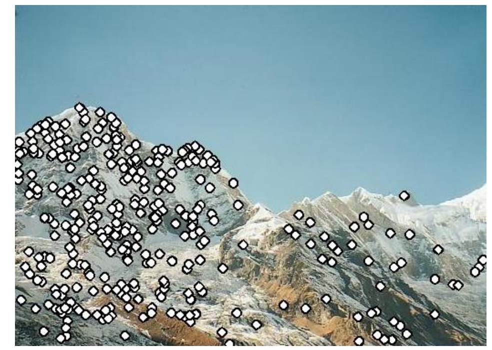
$$\vdots$$

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

How many degrees of freedom in H? 8; αH will produce same homogenous mapping; so let's fix $H_{33} = 1$

How many corresponding points needed? 4

RANSAC for homography fitting

Given 2 images with interest points and candidate matches

Repeat ***k*** times:

- Sample ***n(=4)*** points uniformly at random from left image.
- Fit (homography) warp to these ***n*** points and their correspondences (with SVD)
- Find inliers among the remaining left-image points (i.e., whose warped positions land close to right-image correspondence)
- Return warp with largest inlier set

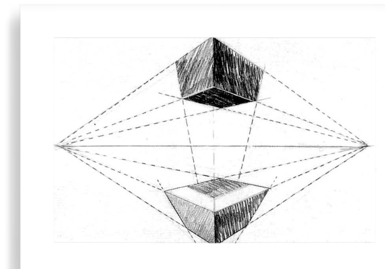
(Optionally return line with optimal least-squares inlier fit)

Source: M. Pollefeys

Agenda

- Pinhole optics
  - Perspective projection (vanishing points, horizon, object height)
  - Camera matrices (intrinsics + extrinsics)
  - Homographies (2 views of plane, rotation)

- Camera models
  - **Properties of camera matrices (DOF, geometric intuition, pixel2rays)**
  - Simplified cameras: orthographic, scaled orthographic, paraperspective, affine
  - Camera calibration (DLT v reprojection error)



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

Recall camera projection

[Using rows x columns]

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

$$= K_{3\times 3} \begin{bmatrix} R_{3\times 3} & T_{3\times 1} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

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

Question:how many DOF in M?

Can any 3x4 matrix be a camera matrix?

Recall camera projection

[Using rows x columns]

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

$$= K_{3\times 3} \begin{bmatrix} R_{3\times 3} & T_{3\times 1} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

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

Question:how many DOF in M?

11

Can any 3x4 matrix be a camera matrix?

Recall camera projection

[Using rows x columns]

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

$$= K_{3\times 3} \begin{bmatrix} R_{3\times 3} & T_{3\times 1} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

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

Question:how many DOF in M?

11

Can any 3x4 matrix be a camera matrix?

No; we'll derive the necessary conditions

Notation

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = M_{3\times 4} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = A_{3\times 3} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + b_{3\times 1}$$

$$M = \begin{bmatrix} \mathbf{m_1}^T \\ \mathbf{m_2}^T \\ \mathbf{m_3}^T \end{bmatrix}, \quad A = \begin{bmatrix} \mathbf{a_1}^T \\ \mathbf{a_2}^T \\ \mathbf{a_3}^T \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_2 \end{bmatrix}$$

Notation & claims

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = M_{3\times 4} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = A_{3\times 3} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + b_{3\times 1}$$

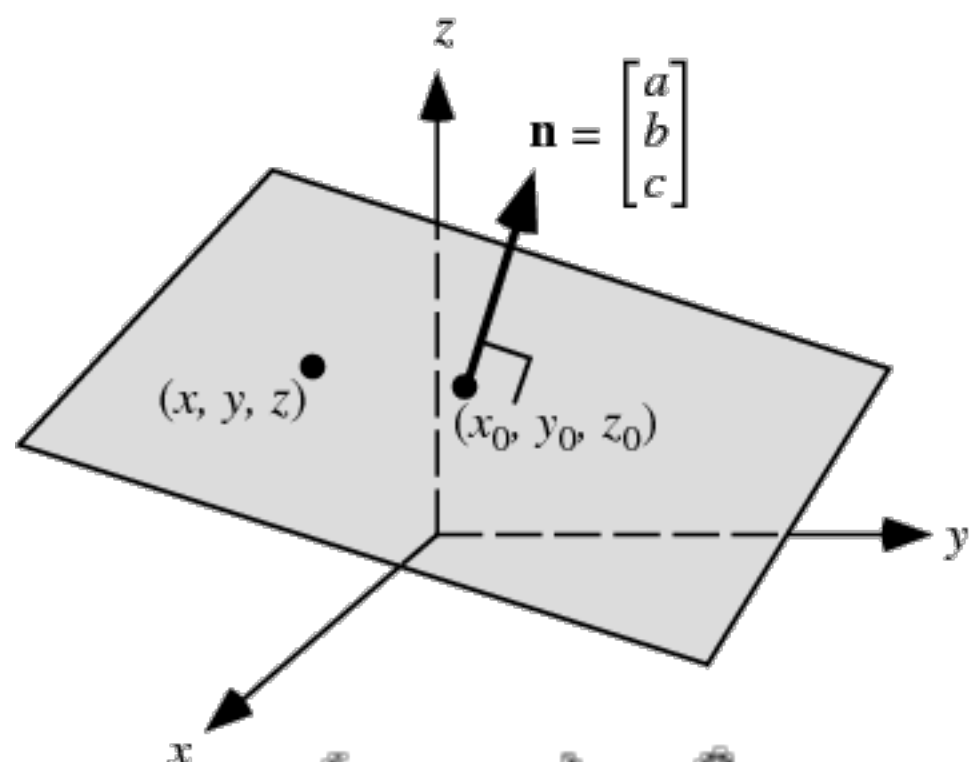
Claims:

1. A 3x4 matrix 'M' can be a camera matrix iff det(A) is not zero (geometric intution later)
2. M is determined only up to a scale factor (easy to show)

$$M = \begin{bmatrix} \mathbf{m_1}^T \\ \mathbf{m_2}^T \\ \mathbf{m_3}^T \end{bmatrix}, \quad A = \begin{bmatrix} \mathbf{a_1}^T \\ \mathbf{a_2}^T \\ \mathbf{a_3}^T \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_2 \end{bmatrix}$$

Recall: eqn of plane

https://mathworld.wolfram.com/Plane.html

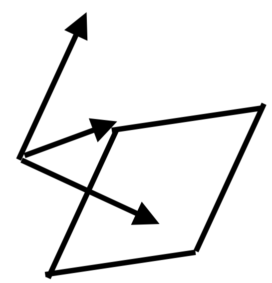


$$\mathbf{n} \cdot (\mathbf{x} - \mathbf{x}_0) = 0,$$

$$ax + by + cz + d = 0,$$

Given (a,b,c,d) with unit-norm (a,b,c):
(a,b,c) = normal of plane
|d| = distance of plane from origin

Building geometric intuitions about camera matrix



$$x = \frac{1}{\lambda}\left(\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_1} + b_1\right)$$

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 & b_1 \\ \mathbf{a}_2 & b_2 \\ \mathbf{a}_3 & b_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \Rightarrow$$

$$y = \frac{1}{\lambda}\left(\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2\right)$$

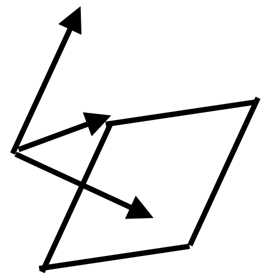
$$\lambda = \begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_3} + b_3$$

Set of 3D points that project to image x = 0:

Set of 3D points that project to image y = 0:

Set of 3D points that project to x = inf or y = inf:

Building geometric intuitions about camera matrix



$$x = \frac{1}{\lambda}(\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_1} + b_1)$$

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 & b_1 \\ \mathbf{a}_2 & b_2 \\ \mathbf{a}_3 & b_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \Rightarrow$$

$$y = \frac{1}{\lambda}(\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2)$$

$$\lambda = \begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_3} + b_3$$

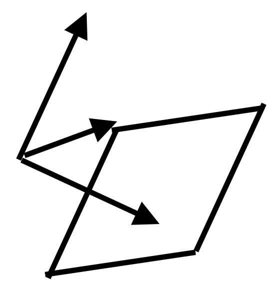
Set of 3D points that project to image x = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_1} + b_1 = 0$

Set of 3D points that project to image y = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2 = 0$

Set of 3D points that project to x = inf or y = inf: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_3} + b_3 = 0$

What do these sets look like in 3D world?

Building geometric intuitions about camera matrix



$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 & b_1 \\ \mathbf{a}_2 & b_2 \\ \mathbf{a}_3 & b_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \Rightarrow$$

$$x = \frac{1}{\lambda}(\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_1} + b_1)$$

$$y = \frac{1}{\lambda}(\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2)$$

$$\lambda = \begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_3} + b_3$$

Set of 3D points that project to image x = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_1} + b_1 = 0$

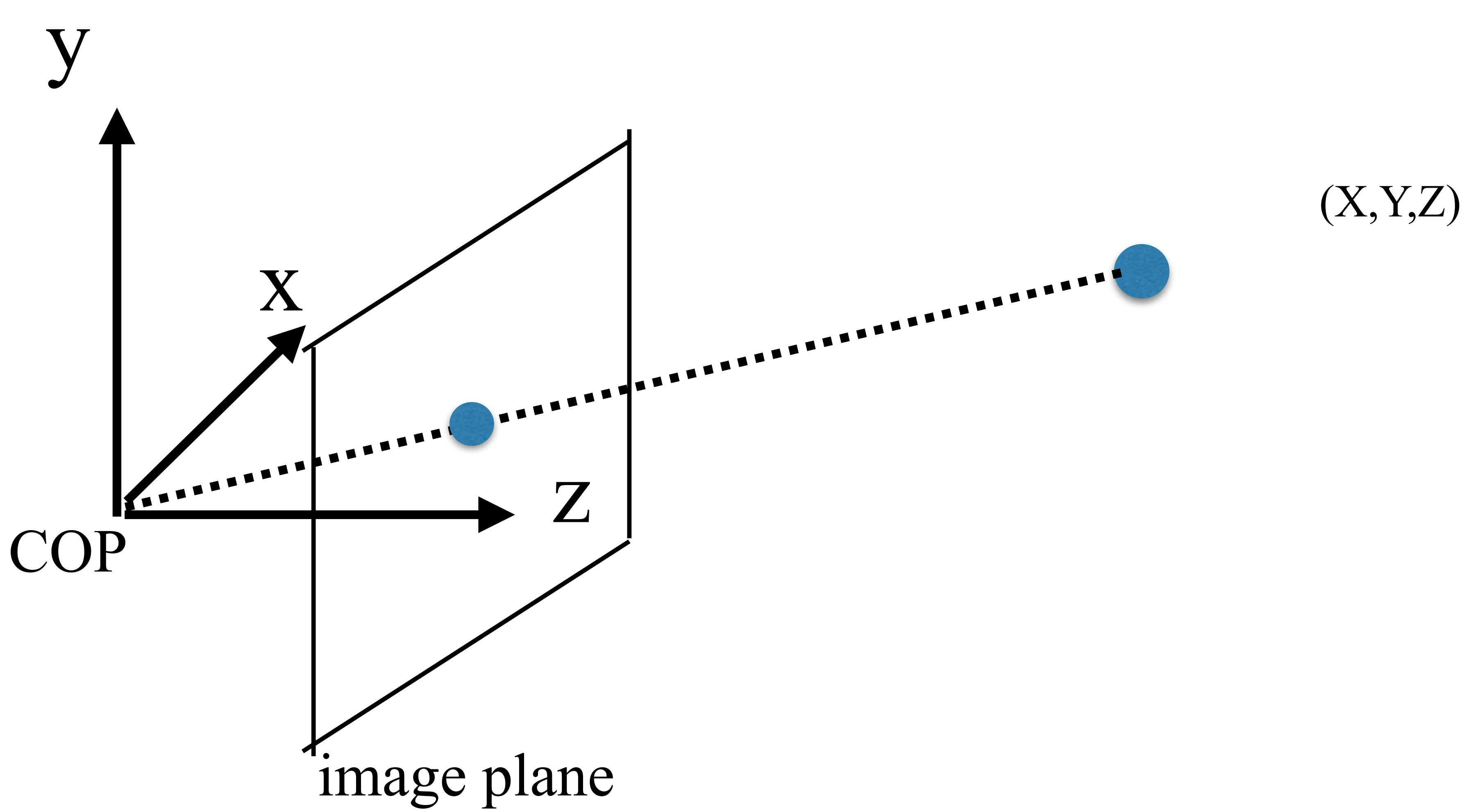
Set of 3D points that project to image y = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2 = 0$

Set of 3D points that project to x = inf or y = inf: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_3} + b_3 = 0$

What do these sets look like in 3D world?

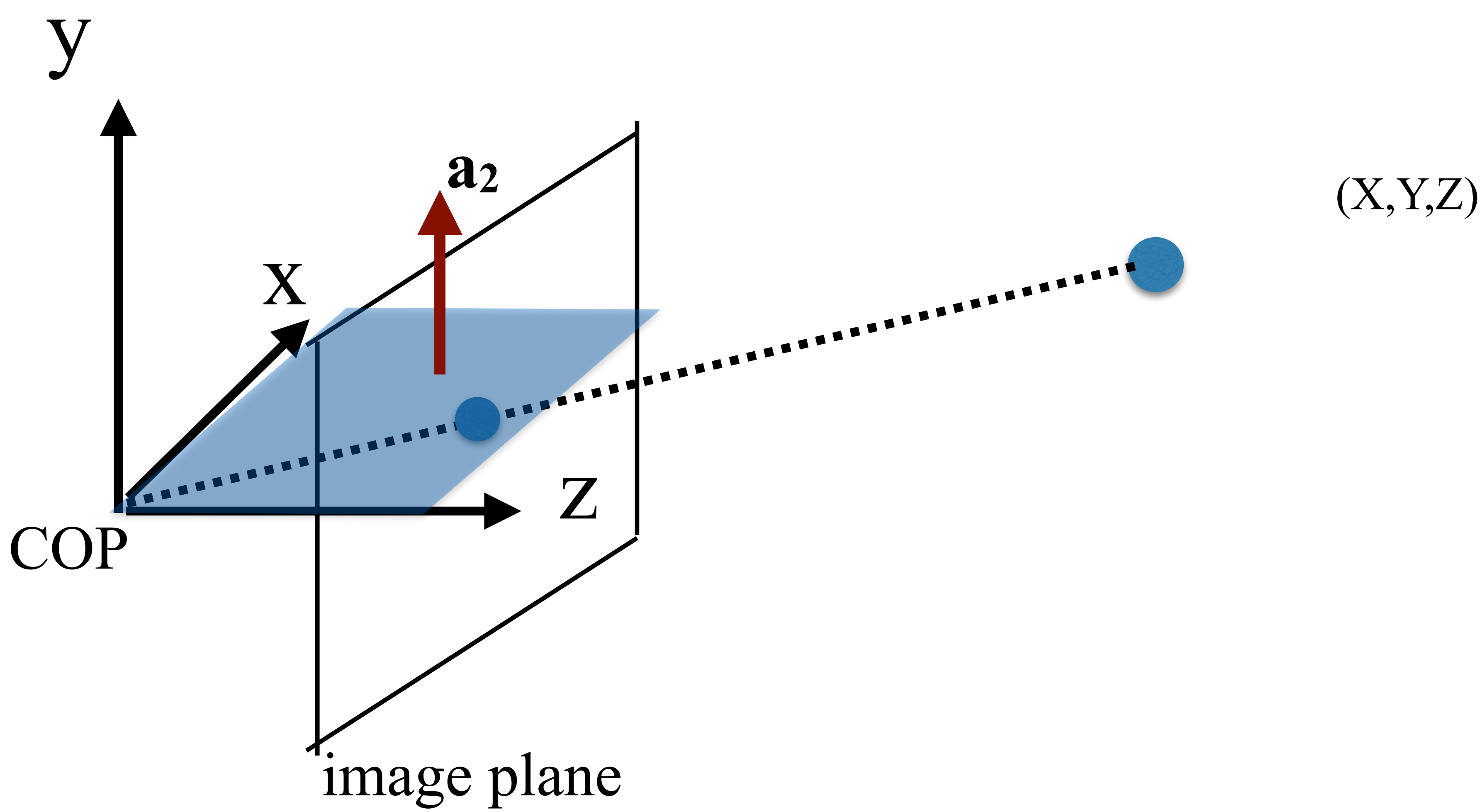
Planes who’s normal vector and distance from origin reveal the structure of the camera matrix (in world coordinates)!

Geometric intution



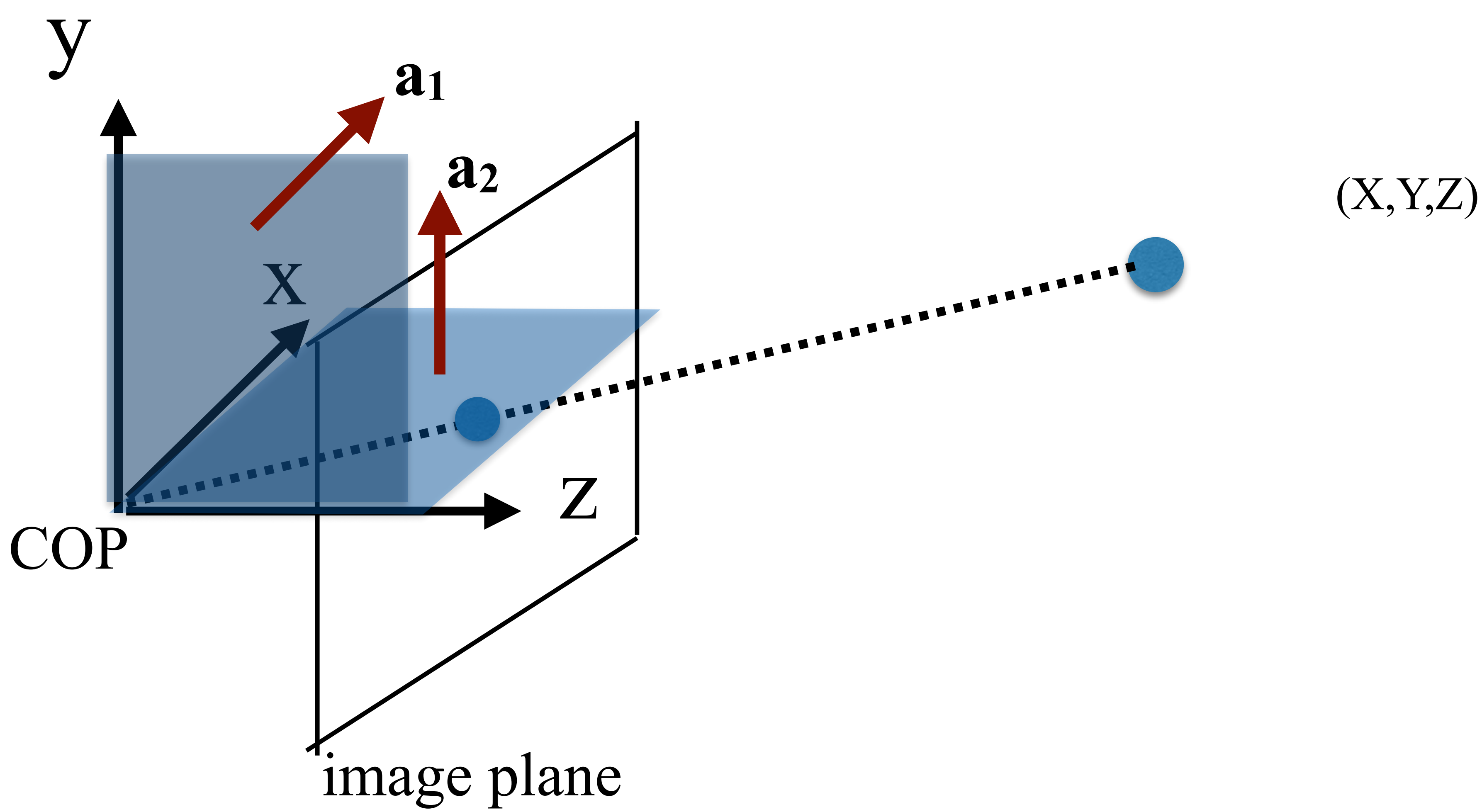
Set of 3D points that project to $y = 0$:

Geometric intution



Set of 3D points that project to y = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2 = 0$

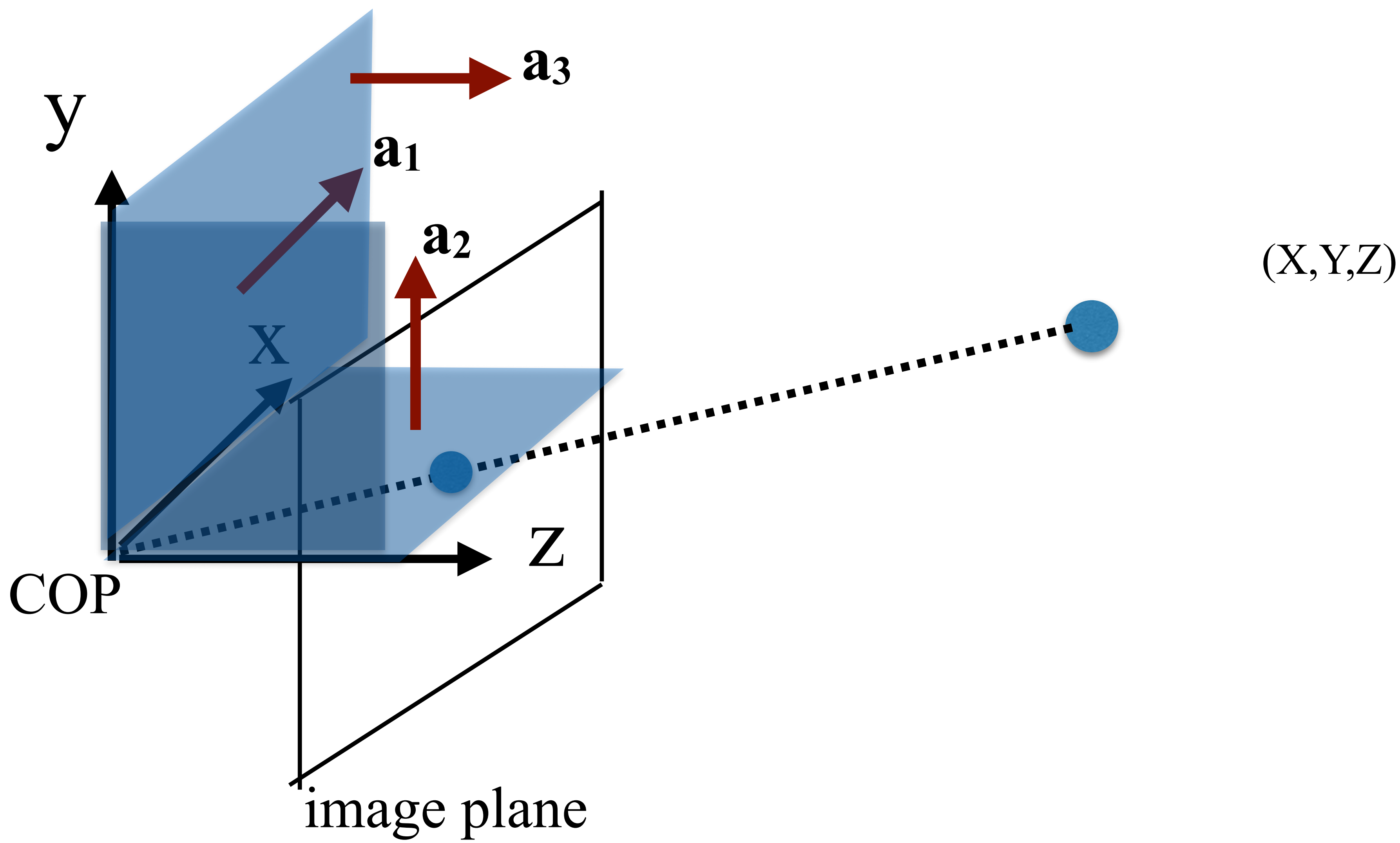
Geometric intution



Set of 3D points that project to x = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_1} + b_1 = 0$

Set of 3D points that project to y = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2 = 0$

Geometric intution

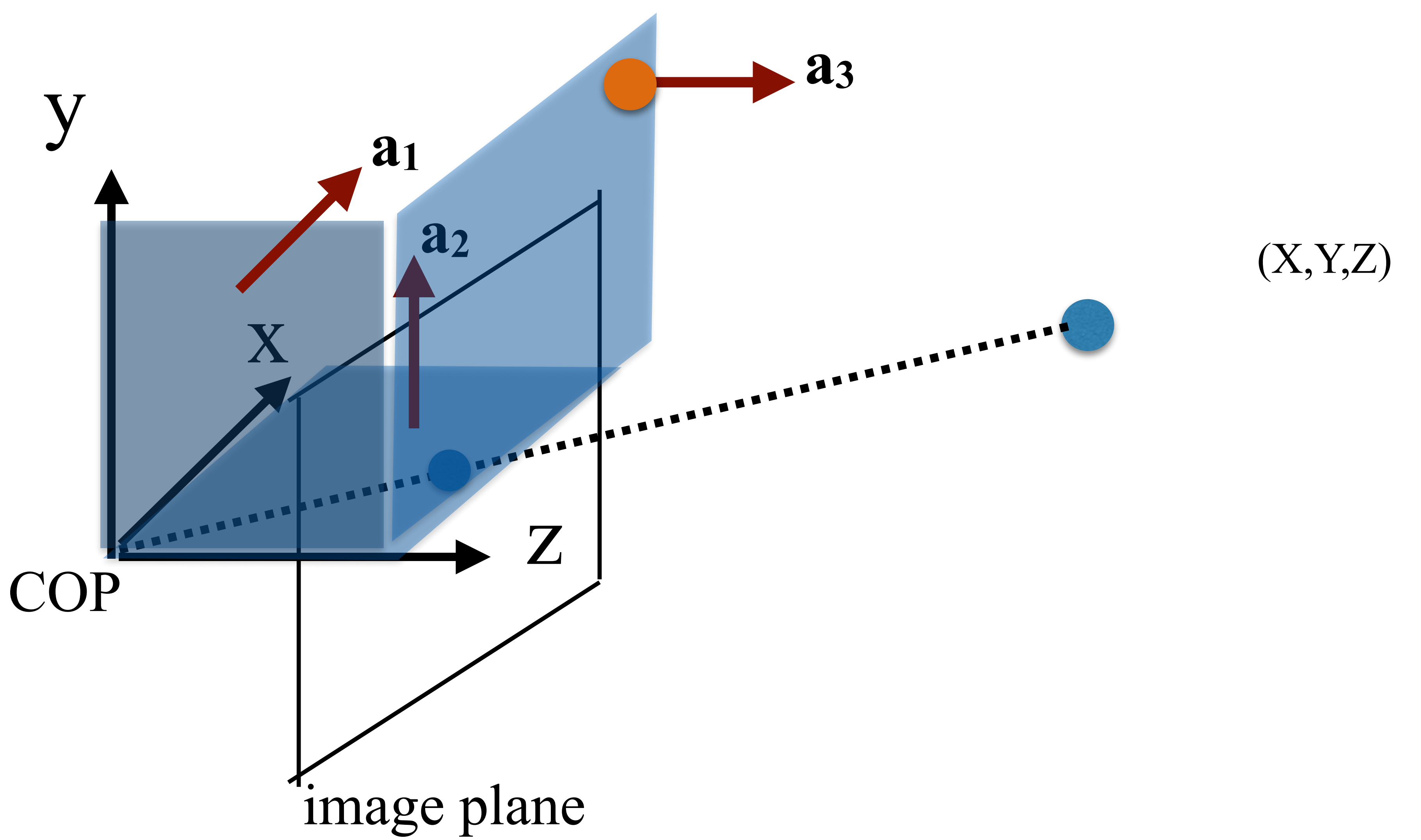


Set of 3D points that project to x = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_1} + b_1 = 0$

Set of 3D points that project to y = 0: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_2} + b_2 = 0$

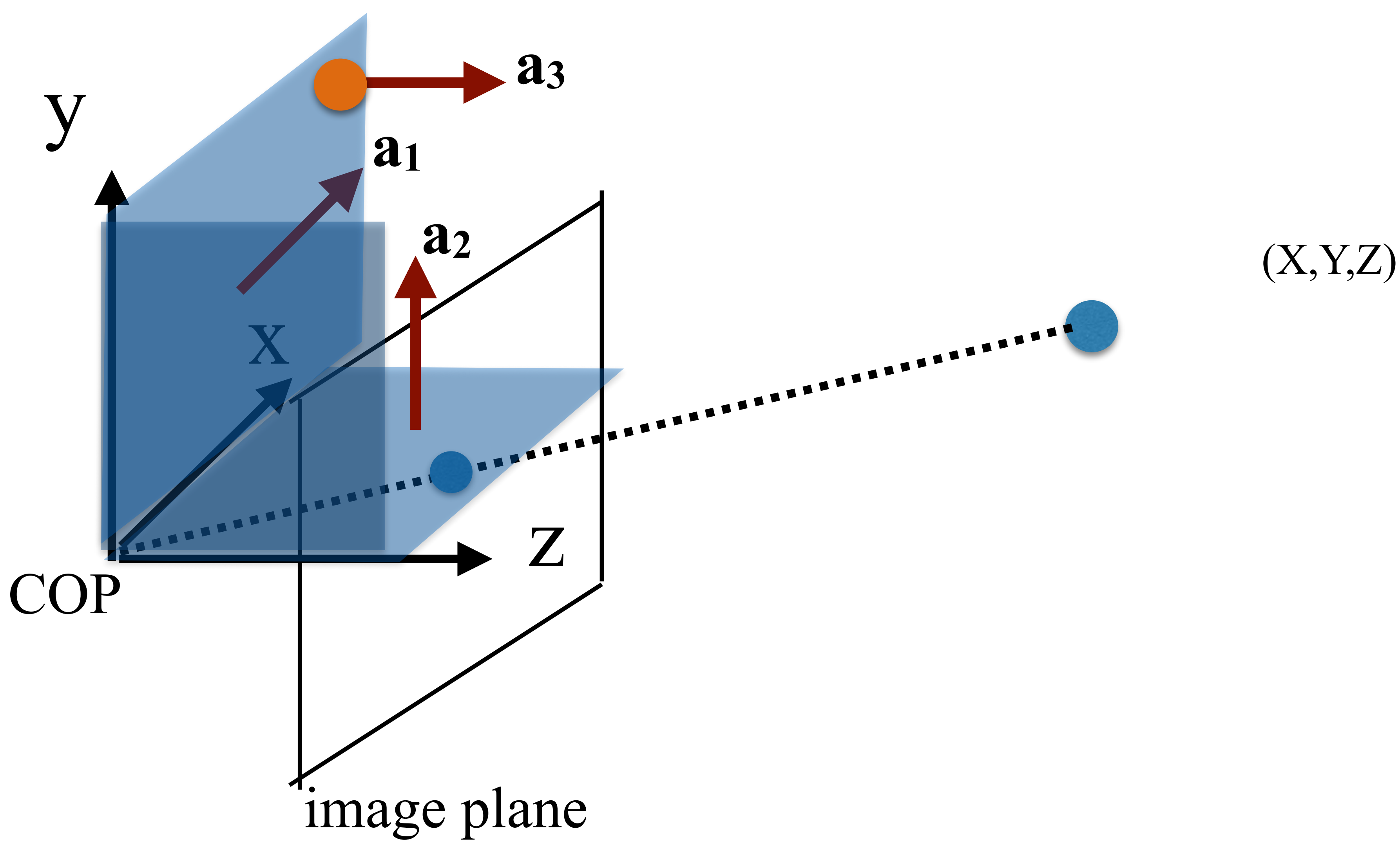
Set of 3D points that project to x = inf or y = inf: $\begin{bmatrix} X & Y & Z \end{bmatrix} \mathbf{a_3} + b_3 = 0$

Geometric intution



Why is $\mathbf{a}_3$ at infinity? Think of image projection of orange point as it shifts to plane z=0

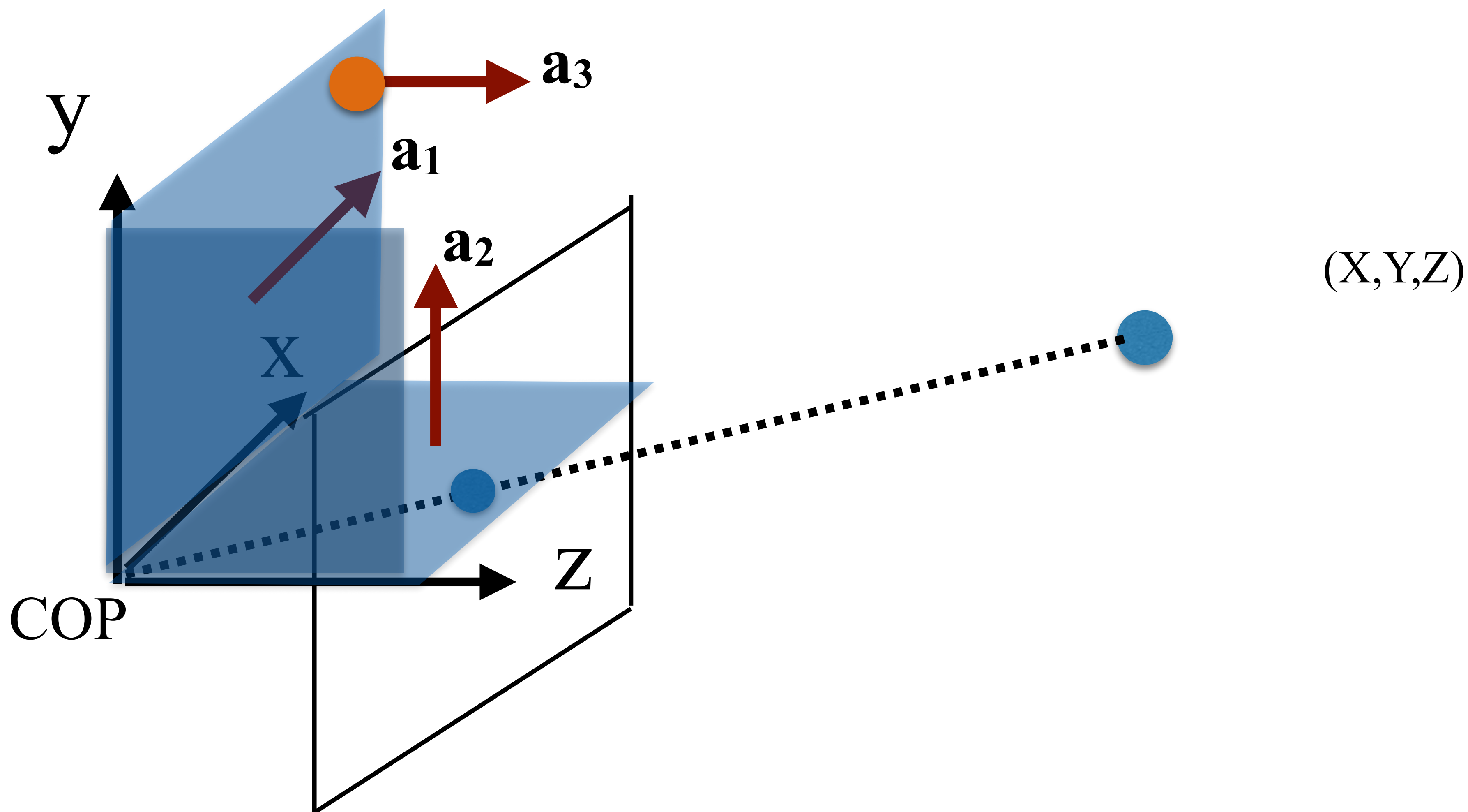
Geometric intution



Why is $\mathbf{a}_3$ at infinity? Think of image projection of orange point as it shifts to plane z=0

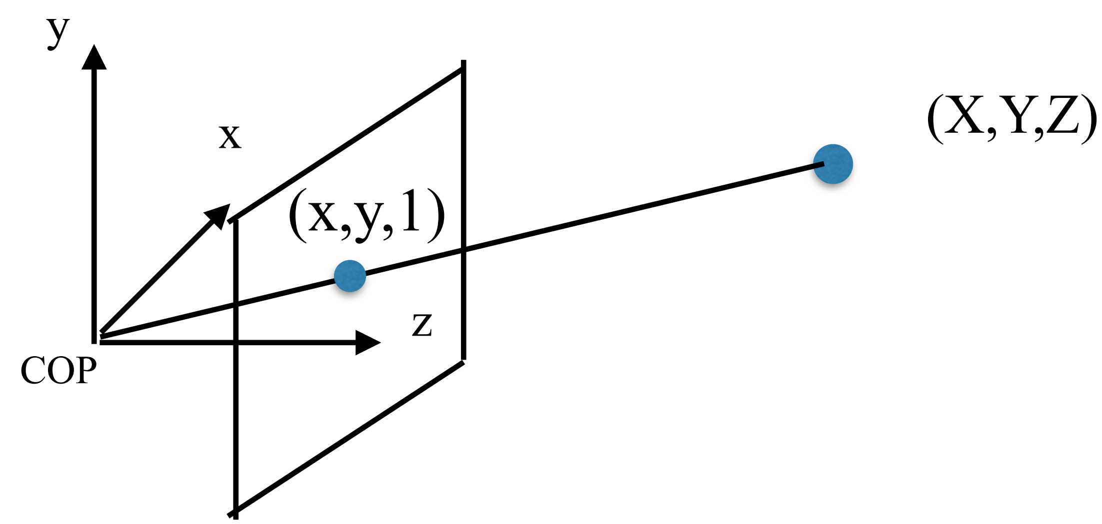
Geometric intution

Det(A_{3x3}) = 0 iff normals are *colinear*



Why is $\mathbf{a}_3$ at infinity? Think of image projection of orange point as it shifts to plane z=0

Other geometric properties

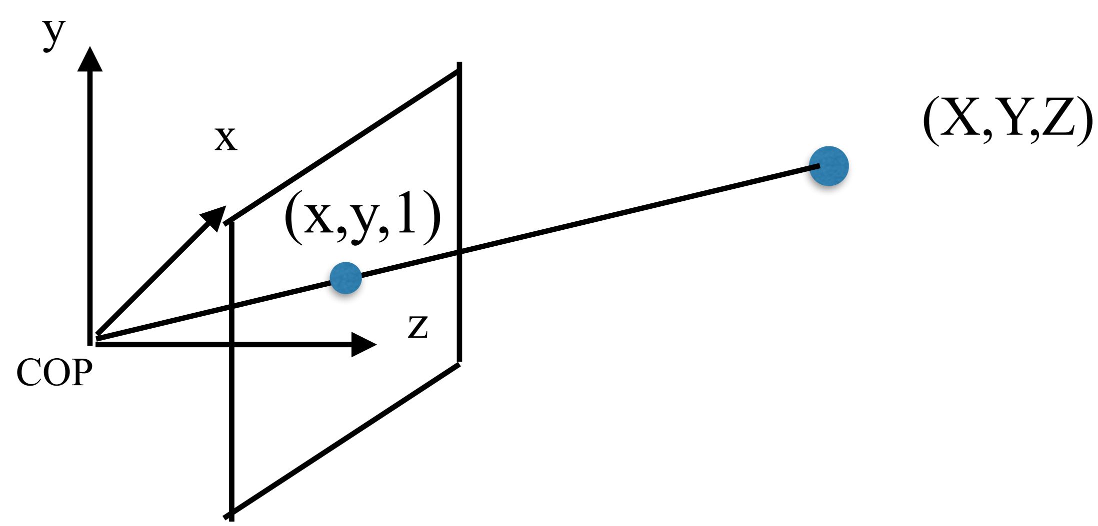


What's set of (X,Y,Z) points that project to same (x,y)?

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = A \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{b}$$

Solve above expression for (X,Y,Z) as a function of (x,y)

Other geometric properties



What's set of (X,Y,Z) points that project to same (x,y)?

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \lambda \mathbf{w} + \mathbf{b_0}$$

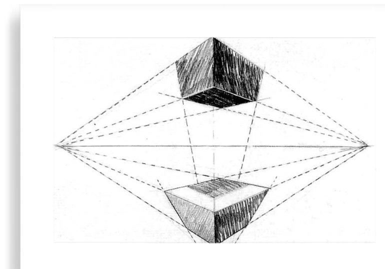
Direction of ray: $\mathbf{w} = A^{-1} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

COP: $\mathbf{b_0} = -A^{-1}\mathbf{b}$

(in world coodinates)

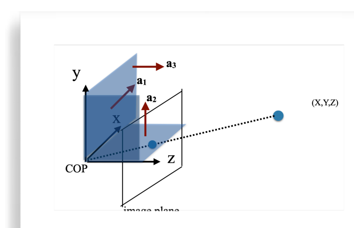
Agenda

- Pinhole optics
    - Perspective projection (vanishing points, horizon, object height)
    - Camera matrices (intrinsics + extrinsics)
    - Homographies (2 views of plane, rotation)

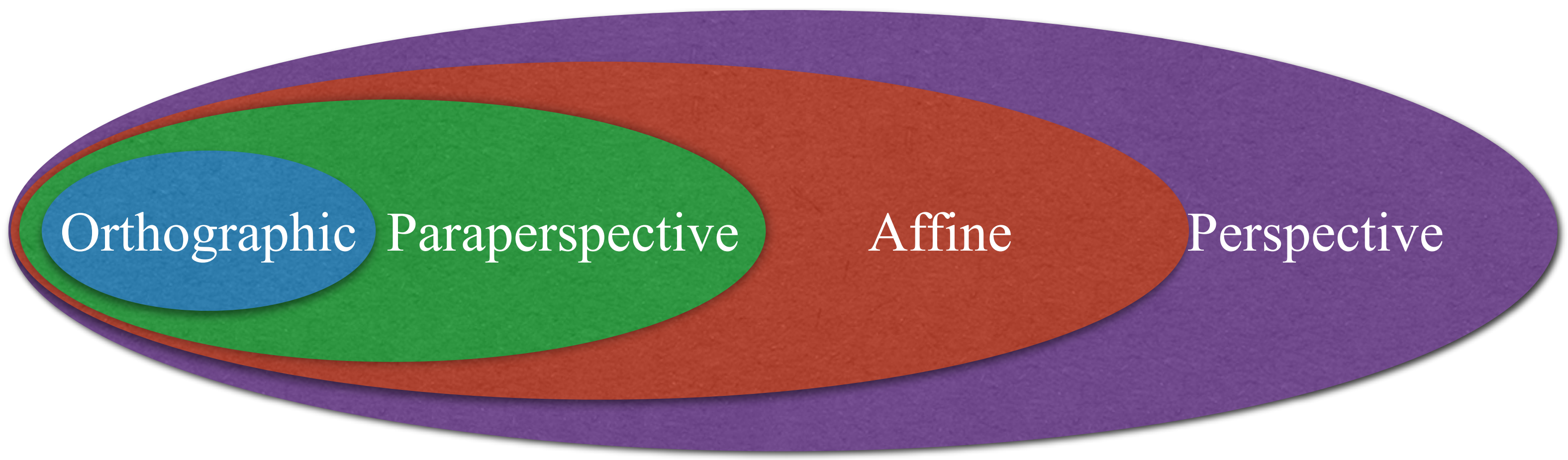


$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Camera models
    - Properties of camera matrices (DOF, geometric intuition, pixel2rays)
    - **Simplified cameras**: orthographic, scaled orthographic, paraperspective, affine
    - Camera calibration (DLT v reprojection error)



Where we are headed…



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

We'll explore *approximations* of M_{3x4} that are easier to work with…
and call out when such approximations hold!

Orthographic projection

"Keep stepping back while increasing focal length"

(as f and Z increase to infinity)

x = fX/Z
y = fY/Z

x =
y =

demo_renderCube.ipynb

Orthographic projection

"Keep stepping back while increasing focal length"

(as f and Z increase to infinity)

$x = fX/Z$
$y = fY/Z$

$x =$
$y =$

demo_renderCube.ipynb

Orthographic projection

"Keep stepping back while increasing focal length"

(as f and Z increase to infinity)

$$x = fX/Z \qquad\qquad x = X$$
$$y = fY/Z \qquad\qquad y = Y$$

demo_renderCube.ipynb

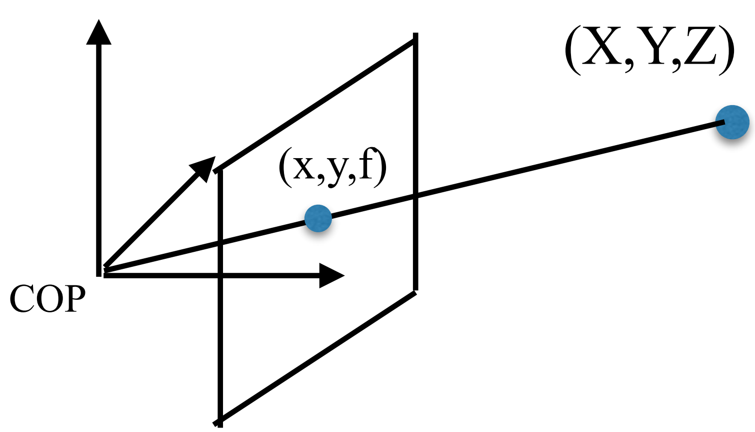
Orthographic projection

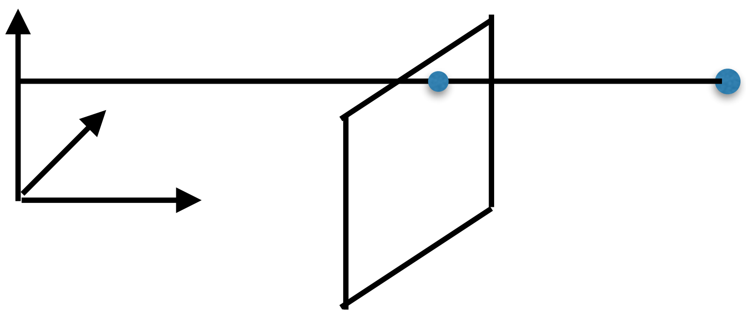
"Keep stepping back while increasing focal length"

(as f and Z increase to infinity)

$$x = fX/Z \qquad x = X$$
$$y = fY/Z \qquad y = Y$$

demo_renderCube.ipynb

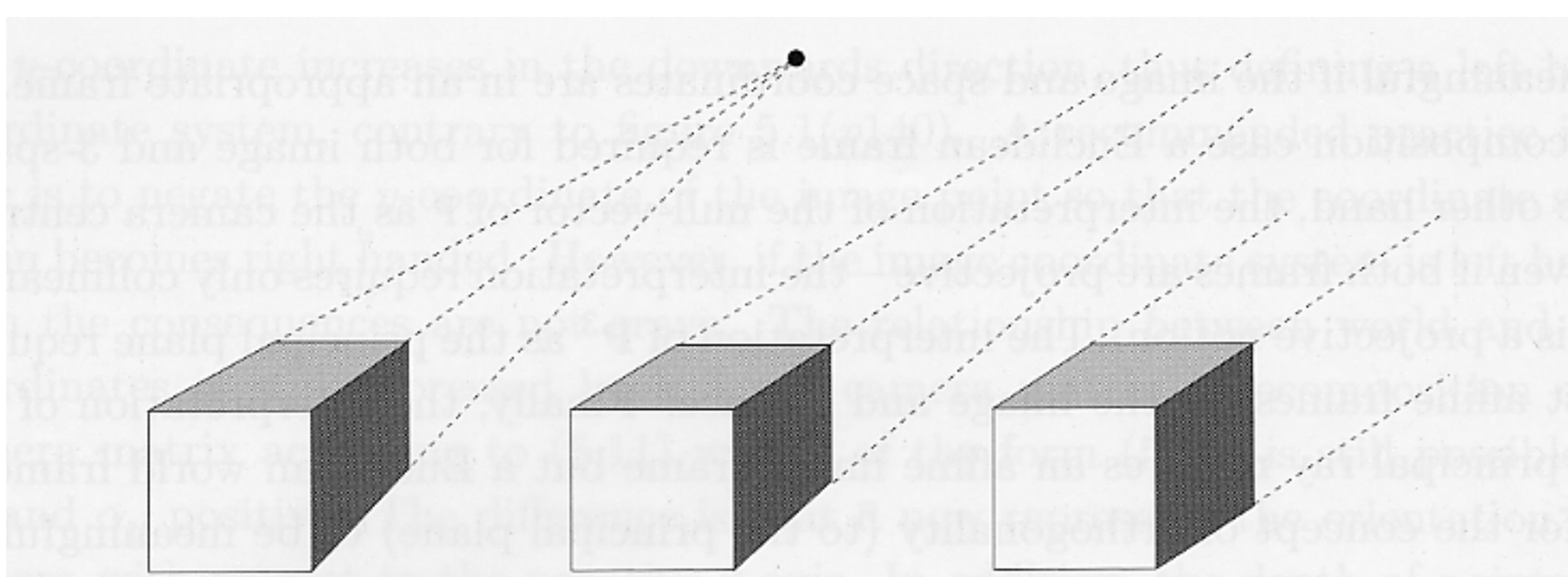




Life would be much simpler; we could trust angles and distances

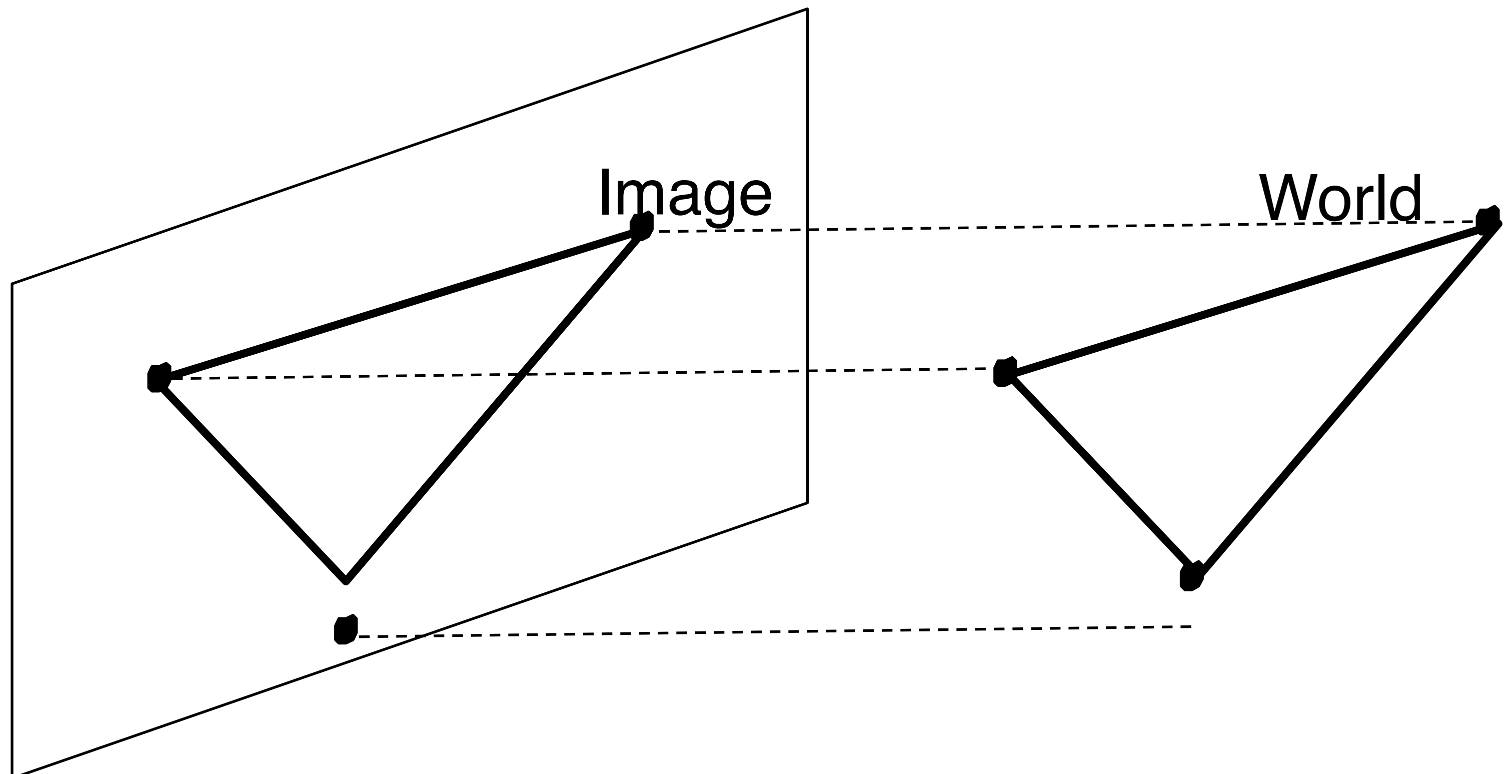
Perspective vs Orthogrpahic

Wide angle Standard Telephoto



(parallel lines project to parallel lines!)

Orthographic Projection



Whats M_{3x4} look like?

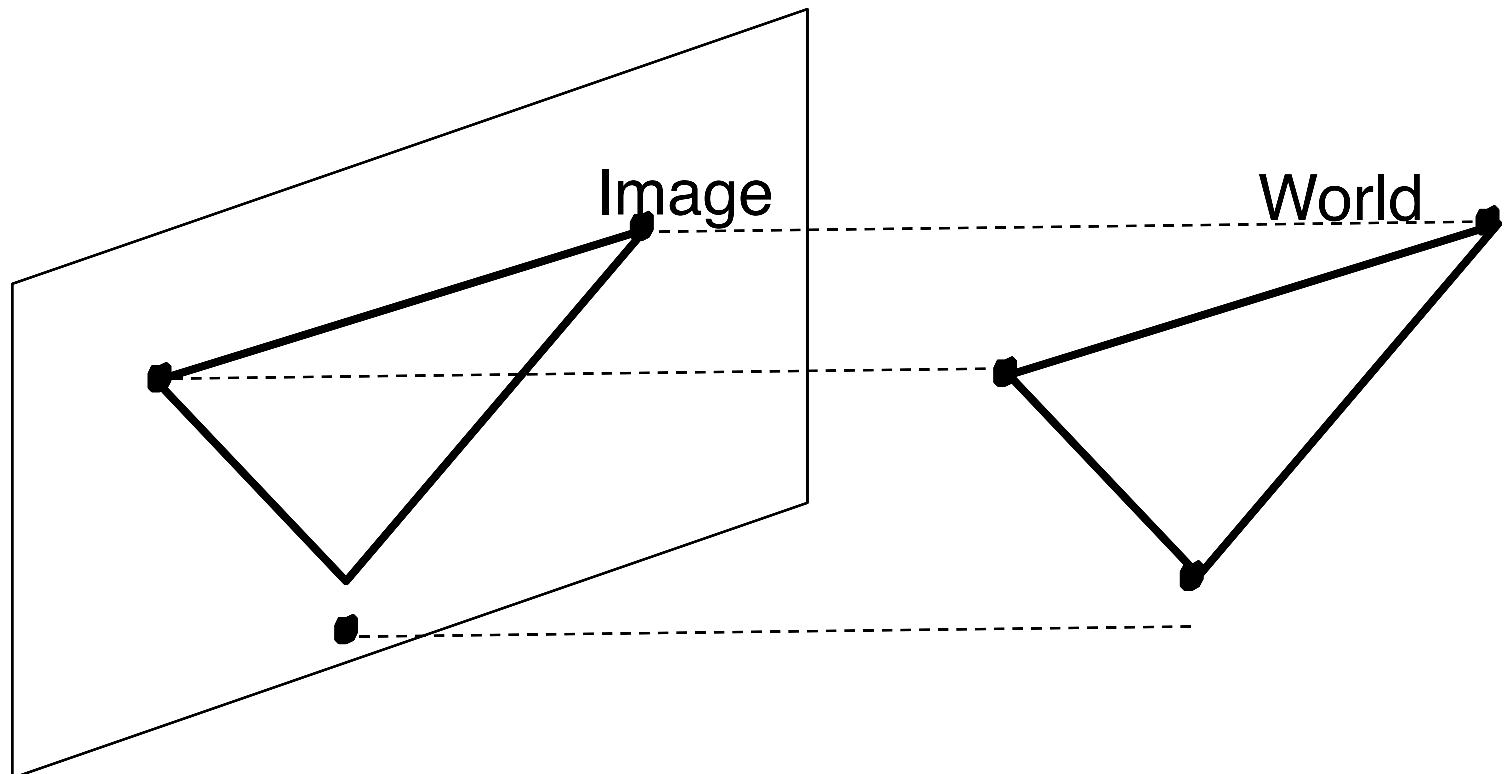
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \quad \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \Rightarrow x = X, y = Y$$

(empty slot denotes 0)

Incredibly convenient because camera projection is *linear*.

Can we make this less restrictive?

Orthographic Projection



Whats M_{3x4} look like?

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

(empty slot denotes 0)

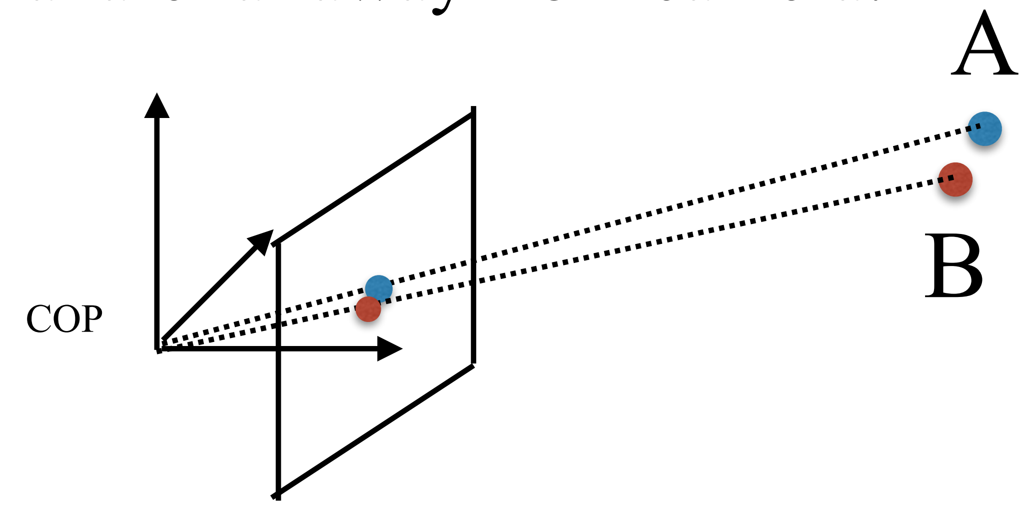
Incredibly convenient because camera projection is *linear*.

Can we make this less restrictive?

Scaled orthographic or "weak perspective" projection

Consider two points (A,B) at different depths that are far away from camera:

$$\begin{bmatrix} A_x \\ A_y \\ Z \end{bmatrix} \qquad \begin{bmatrix} B_x \\ B_y \\ Z + \Delta Z \end{bmatrix}$$

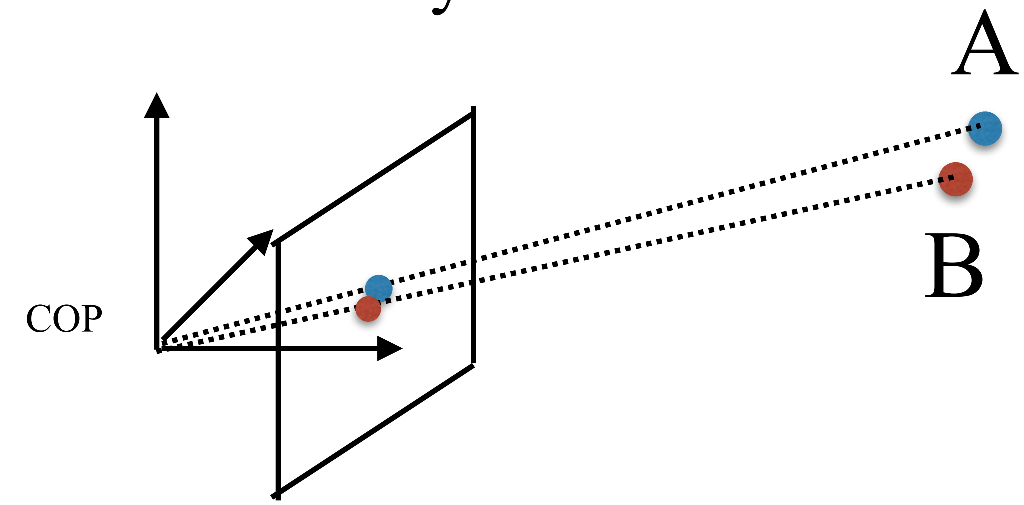


if Z >> ΔZ, what happens to their image projections (e.g., a_x and b_x)?

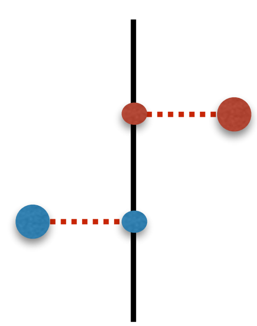
Scaled orthographic or “weak perspective” projection

Consider two points (A,B) at different depths that are far away from camera:

$$\begin{bmatrix} A_x \\ A_y \\ Z \end{bmatrix} \qquad \begin{bmatrix} B_x \\ B_y \\ Z + \Delta Z \end{bmatrix}$$



if Z >> ΔZ, what happens to their image projections (e.g., a_x and b_x)?



$$a_x = \frac{f A_x}{Z} = \alpha A_x$$

$$b_x = \frac{f B_x}{Z + \Delta Z}$$

$$\approx \frac{f B_x}{Z} = \alpha B_x \quad \text{for} \quad \Delta Z \ll Z$$

We can approximate sets of such points with a scaled orthographic model
(that projects all points to plane of constant depth)

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

Locally scaled-orthographic

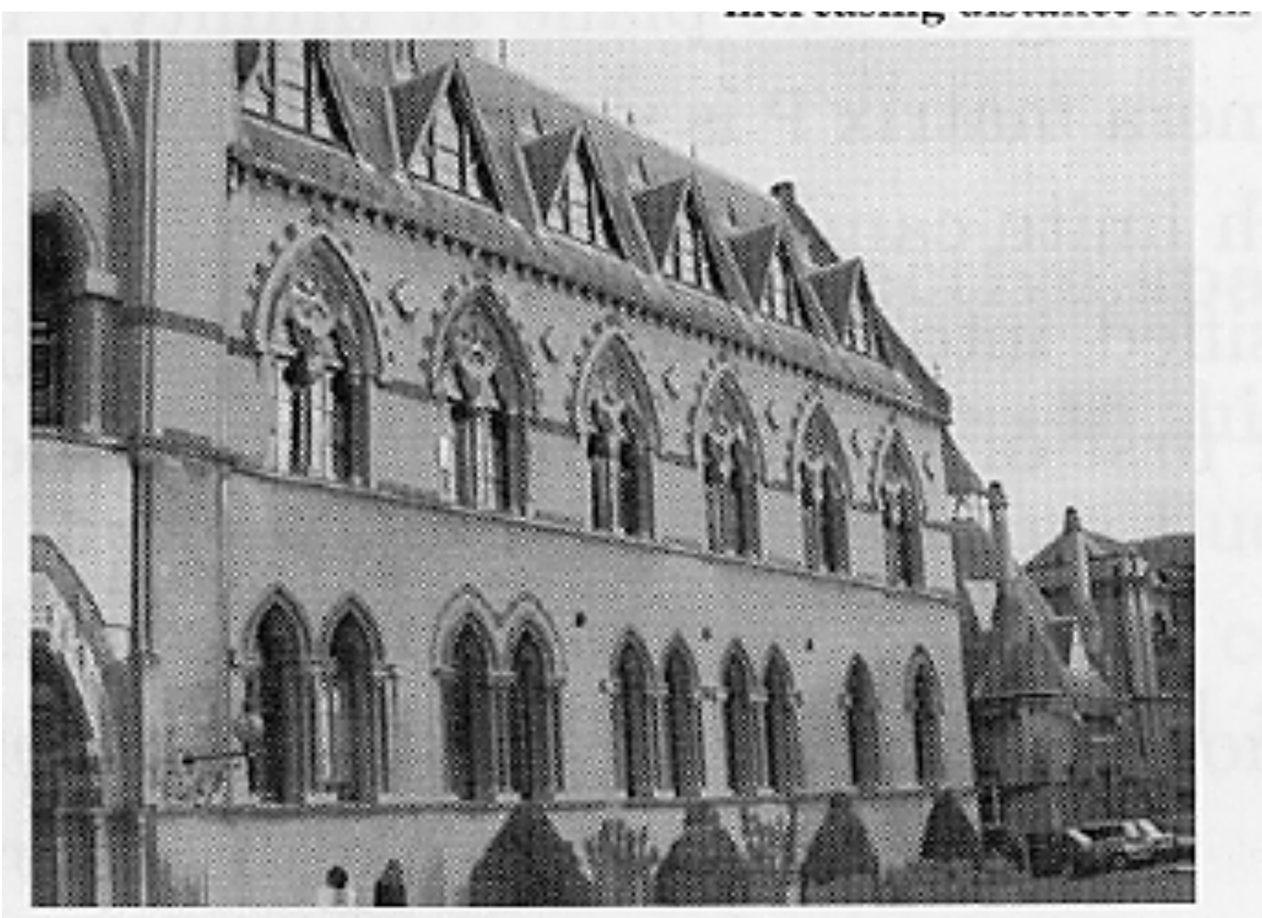
Compute different scale factors for different objects (α_{person} and α_{tower})

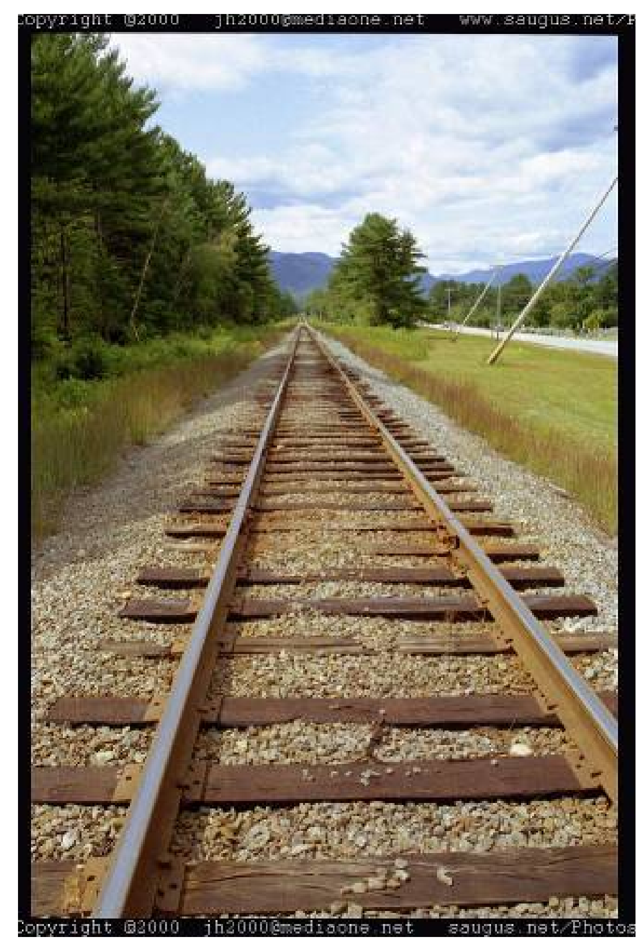
Locally scaled-orthographic

This is why searching over object scale (with an image pyramid) tends to be quite effective

Where is scaled-orthographic a poor model?

(when changes in depth of object ΔZ are large relative to distance from camera Z)





Intermeddiate model between perspective and scaled orthographic: *paraperspective*

Motivation: objects at image border are viewed from different angle

… but scaled orthographic doesn't capture this!

CVPR17

3D Bounding Box Estimation Using Deep Learning and Geometry

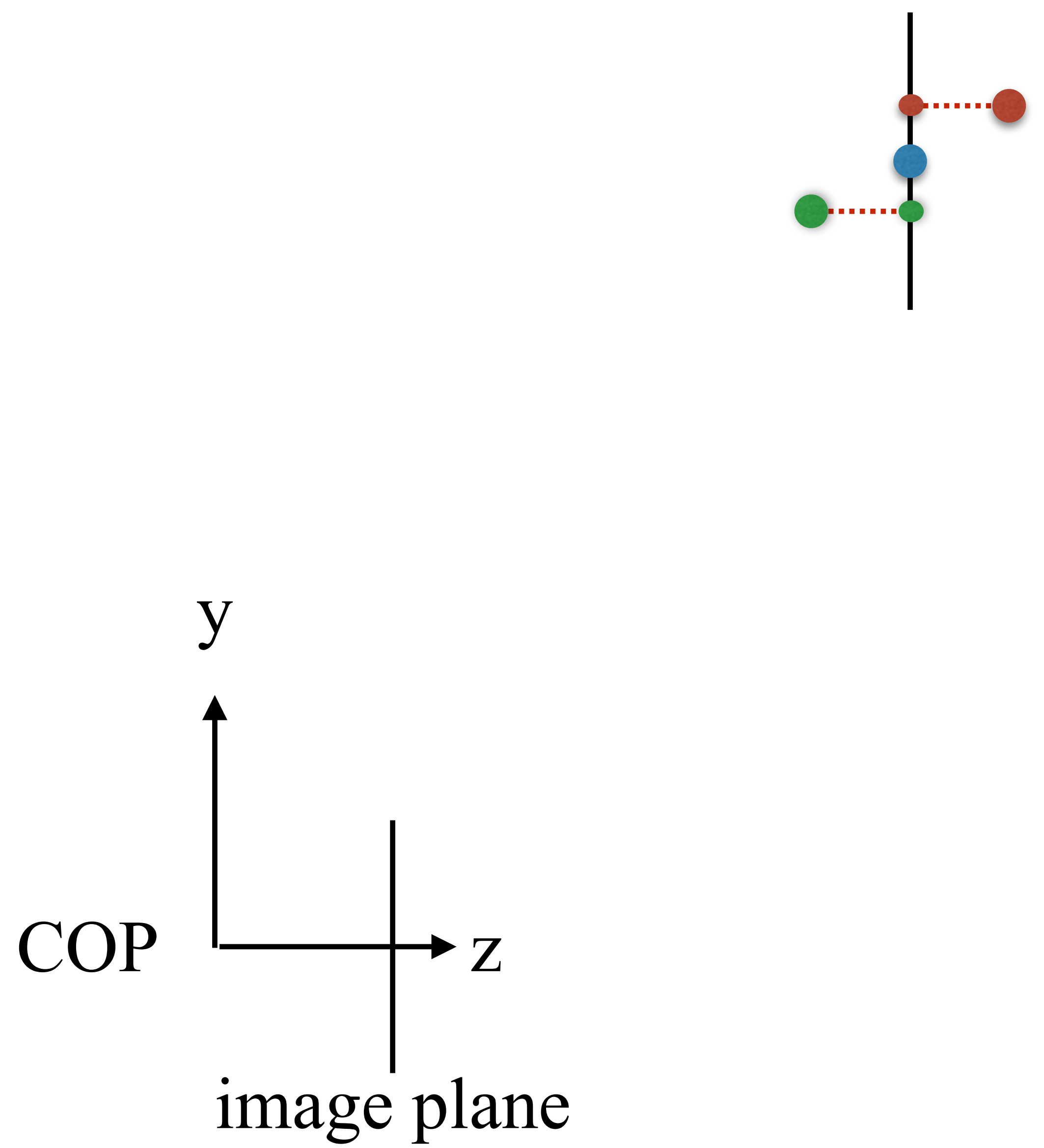
Arsalan Mousavian*
George Mason University
amousavi@gmu.edu

Dragomir Anguelov
Zoox, Inc.
drago@zoox.com

John Flynn
Zoox, Inc.
john.flynn@zoox.com

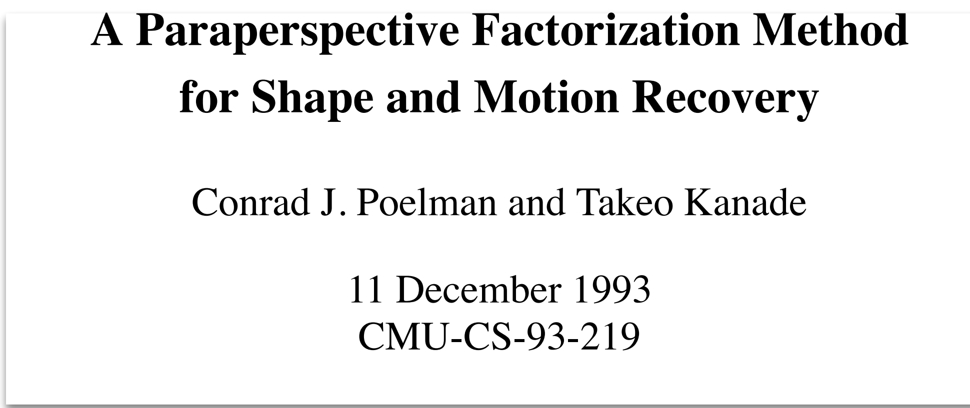
Jana Košecká
George Mason University
kosecka@gmu.edu

Intermeddiate model between perspective and scaled orthographic: *paraperspective*



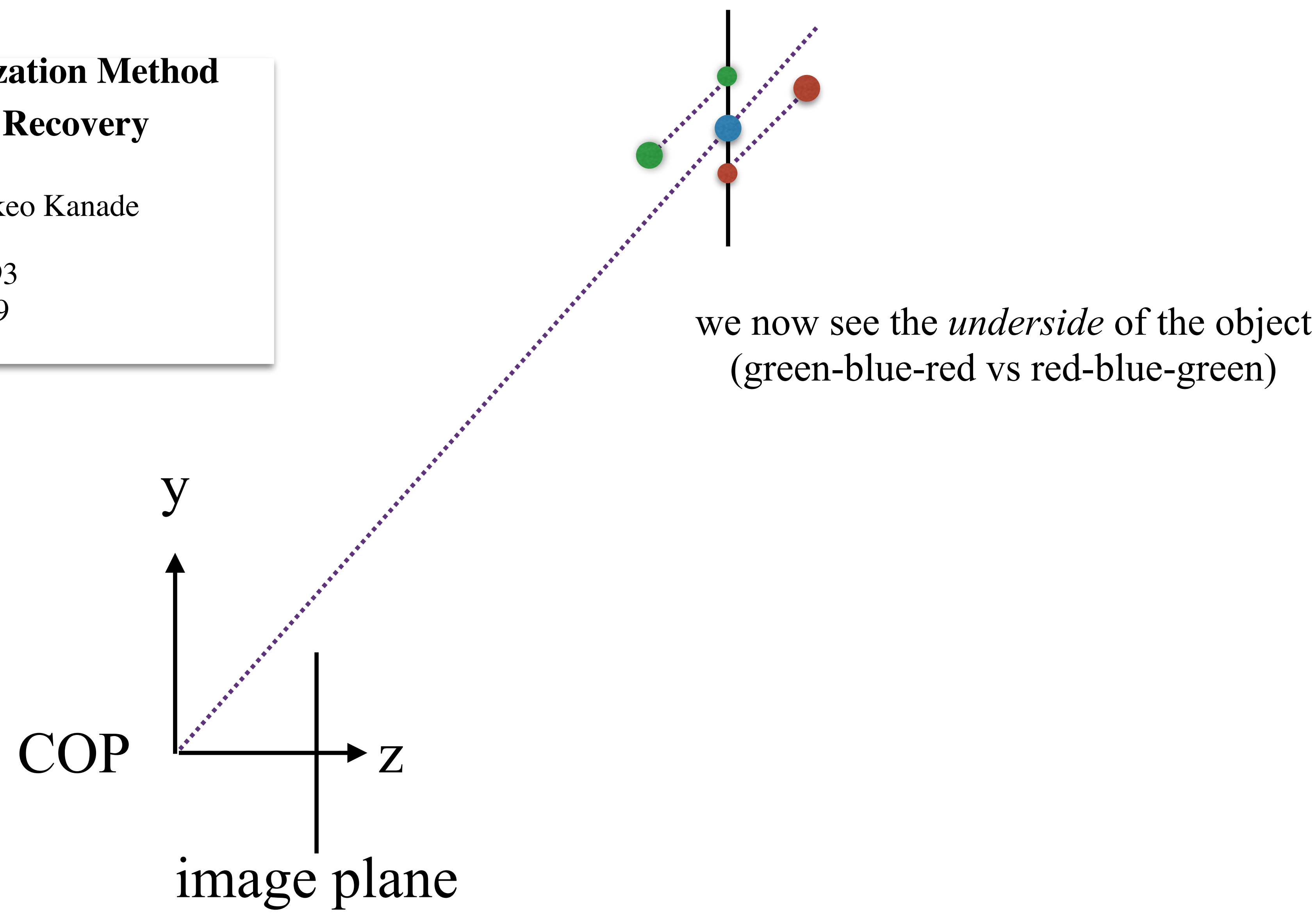
Scaled orthographic: project all points to plane aligned at average object depth

Intermeddiate model between perspective and scaled orthographic: *paraperspective*


A Paraperspective Factorization Method for Shape and Motion Recovery

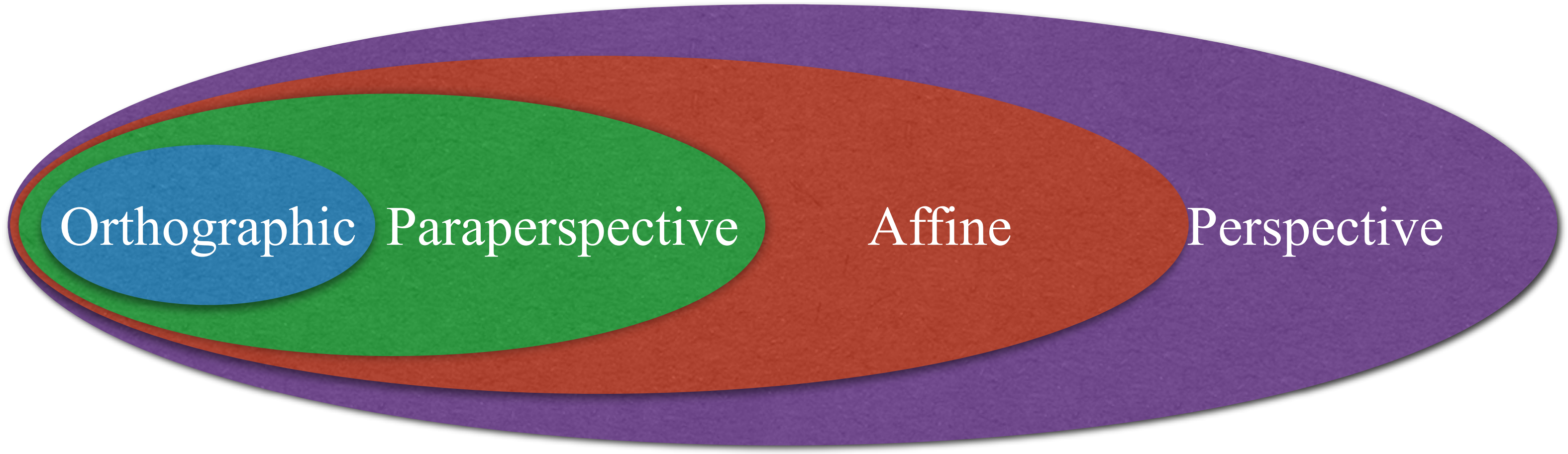
Conrad J. Poelman and Takeo Kanade

11 December 1993
CMU-CS-93-219



Paraperspective: project all points to plane aligned at average object depth, *along direction of ray from COP to object centroid*

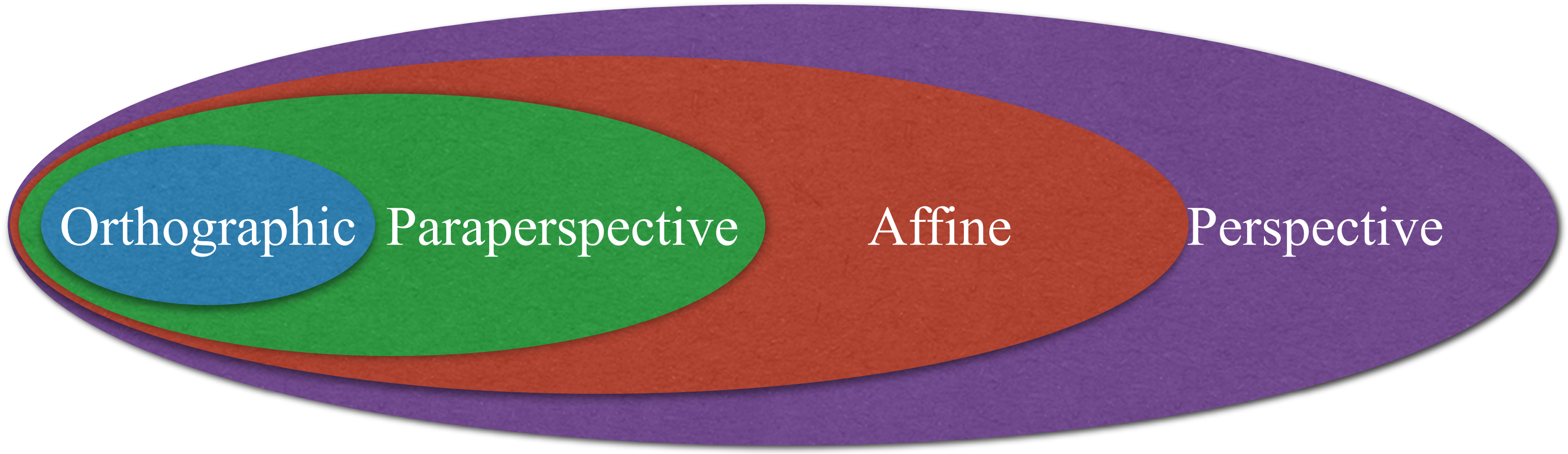
Thought experiment



- (Scaled) Orthographic and Paraperspective (eqns are ommitted cause they are hairy) are attractive since we don't need to " divide by Z"
- What's the most general such "linear" camera?

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

Thought experiment



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

- (Scaled) Orthographic and Paraperspective (eqns are ommitted cause they are hairy) are attractive since we don’t need to “ divide by Z”
- What’s the most general such “linear” camera?

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

Affine cameras

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ & & & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \mathbf{m_3}^T = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{A}\mathbf{X} + \mathbf{b}$$

- 2D points = linear projection of 3D points (+ 2D translation)
- Projection defined by 8 parameters
- (b1,b2) is the image of the world origin (0,0,0)
- parallel 3D lines project to parallel 2D lines

Can affine cameras model paraperspective projection?

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ & & & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\mathbf{m_3}^T = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{AX} + \mathbf{b}$$

Can affine cameras model paraperspective projection?

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ & & & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\mathbf{m_3}^T = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{AX} + \mathbf{b}$$

Yes!

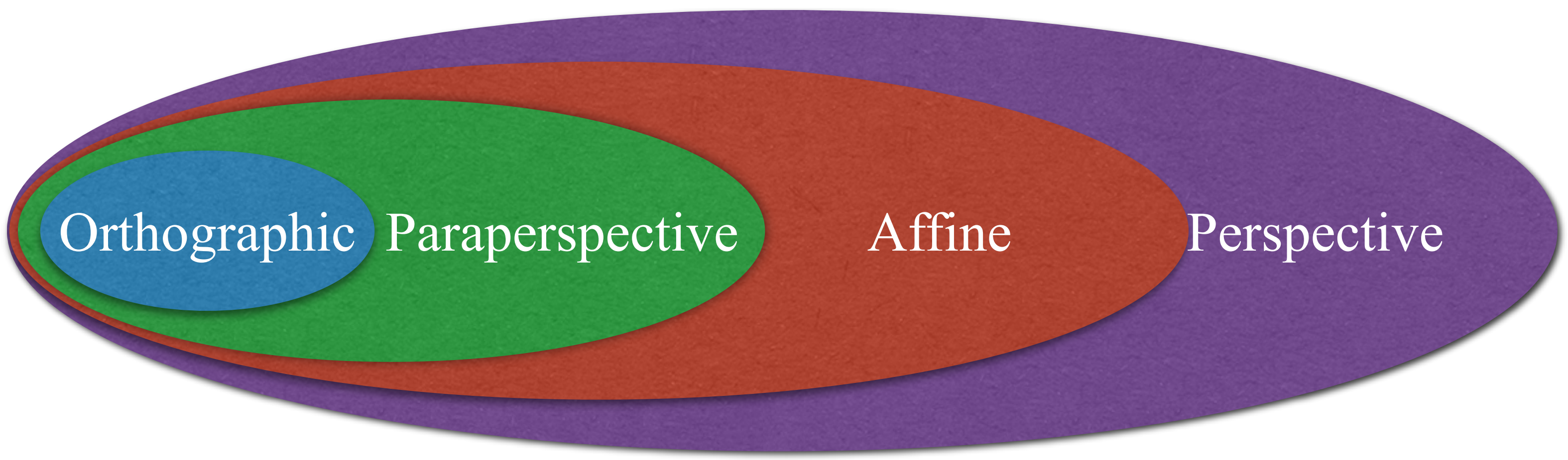
Consider a (3D rigid-body transformation + stretch) + orthographic projection + 2D affine transformation

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

(empty slot denotes 0,
while dot denotes typically nonzero value)

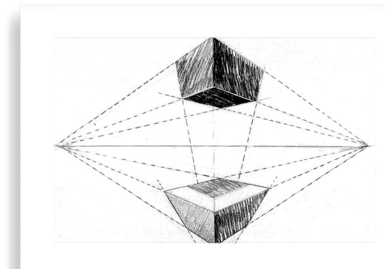
Turns out, affine cameras are even more general in that that can model 3D affine transformations

Simplified camera models



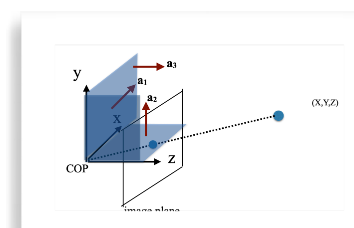
Agenda

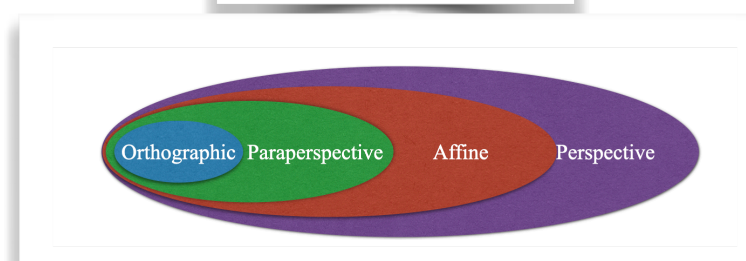
- Pinhole optics
  - Perspective projection (vanishing points, horizon, object height)
  - Camera matrices (intrinsics + extrinsics)
  - Homographies (2 views of plane, rotation)



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

- Camera models
  - Properties of camera matrices (DOF, geometric intuition, pixel2rays)
  - Simplified cameras: orthographic, scaled orthographic, paraperspective, affine
  - **Camera calibration (DLT v reprojection error)**

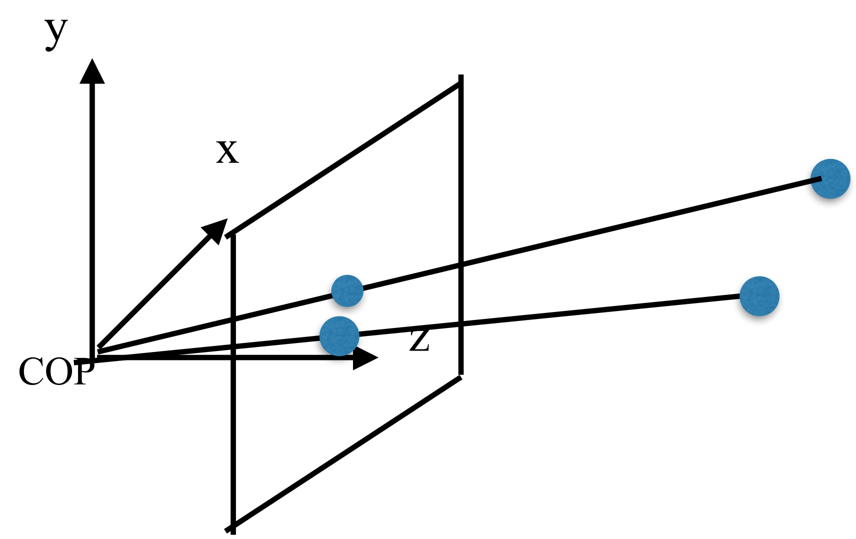




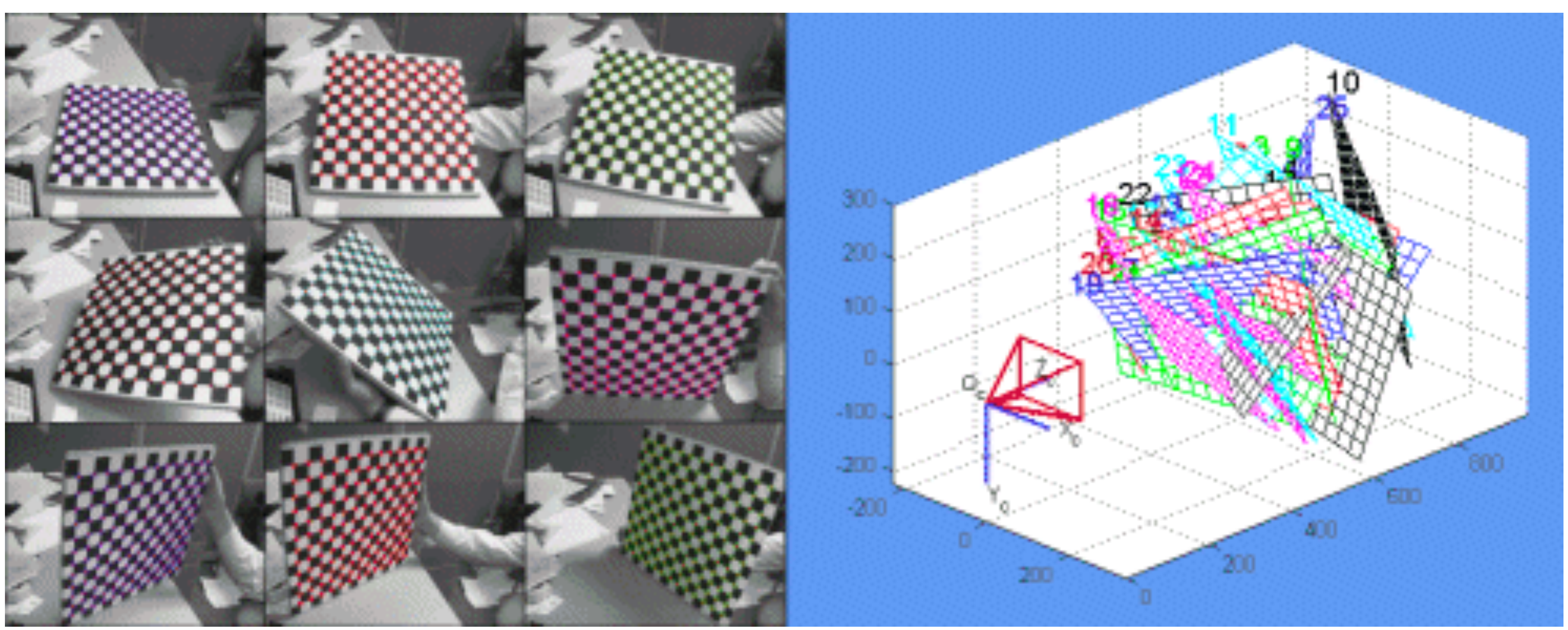
PnP = Perspective n-Point

Calibration: Recover camera matrix M_{3X4} from 3D scene points $\mathbf{X_1,..,X_N}$ and the corresponding 2D projections in the image plane $\mathbf{x_1,..,x_N}$

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$



$$\begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$



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

Write relation between image point (x,y), camera projection matrix M, and 3D point (X,Y,Z) :

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

Write non-linear relations between coordinates:

$$x_i =$$

$$y_i =$$

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

Write relation between image point (*x,y*), camera projection matrix M, and 3D point (*X,Y,Z*) :

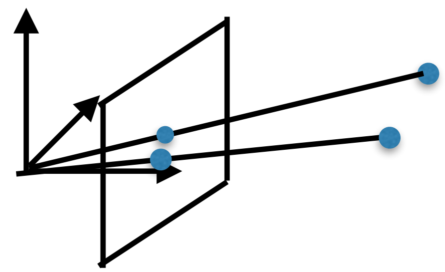
$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Write non-linear relations between coordinates:

$$x_i = \frac{m_{11}X_i + m_{12}Y_i + m_{13}Z_i + m_{14}}{m_{31}X_i + m_{32}Y_i + m_{33}Z_i + m_{34}}$$

$$y_i = \frac{m_{21}X_i + m_{22}Y_i + m_{23}Z_i + m_{14}}{m_{31}X_i + m_{32}Y_i + m_{33}Z_i + m_{34}}$$

Estimating a camera matrix

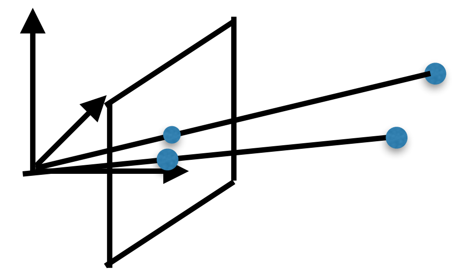


Given knowns $\mathbf{X}_i$ and $\mathbf{x}_i$, write constraints as linear in *unknowns* in M

$$(\qquad x_i = m_{11}$$

$$\vdots$$

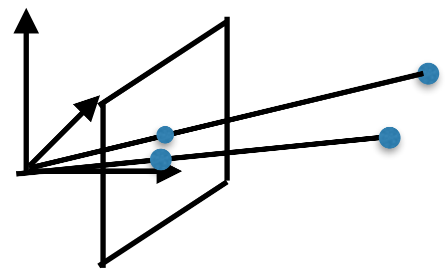
Estimating a camera matrix



Given knowns $\mathbf{X}_i$ and $\mathbf{x}_i$, write constraints as linear in *unknowns* in M

$$(m_{31}X_i + m_{32}Y_i + m_{33}Y_i + m_{34})x_i = m_{11}X_i + m_{12}Y_i + m_{13}Z_i + m_{14}$$
$$\vdots$$

Estimating a camera matrix



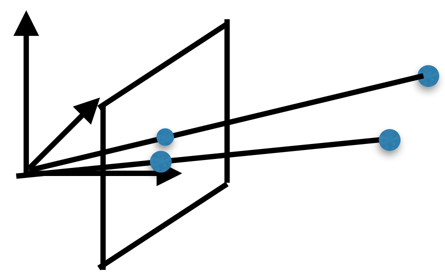
Given knowns $\mathbf{X}_i$ and $\mathbf{x}_i$, write constraints as linear in *unknowns* in M

$$(m_{31}X_i + m_{32}Y_i + m_{33}Y_i + m_{34})x_i = m_{11}X_i + m_{12}Y_i + m_{13}Z_i + m_{14}$$
$$\vdots$$

$$LM(:) = \begin{bmatrix} 0 \\ 0 \\ \ldots \end{bmatrix}$$ Homogenous linear system

What's the size of L?
How many degrees of freedom in M?
How many corresponding points needed?
In noise-free case, L will have a non-zero null-space

Estimating a camera matrix



Given knowns $\mathbf{X}_i$ and $\mathbf{x}_i$, write constraints as linear in *unknowns* in M

$$(m_{31}X_i + m_{32}Y_i + m_{33}Y_i + m_{34})x_i = m_{11}X_i + m_{12}Y_i + m_{13}Z_i + m_{14}$$
$$\vdots$$

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

Homogenous linear system

What's the size of L? 2N by 12

How many degrees of freedom in M? 11 (6 with known intrinics)

How many corresponding points needed? 6 (3 with known intrinics)

In noise-free case, L will have a non-zero null-space

What about noisy case?

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

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

Min right singular vector of L (or eigenvector of $L^T L$)

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

What about noisy case?

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

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

Min right singular vector of L (or eigenvector of $L^T L$)

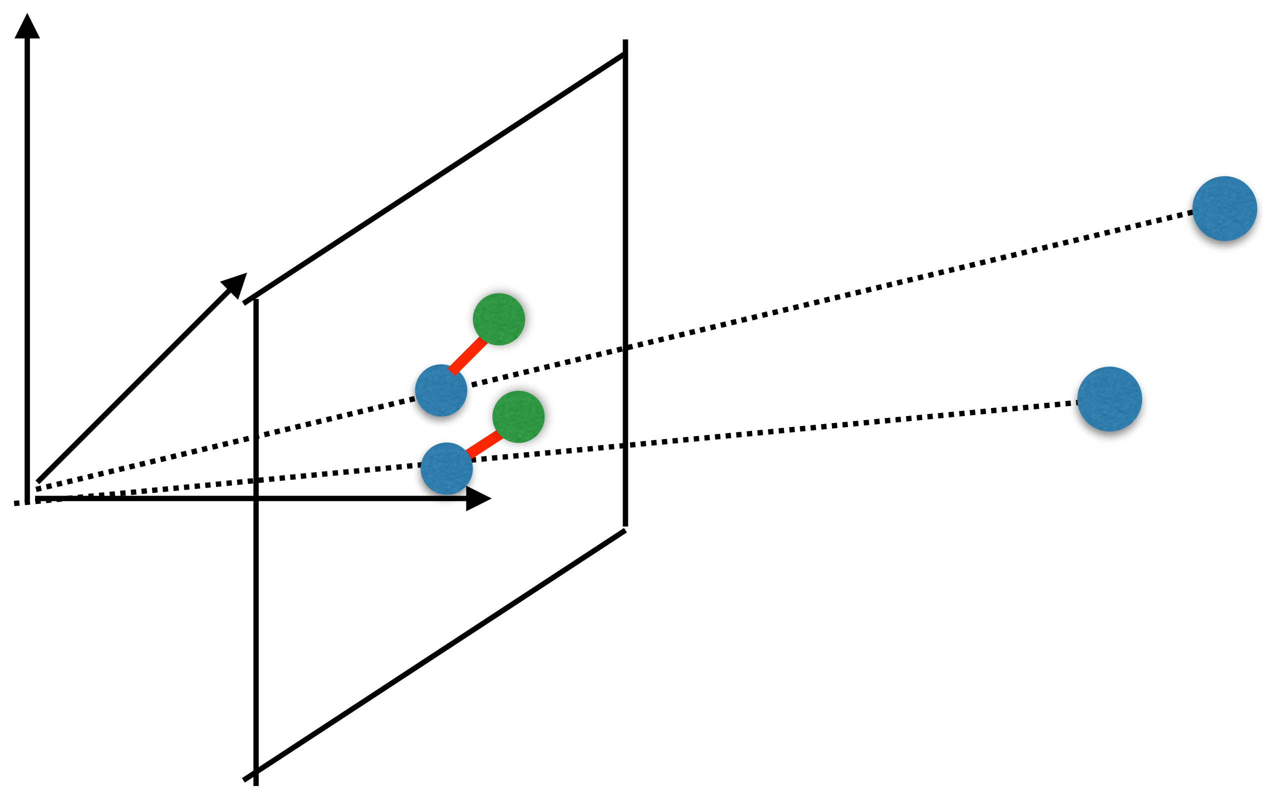
Aside: this technique of linearizing a nonlinear equation resulting from perspective projection is known as a *direct linear transform*

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

Is this the right error to minimize?

If not, what is?

Ideal error: image reprojection error



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

Initialize nonlinear optimization with “algebraic” solution

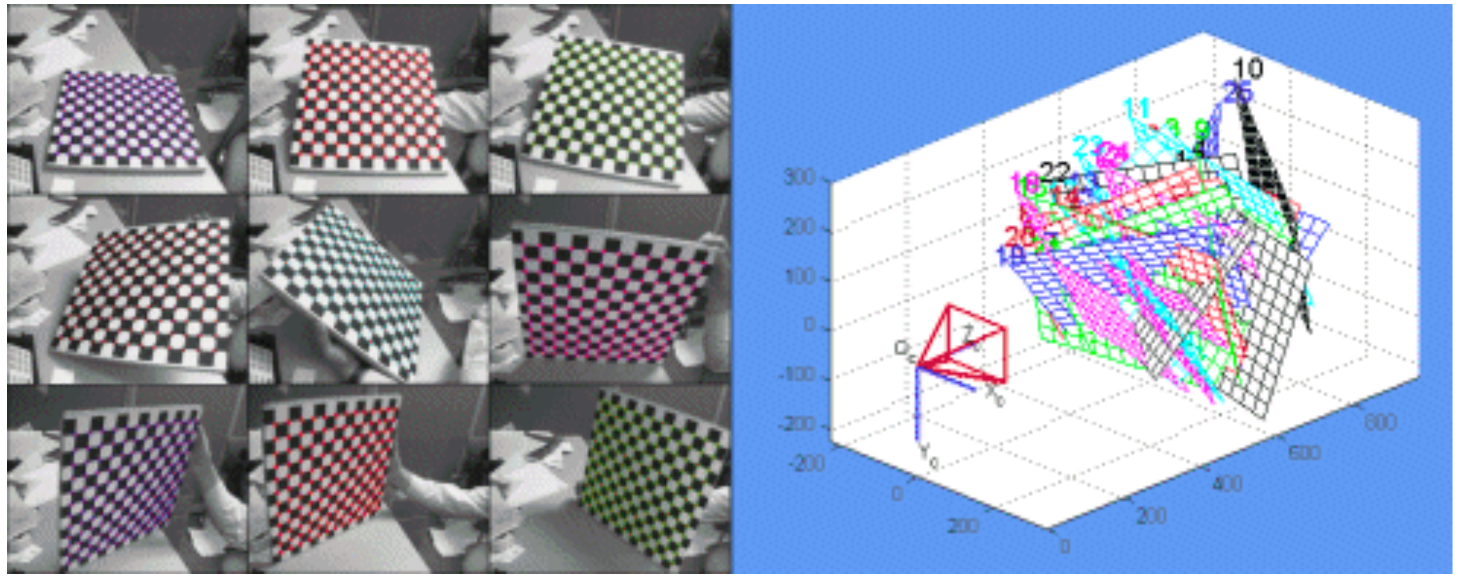
(Observation also holds when applying DLT for computing homographies)

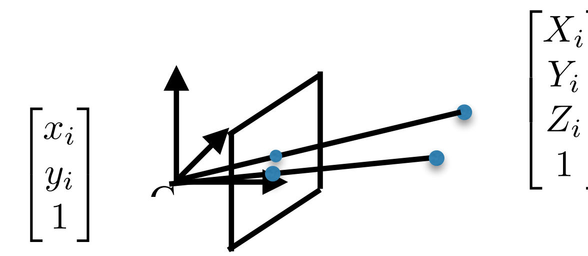
Overall approach

Minimize reprojection error: Error(M,k's)

Initialize with algebraic solution
(various approaches in literature based on various assumptions)

Calibrating the intrinsics K

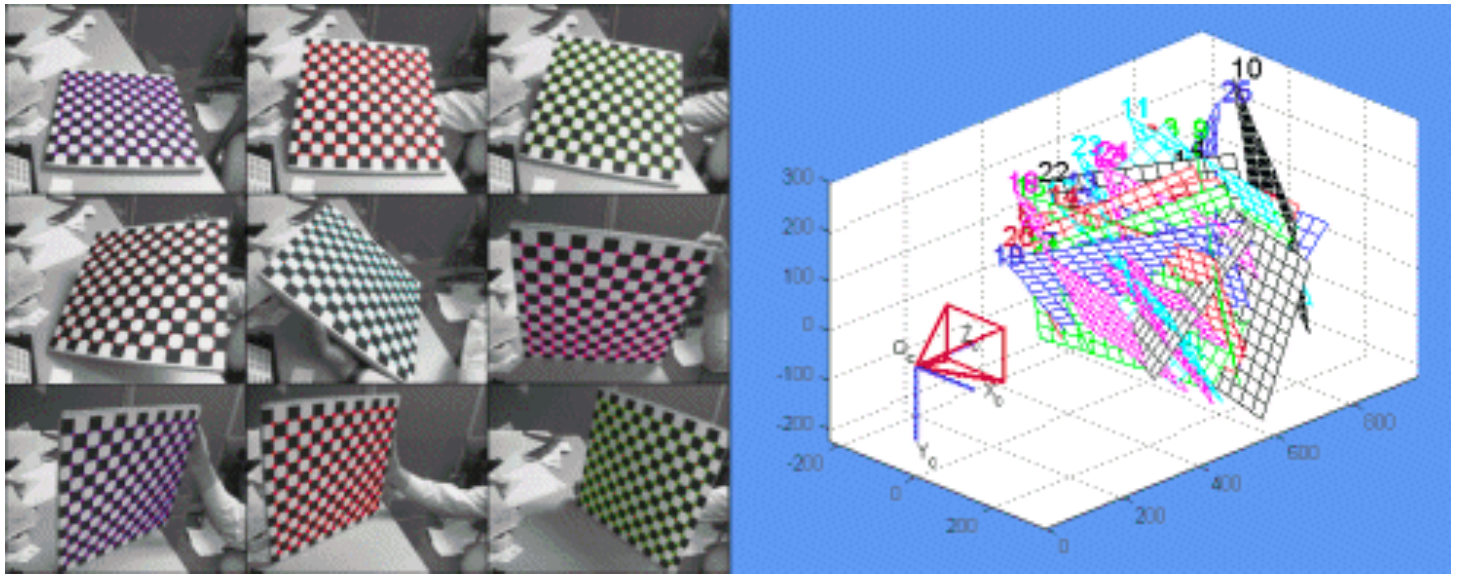


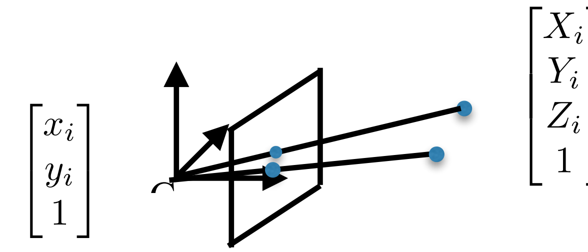


1. Simple math, complex procedure

Wave calibration object with known pose (e.g., via a motion-controlled platform) infront of camera. Solve for intrinsics K that minimize reprojection error

Calibrating the intrinsics K





1. Simple math, complex procedure

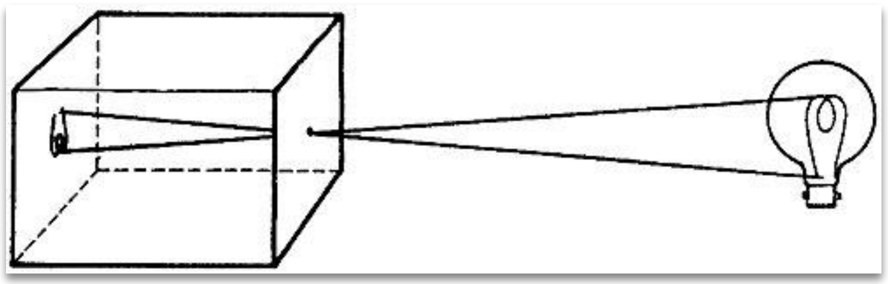
Wave calibration object with known pose (e.g., via a motion-controlled platform) infront of camera. Solve for intrinsics K that minimize reprojection error

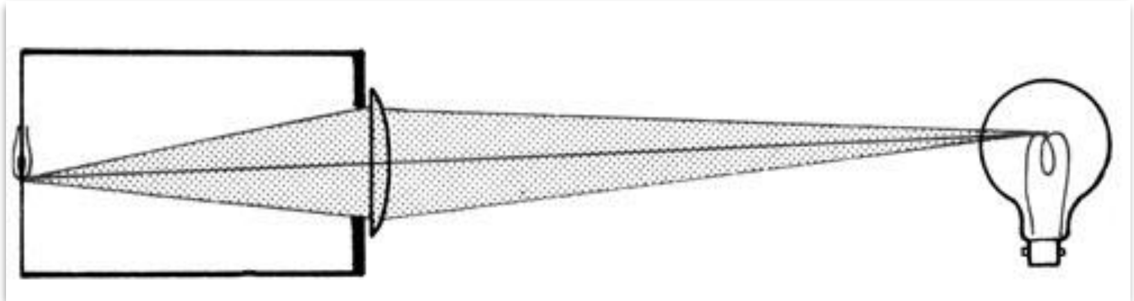
2. Complex math, simpler procedure

Wave calibration object with unknown pose. Model the projection of each frame with frame-specific R,t and fixed K. Solve for unknowns$\{K, R_1, \mathbf{T}_1, R_2, \mathbf{T}_2, \ldots\}$ that minimize reprojection error

$$\begin{bmatrix} x_{ij} \\ y_{ij} \\ 1 \end{bmatrix} \equiv K \begin{bmatrix} R_j & T_j \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

Last component of camera calibration: Lens Distortions





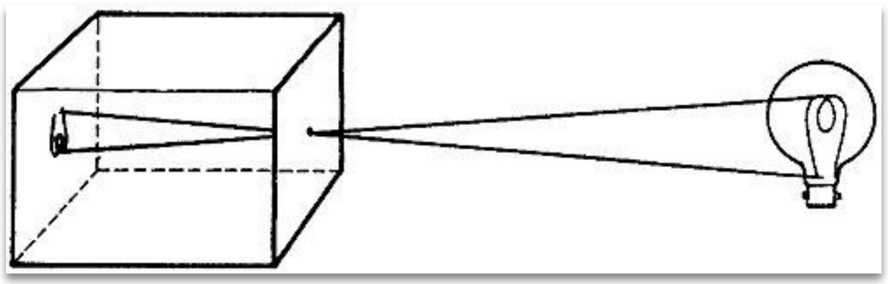
make use of lenses to grab more photons (better signal-to-noise for film sensor)

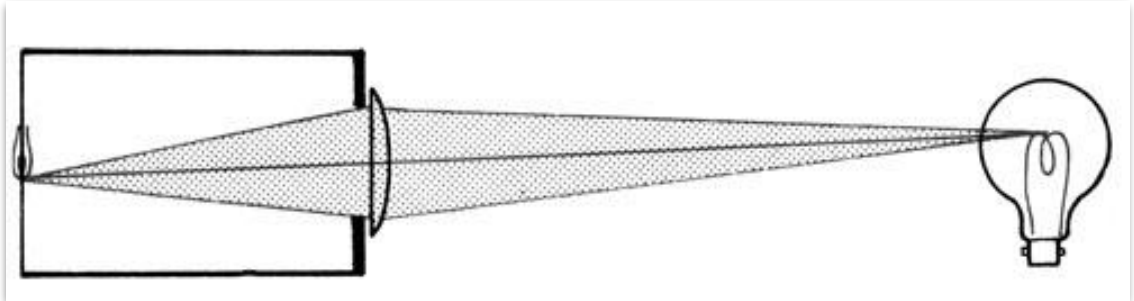
Cons: objects outside particular depth will be blurred (limited “depth-of-field”) and radial distortion from lens

Lots of cool math with “thin lens model”; changing the focal length changes the depth at which the world will be in focus!

See class on computational photography

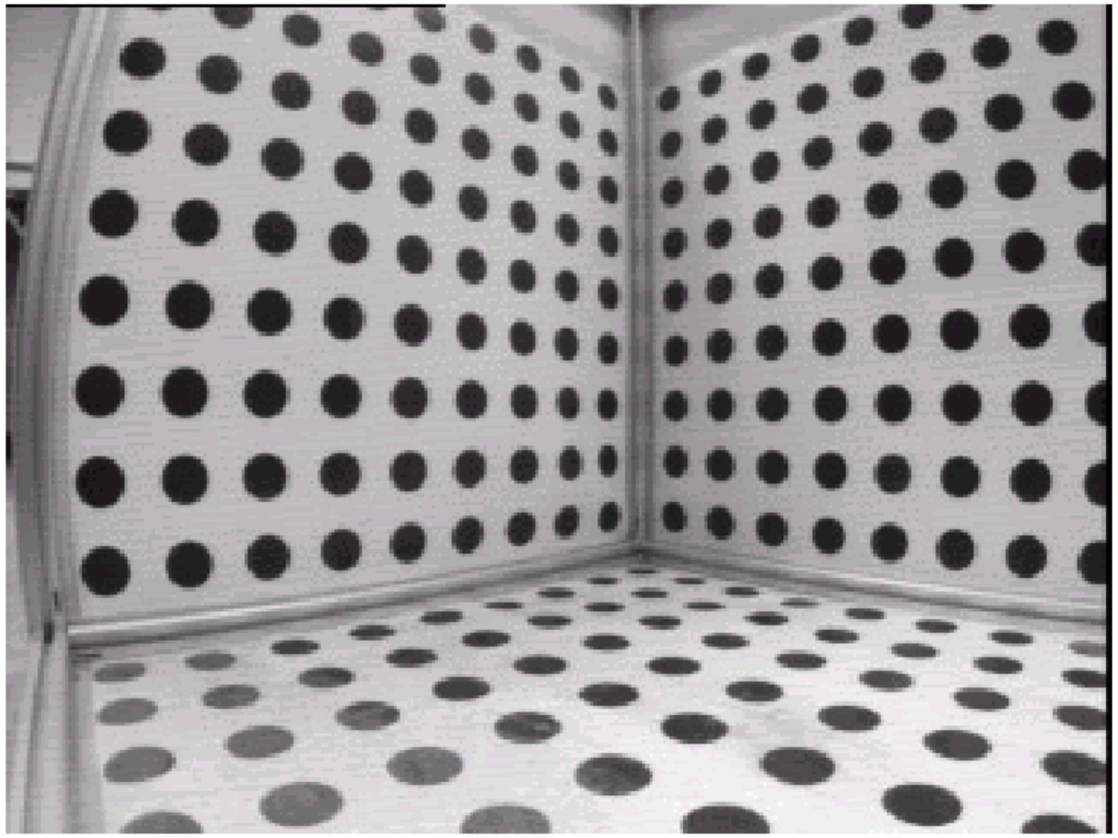
Last component of camera calibration: Lens Distortions



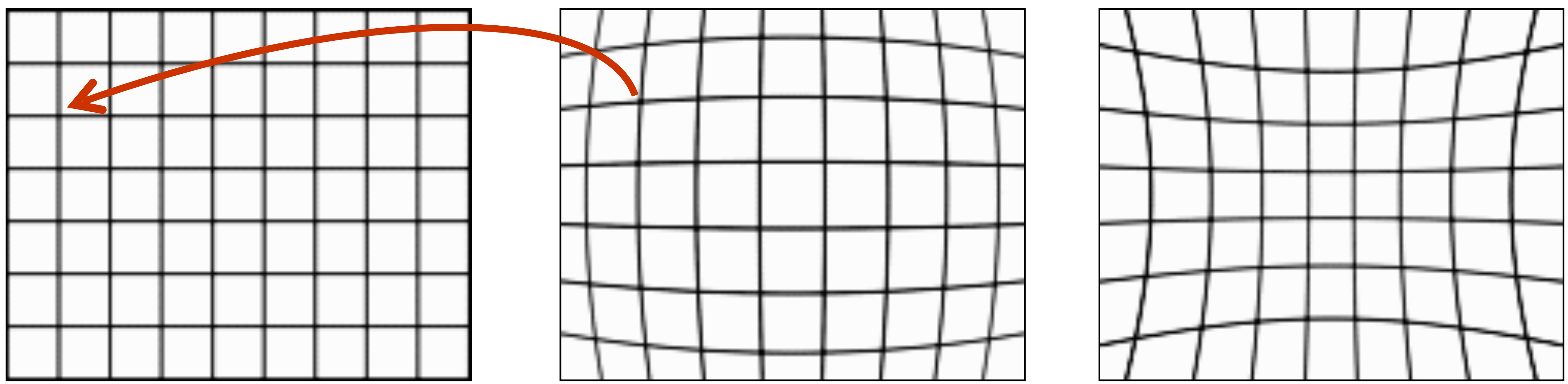


make use of lenses to grab more photons (better signal-to-noise for film sensor)

Challenge: radial distortion

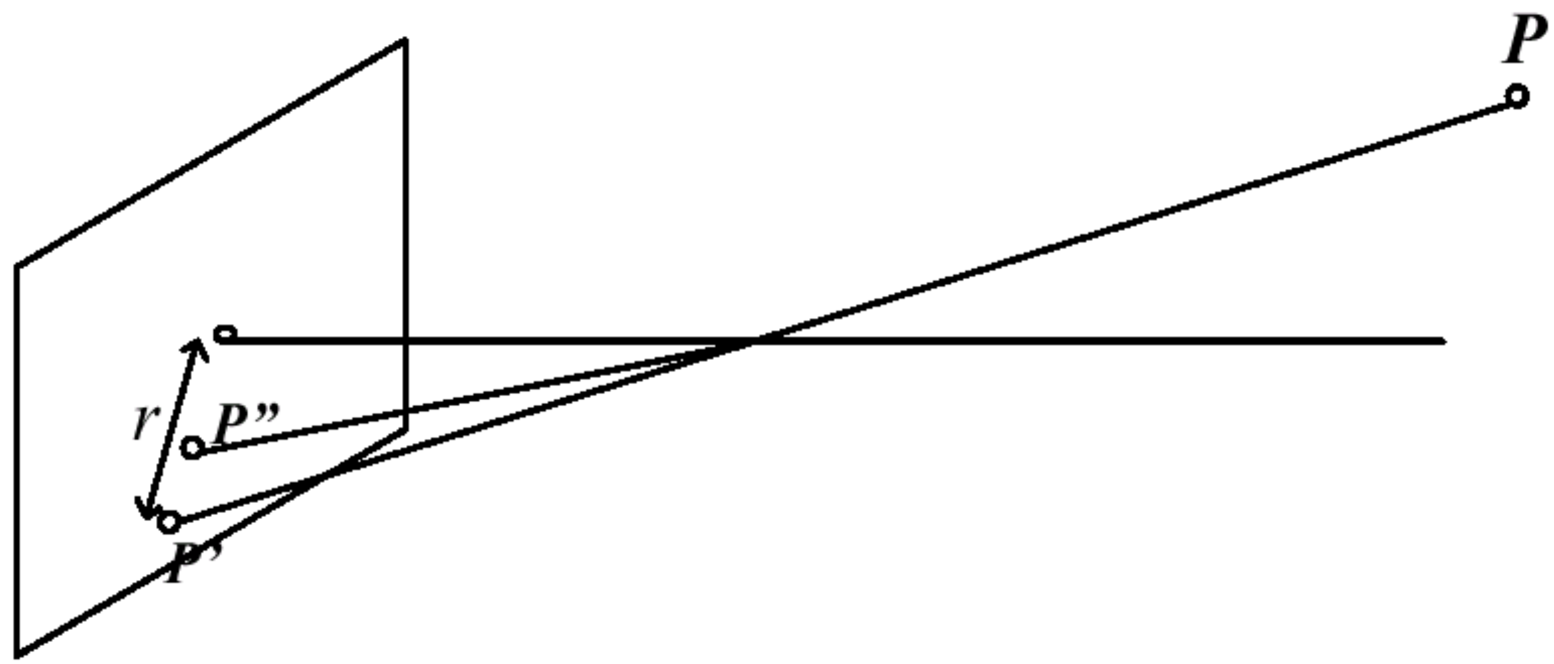


Radial Lens Distortions



No Distortion | Barrel Distortion | Pincushion Distortion

Radial Distortion Model



Ideal:

$$x' = f\frac{x}{z}$$

$$y' = f\frac{y}{z}$$

Distorted:

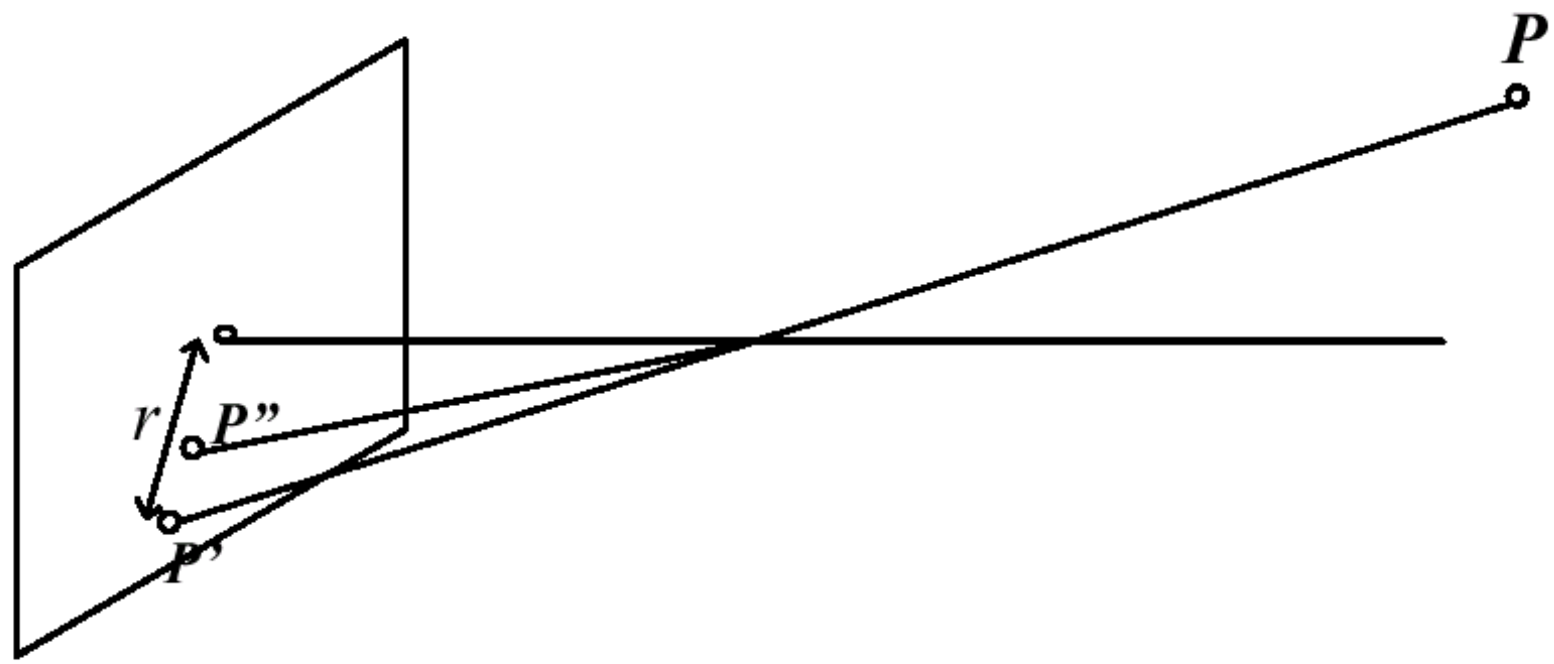
$$x'' = \frac{1}{\lambda}x'$$

$$y'' = \frac{1}{\lambda}y'$$

$$\lambda = 1 + k_1 r^2 + k_2 r^4 + \cdots$$

Given correspondences on warped and ideal (unwarped) grid, compute $(k_1, k_2, ..)$ that minimize..

Radial Distortion Model



Ideal:

$$x' = f\frac{x}{z}$$

$$y' = f\frac{y}{z}$$

Distorted:

$$x'' = \frac{1}{\lambda}x'$$

$$y'' = \frac{1}{\lambda}y'$$

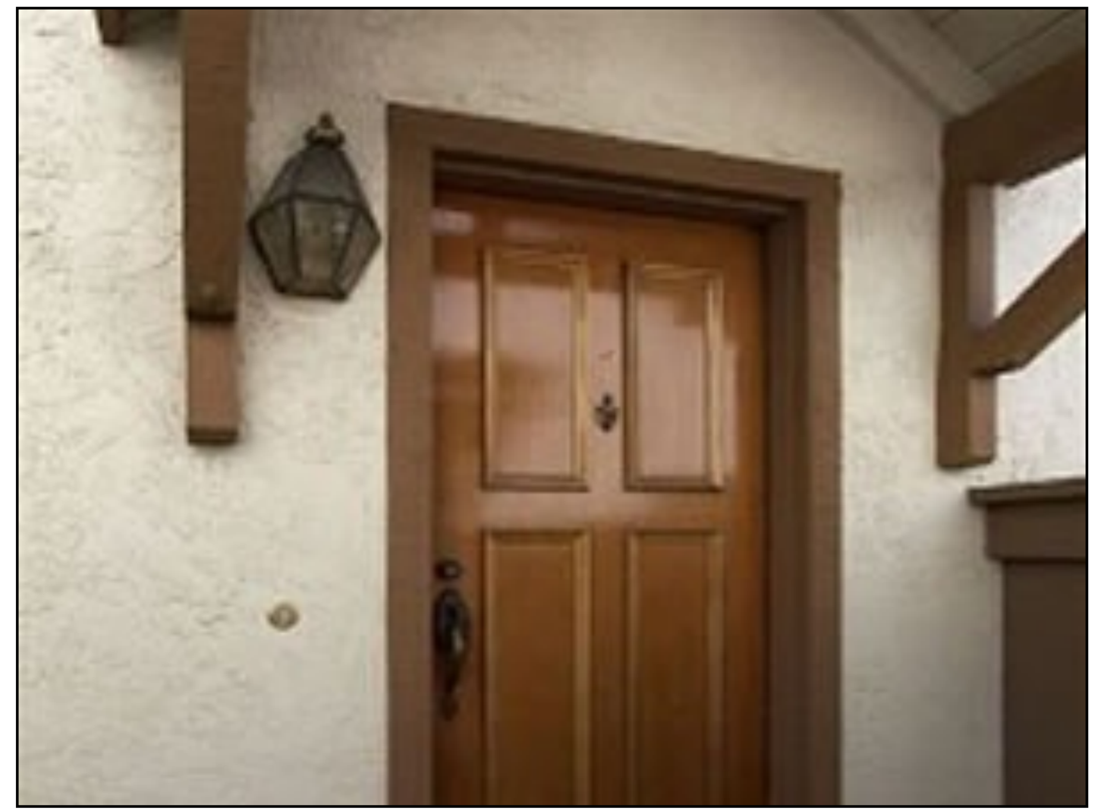
$$\lambda = 1 + k_1 r^2 + k_2 r^4 + \cdots$$

Given correspondences on warped and ideal (unwarped) grid, compute (k_1,k_2,..) that minimize..

$$\sum_i (x_i'' - x_i)^2 + (y_i'' - y_i)^2$$

Correcting Radial Lens Distortions

Before

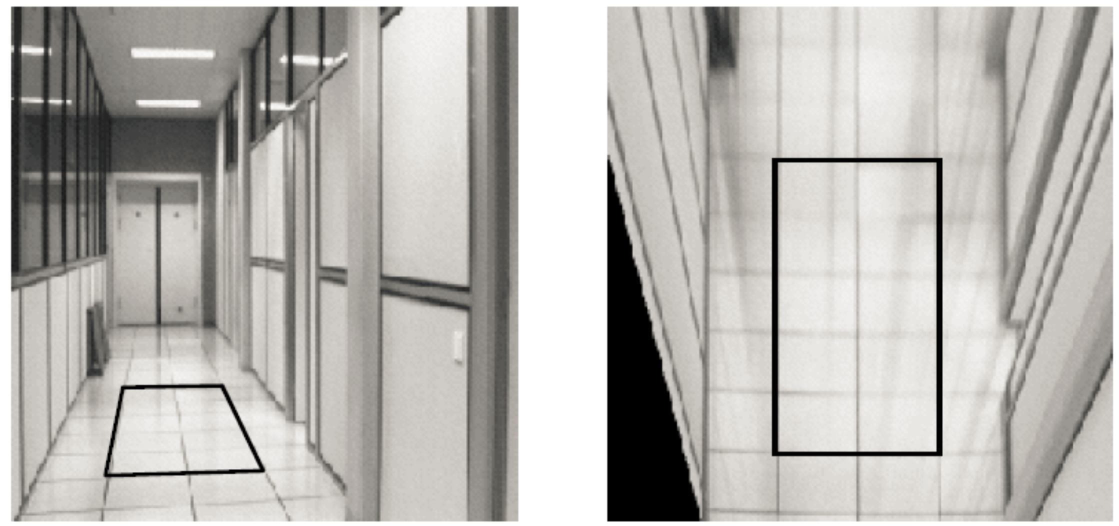


After

http://www.grasshopperonline.com/barrel_distortion_correction_software.html

(Another) application of DLT: Estimating homographies

Given (x_1,y_1) and H, how do we compute (x_2,y_2)?

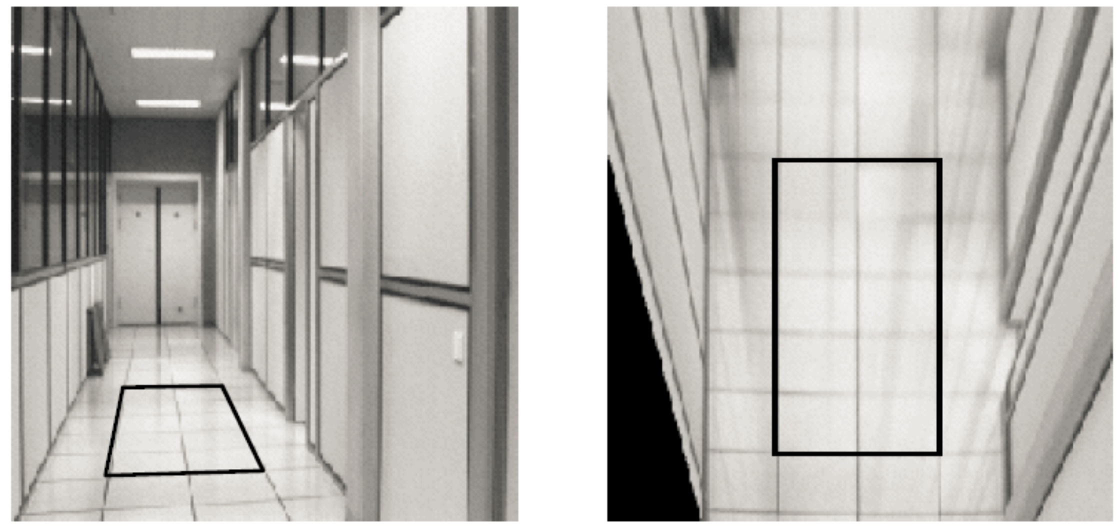


from Hartley & Zisserman

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

(Another) application of DLT: Estimating homographies

Given (x_1,y_1) and H, how do we compute (x_2,y_2)?



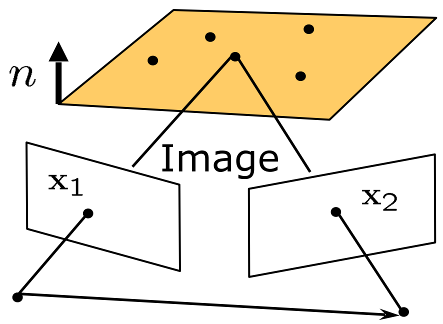
from Hartley & Zisserman

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

$$x_2 = \frac{\lambda x_2}{\lambda} = \frac{a x_1 + b y_1 + c}{g x_1 + h y_1 + i}$$

Estimating Homographies

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

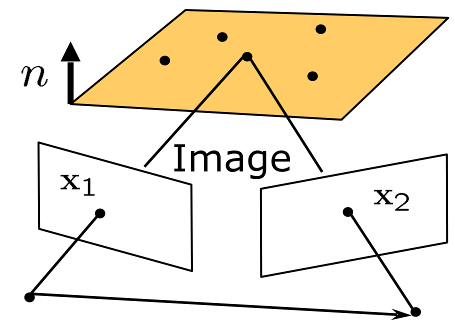


$$x_2(gx_1 + hy_1 + i) = ax_1 + by_1 + c$$

$$\vdots$$

Estimating Homographies

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



$$x_2(gx_1 + hy_1 + i) = ax_1 + by_1 + c$$

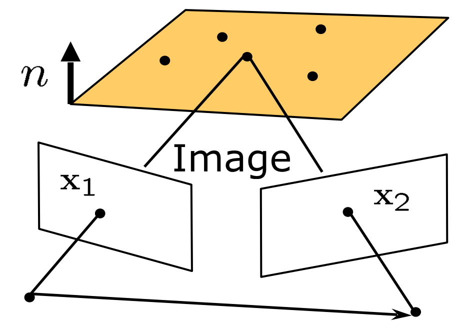
$$\vdots$$

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

How many degrees of freedom in H?
How many corresponding points needed?

Estimating Homographies

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



$$x_2(gx_1 + hy_1 + i) = ax_1 + by_1 + c$$

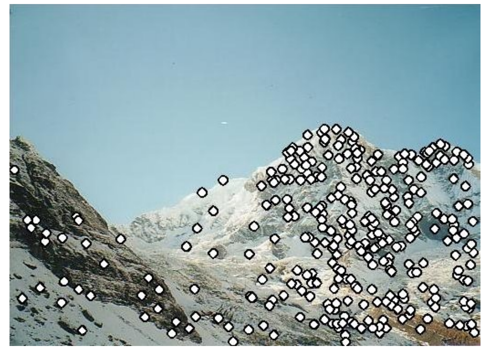
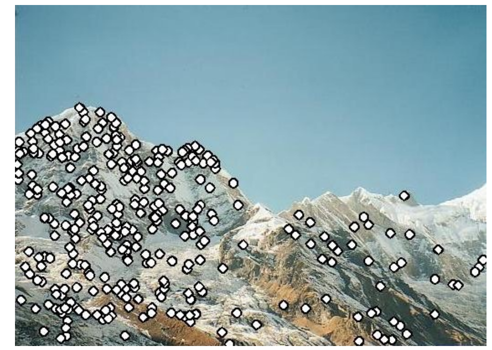
$$\vdots$$

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

How many degrees of freedom in H? 8

How many corresponding points needed? 4

RANSAC for homography fitting

Given 2 images with interest points and candidate matches

Repeat ***k*** times:

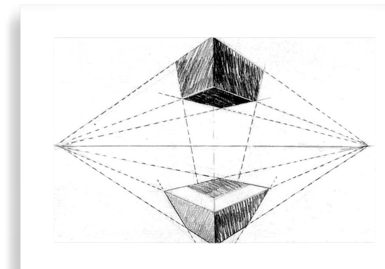
- Draw ***n*** points uniformly at random (from left image)
- Fit (homography) warp to these ***n*** points and their correspondences
- Find inliers among the remaining left-image points (i.e., whose warped positions land close to right-image correspondence)
- Return warp with largest inlier set

(Optionally return line with optimal least-squares inlier fit)

Source: M. Pollefeys

Agenda

- Pinhole optics
  - Perspective projection (vanishing points, horizon, object height)
  - Camera matrices (intrinsics + extrinsics)
  - Homographies (2 views of plane, rotation)



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

- Camera models
  - Properties of camera matrices (DOF, geometric intuition, pixel2rays)
  - Simplified cameras: orthographic, scaled orthographic, paraperspective, affine
  - Camera calibration (DLT v reprojection error)