CS 4495 Computer Vision
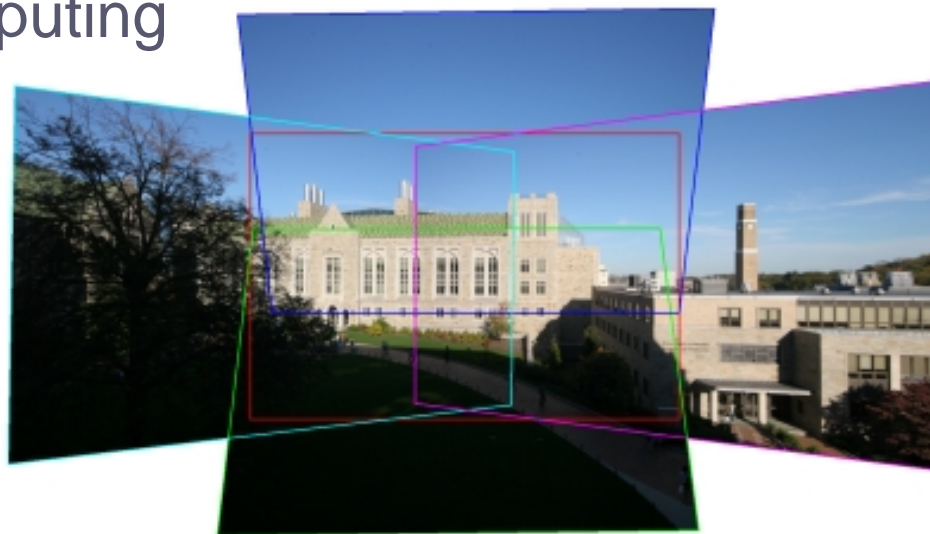
N-Views (1) – Homographies and Projection

Aaron Bobick

School of Interactive Computing

Administrivia

- PS 2:
  - Get SDD and Normalized Correlation working for a given windows size – say 5x5. Then try on a few window sizes.
  - If too slow, resize the images (imresize) and get your code working. Then try on full images (which are reduced already!).
  - Results not perfect on even test images? Should they be?
  - Yes you can use normxcorr2 (it did say this!)
  - Some loops are OK. For SSD you might have 3 nested loops (row, col, disp) but shouldn't be looping over pixels.

- Now: Multiple-views
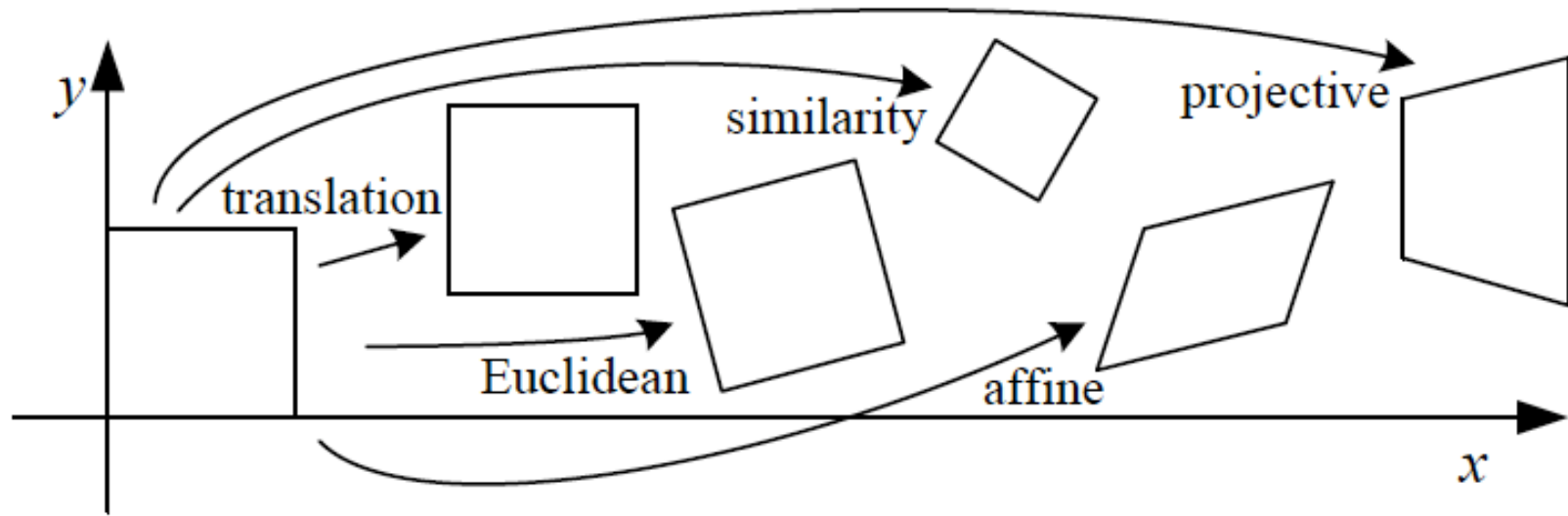 - FP 7.1, 8 (all)

- Today: First half of 2-Views …

Two views…and two lectures

- Projective transforms from image to image

- Some more projective geometry
  - Points and lines and planes

- Two arbitrary views of the same scene
  - Calibrated – "Essential Matrix"
  - Two uncalibrated cameras "Fundamental Matrix"
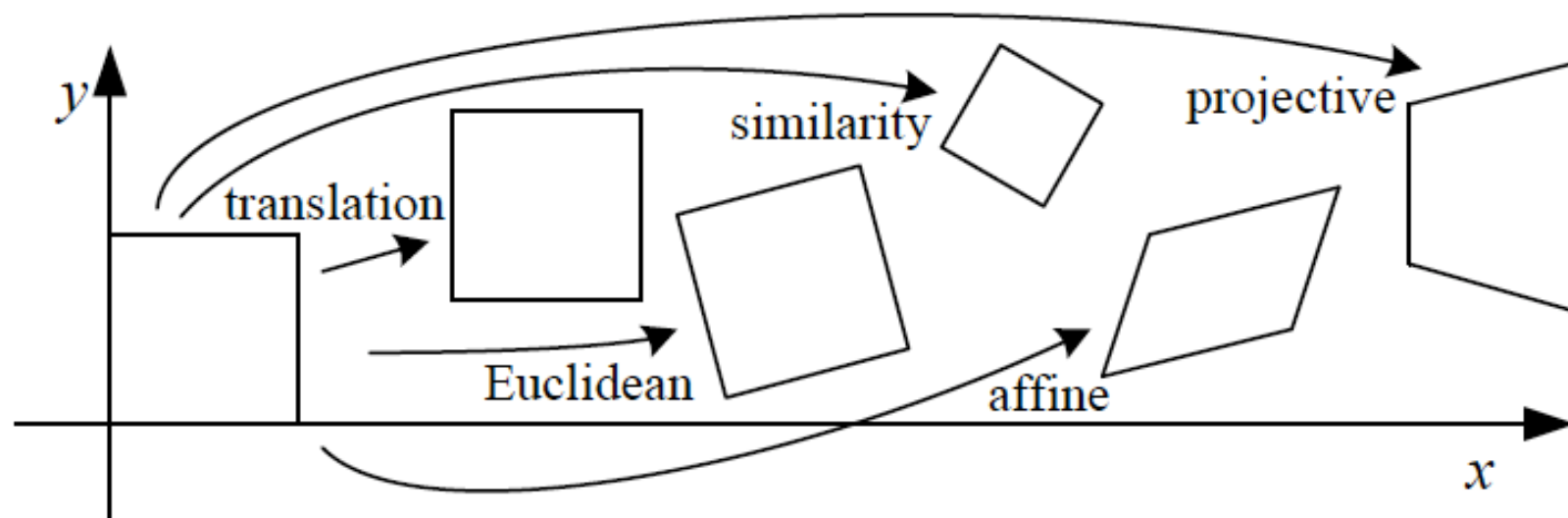   - Gives epipolar lines

2D Transformations
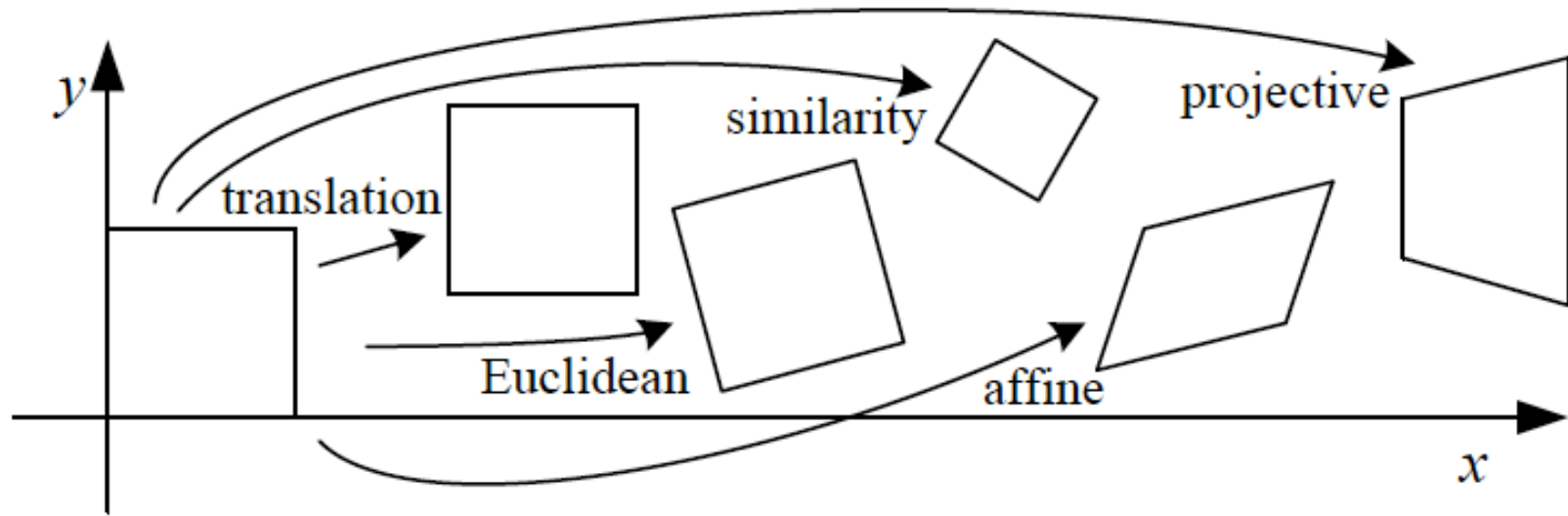
2D Transformations

Example: translation

$$x' = x + t$$

2D Transformations

Example: translation

$$x' = x + t \qquad x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x}$$

2D Transformations

Homogenous vector

Example: translation

$$x' = x + t \qquad x' = \begin{bmatrix} I & t \end{bmatrix} \bar{x} \qquad \bar{x}' = \begin{bmatrix} I & t \\ 0^T & 1 \end{bmatrix} \bar{x}$$

[BTW: Now we can chain transformations]

Projective Transformations

- *Projective* transformations: for 2D images it's a 3x3 matrix applied to homogenous coordinates

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$
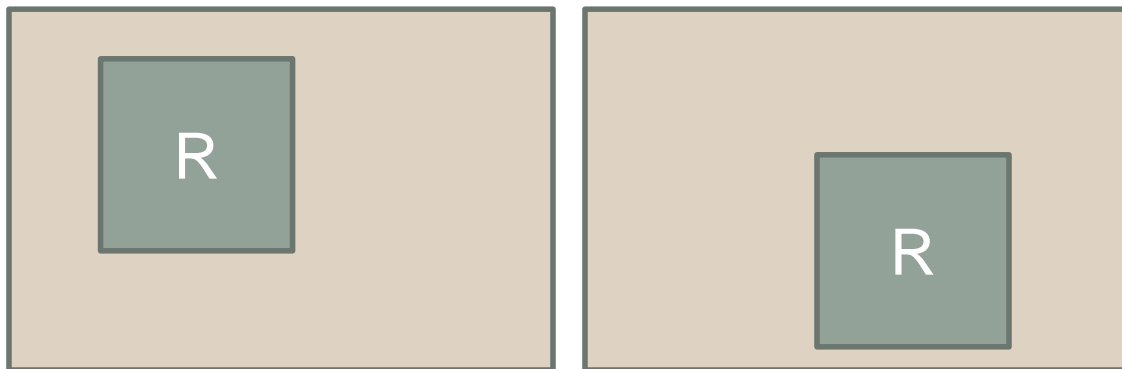
Special Projective Transformations

- **Translation**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **Preserves:**
  - Lengths/Areas
  - Angles
  - Orientation
  - Lines

R

R

Special Projective Transformations

• **Euclidean (Rigid body)**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

• **Preserves:**
  - Lengths/Areas
  - Angles
  - Lines

Special Projective Transformations

- **Similarity (trans, rot, scale) transform**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a\cos(\theta) & -a\sin(\theta) & t_x \\ a\sin(\theta) & a\cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

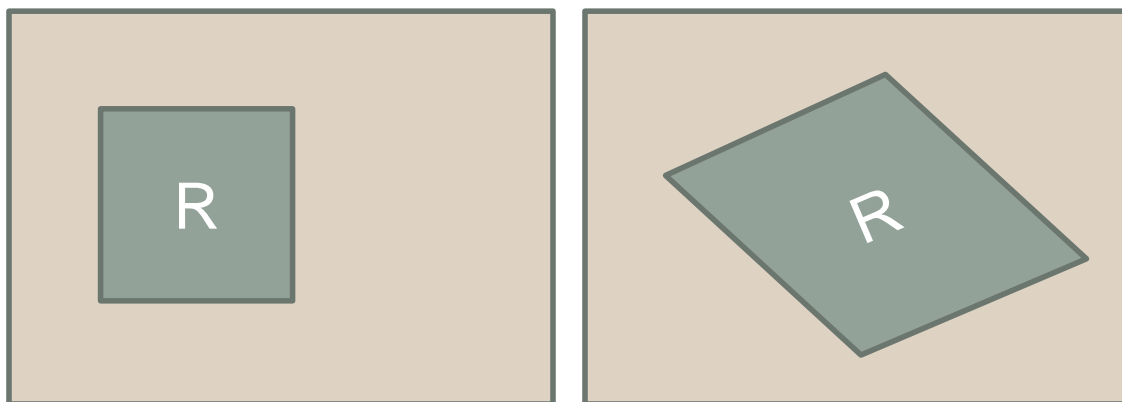- **Preserves:**
  - Ratios of Areas
  - Angles
  - Lines

Special Projective Transformations

- **Affine transform**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **Preserves:**
  - Parallel lines
  - Ratio of Areas
  - Lines

Projective Transformations

- Remember, these are homogeneous coordinates

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \simeq \begin{bmatrix} sx' \\ sy' \\ s \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & s \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$
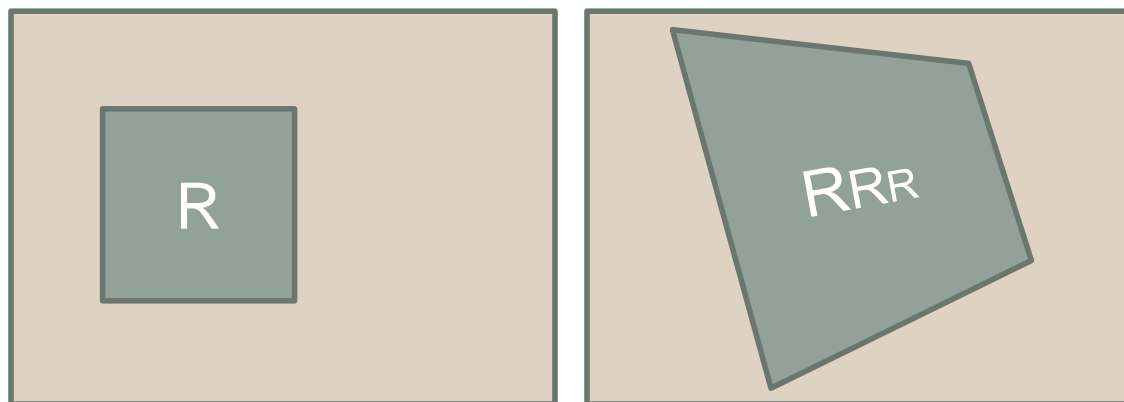
Projective Transformations

- General **projective transform** (or *Homography)*

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \simeq \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$
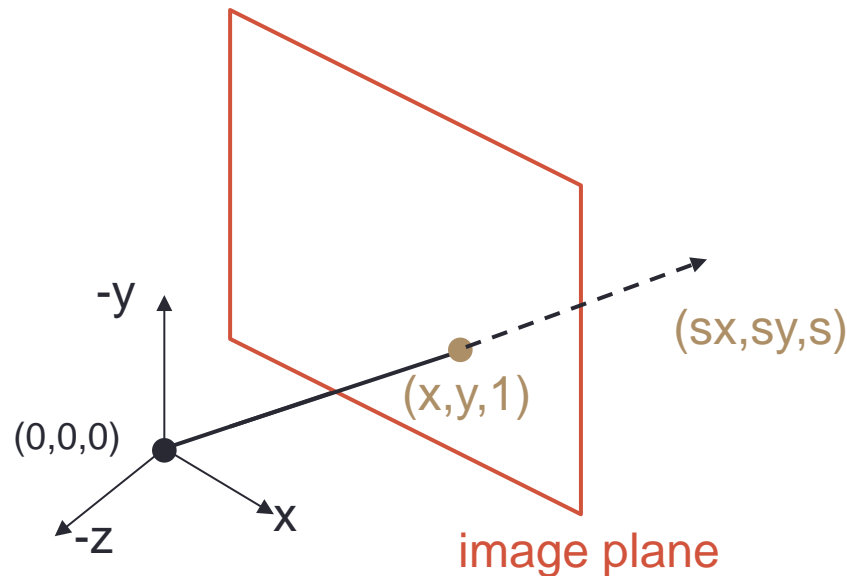
- **Preserves:**
  - Lines

  - Also cross ratios (maybe later)

The projective plane

- What is the geometric intuition of using homogenous coordinates ?

    - a point in the image is a *ray* in projective space

- Each *point* (x,y) on the plane (at z=1) is represented by a *ray* (sx,sy,s)

   – all points on the ray are equivalent: $(x, y, 1) \cong (sx, sy, s)$

Image reprojection

The irrelevant world!

- ● Basic question
  - • How to relate two images from the same camera center?
    - • how to map a pixel from projective plane PP1 to PP2
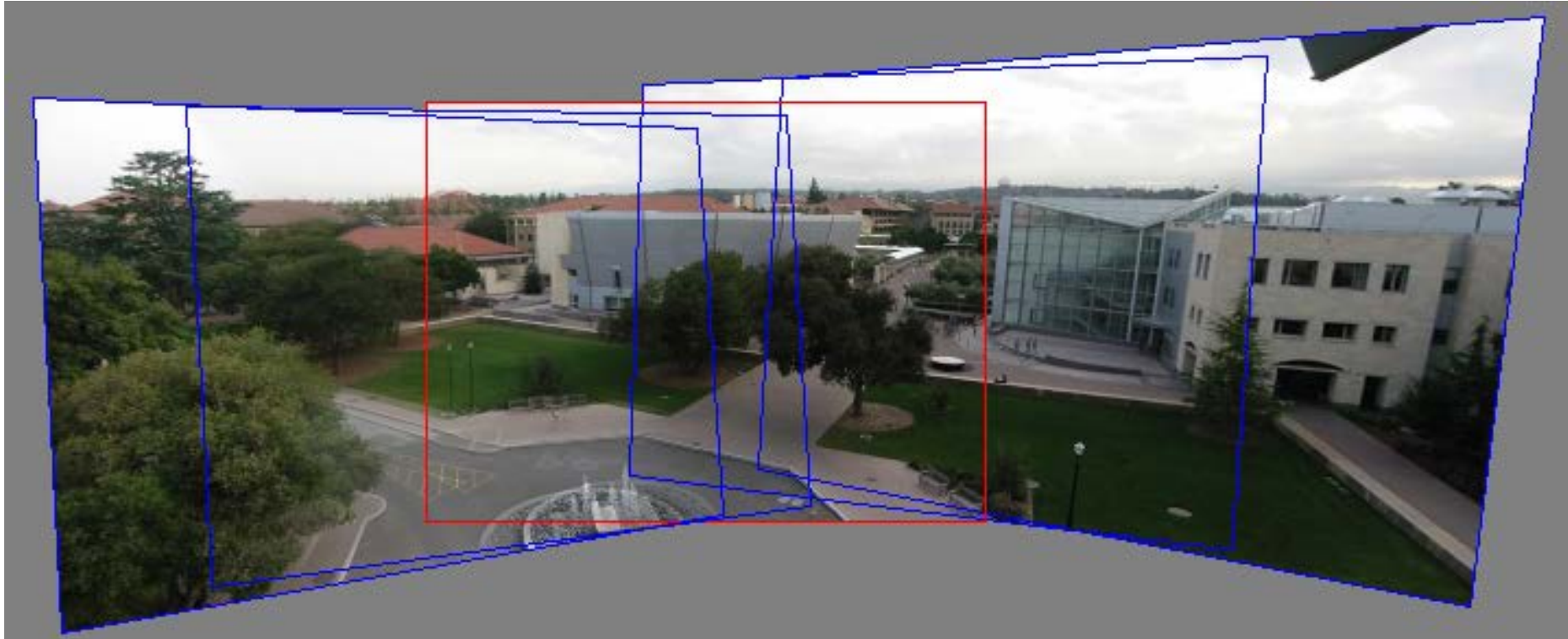
PP2

Answer

- • Cast a ray through each pixel in PP1
- • Draw the pixel where that ray intersects PP2

PP1

Observation:
Rather than thinking of this as a 3D reprojection, think of it as a 2D **image warp** from one image to another.

Source: Alyosha Efros
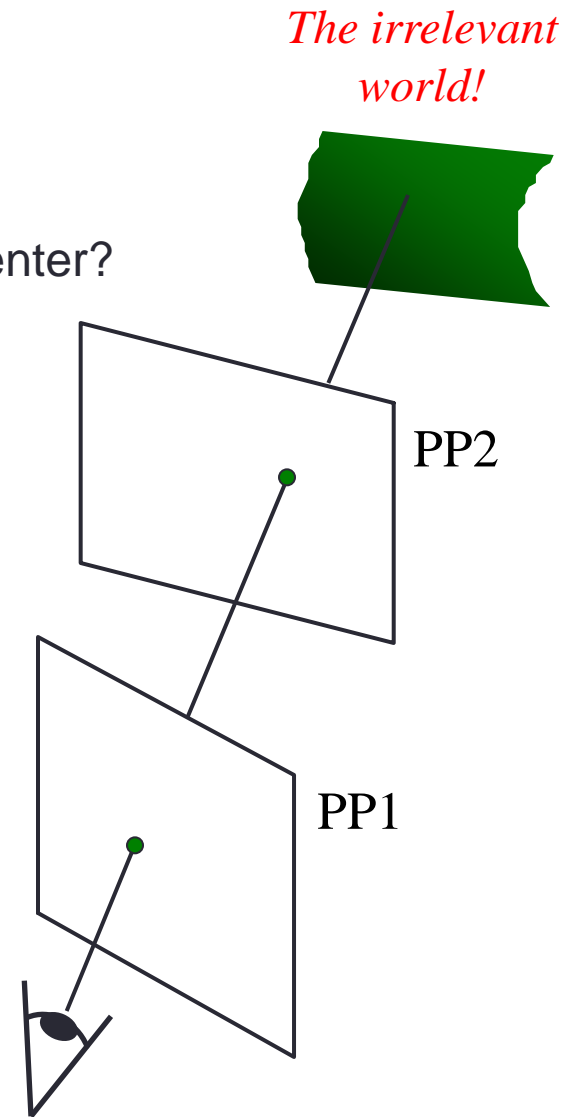
Application: Simple mosaics

How to stitch together a panorama (a.k.a. mosaic)?

- Basic Procedure
  - Take a sequence of images from the same position
    - Rotate the camera about its optical center
  - Compute transformation between second image and first
  - Transform the second image to overlap with the first
  - Blend the two together to create a mosaic
  - (If there are more images, repeat)

- …but wait, why should this work at all?
  - What about the 3D geometry of the scene?
  - Why aren't we using it?

Image reprojection
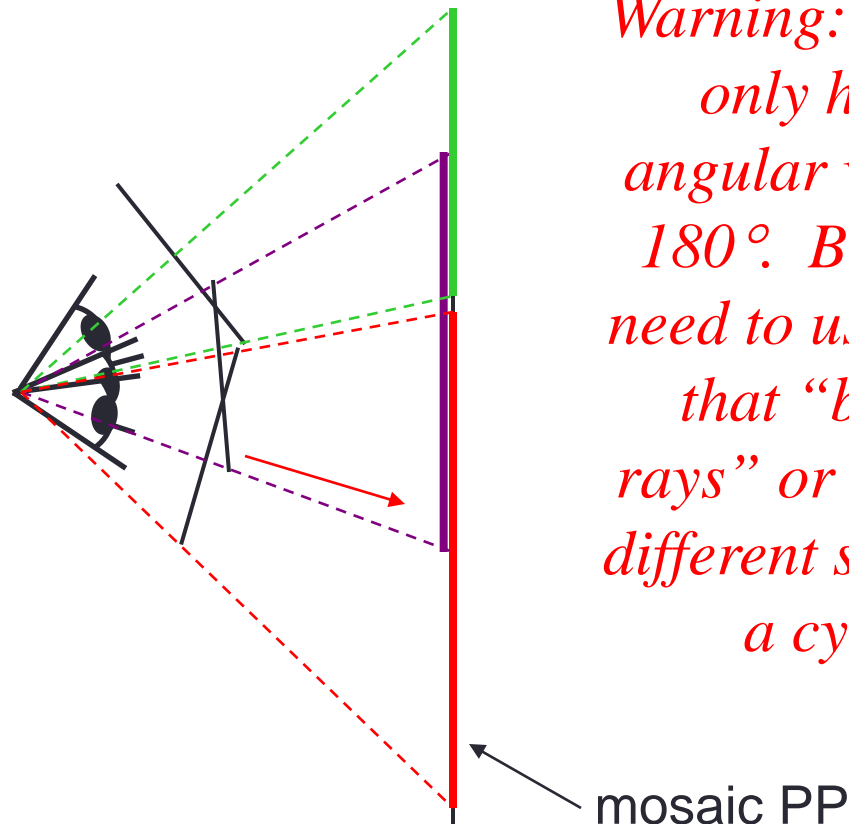
Warning: This model only holds for angular views up to 180°. Beyond that need to use sequence that "bends the rays" or map onto a different surface, say, a cylinder.

mosaic PP

- The mosaic has a natural interpretation in 3D
  - The images are reprojected onto a common plane
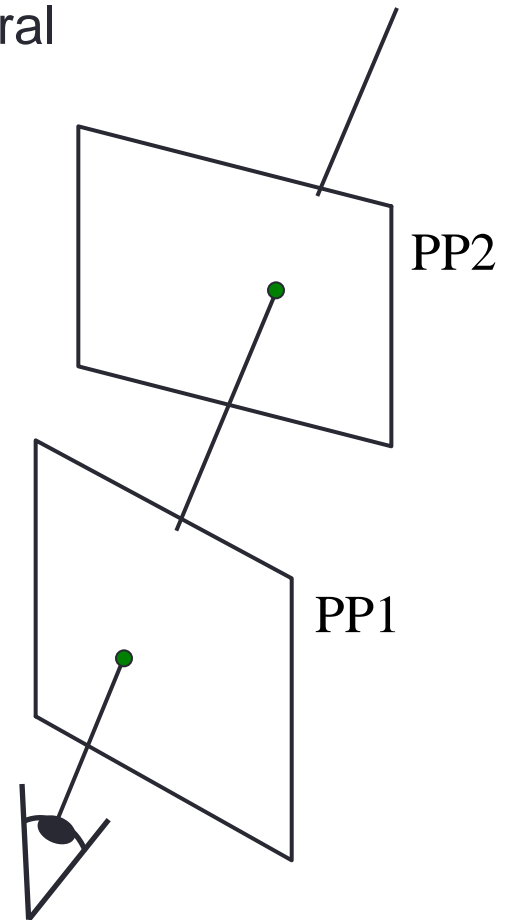 - The mosaic is formed on this plane

Mosaics

Obtain a wider angle view by combining multiple images *all of which are taken from the same camera center*.
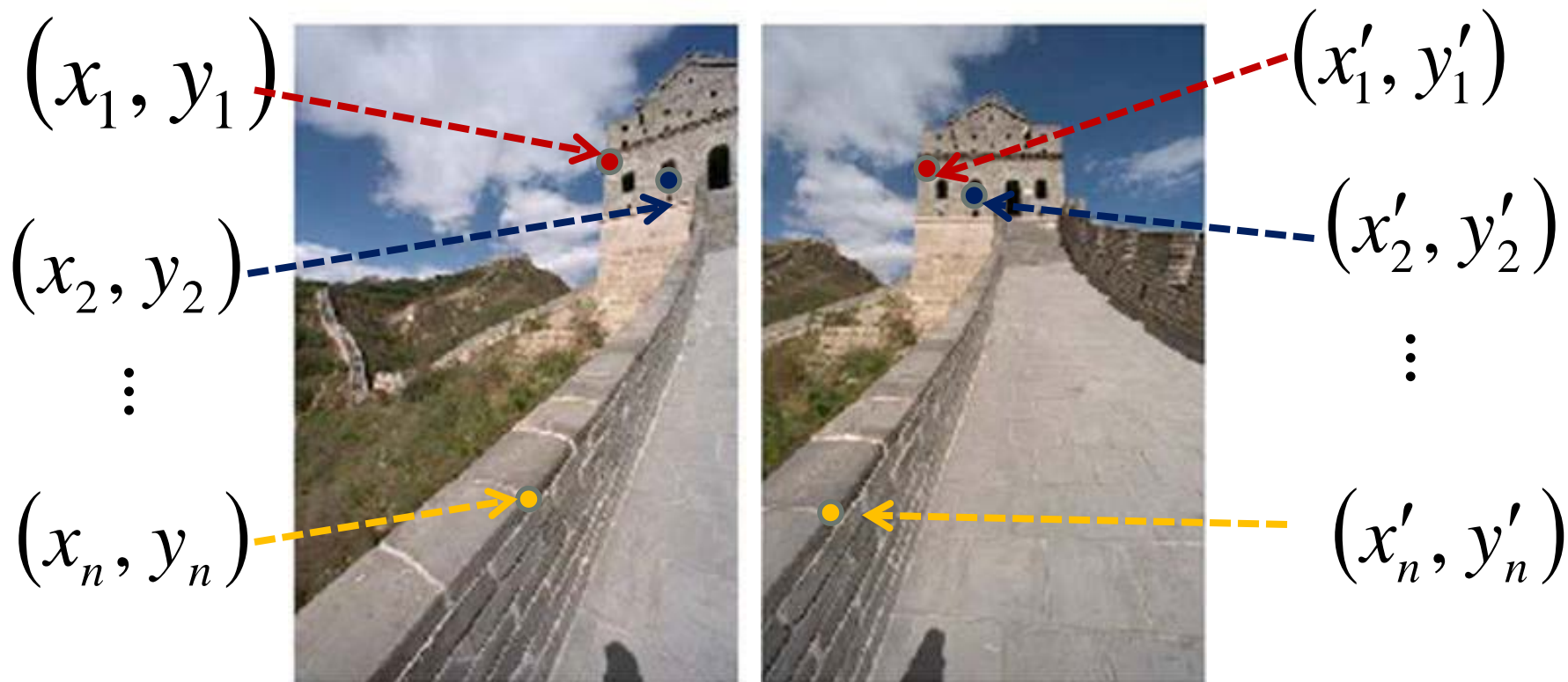
Image reprojection: Homography

- A projective transform is a mapping between any two PPs with the same center of projection
  - rectangle should map to arbitrary quadrilateral
  - parallel lines aren't
  - but must preserve straight lines
- called **Homography**

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

p' H p

PP2

PP1

Source: Alyosha Efros

Homography

(x_1, y_1)

(x_2, y_2)

$\vdots$

(x_n, y_n)

(x_1', y_1')

(x_2', y_2')

$\vdots$

(x_n', y_n')

To **compute** the homography given pairs of corresponding points in the images, we need to set up an equation where the parameters of **H** are the unknowns…

Solving for homographies

$$\mathbf{p'} = \mathbf{Hp}$$

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Can set scale factor i=1. So, there are 8 unknowns.
- Set up a system of linear equations $\mathbf{Ah} = \mathbf{b}$
- where vector of unknowns h = [a,b,c,d,e,f,g,h]T
- Need at least 4 points for 8 eqs, but the more the better…
- Solve for h. If overconstrained, solve using least-squares:

$$\min \left\| \mathbf{Ah} - \mathbf{b} \right\|^2$$

- Look familiar?   (If don't set i to 1 can use SVD)
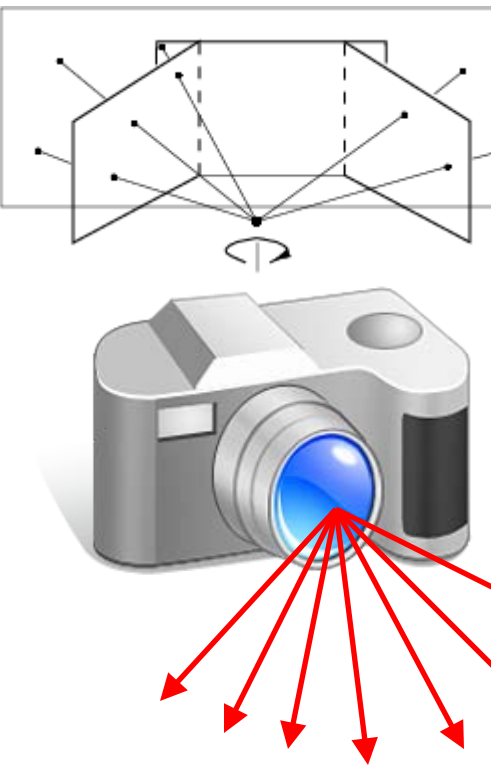- >> help mldivide

Homography

(x, y)

$\left(\dfrac{wx'}{w}, \dfrac{wy'}{w} \right) = (x', y')$

To **apply** a given homography **H**

- Compute **p'** = **Hp** (regular matrix multiply)
- Convert **p'** from homogeneous to image coordinates

$$
\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}
$$

$$\mathbf{p'} \qquad\qquad \mathbf{H} \qquad\quad \mathbf{p}$$
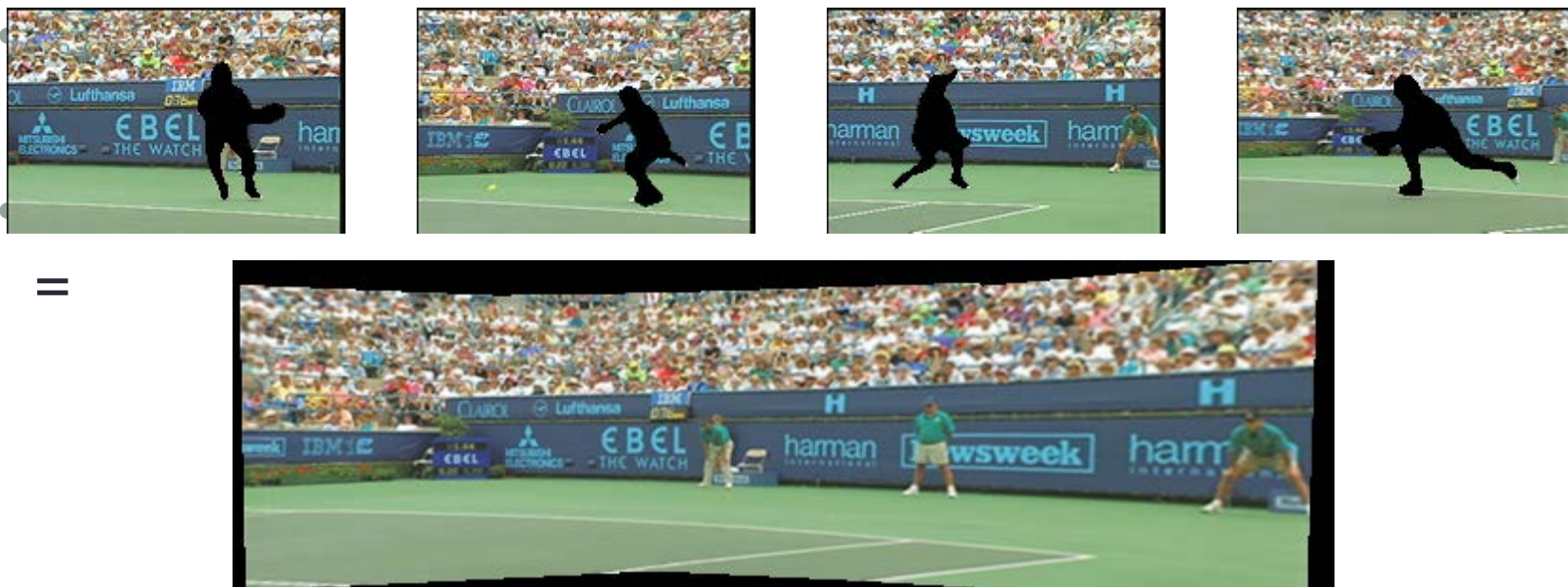
Mosaics

image from S. Seitz

Combine images with the computed homographies…

Mosaics for Video Coding

- Convert masked images into a background sprite for content-based coding

=

Homographies and 3D planes

- Remember this:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \simeq \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Suppose the 3D points are on a plane:

$$aX + bY + cZ + d = 0$$

Homographies and 3D planes
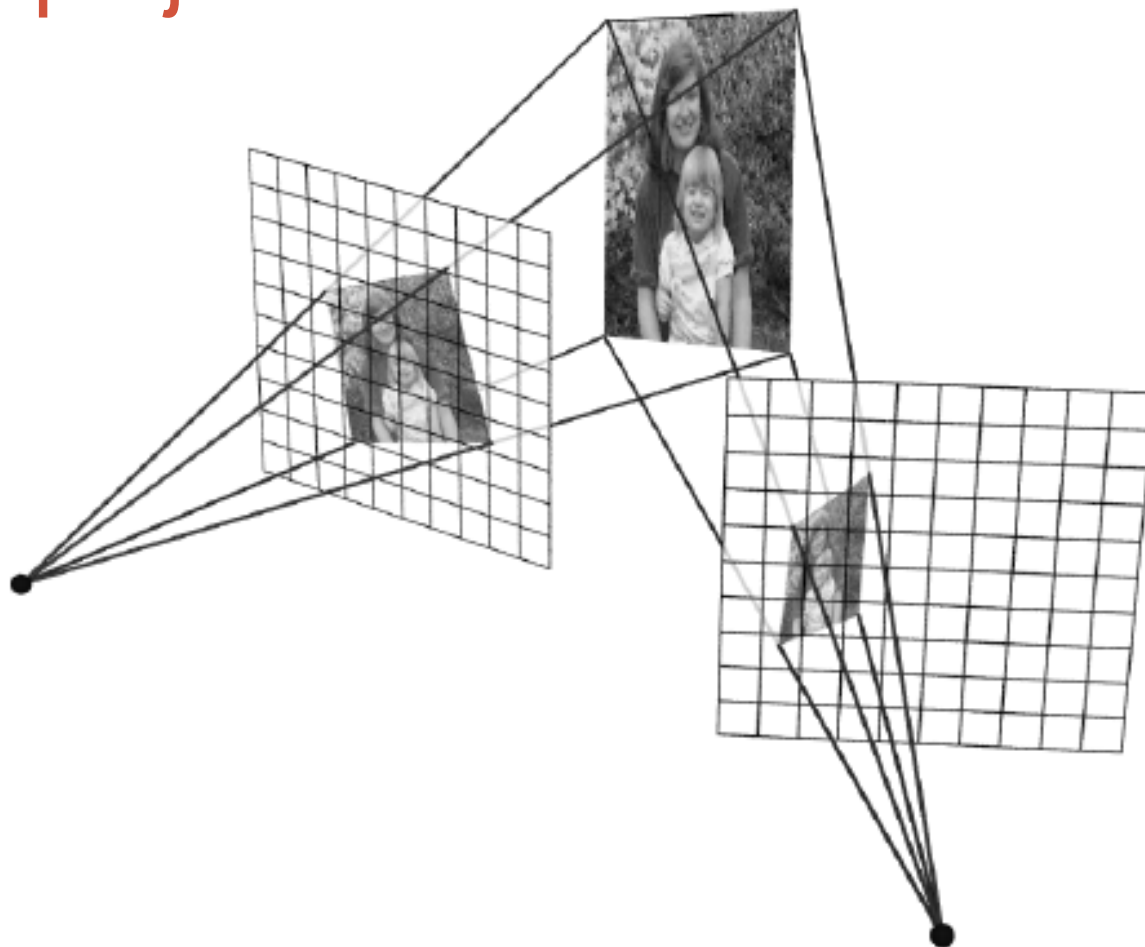
- On the plane *[a b c d]* can replace Z:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \simeq \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ (aX + bY + d)/(-c) \\ 1 \end{bmatrix}$$

- So, can put the Z coefficients into the others:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \simeq \begin{bmatrix} m'_{00} & m'_{01} & 0 & m'_{03} \\ m'_{10} & m'_{11} & 0 & m'_{13} \\ m'_{20} & m'_{21} & 0 & m'_{23} \end{bmatrix} \begin{bmatrix} X \\ Y \\ (aX + bY + d)/(-c) \\ 1 \end{bmatrix}$$

3x3 Homography!

Image reprojection

- Mapping between planes is a homography. Whether a plane in the world to the image or between image planes.
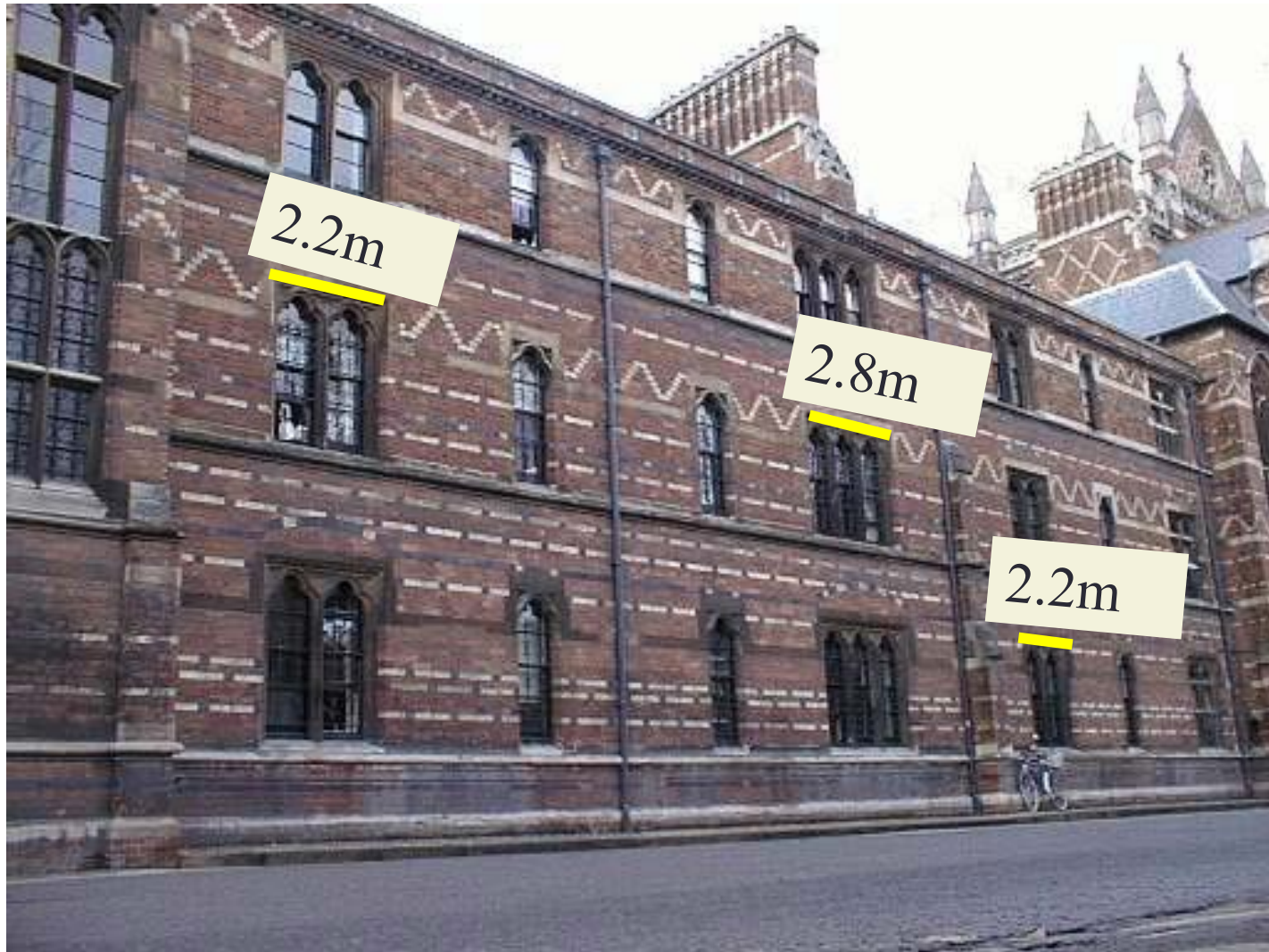
What else: Rectifying Slanted Views of Planes
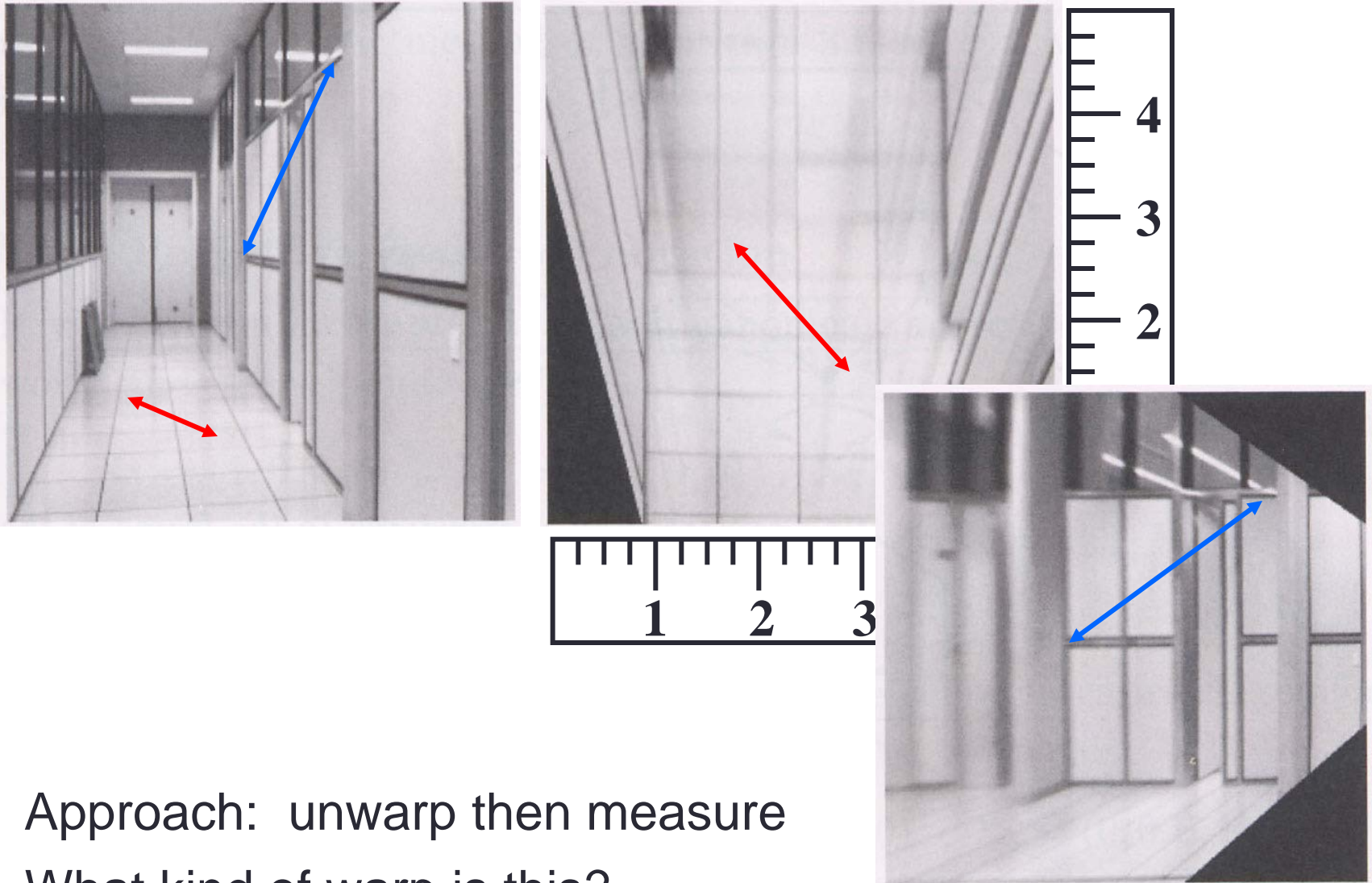
Rectifying slanted views

Corrected image (front-to-parallel)

Measuring distances
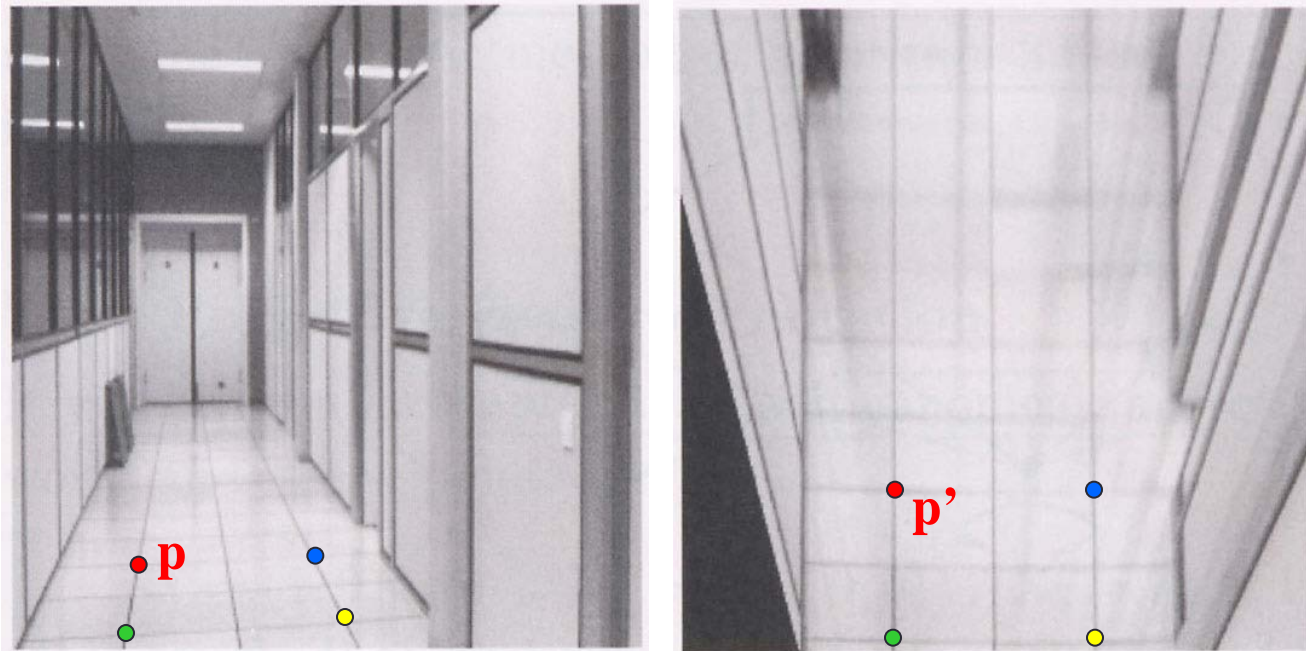
Measurements on planes

Approach: unwarp then measure

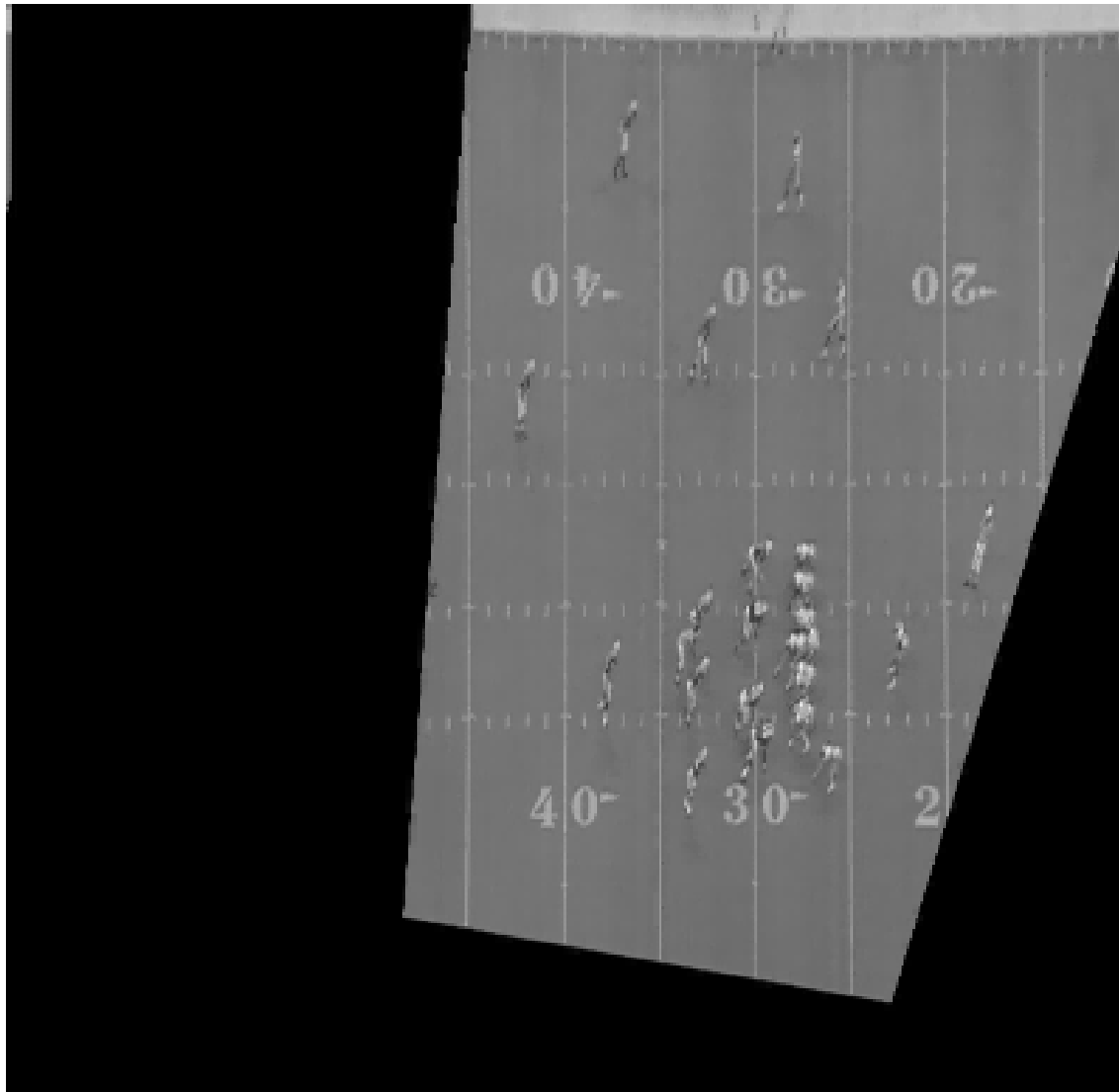What kind of warp is this?
Homography…

Image rectification

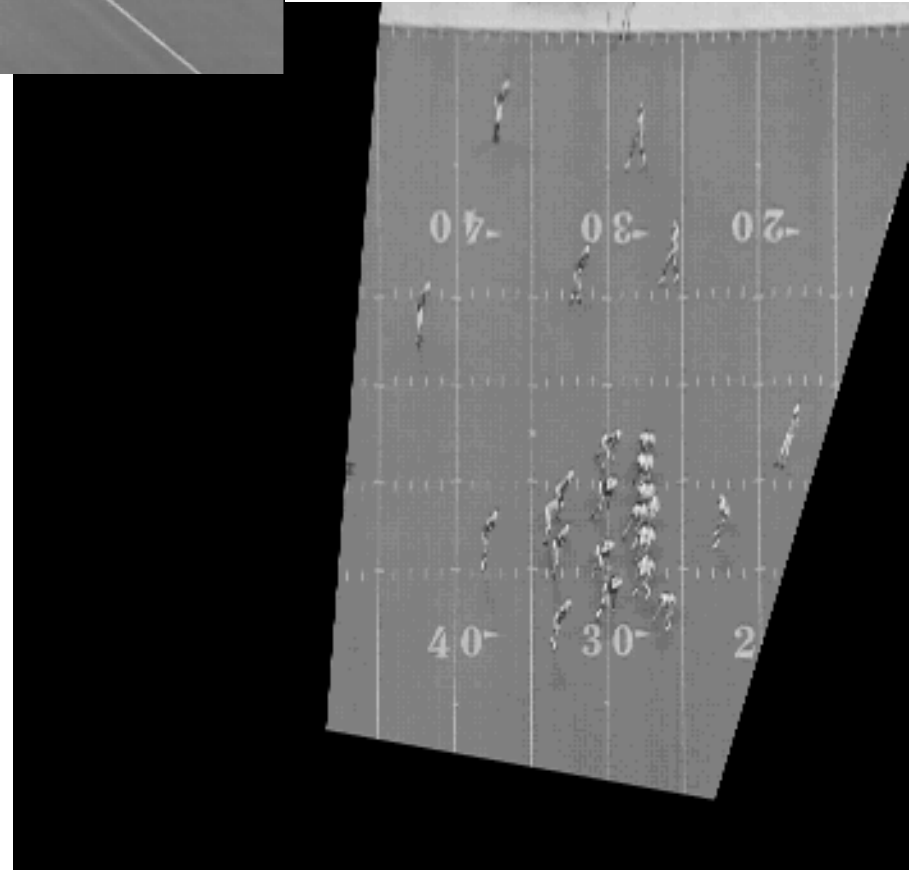A planar rectangular grid in the scene. Map it into a
rectangular grid in the image.

Some other images of rectangular grids…

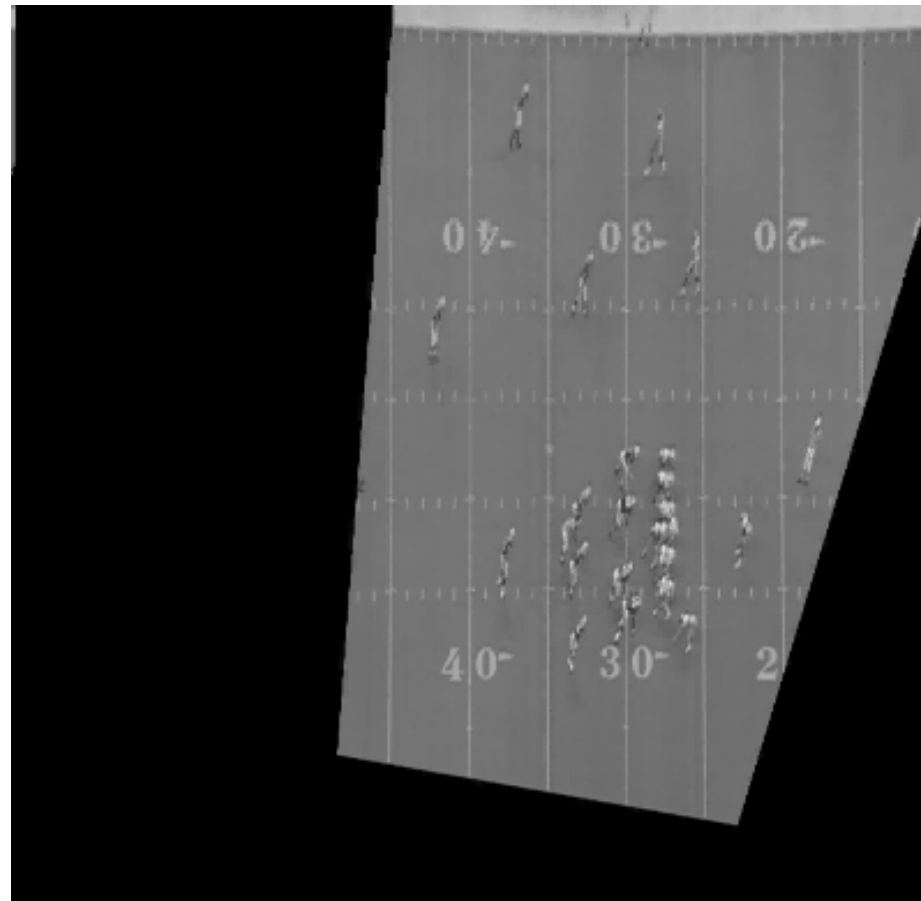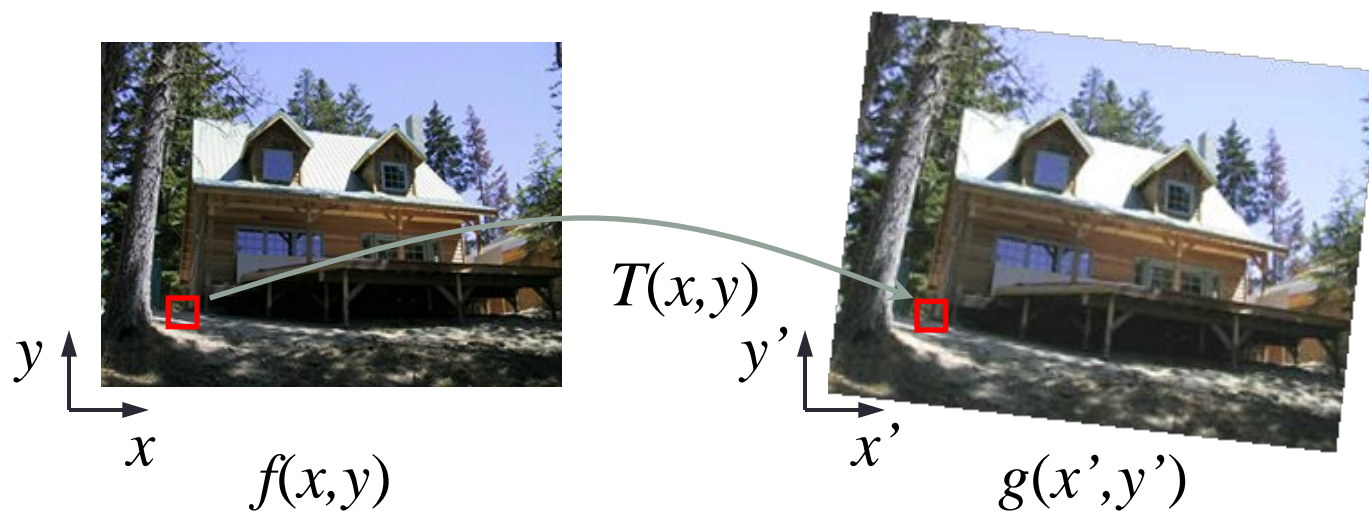# Who needs a blimp?

Same pixels – via a homography

Image warping

$T(x,y)$
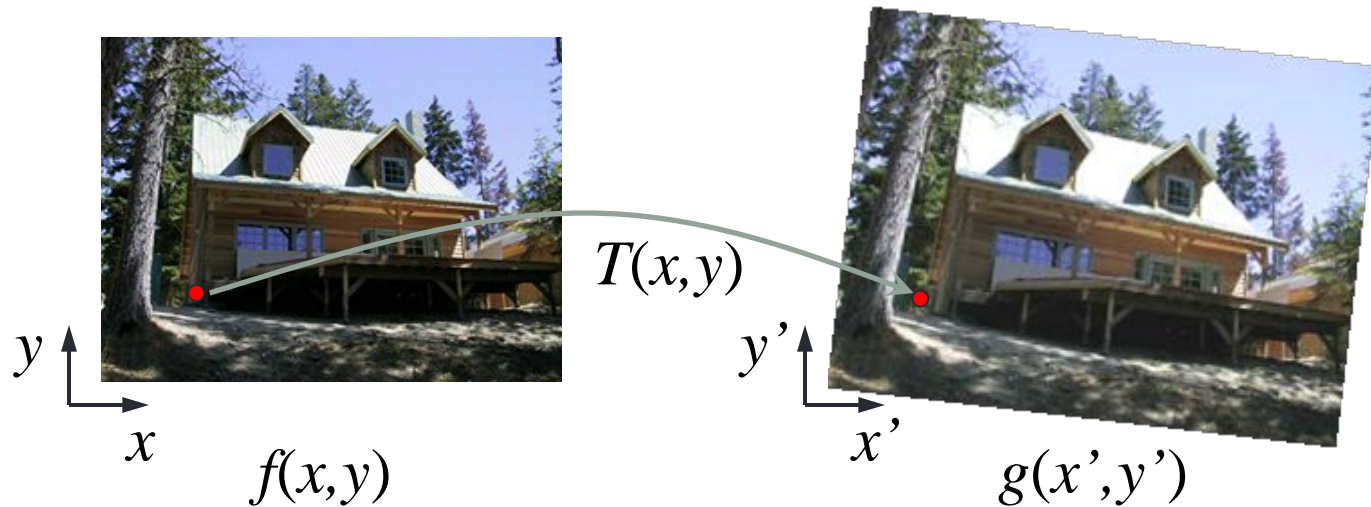
y ↑
x
$f(x,y)$

y' ↑
x'
$g(x',y')$

Given a coordinate transform and a source image *f(x,y)*, how do we compute a transformed image *g(x',y') = f(T(x,y))*?

Forward warping

$T(x,y)$

y

x

$f(x,y)$

y'

x'

$g(x',y')$

- Send each pixel $f(x,y)$ to its corresponding location
-         $(x',y') = T(x,y)$ in the second image

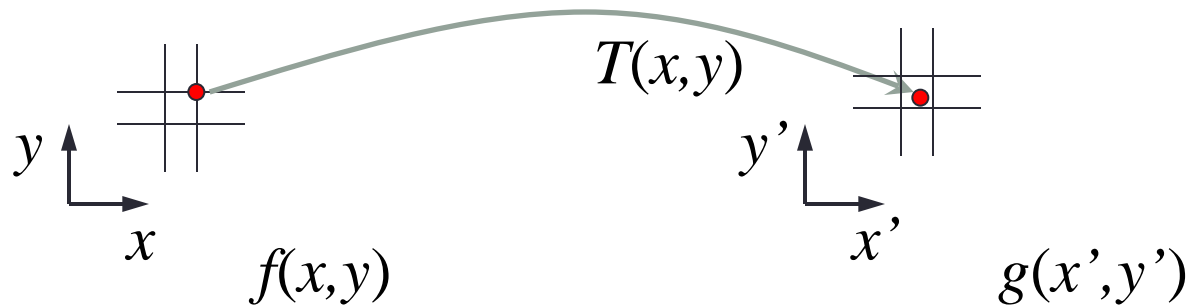Q: what if pixel lands "between" two pixels?
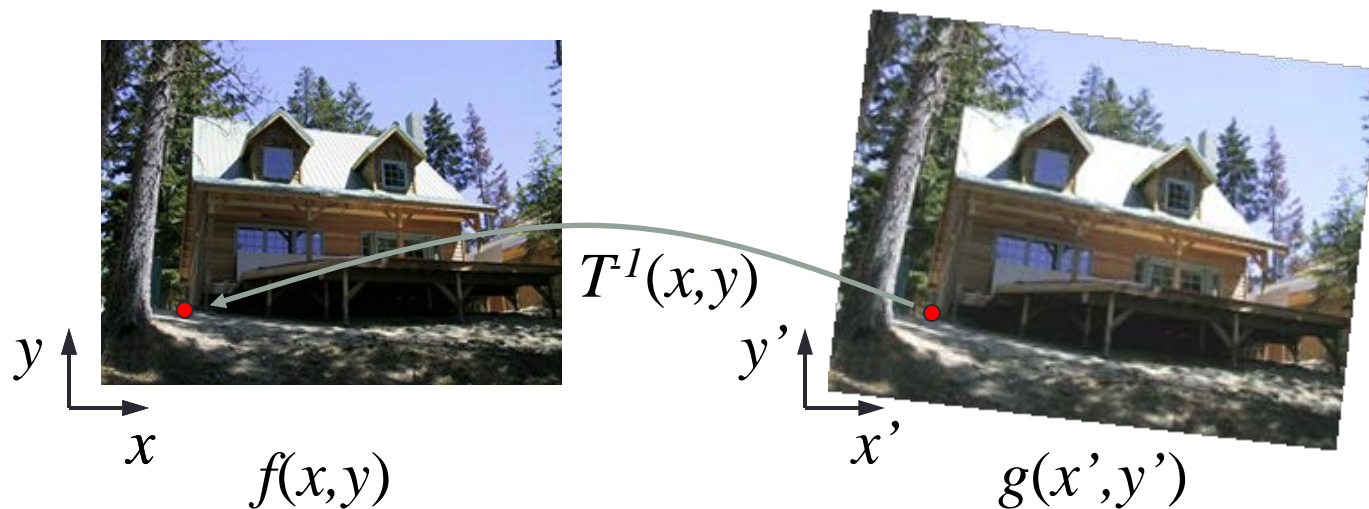
Forward warping

- Send each pixel *f(x,y)* to its corresponding location
- $(x',y') = T(x,y)$ in the second image

Q: what if pixel lands "between" two pixels?

A: distribute color among neighboring pixels (x',y')

– Known as "splatting"
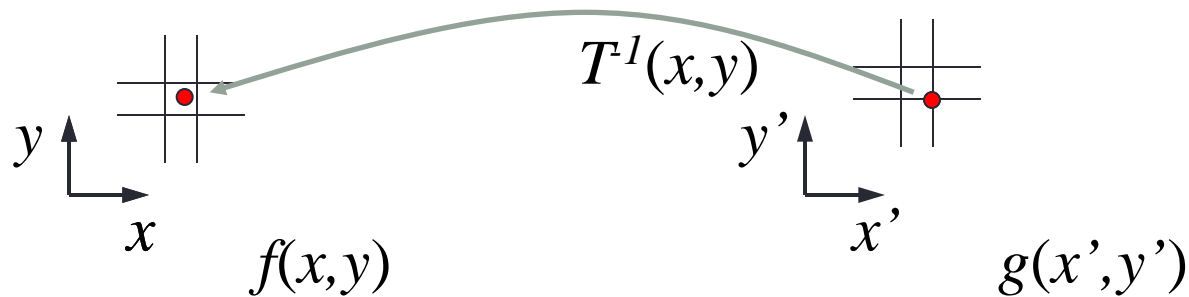
Inverse warping

$$T^{-1}(x,y)$$

y $f(x,y)$ x

y' x' $g(x',y')$

Get each pixel $g(x',y')$ from its corresponding location

$(x,y) = T^{-1}(x',y')$ in the first image

Q: what if pixel comes from "between" two pixels?

Inverse warping

$T^{-1}(x,y)$

y

x

$f(x,y)$

y'

x'

$g(x',y')$

Get each pixel $g(x',y')$ from its corresponding location

$(x,y) = T^{-1}(x',y')$ in the first image
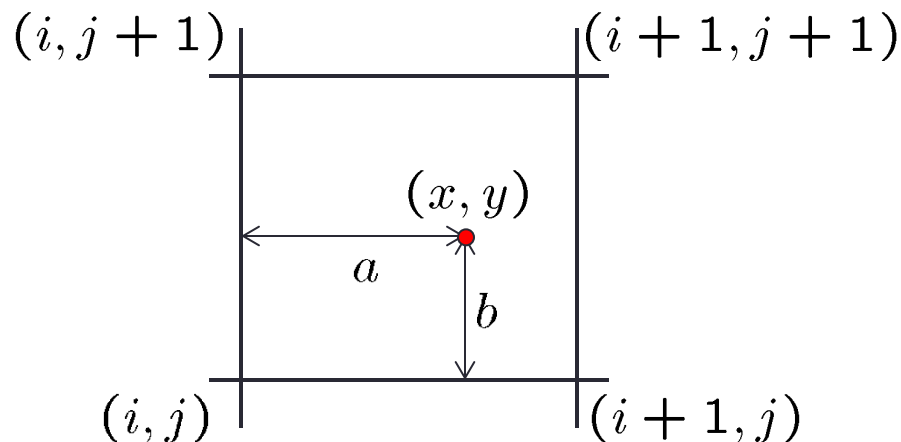
Q: what if pixel comes from "between" two pixels?

A: *Interpolate* color value from neighbors

– nearest neighbor, bilinear…

`>> help interp2`

Bilinear interpolation

Sampling at *f(x,y):*

$$f(x, y) = \quad (1 - a)(1 - b) \quad f[i, j]$$
$$+a(1 - b) \quad f[i + 1, j]$$
$$+ab \quad f[i + 1, j + 1]$$
$$+(1 - a)b \quad f[i, j + 1]$$

Recap: How to stitch together a panorama (a.k.a. mosaic)?

- Basic Procedure
 - Take a sequence of images from the same position
    - Rotate the camera about its optical center
  - Compute transformation (homography) between second image and first using corresponding points.
  - Transform the second image to overlap with the first.
  - Blend the two together to create a mosaic.
  - (If there are more images, repeat)

Source: Steve Seitz

Why projective geometry?

The projective plane

- What is the geometric intuition of using homogenous coordinates ?

  - a point in the image is a *ray* in projective space
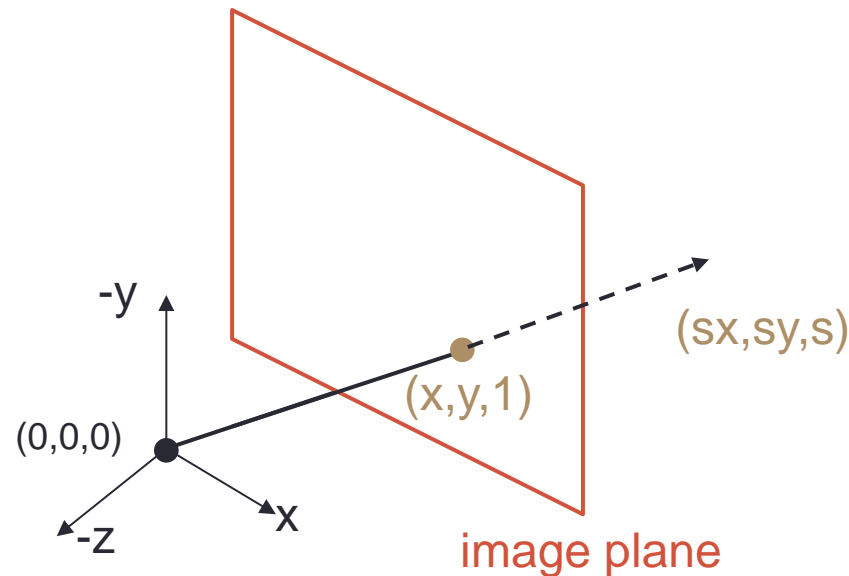
- Each *point* (x,y) on the plane is represented by a *ray* (sx,sy,s)

  – all points on the ray are equivalent: $(x, y, 1) \cong (sx, sy, s)$
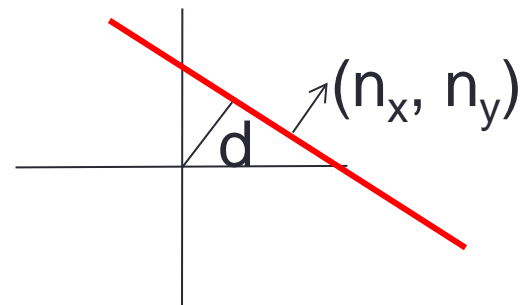
Homogeneous coordinates

2D Points:

$$p = \begin{bmatrix} x \\ y \end{bmatrix} \longrightarrow p' = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad p' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \longrightarrow p = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$$

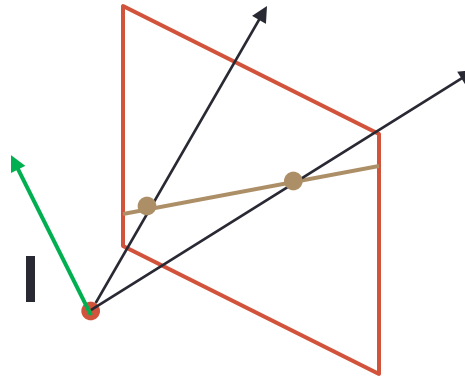2D Lines: $\quad ax + by + c = 0$

$$\begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \qquad l = \begin{bmatrix} a & b & c \end{bmatrix} \Rightarrow \begin{bmatrix} n_x & n_y & d \end{bmatrix}$$

(n_x, n_y)

d

Projective lines

- What does a line in the image correspond to in projective space?

- A line is a *plane* of rays through origin
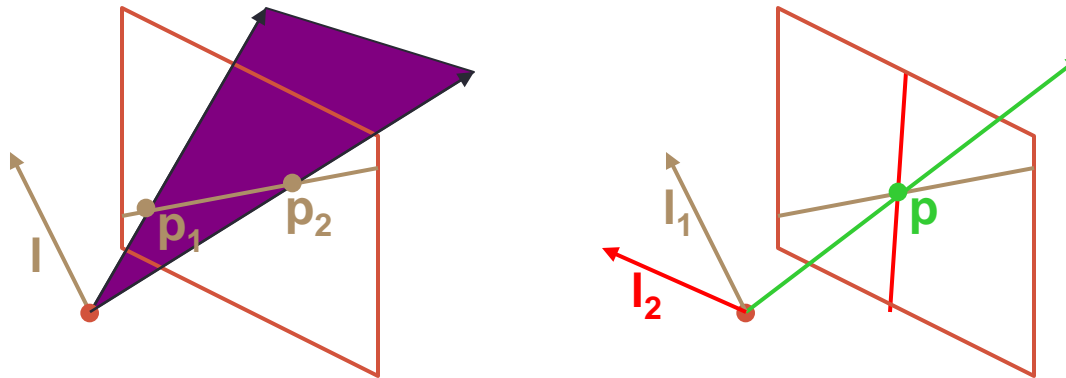 - all rays (x,y,z) satisfying: ax + by + cz = 0

$$0 = \begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

in vector notation:

l   **p**

- A line is also represented as a homogeneous 3-vector **l**

Point and line duality

- A line **l** is a homogeneous 3-vector
- It is $\perp$ to every point (ray) **p** on the line: $\mathbf{l}^T\mathbf{p}=0$

What is the line **l** spanned by rays $\mathbf{p_1}$ and $\mathbf{p_2}$?

- **l** is $\perp$ to $\mathbf{p_1}$ and $\mathbf{p_2}$ $\Rightarrow$ **l** = $\mathbf{p_1} \times \mathbf{p_2}$
- **l** is the plane normal

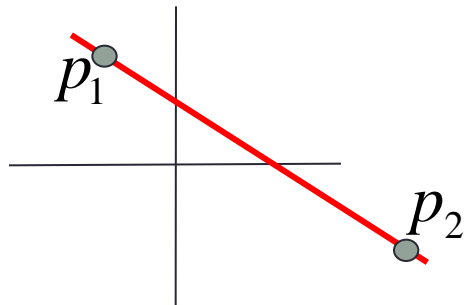What is the intersection of two lines $\mathbf{l_1}$ and $\mathbf{l_2}$?

- **p** is $\perp$ to $\mathbf{l_1}$ and $\mathbf{l_2}$ $\Rightarrow$ **p** = $\mathbf{l_1} \times \mathbf{l_2}$

Points and lines are *dual* in projective space

- given any formula, can switch the meanings of points and lines to get another formula
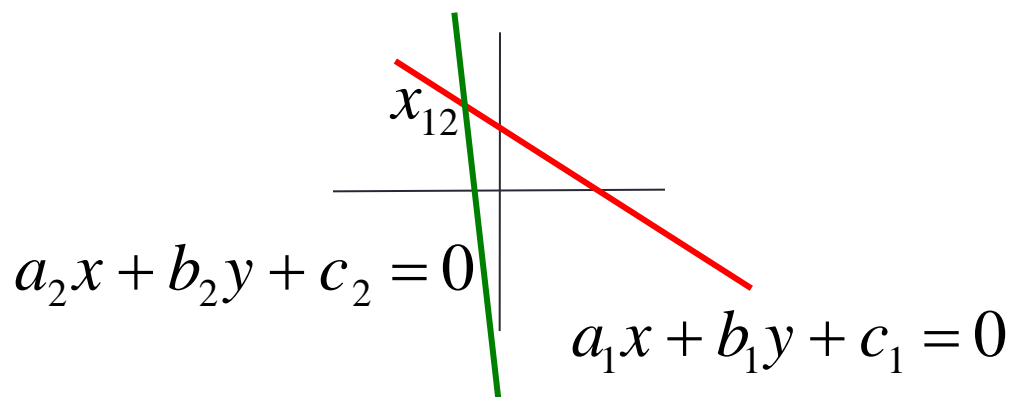
Homogeneous coordinates

Line joining two points:

$$ax + by + c = 0$$

$$p_1 = \begin{bmatrix} x_1 & y_1 & 1 \end{bmatrix}$$

$$p_2 = \begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix}$$

$$l = p_1 \times p_2$$

Homogeneous coordinates

Intersection between two lines:

$$a_2x + b_2y + c_2 = 0$$

$$a_1x + b_1y + c_1 = 0$$

x_{12}

$$l_1 = \begin{bmatrix} a_1 & b_1 & c_1 \end{bmatrix}$$

$$l_2 = \begin{bmatrix} a_2 & b_2 & c_2 \end{bmatrix}$$

$$x_{12} = l_1 \times l_2$$

Ideal points and lines
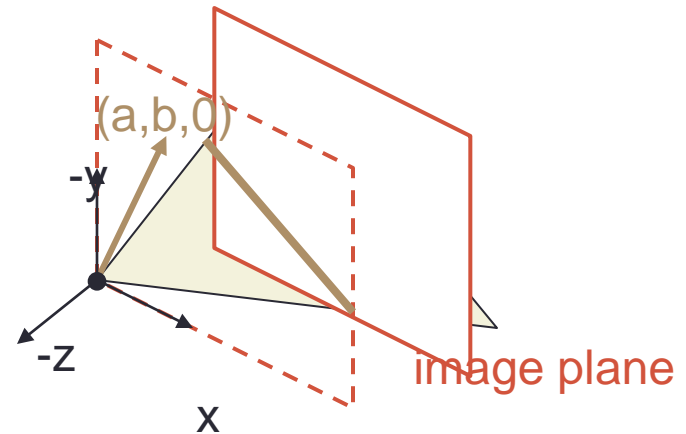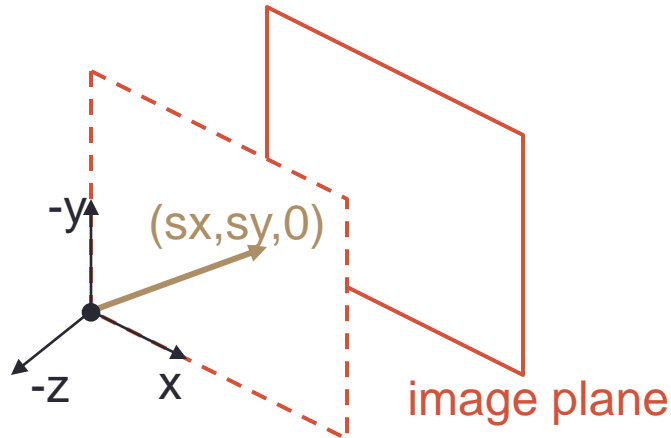
- ## Ideal point ("point at infinity")
  - p $\cong$ (x, y, 0) – parallel to image plane
  - It has infinite image coordinates

Ideal line

- l $\cong$ (a, b, 0) – normal is parallel to (is in!) image plane
- Corresponds to a line in the image (finite coordinates)
  - goes through image origin (*principle point*)

3D projective geometry

- These concepts generalize naturally to 3D
- Recall the equation of a plane:

$$aX + bY + cZ + d = 0$$

- Homogeneous coordinates
  - Projective 3D points have four coords: $\mathbf{p} = (wX, wY, wZ, w)$
- Duality
  - A plane $\mathbf{N}$ is also represented by a 4-vector $\mathbf{N} = (a,b,c,d)$
  - Points and planes are dual in 3D: $\mathbf{N^T p} = \mathbf{0}$
- Projective transformations
  - Represented by 4x4 matrices T: P' = TP

3D to 2D: "perspective" projection

- Matrix Projection:
$$\mathbf{p} = \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi P}$$

You've already seen this.

What is *not* preserved under perspective projection?

What IS preserved?

What's next…

- Today – more projective geometry, the duality between points and lines in projective space

- Tuesday– using the projective geometry revisit 2-views:
  - Essential and Fundamental matrices