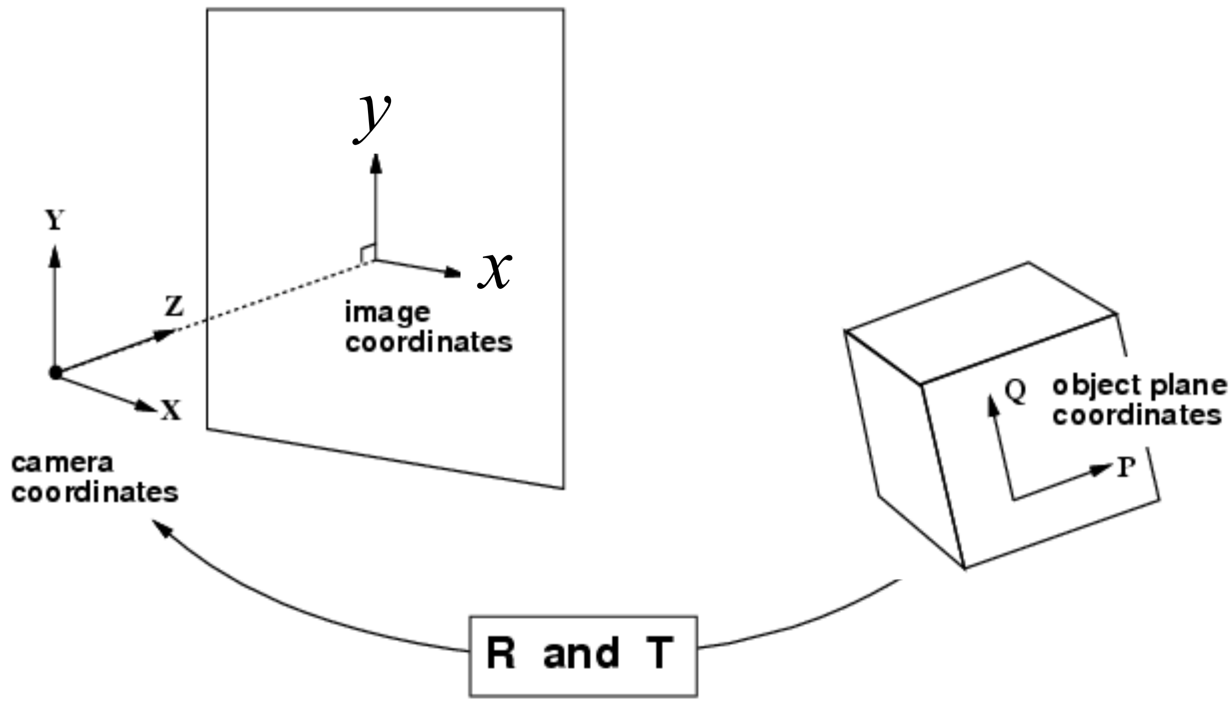
Robert Collins
CMPEN454

Planar Homographies

Motivation: Points on Planar Surface



We want to derive a transformation that maps points on a 2d planar surface into 2d image points.

Recall: Camera Projection

$$\underbrace{\begin{pmatrix} u' \\ v' \\ w' \end{pmatrix}}_{\text{homogeneous pixel coords}} = \underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}}_{\text{film to pixel}} \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{3d-to-2d projection}} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{world to camera}} \underbrace{\begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}}_{\text{3d world coords}}$$

$$\downarrow$$

$$u = u'/w'$$
$$v = v'/w'$$

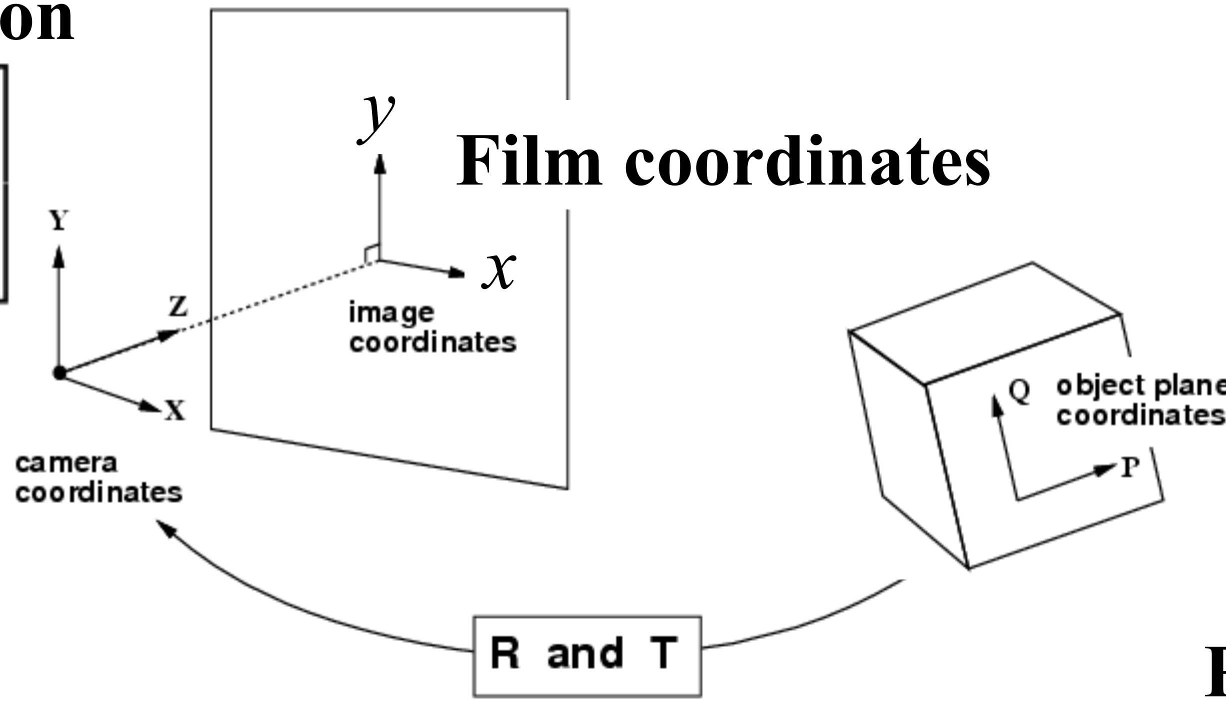
2d pixel coords

Projection of Points on Planar Surface

Perspective projection

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

Film coordinates



Point on plane

$$\begin{bmatrix} p \\ q \\ 0 \\ 1 \end{bmatrix}$$

Rotation + Translation

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Robert Collins
CMPEN454

Projection of Planar Points

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ q \\ \boxed{0} \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

Projection of Planar Points (cont)

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} fr_{11} & fr_{12} & ft_x \\ fr_{21} & fr_{22} & ft_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

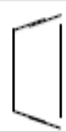
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

Homography H (planar projective transformation)

Robert Collins
CMPEN454

Background Info

2d-to-2d geometric transformations represented as 3x3 matrices in homogeneous coords

Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\left[\, I \mid t \,\right]_{2\times 3}$	2	orientation $+\cdots$	
rigid (Euclidean)	$\left[\, R \mid t \,\right]_{2\times 3}$	3	lengths $+\cdots$	
similarity	$\left[\, sR \mid t \,\right]_{2\times 3}$	4	angles $+\cdots$	
affine	$\left[\, A \,\right]_{2\times 3}$	6	parallelism $+\cdots$	
projective	$\left[\, H \,\right]_{3\times 3}$	8	straight lines	

$$\begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$ **Euclidean**

$$\begin{bmatrix} sR & t \\ 0 & 1 \end{bmatrix}$$ **similarity**

$$\begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix}$$ **affine**

$$\begin{bmatrix} A & b \\ c^T & d \end{bmatrix}$$ **Projective**
(homography)

Projection of Planar Points (cont)

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

Homography H (planar projective transformation)

Punchline: For planar surfaces, 3D to 2D perspective projection reduces to a 2D to 2D transformation.

Punchline2: This transformation is INVERTIBLE!

Special Case : Frontal Plane

What if the planar surface is perpendicular to the optic axis (Z axis of camera coord system)?

Then world rotation matrix simplies:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ q \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Frontal Plane

So the homography for a frontal plane simplies:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} fr_{11} & fr_{12} & ft_x \\ fr_{21} & fr_{22} & ft_y \\ r_{31} & r_{32} & t_z \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f\cos\theta & -f\sin\theta & ft_x \\ f\sin\theta & f\cos\theta & ft_y \\ 0 & 0 & t_z \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

Similarity Transformation

(preserves parallelism and angles)

What about Pixel Coords?

Internal camera parameters

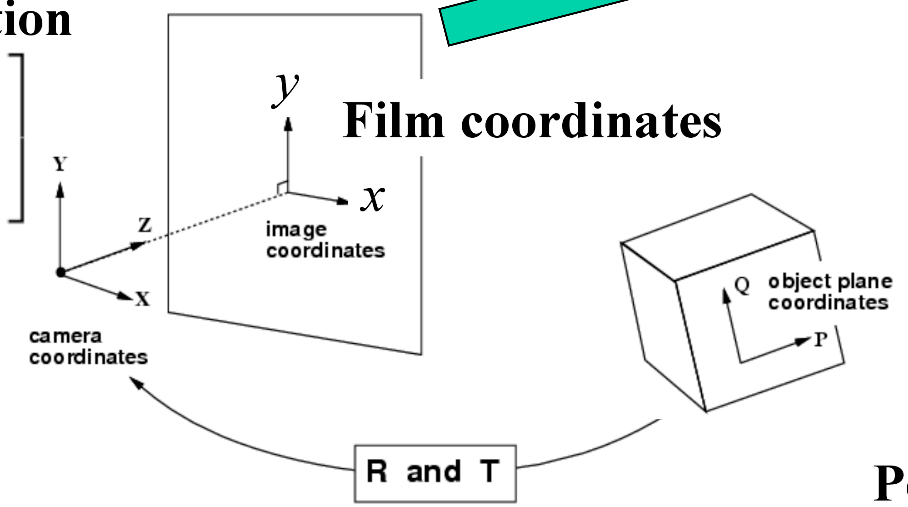
$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix}$$

pixels u,v

Perspective projection

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

Film coordinates



$$\begin{bmatrix} p \\ q \\ 0 \\ 1 \end{bmatrix}$$

Point on plane

Rotation + Translation

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Include 3x3 Affine Transform to Pixel Coords

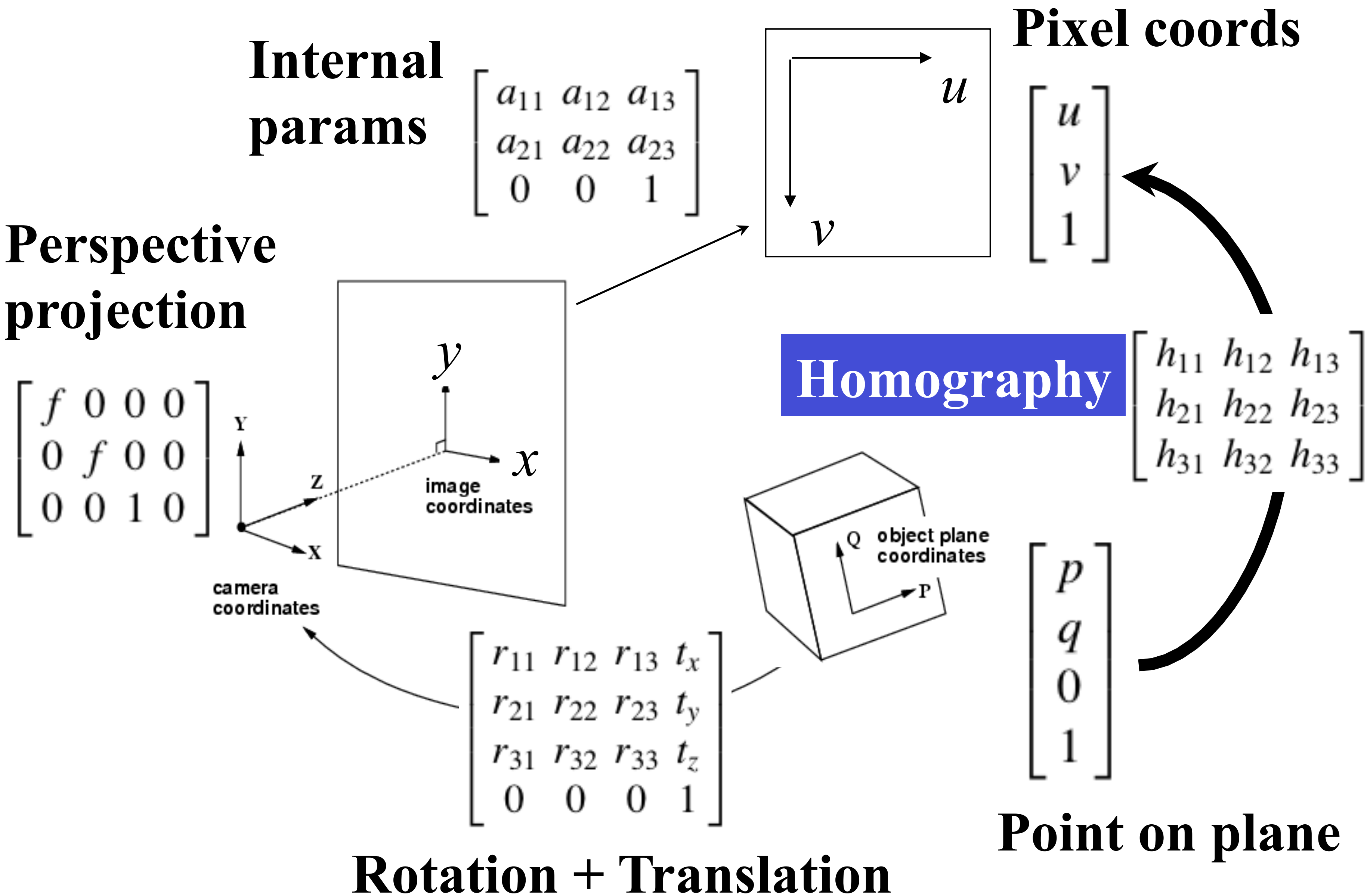
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

Homography H (planar projective transformation)

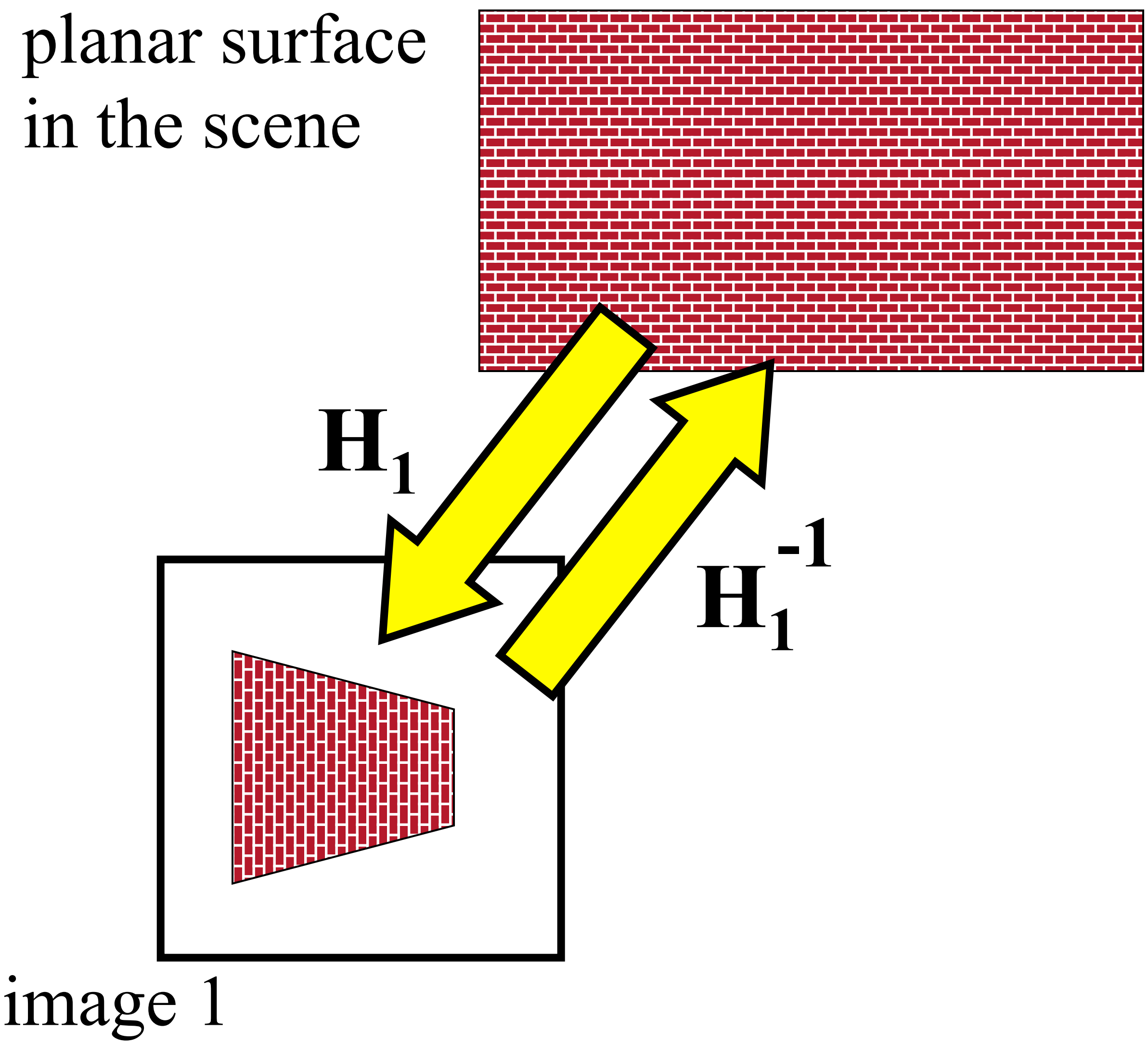
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} h'_{11} & h'_{12} & h'_{13} \\ h'_{21} & h'_{22} & h'_{23} \\ h'_{31} & h'_{32} & h'_{33} \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

It is still a homography (and still invertible)

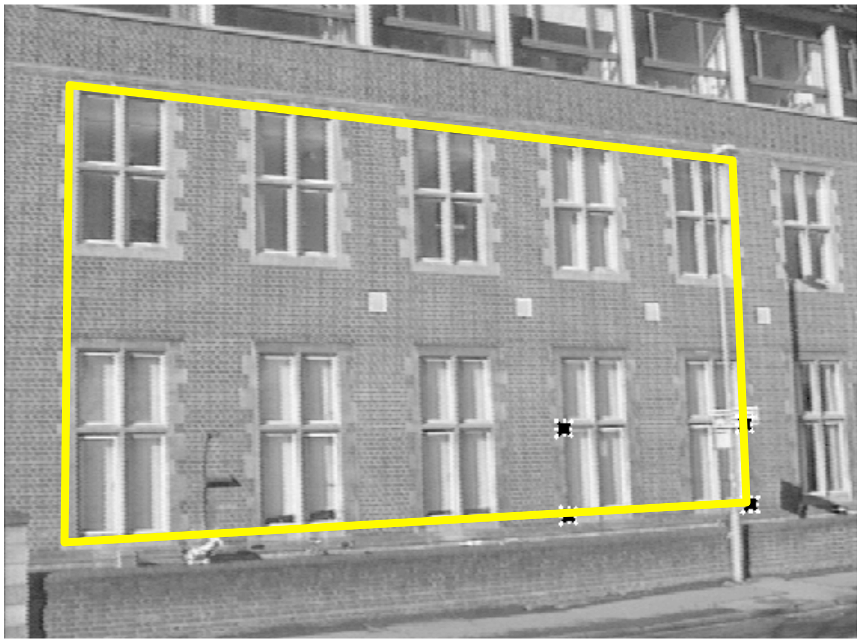
Summary: Planar Projection

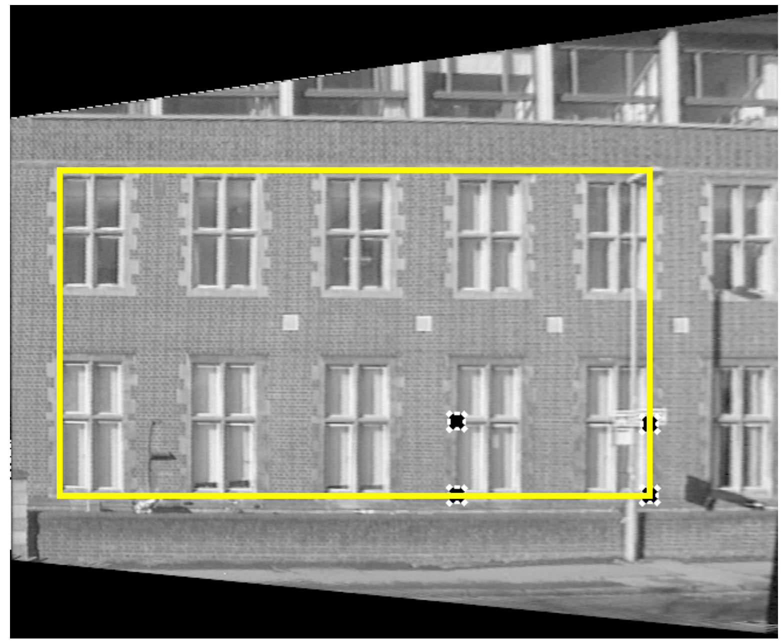


Images of Planar Surfaces



Applying Homographies to Remove Perspective Distortion

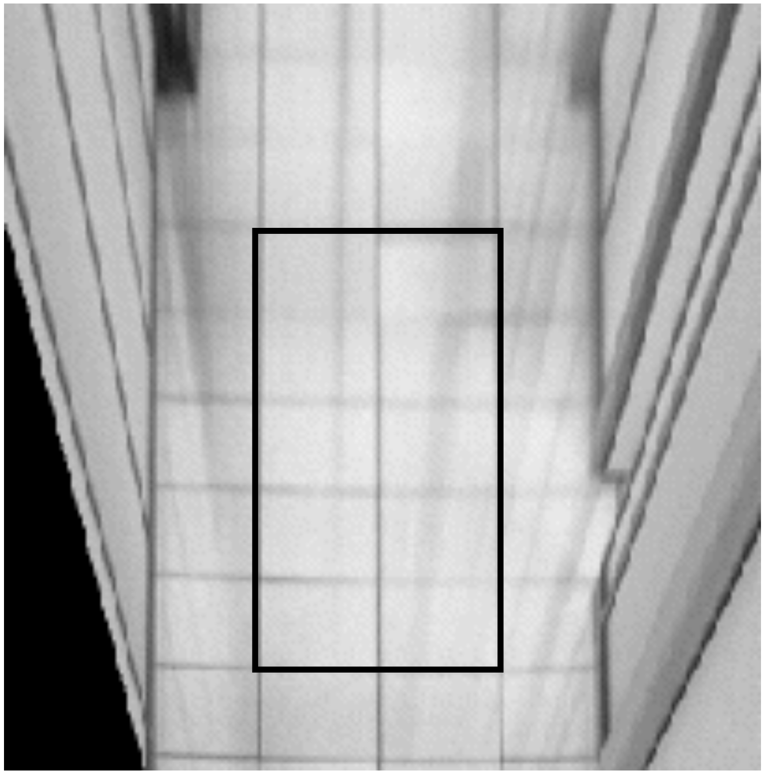




from Hartley & Zisserman

4 point correspondences suffice for
the planar building facade

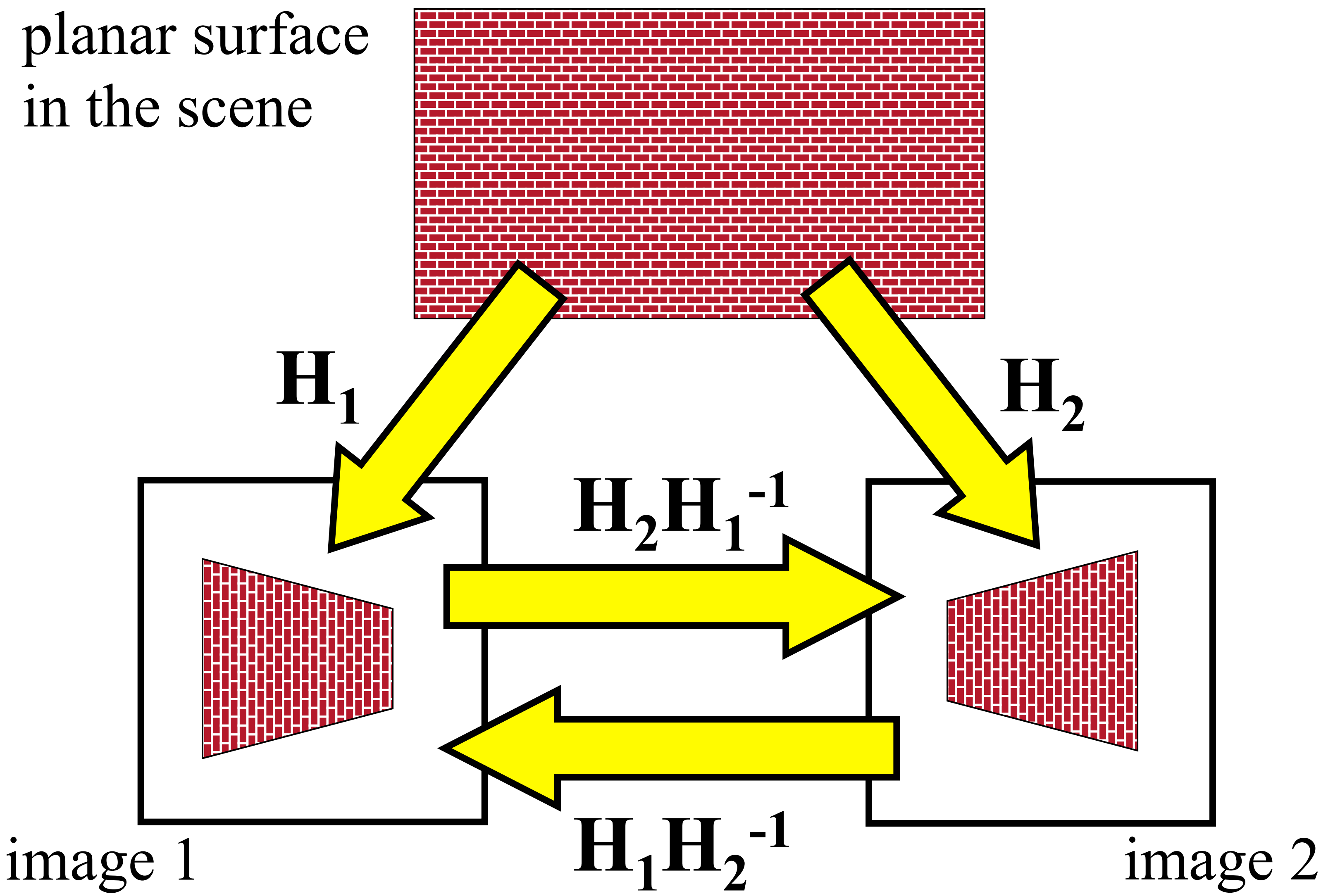
Homographies for Bird's-eye Views



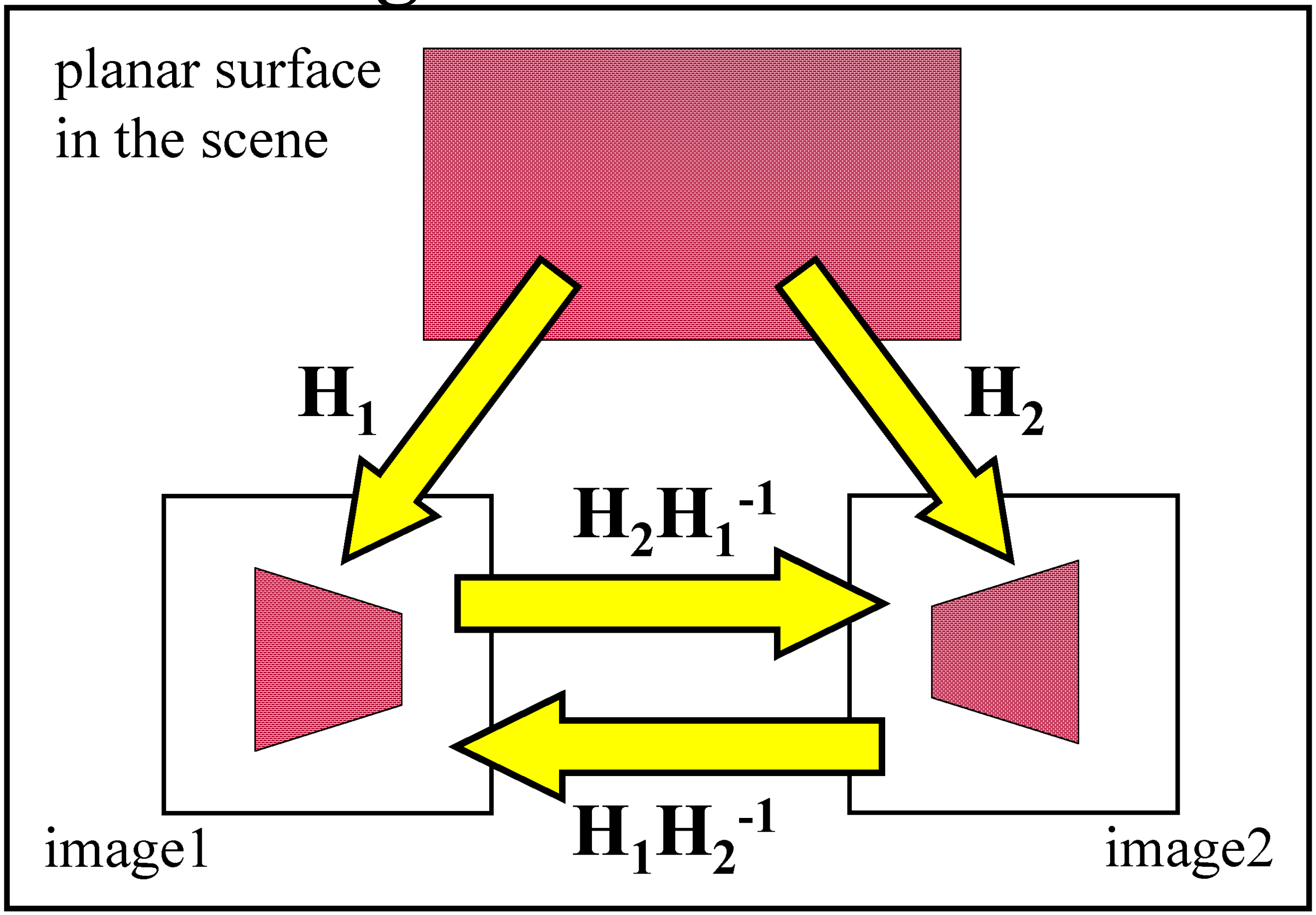
from Hartley & Zisserman

Planewarp demo

Images of Planar Surfaces

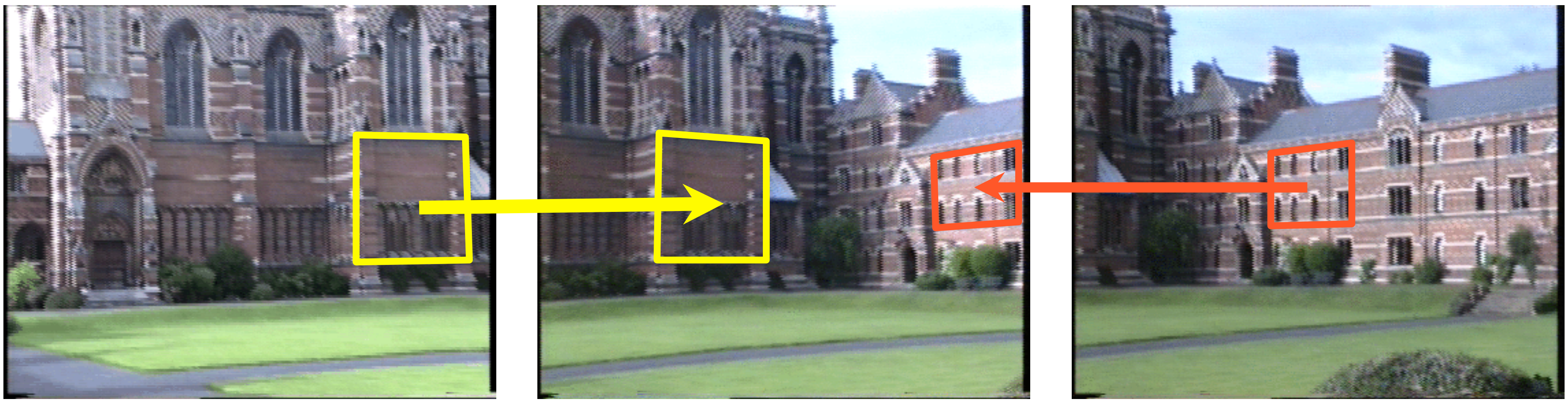


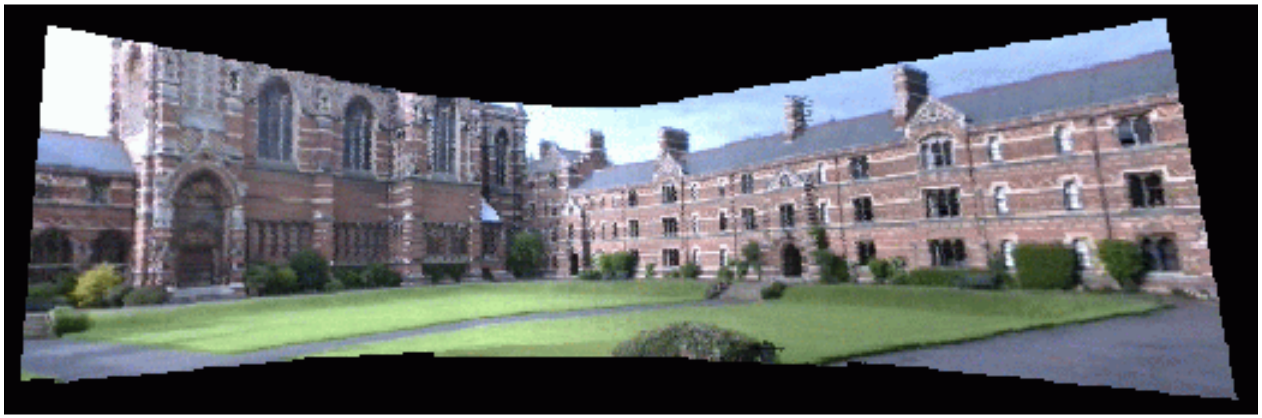
Images of Planar Surfaces



Note that mapping from one image of a plane to another is also just a 3x3 homography H.

Homographies for Mosaicing





from Hartley & Zisserman

Two Practical Issues

How to estimate the homography given four or more point correspondences, e.g. pi, qi -> ui, vi, for i=1,2,3,4... ?

How to (un)warp image pixel values to produce a new picture ?

Estimating a Homography

Matrix Form:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Algebraic Equations:

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

Degrees of Freedom?

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

There are 9 numbers $h_{11},\ldots,h_{33}$, so are there 9 DOF?

No. Note that we can multiply all h_{ij} by nonzero k without changing the equations:

$$x' = \frac{kh_{11}x + kh_{12}y + kh_{13}}{kh_{31}x + kh_{32}y + kh_{33}} \qquad \Rightarrow \qquad x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}$$

$$y' = \frac{kh_{21}x + kh_{22}y + kh_{23}}{kh_{31}x + kh_{32}y + kh_{33}} \qquad \qquad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

Enforcing 8 DOF

One approach: Set $h_{33} = 1$.

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1}$$

Second approach: Impose unit vector constraint

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

Subject to the constraint: $h_{11}^2 + h_{12}^2 + h_{13}^2 + h_{21}^2 + h_{22}^2 + h_{23}^2 + h_{31}^2 + h_{32}^2 + h_{33}^2 = 1$

L.S. using Algebraic Distance

Setting $h_{33} = 1$

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1}$$

Multiplying through by denominator

$$(h_{31}x + h_{32}y + 1)x' = h_{11}x + h_{12}y + h_{13}$$

$$(h_{31}x + h_{32}y + 1)y' = h_{21}x + h_{22}y + h_{23}$$

Rearrange

$$h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx' = x'$$

$$h_{21}x + h_{22}y + h_{23} - h_{31}xy' - h_{32}yy' = y'$$

Algebraic Distance, h_{33}=1 (cont)

$$\begin{array}{ccccc}
 & \mathbf{2N \times 8} & \mathbf{8 \times 1} & & \mathbf{2N \times 1} \\
\begin{array}{l} \textbf{Point 1} \\ \\ \textbf{Point 2} \\ \\ \textbf{Point 3} \\ \\ \textbf{Point 4} \\ \\ \end{array} &
\begin{bmatrix}
x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 \\
0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 \\
x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x'_2 & -y_2x'_2 \\
0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y'_2 & -y_2y'_2 \\
x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3x'_3 & -y_3x'_3 \\
0 & 0 & 0 & x_3 & y_3 & 1 & -x_3y'_3 & -y_3y'_3 \\
x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4x'_4 & -y_4x'_4 \\
0 & 0 & 0 & x_4 & y_4 & 1 & -x_4y'_4 & -y_4y'_4
\end{bmatrix} &
\begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} & = &
\begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \\ x'_4 \\ y'_4 \end{bmatrix} \\
\textbf{additional points} & \vdots & & & \vdots
\end{array}$$

Algebraic Distance, h_{33}=1 (cont)

Linear equations

$$\underset{2N\times 8}{A}\;\underset{8\times 1}{h} = \underset{2N\times 1}{b}$$

Solve:

$$\underset{8\times 2N}{A^T}\;\underset{2N\times 8}{A}\;\underset{8\times 1}{h} = \underset{8\times 2N}{A^T}\;\underset{2N\times 1}{b}$$

$$\overbrace{(A^T\;A)}^{8\times 8}\;\underset{8\times 1}{h} = \overbrace{(A^T\;b)}^{8\times 1}$$

$$h = (A^T\;A)^{-1}\,(A^T\;b)$$

Matlab: $h = A \backslash b$

Two Practical Issues

How to estimate the homography given four or more point correspondences, e.g. pi, qi -> ui, vi, for i=1,2,3,4... ?

How to (un)warp image pixel values to produce a new picture ?

Warping & Bilinear Interpolation

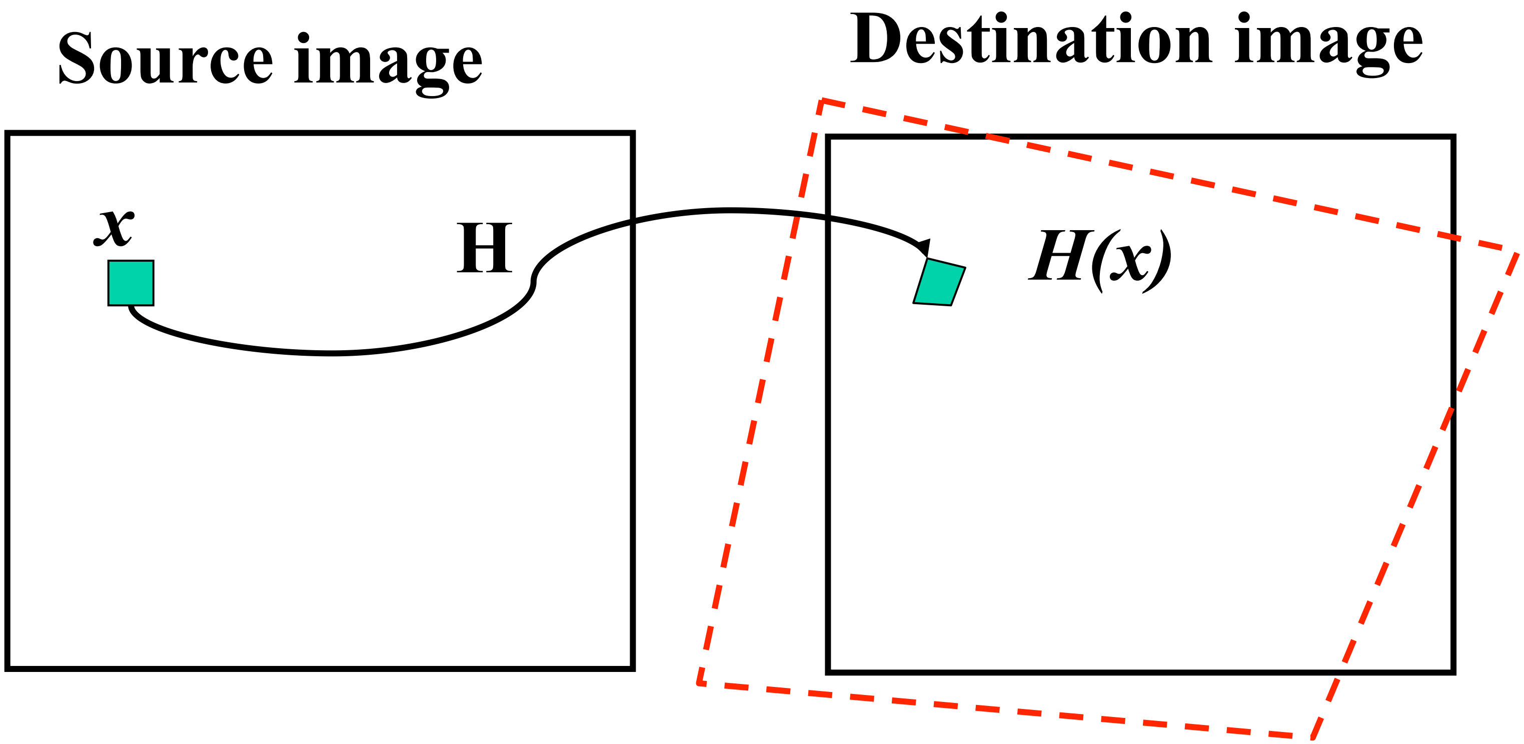
Given a homography between two images, (coordinate systems) we want to "warp" one image into the coordinate system of the other.

We will call the coordinate system where we are mapping from the "source" image

We will call the coordinate system we are mapping to the "destination" image.

Robert Collins
CMPEN454

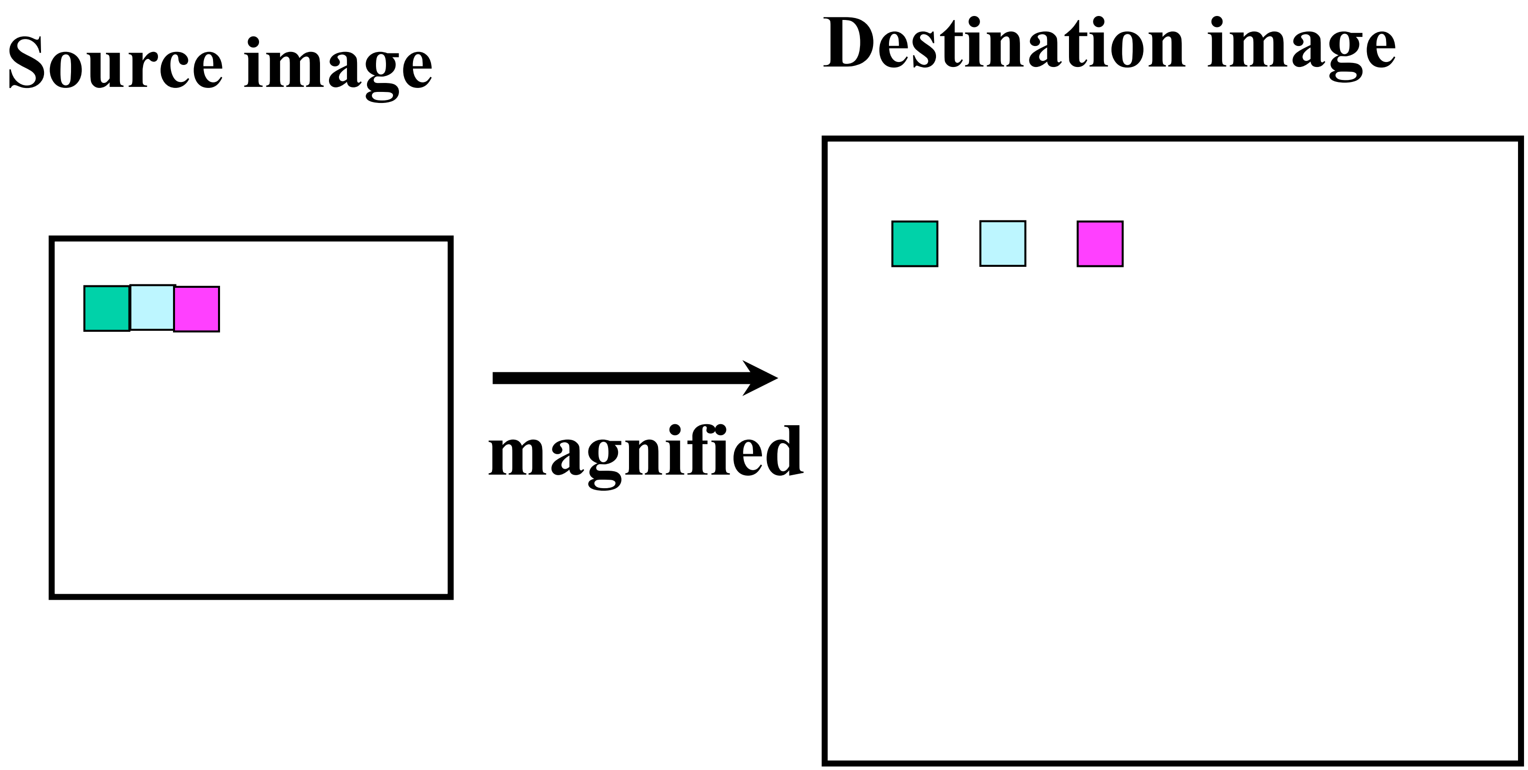
Forward Warping



- For each pixel *x* in the source image
- Determine where it goes as *H(x)*
- Color the destination pixel

Problems?

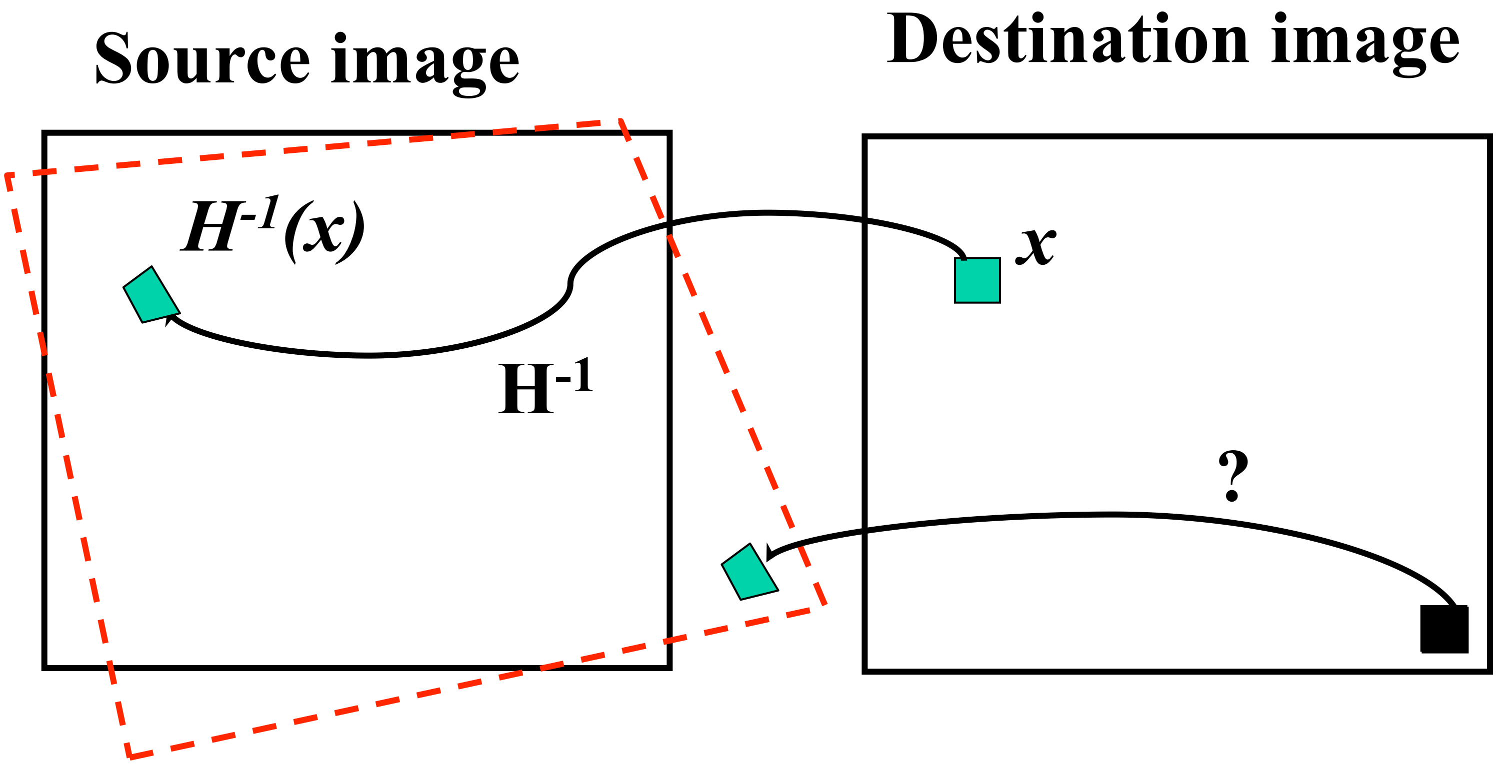
Forward Warping Problem



Can leave gaps!

Robert Collins
CMPEN454

Backward Warping (No gaps)

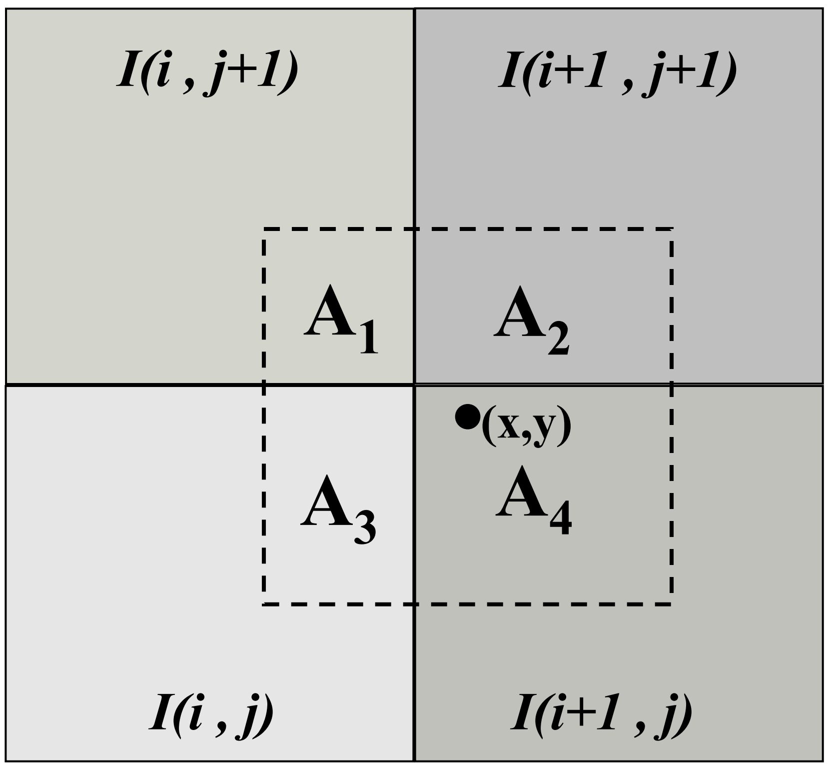


- For each pixel x in the destination image
- Determine where it comes from as $H^{-1}(x)$
- Get color from that location

Bilinear interpolation

What do we mean by "get color from that location"?

Consider grey values. **What is intensity at (x,y)?**



Bilinear Interpolation:

Weighted average

$$I(x,y) = A3*I(i,j) + A4*I(i+1,j) + A2*I(i+1,j+1) + A1*I(i,j+1)$$

Matlab's Interp2

interp2 in MATLAB does image warping, using bilinear interpolation

```
Sample code:

im = double(imread('cameraman.tif'));
size(im)
ans =
   256   256

[xi, yi] = meshgrid(1:256, 1:256);

h = [put your 3x3 homography matrix here];

h = inv(h);   %TAKE INVERSE FOR USE WITH INTERP2
xx = (h(1,1)*xi+h(1,2)*yi+h(1,3))./(h(3,1)*xi+h(3,2)*yi+h(3,3));
yy = (h(2,1)*xi+h(2,2)*yi+h(2,3))./(h(3,1)*xi+h(3,2)*yi+h(3,3));
foo = uint8(interp2(im,xx,yy));
figure(1); imshow(foo)
```