Mosaicing

many slides are from Collins, CSE486 (Penn State) but some are from Efros, Seitz, Szeliski, Criminisci, Brown and Lowe

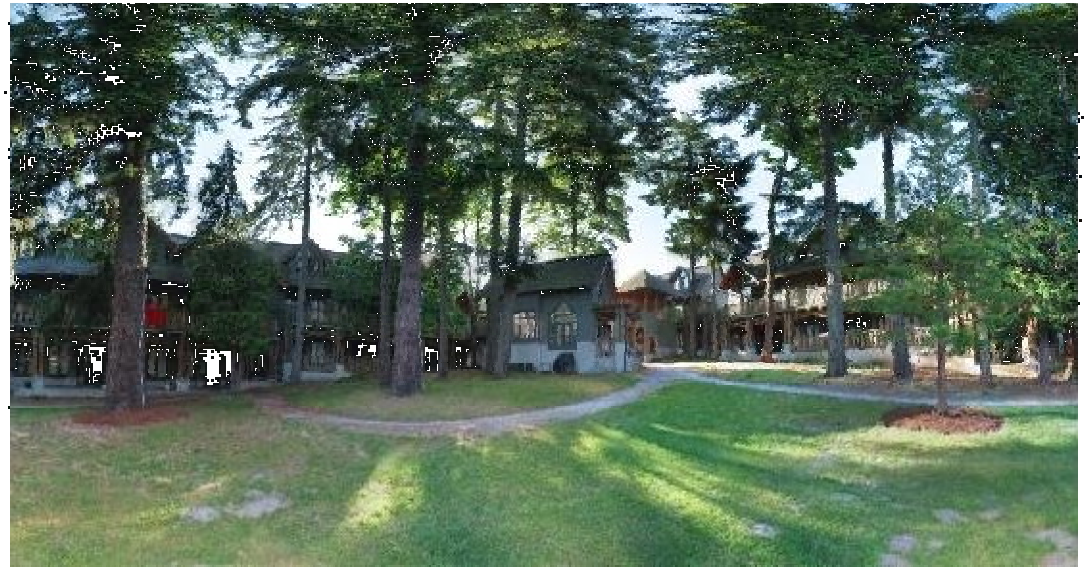
Why Make Mosaics?

- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°

Slide from Brown & Lowe

Why Make Mosaics?

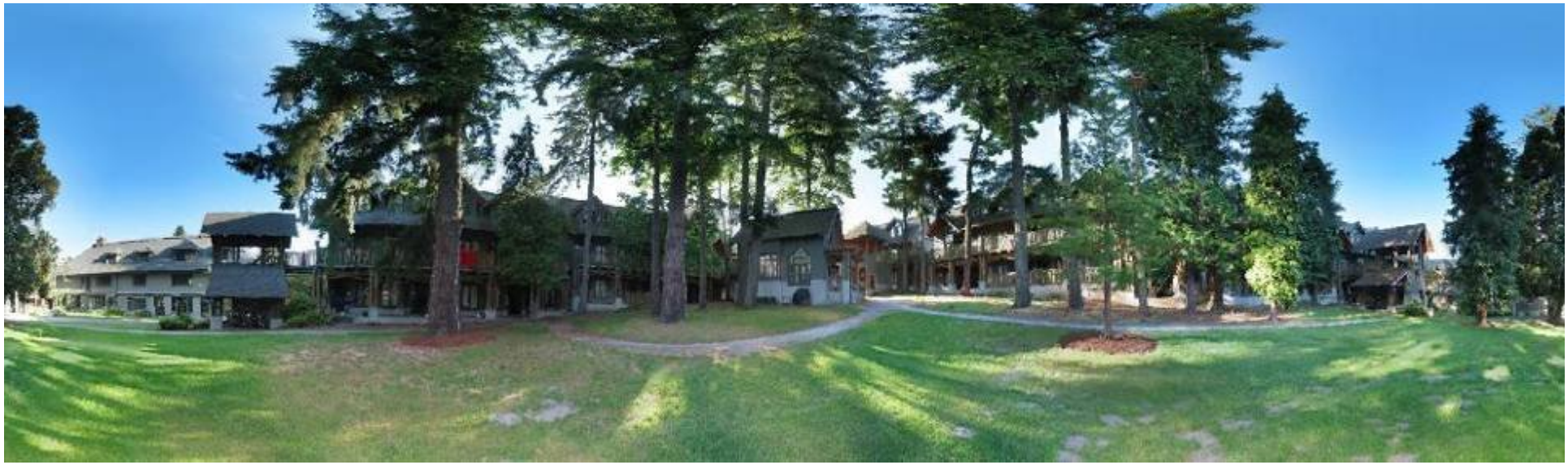
- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°
  - Human FOV = 200 x 135°



Slide from Brown & Lowe

Why Make Mosaics?

- Are you getting the whole picture?
  - Compact Camera FOV = 50 x 35°
  - Human FOV = 200 x 135°
  - Panoramic Mosaic = 360 x 180°



Slide from Brown & Lowe

Mosaics: stitching images together

virtual wide-angle camera

Efros

Note: Alignment uses homographies!

Translations are not enough to align the images

left on top

right on top

Efros

Overview

what images are related by a homography:
  images of a planar surface (arbitrary perspective cameras)
  images taken from same optical center (rotating, zooming)

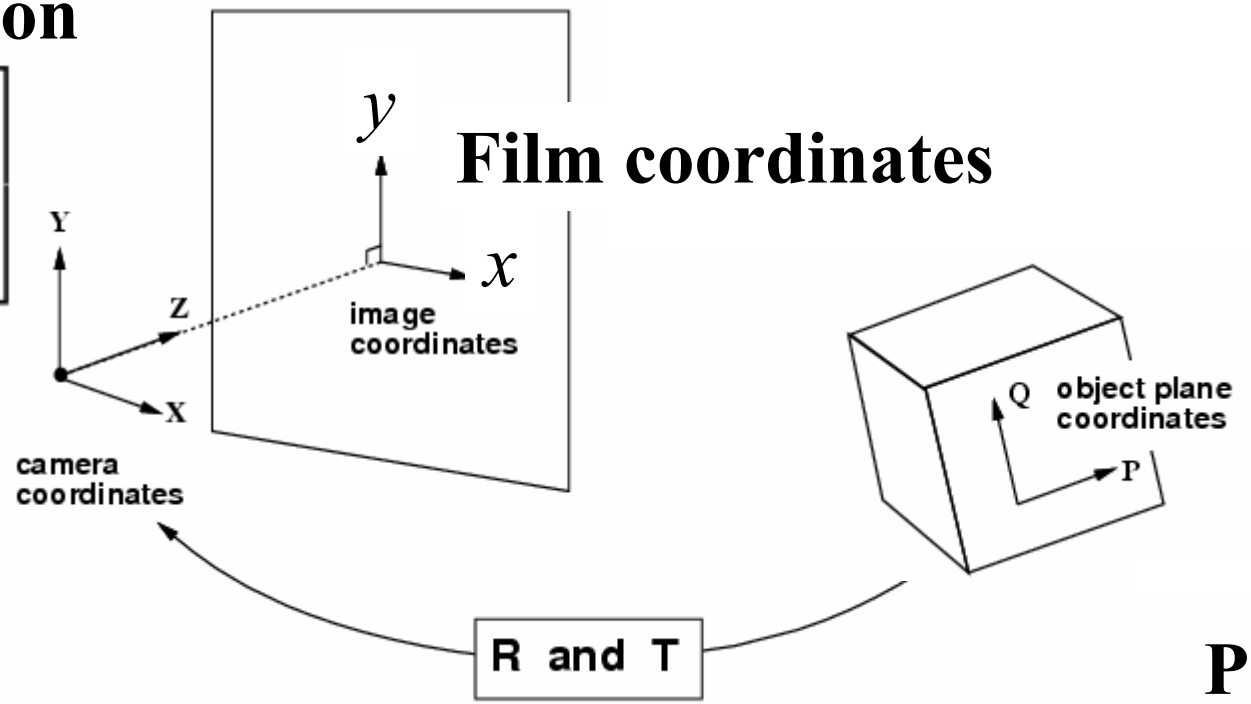
given matches:
  estimating homography, warping, blending

computing matches
  Harris corners + RANSAC
  SIFT keys / Autostitch

Projection of Points on Planar Surface

Perspective projection

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



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

Point on plane

Rotation + Translation

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

Projection of Planar Points

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

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

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

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)

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!

Even True for Uncalibrated Cameras

Internal params

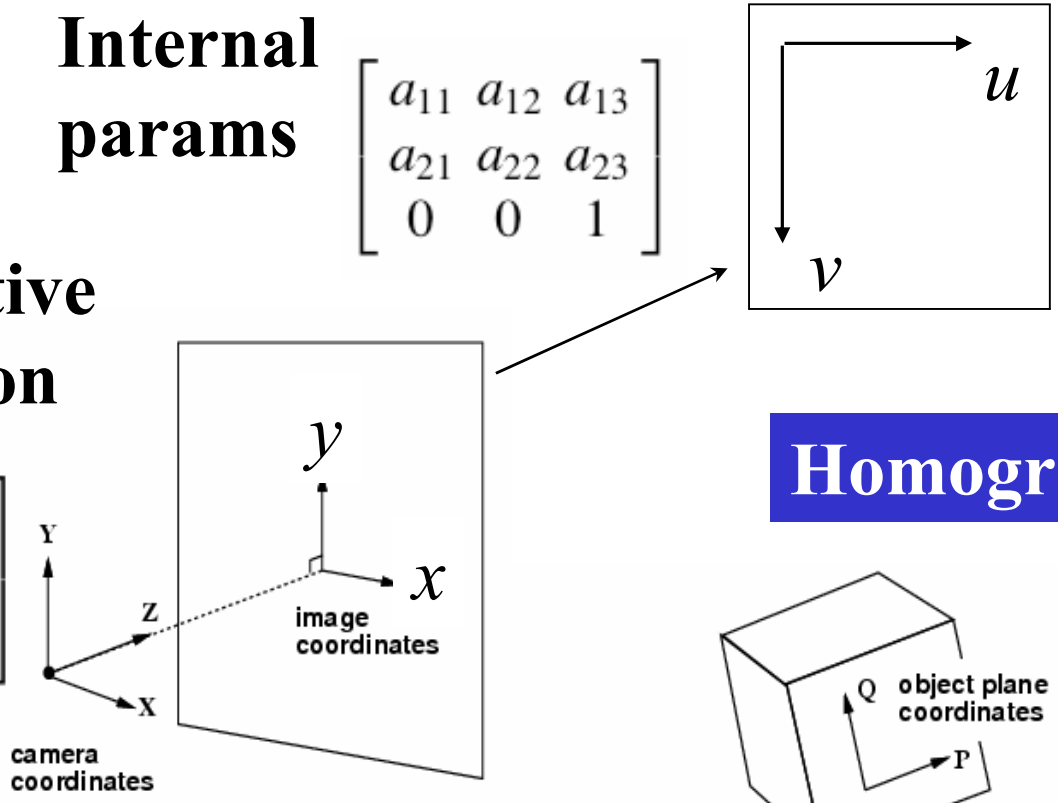
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

Pixel coords

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Perspective projection

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



Homography

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

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

Rotation + Translation

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

Point on plane

Even True for Uncalibrated Cameras

Internal params

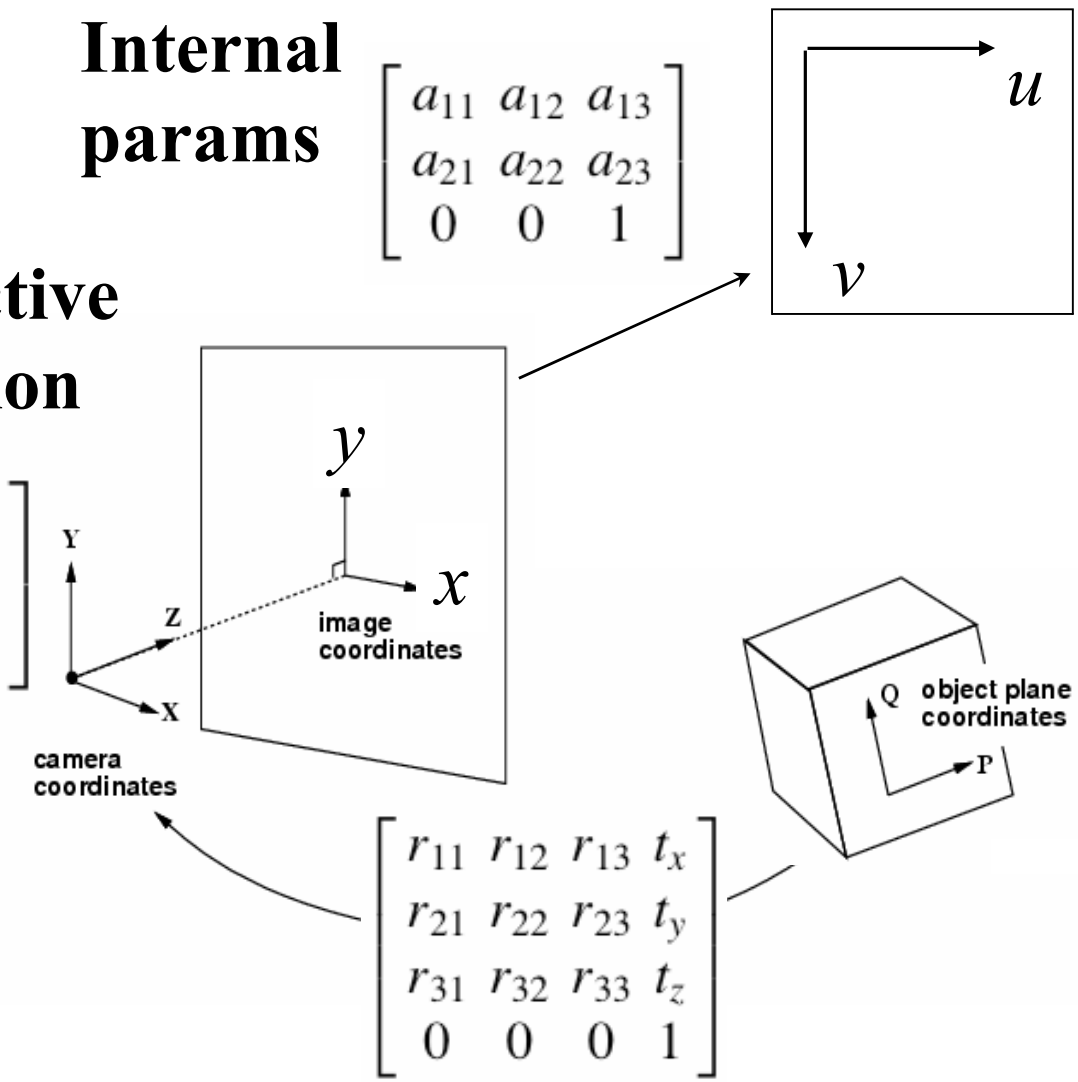
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

Pixel coords

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Perspective projection

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



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

Rotation + Translation

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

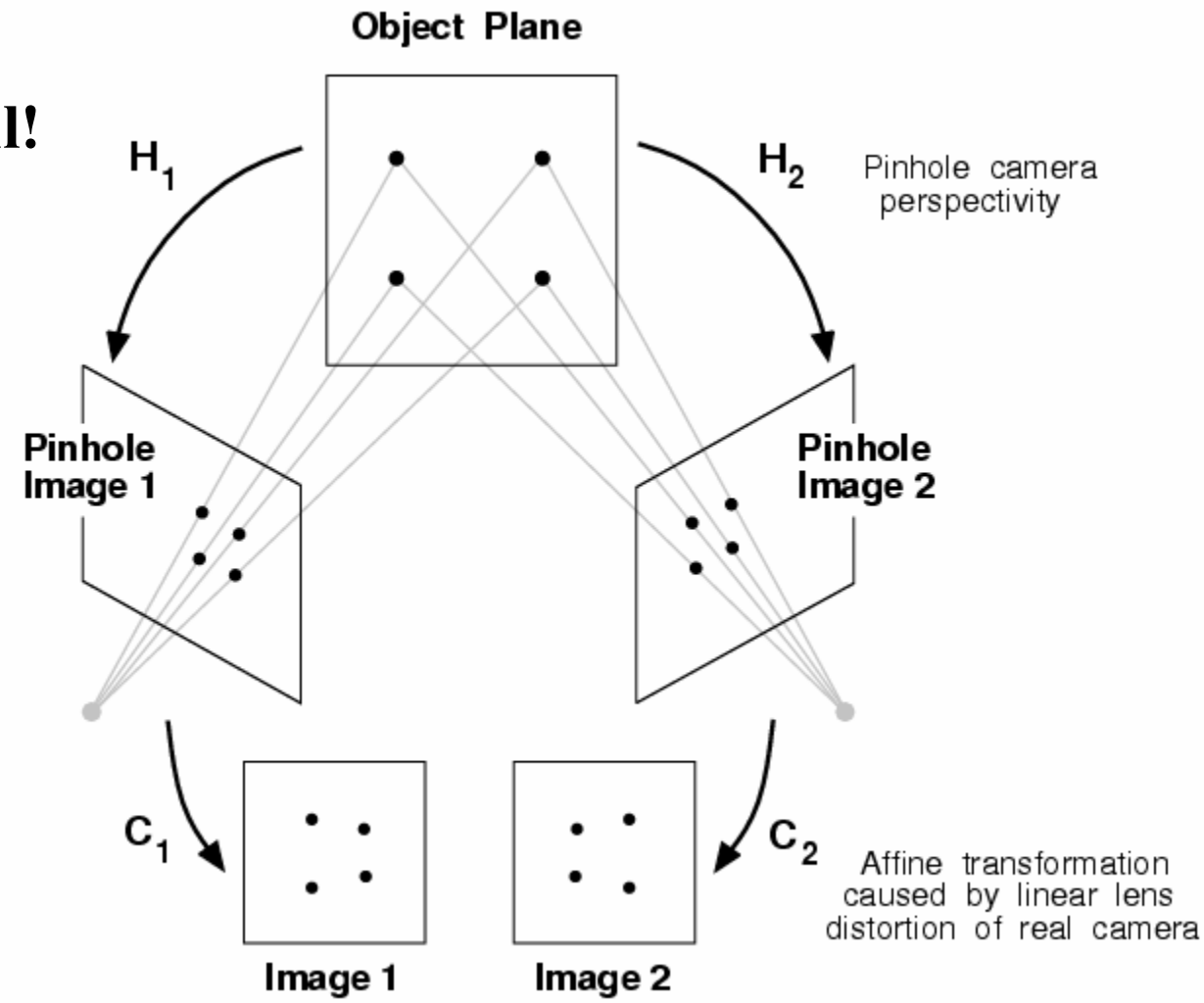
Point on plane

H

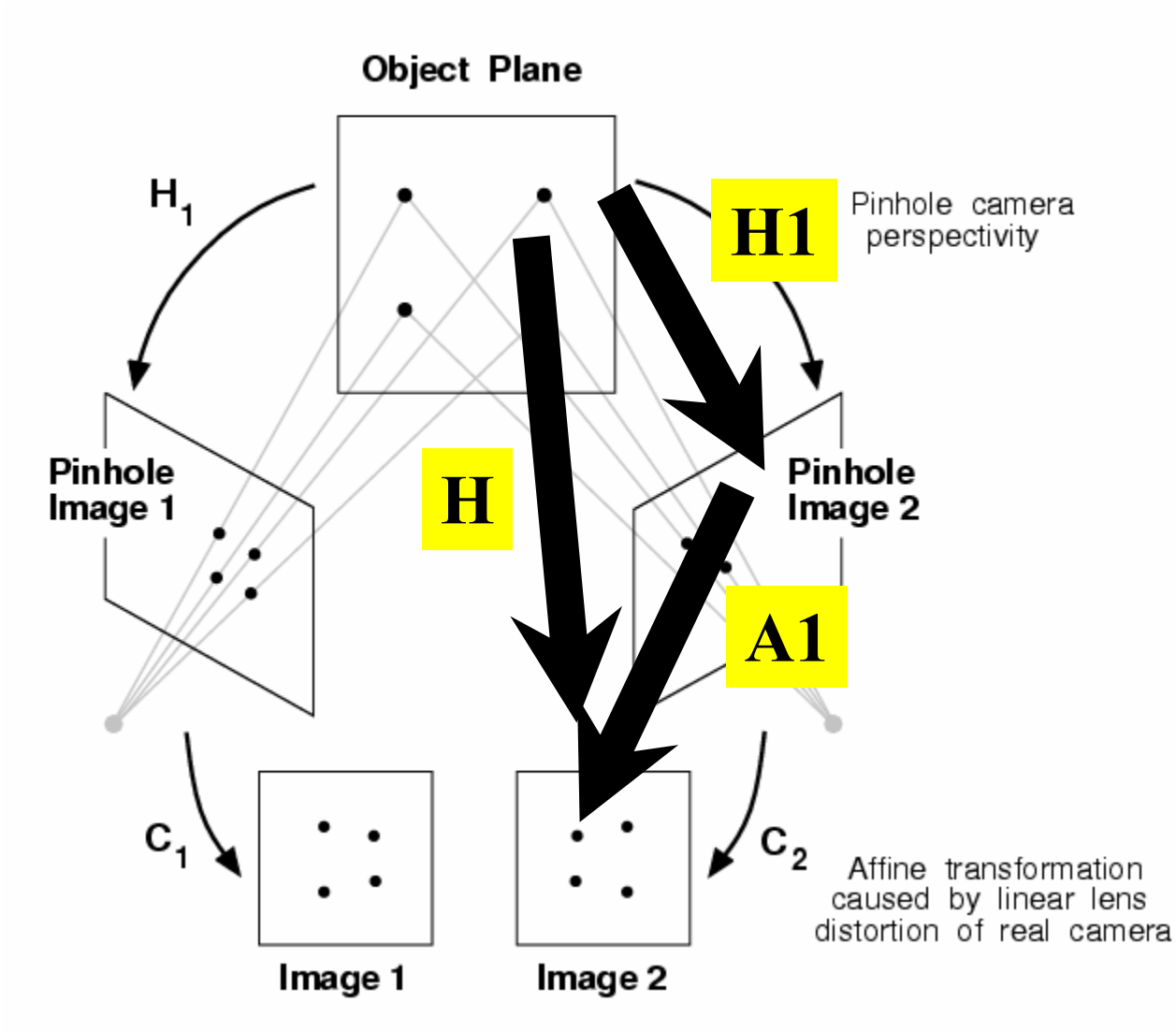
H^{-1}

Planar Projection Diagram

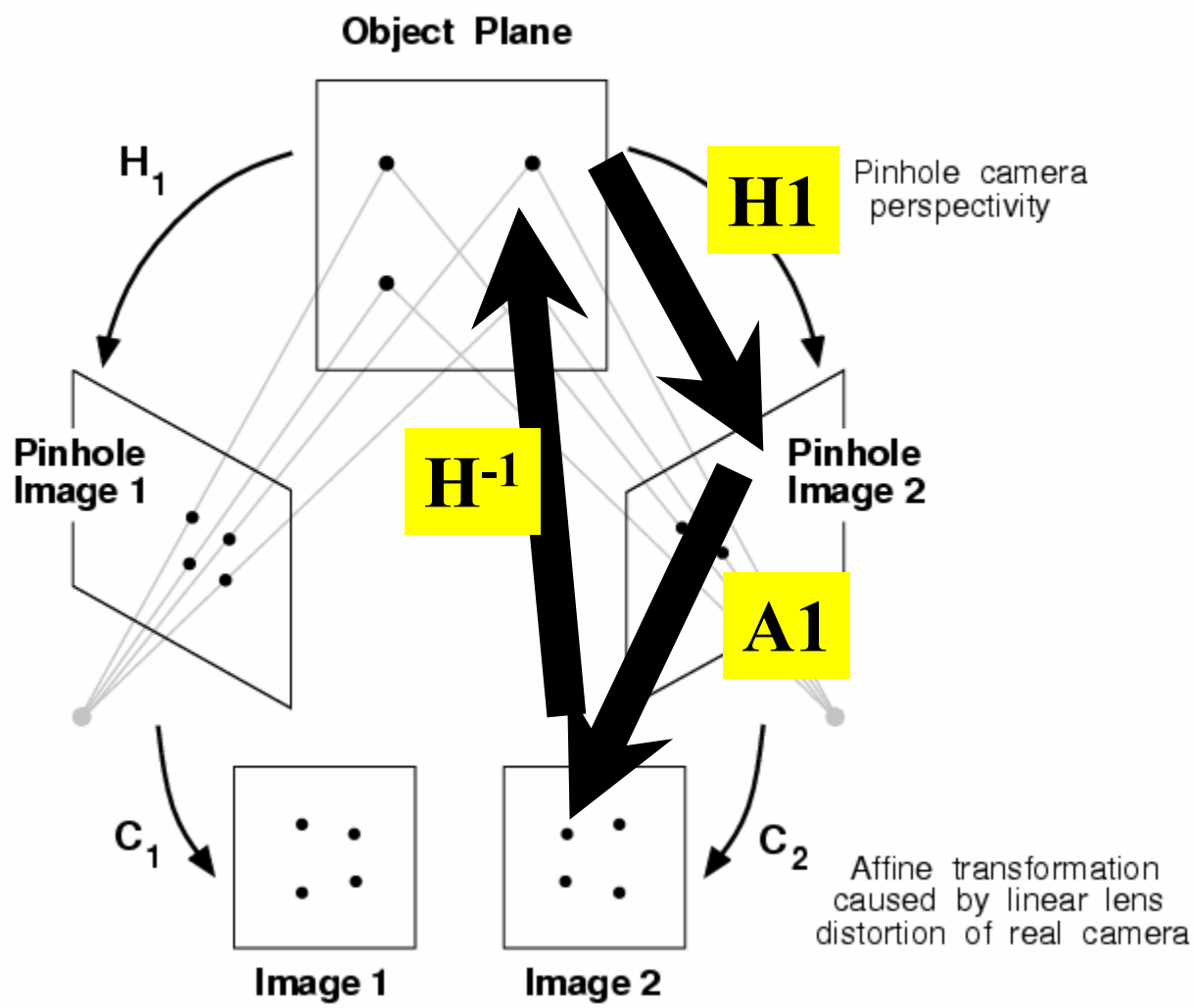
Here's where transformation groups get useful!



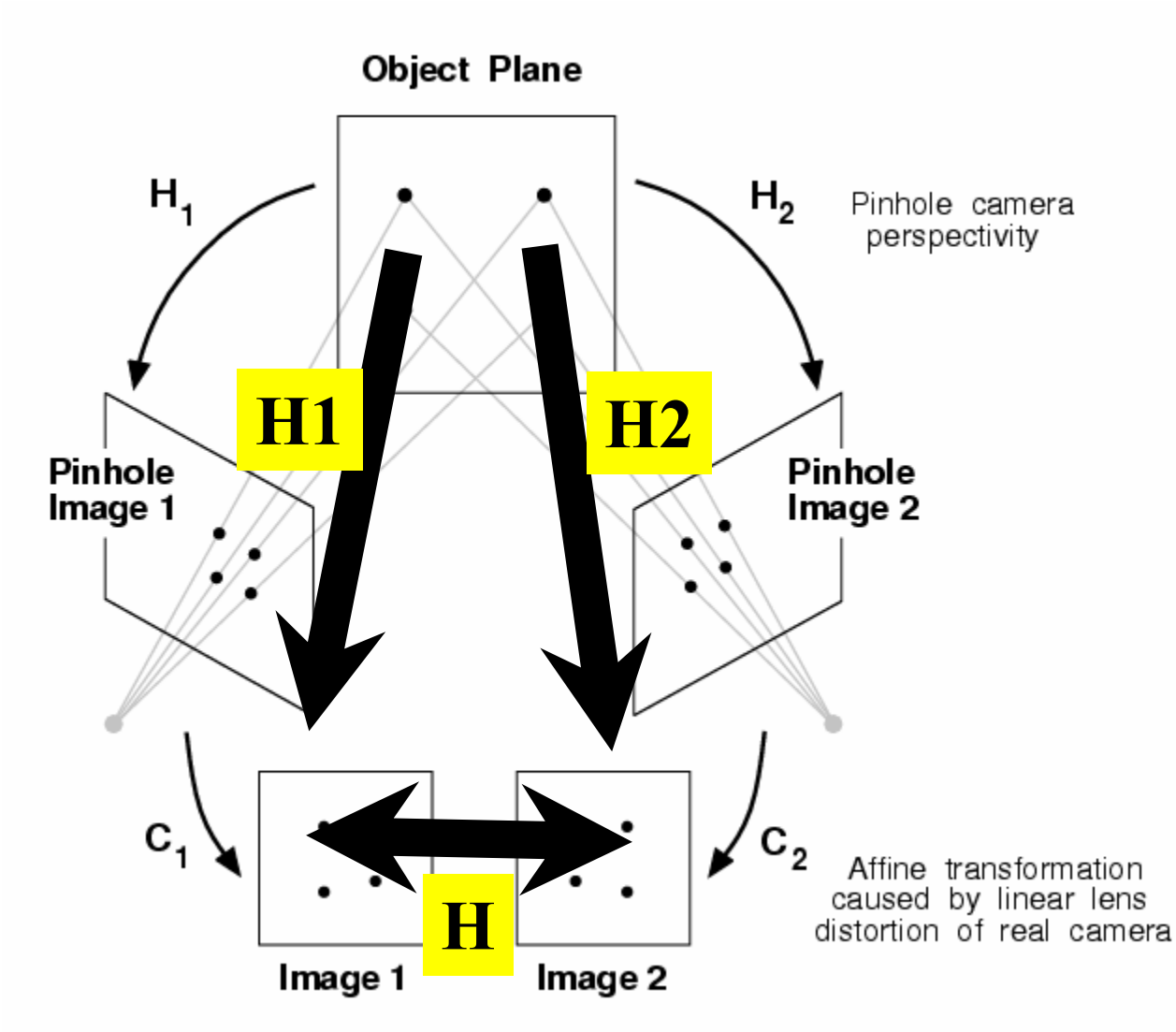
General Planar Projection



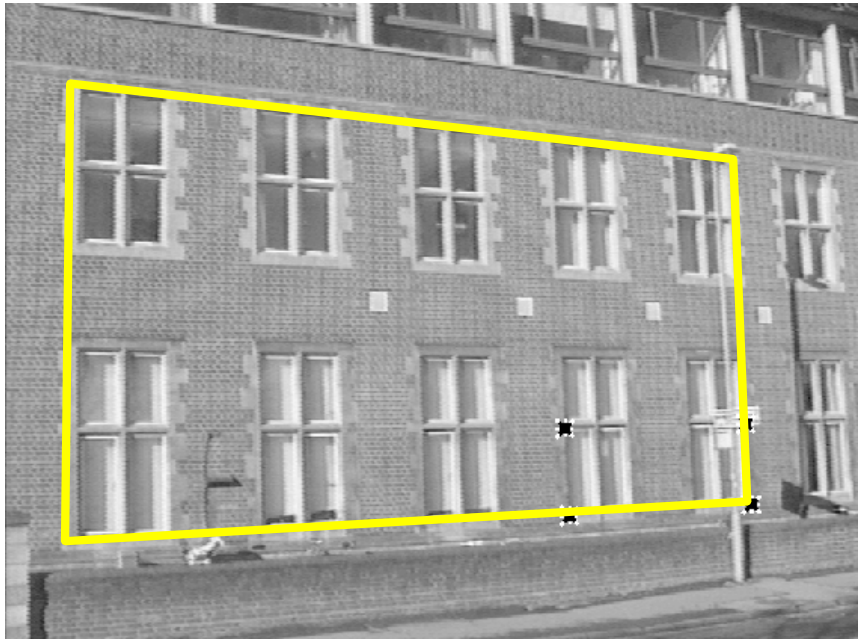
General Planar Projection

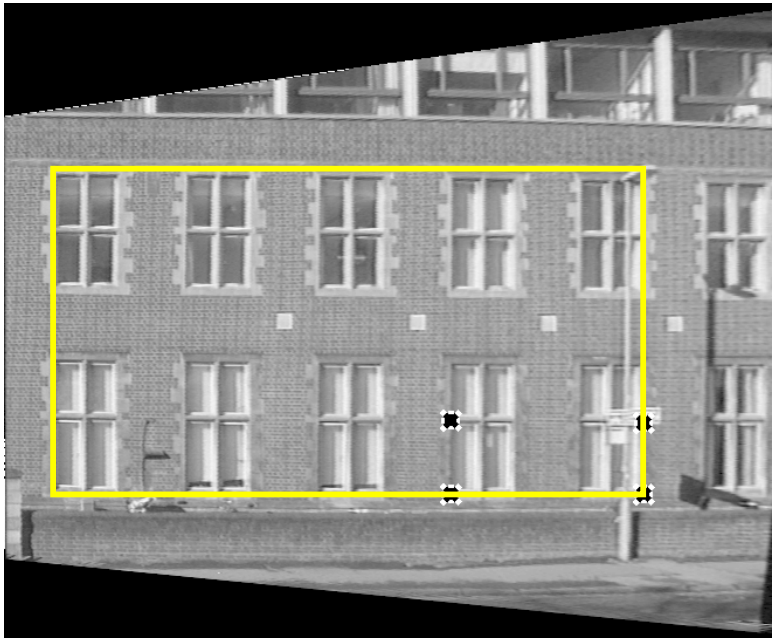


General Planar Projection



Applying Homographies to Remove Perspective Distortion

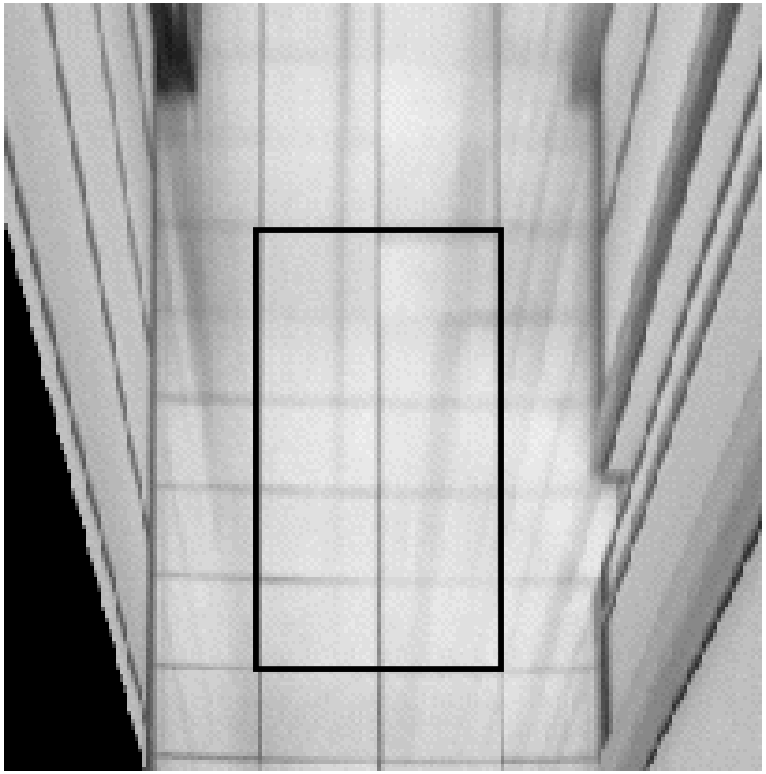




from Hartley & Zisserman

4 point correspondences suffice for the planar building facade

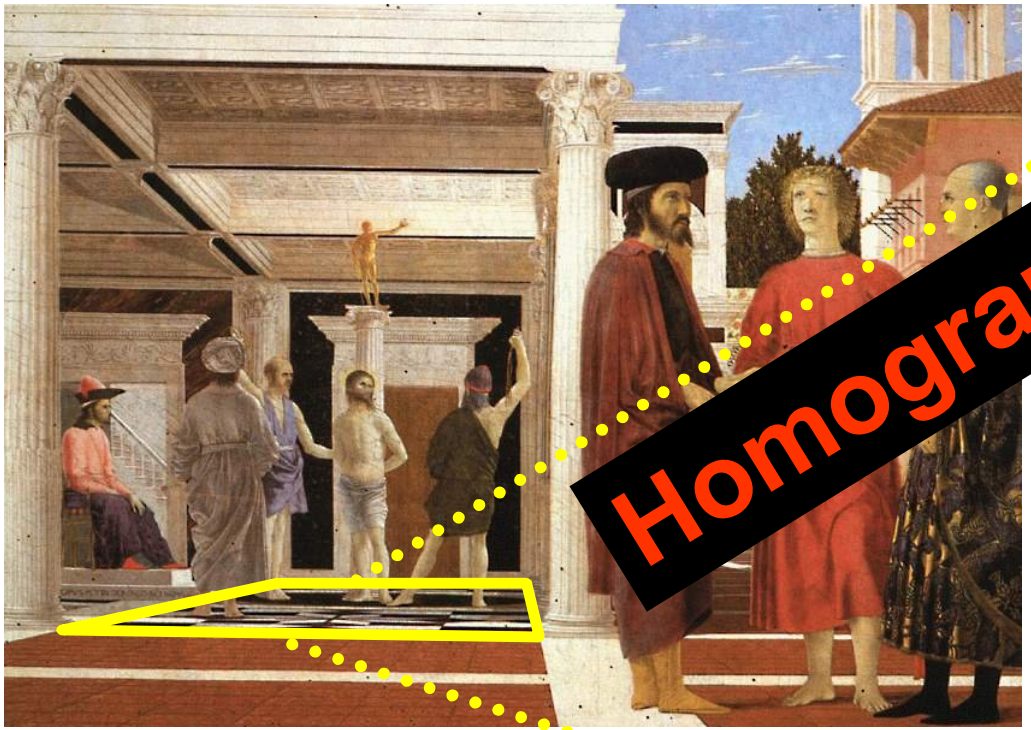
Homographies for Rectification



from Hartley & Zisserman

Analysing patterns and shapes

What is the shape of the b/w floor pattern?



Homography

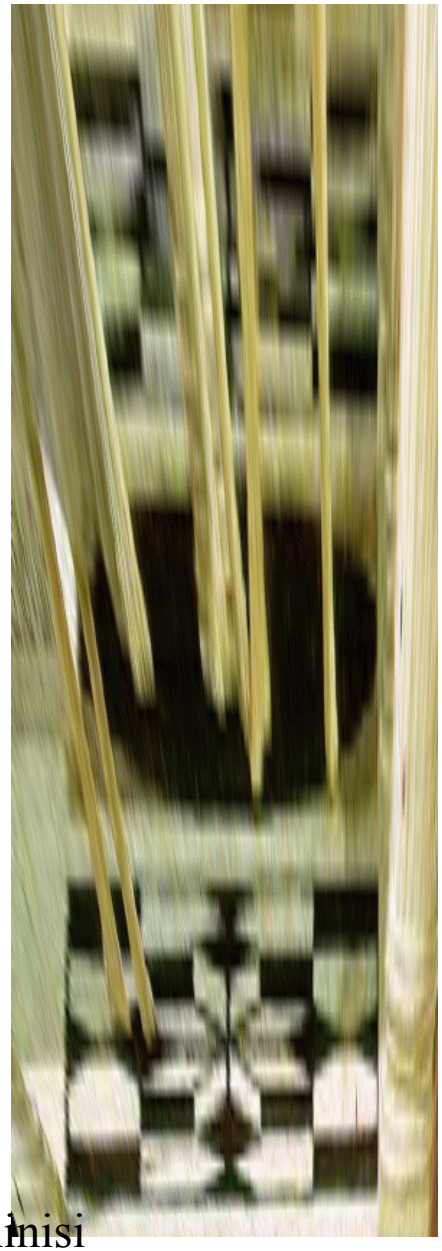
The floor (enlarged)

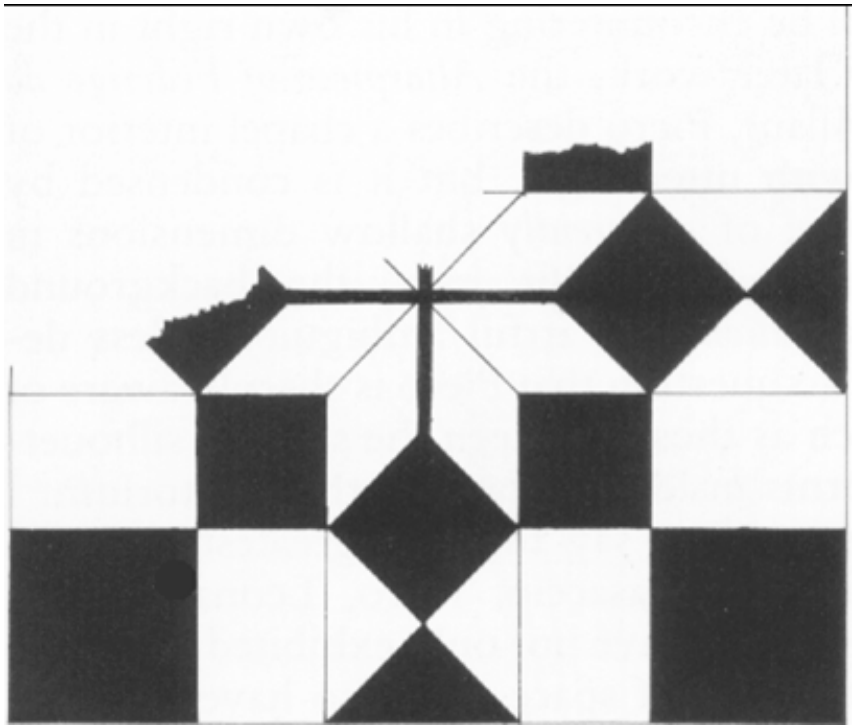
Automatically rectified floor

Slide from Criminisi

Analysing patterns and shapes

Automatic rectification



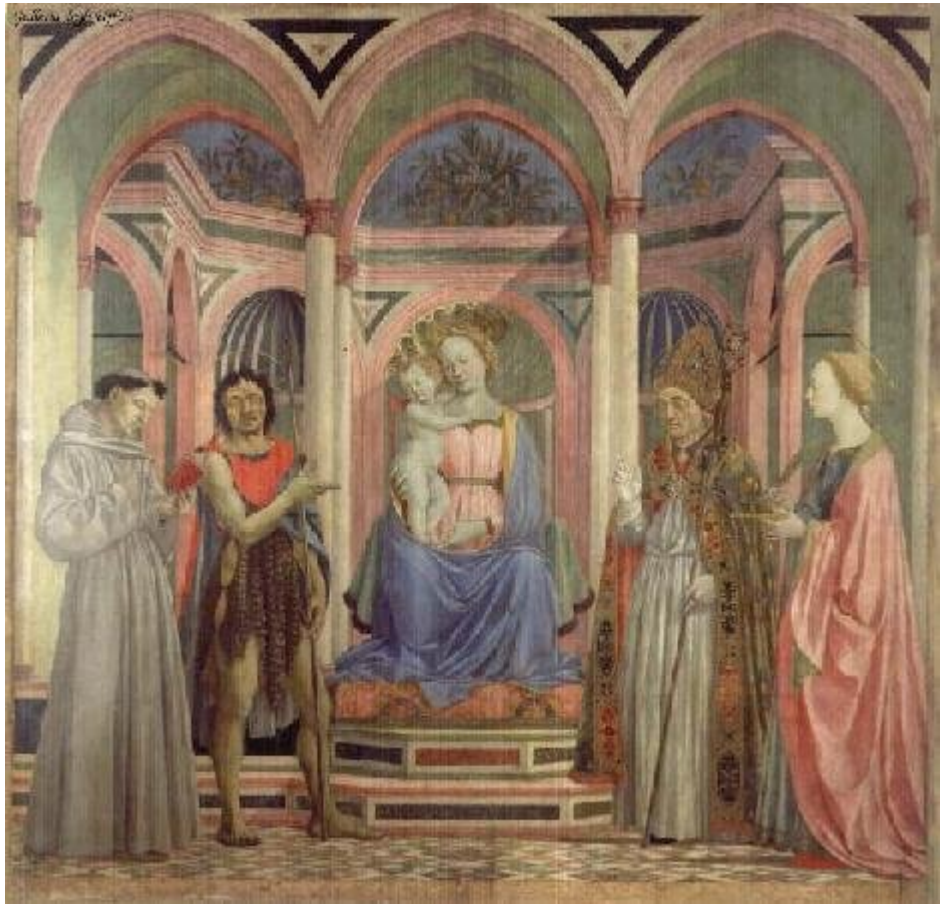


From Martin Kemp *The Science of Art*
(manual reconstruction)

2 patterns have been discovered !

Slide from Criminisi

Analysing patterns and shapes



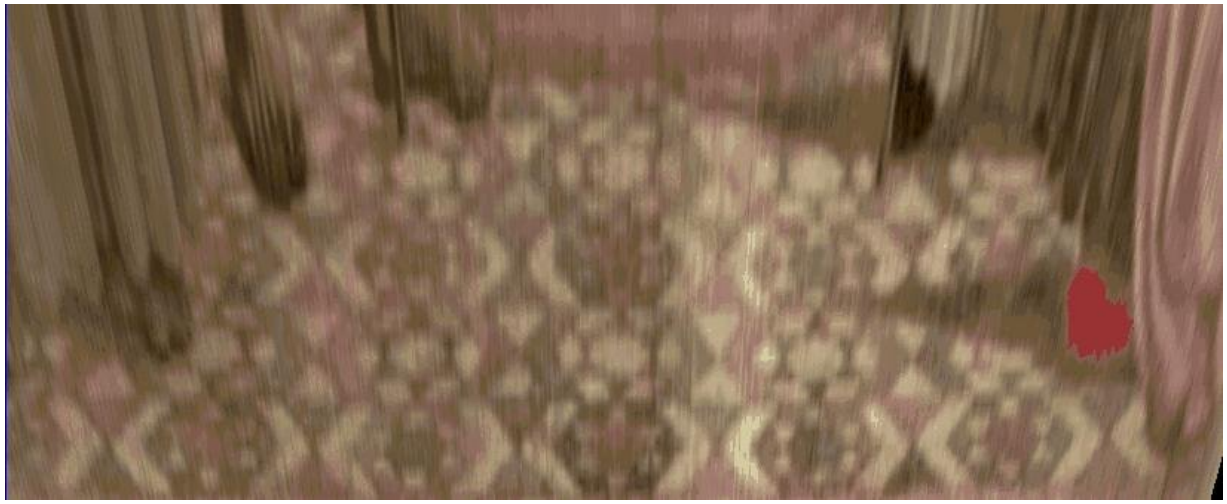
St. Lucy Altarpiece, **D. Veneziano**

Slide from Criminisi

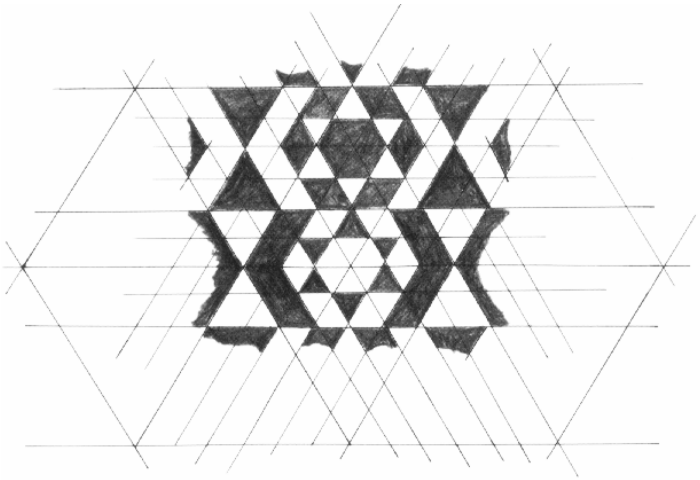
What is the (complicated) shape of the floor pattern?

Automatically rectified floor

Analysing patterns and shapes



Automatic rectification



From Martin Kemp, *The Science of Art* *(manual reconstruction)*

Slide from Criminisi

Julian Beever: Manual Homographies

http://users.skynet.be/J.Beever/pave.htm

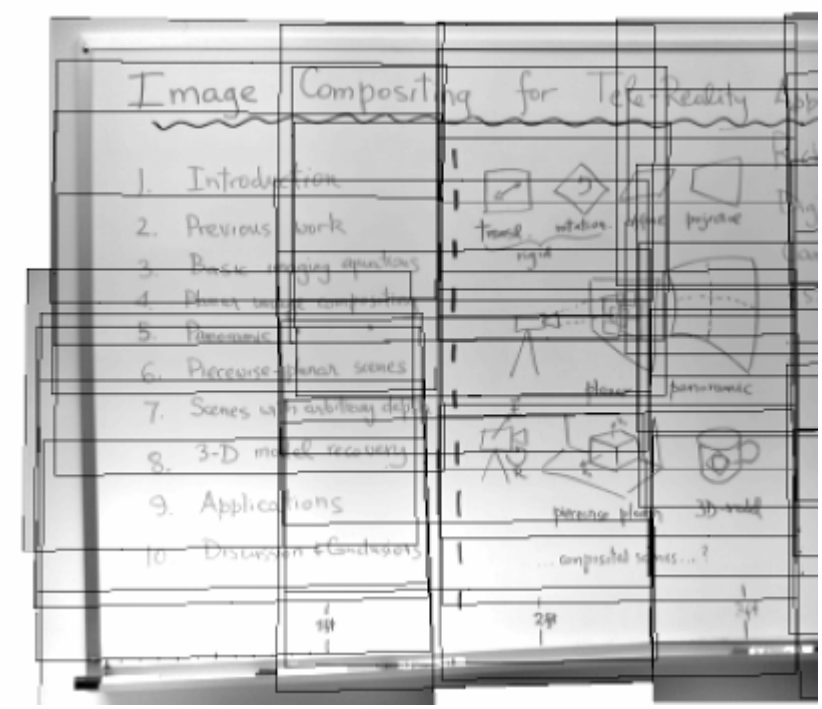
Fun with homographies

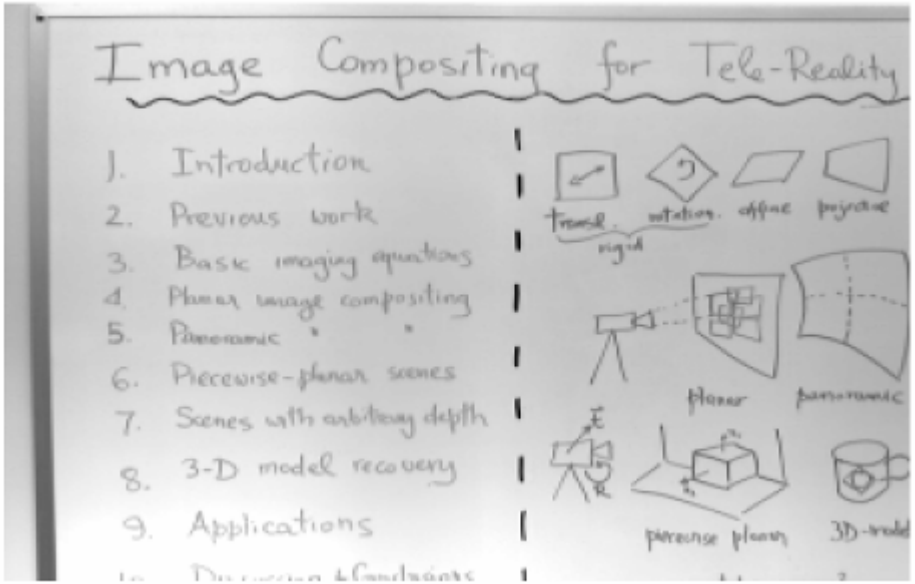
Original image

St.Petersburg
photo by A. Tikhonov

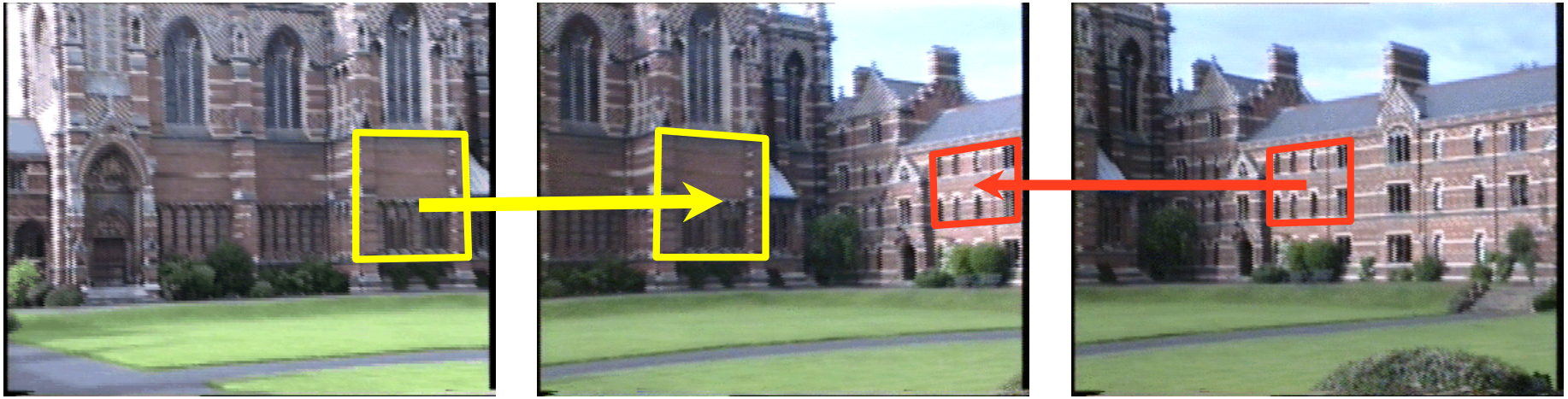
Rectifying different building faces

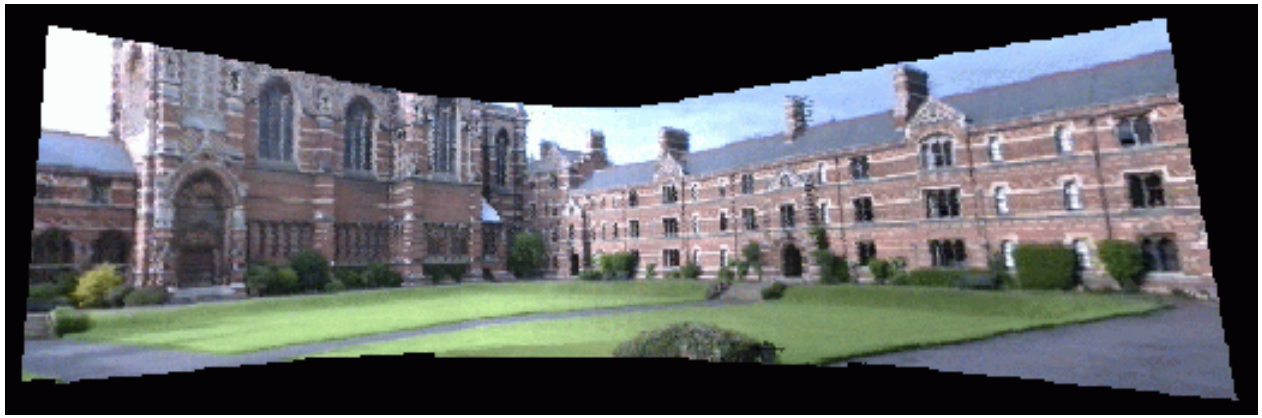
Planar mosaic





Homographies for Mosaicing





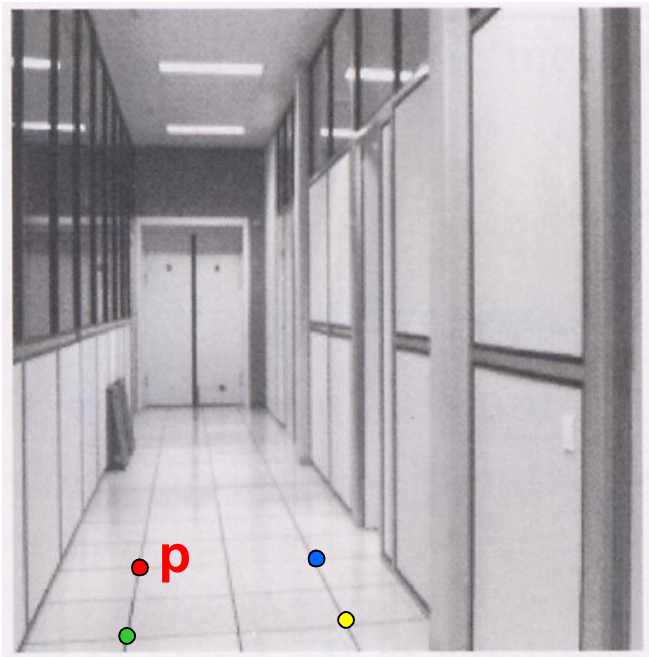
from Hartley & Zisserman

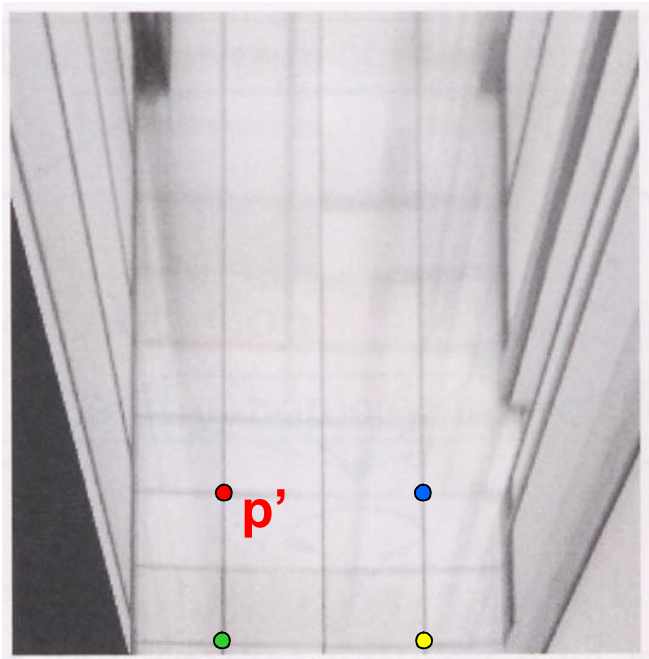
Two Practical Issues

How to estimate the homography given four or more point correspondences

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

Estimating a Homography





To unwarp (rectify) an image

- Find the homography **H** given a set of **p** and **p'** pairs
- How many correspondences are needed?
- Tricky to write H analytically, but we can <u>solve</u> for it!
  - Find such H that "best" transforms points p into p'
  - Use least-squares!

Estimating a Homography

Matrix Form:

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

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:

$$x' = \frac{kh_{11}x + kh_{12}y + kh_{13}}{kh_{31}x + kh_{32}y + kh_{33}} \qquad y' = \frac{kh_{21}x + kh_{22}y + kh_{23}}{kh_{31}x + kh_{32}y + kh_{33}}$$

$$\Rightarrow$$

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \qquad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

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)

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

additional points

$$
\begin{array}{ccc}
\vdots & & \vdots
\end{array}
$$

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

Linear equations

$$\overset{2Nx8}{\mathbf{A}} \quad \overset{8x1}{\mathbf{h}} \quad = \quad \overset{2Nx1}{\mathbf{b}}$$

Solve:

$$\overset{8x2N}{\mathbf{A}^T} \quad \overset{2Nx8}{\mathbf{A}} \quad \overset{8x1}{\mathbf{h}} \quad = \quad \overset{8x2N}{\mathbf{A}^T} \quad \overset{2Nx1}{\mathbf{b}}$$

$$\overbrace{(\mathbf{A}^T \quad \mathbf{A})}^{8x8} \quad \overset{8x1}{\mathbf{h}} \quad = \quad \overbrace{(\mathbf{A}^T \quad \mathbf{b})}^{8x1}$$

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

Matlab: $\mathbf{h} \; = \; \mathbf{A} \backslash \mathbf{b}$

Caution: Numeric Conditioning

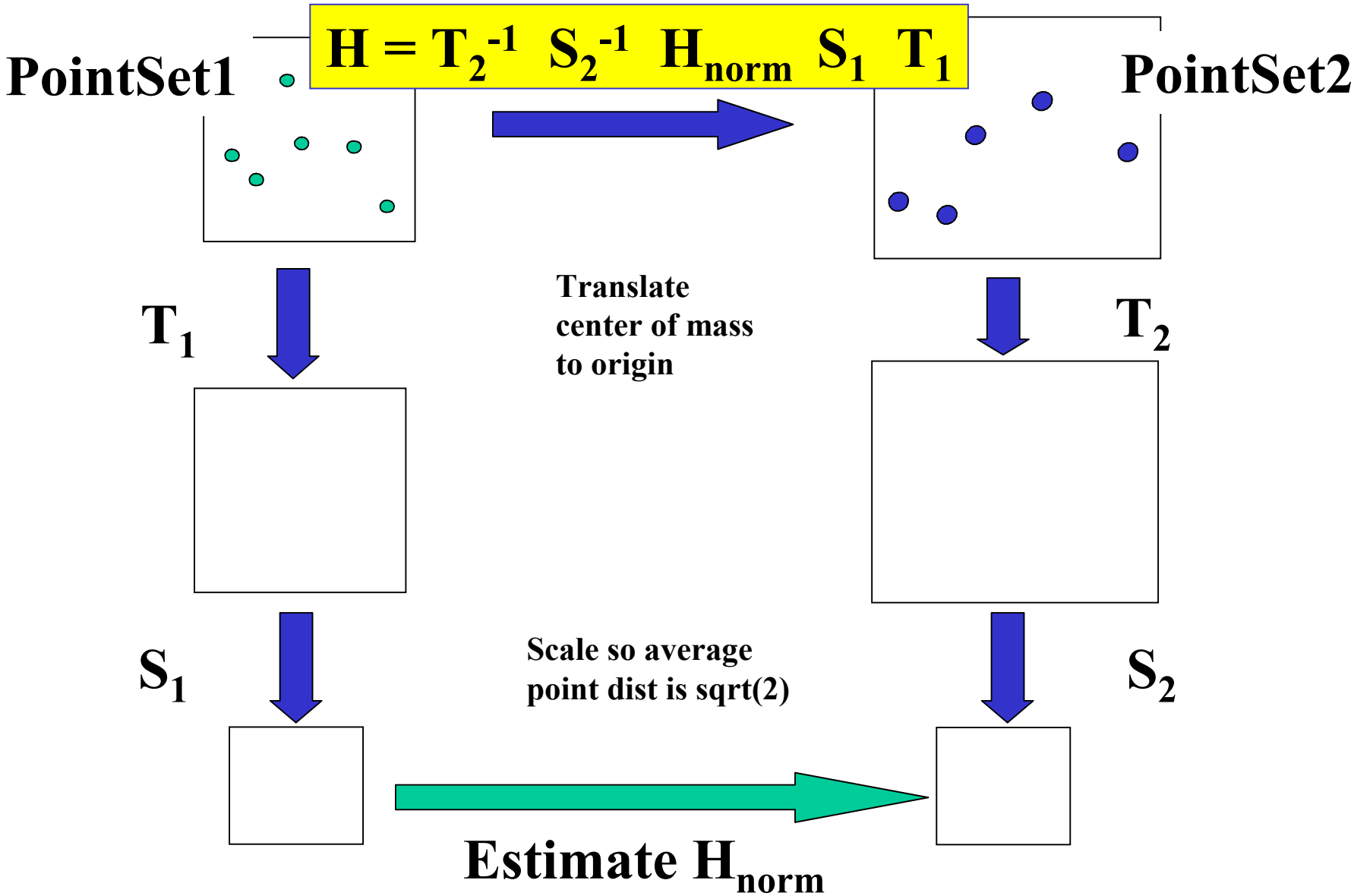
R.Hartley: "In Defense of the Eight Point Algorithm"

Observation: Linear estimation of projective transformation parameters from point correspondences often suffer from poor "conditioning" of the matrices involves. This means the solution is sensitive to noise in the points (<u>even if there are no outliers</u>).

To get better answers, precondition the matrices by performing a normalization of each point set by:

- translating center of mass to the origin
- scaling so that average distance of points from origin is sqrt(2).
- do this normalization to each point set independently

Hartley's PreConditioning



A More General Approach

What might be wrong with setting $h_{33} = 1$?

If h_{33} actually = 0, we can't get the right answer.

Algebraic Distance, ||h||=1

||h|| = 1

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

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

Multiplying through by denominator

$$(h_{31}x + h_{32}y + h_{33})x' = h_{11}x + h_{12}y + h_{13}$$

$$(h_{31}x + h_{32}y + h_{33})y' = h_{21}x + h_{22}y + h_{23}$$

Rearrange

$$h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx' - h_{33}x' = 0$$

$$h_{21}x + h_{22}y + h_{23} - h_{31}xy' - h_{32}yy' - h_{33}y' = 0$$

Algebraic Distance, ||h||=1 (cont)

2N x 9 **9 x 1** **2N x 1**

4 POINTS

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_1' & -y_1x_1' & -x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1' & -y_1y_1' & -y_1' \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \\ \\ \\ \\ \\ \\ \end{bmatrix}$$

additional points

$\vdots$ $\vdots$

Algebraic Distance, ||h||=1 (cont)

Homogeneous equations

$$\overset{2N\times 9}{\mathbf{A}}\ \overset{9\times 1}{\mathbf{h}} = \overset{2N\times 1}{\mathbf{0}}$$

Solve:

$$\overset{9\times 2N}{\mathbf{A}^T}\ \overset{2N\times 9}{\mathbf{A}}\ \overset{9\times 1}{\mathbf{h}} = \overset{9\times 2N}{\mathbf{A}^T}\ \overset{2N\times 1}{\mathbf{0}}$$

$$\overbrace{(\mathbf{A}^T\ \mathbf{A})}^{9\times 9}\ \overset{9\times 1}{\mathbf{h}} = \overset{9\times 1}{\mathbf{0}}$$

SVD of $\mathbf{A}^T\mathbf{A} = \mathbf{U}\ \mathbf{D}\ \mathbf{U}^T$

Let h be the column of U (unit eigenvector) associated with the smallest eigenvalue in D. (if only 4 points, that eigenvalue will be 0)

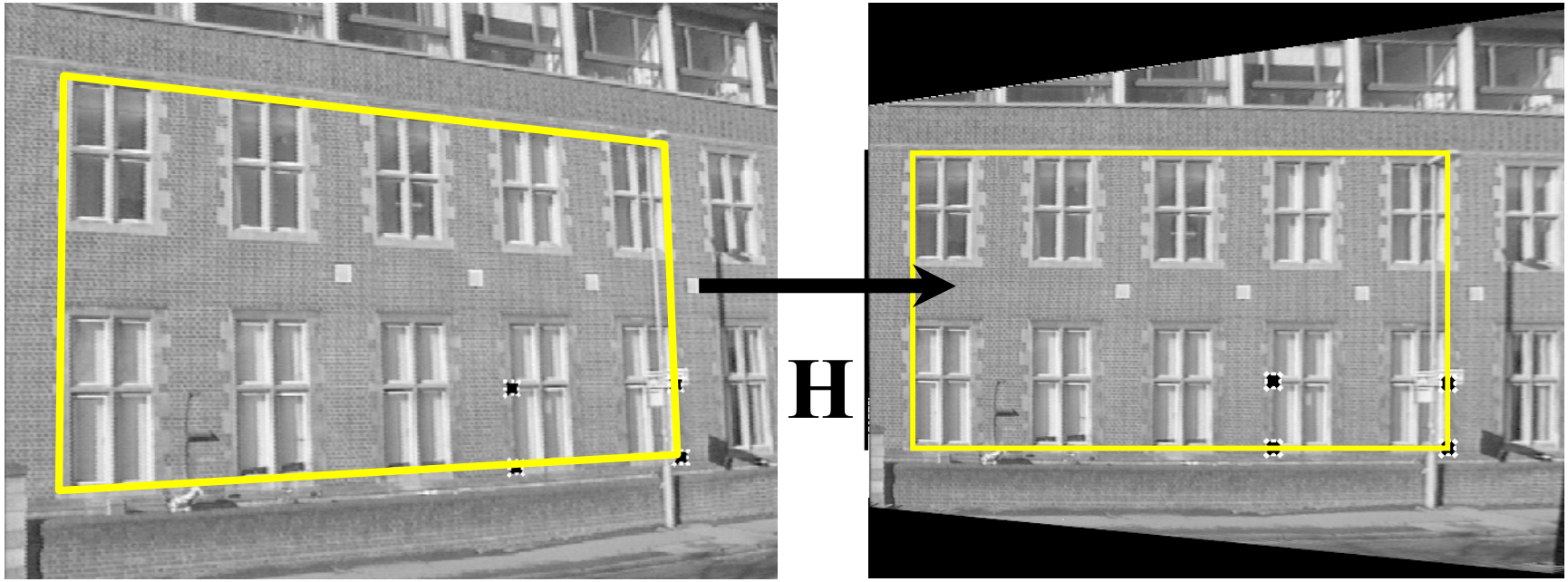
Warping & Bilinear Interpolation

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.

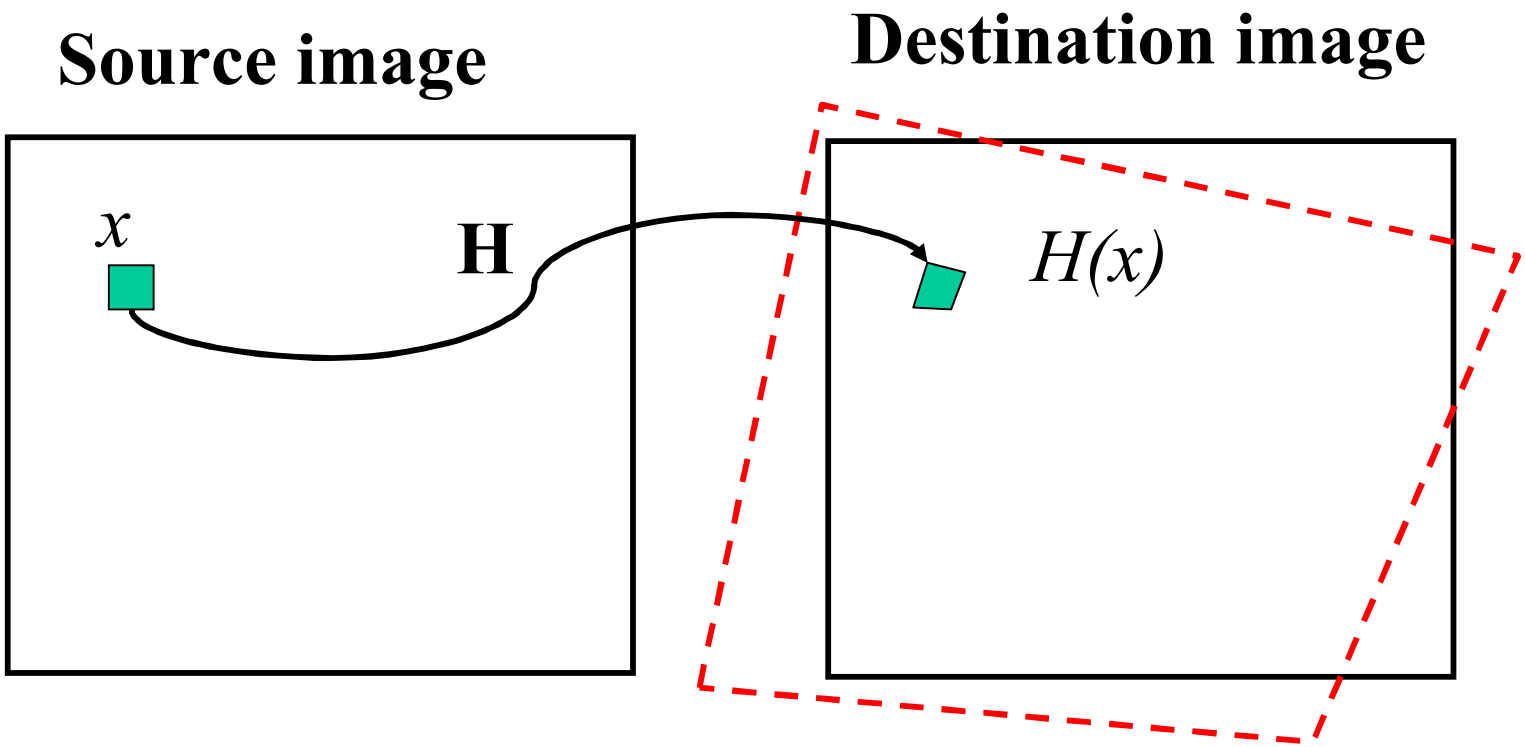
Projective Warping Example



from Hartley & Zisserman

Source Image **Destination image**

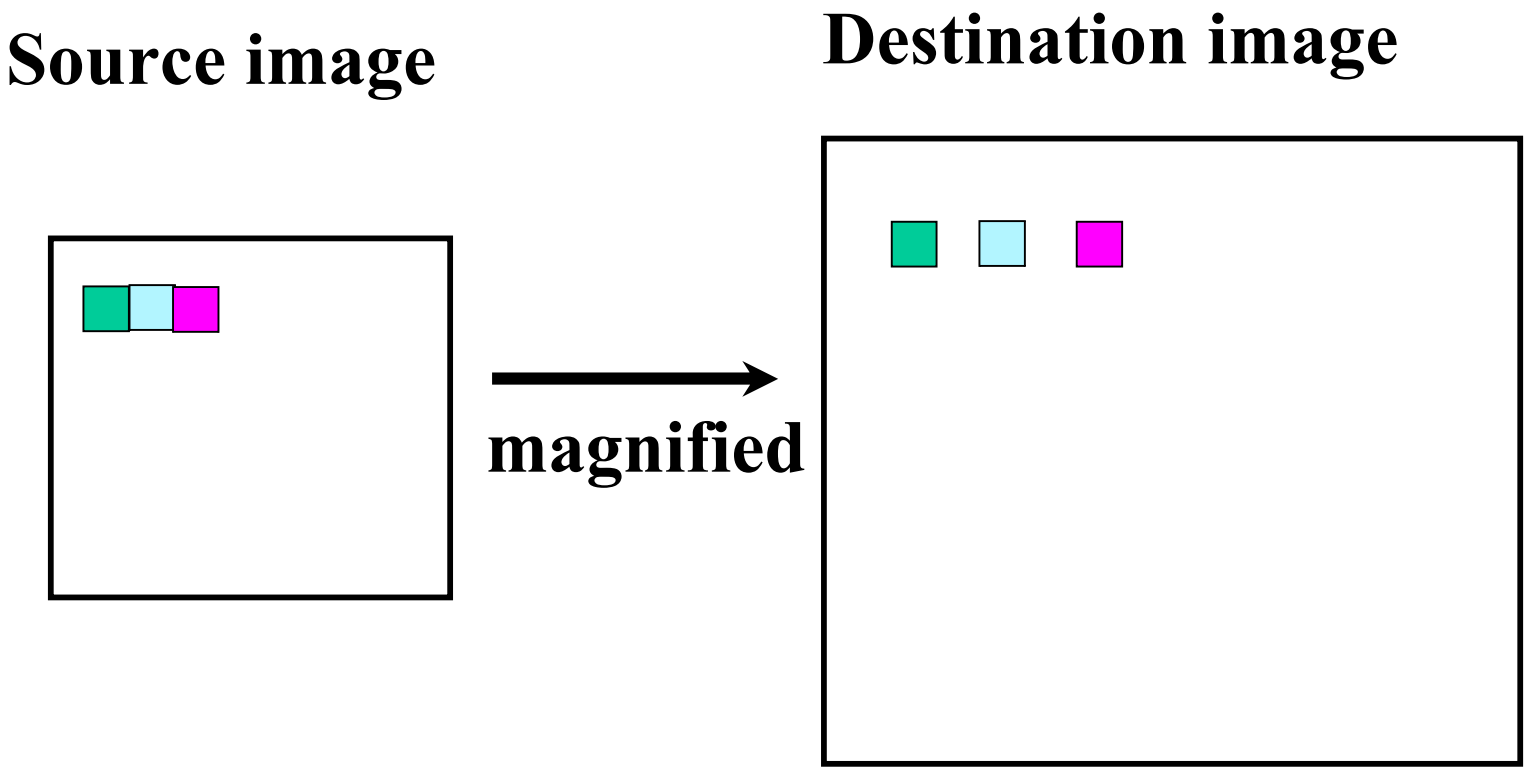
Forward Warping



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

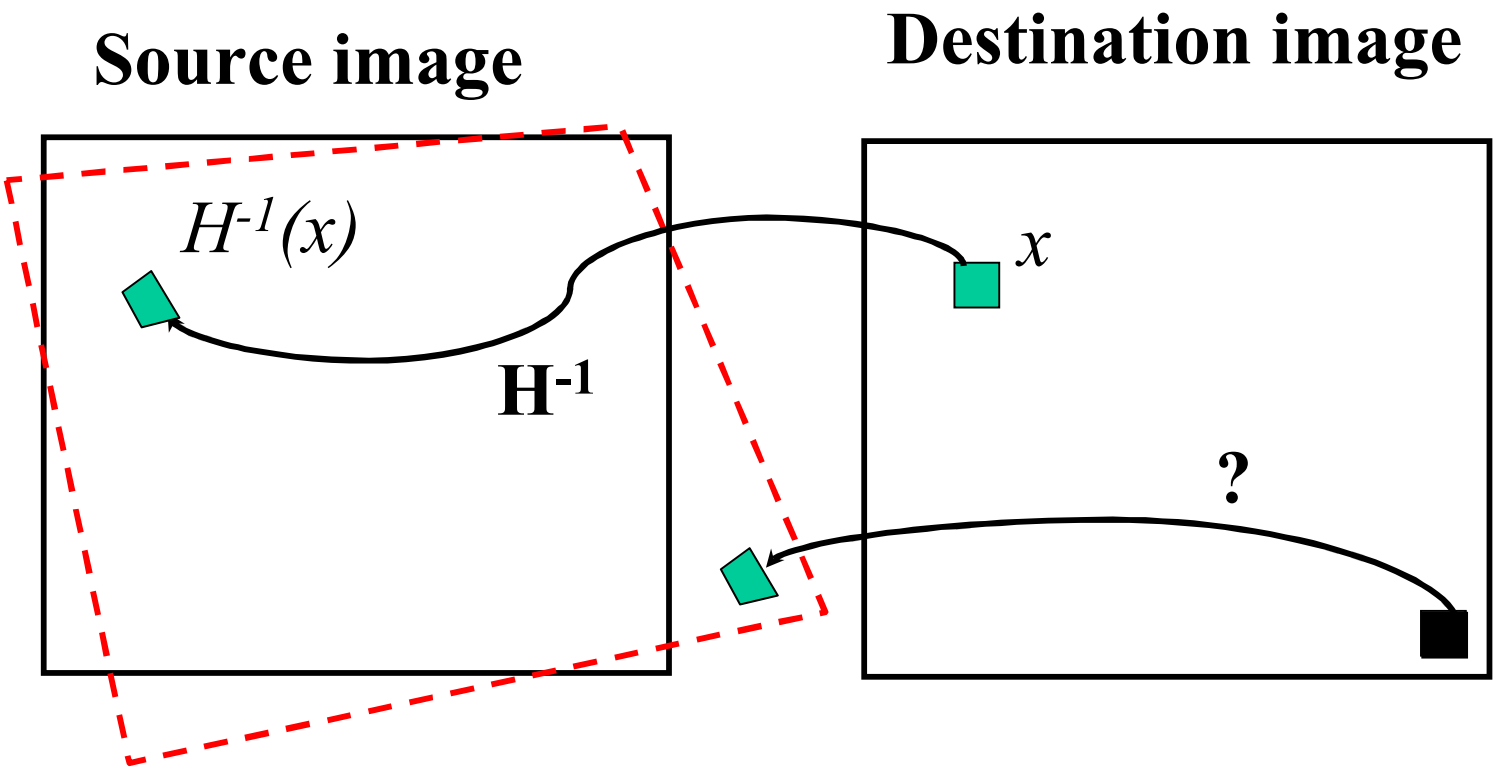
Problems?

Forward Warping Problem



Can leave gaps!

Backward Warping (No gaps)

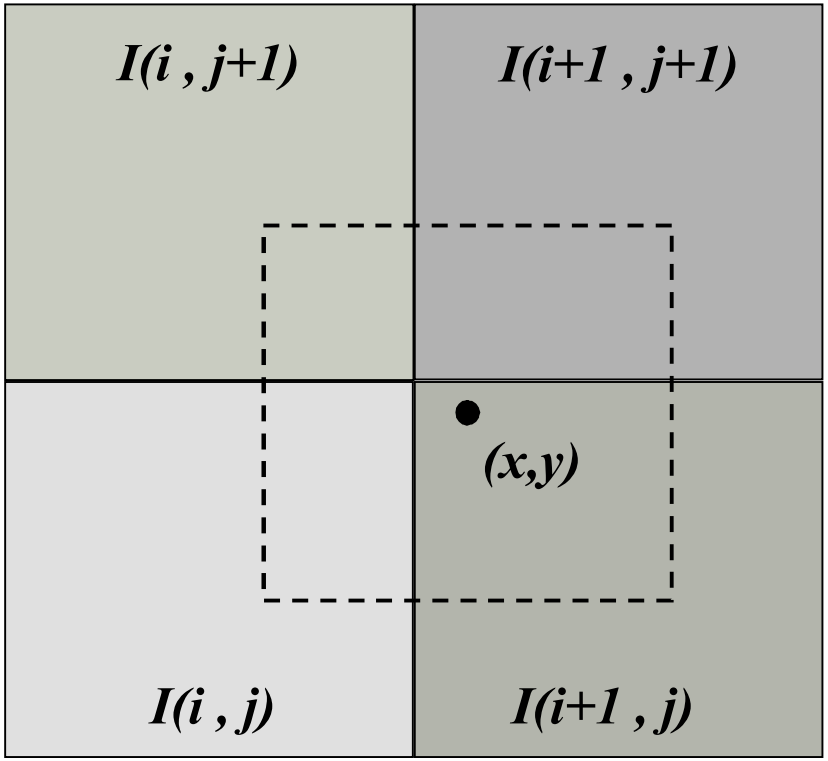


- For each pixel x in the destination image
- Determine where it comes from as $H^{-1}(x)$
- Get color from that location

Interpolation

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

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



Nearest Neighbor:

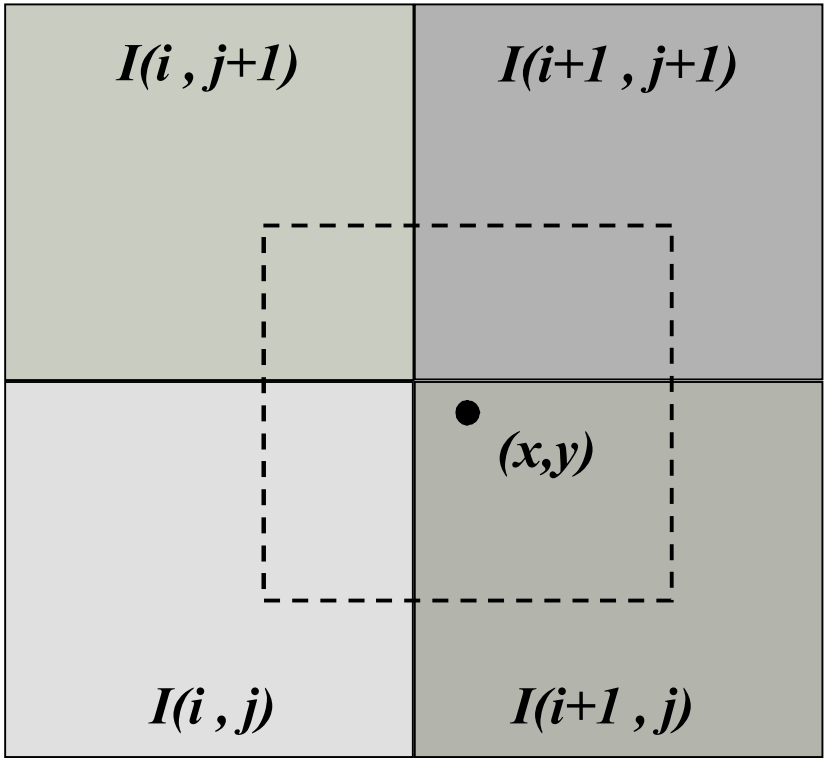
Take color of pixel with closest center.

$$\mathbf{I(x,y) = I(i+1,j)}$$

Bilinear interpolation

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

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



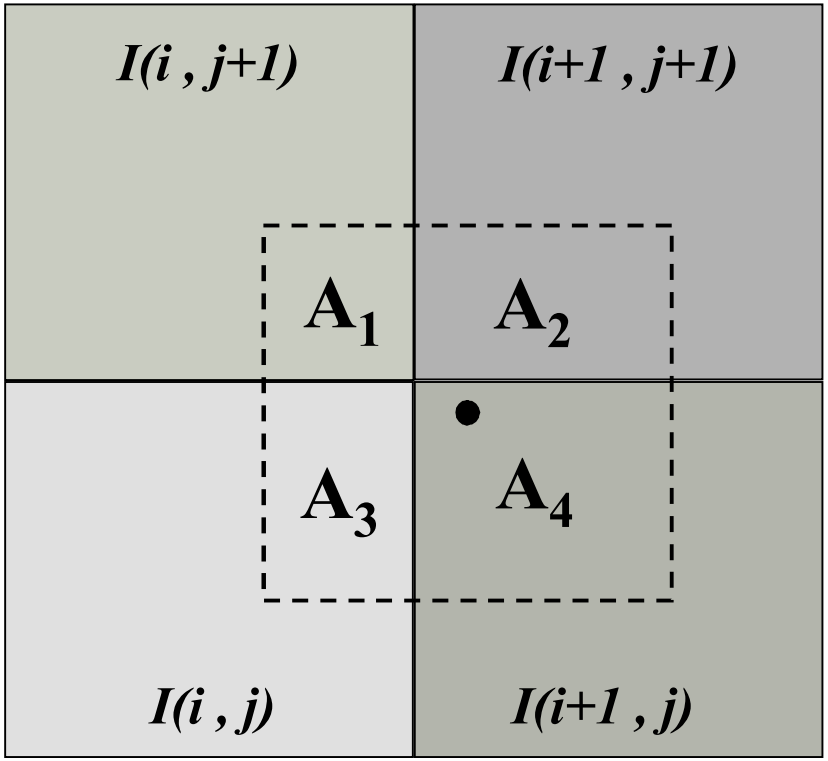
Bilinear Interpolation:

Weighted average

Bilinear interpolation

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

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



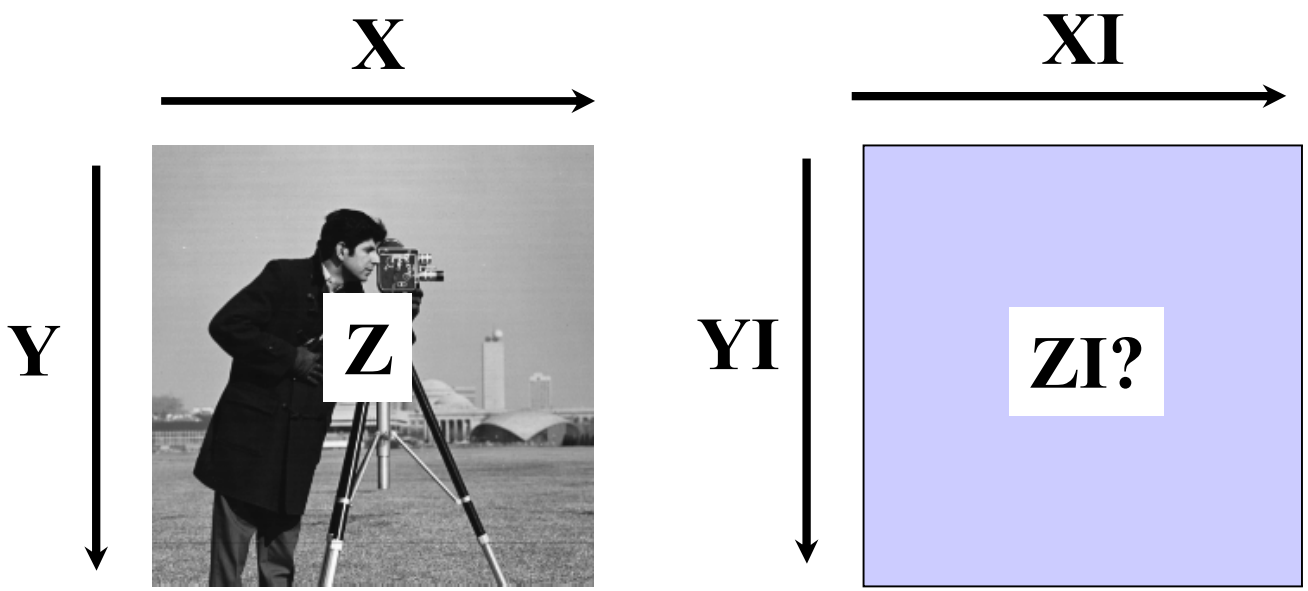
Bilinear Interpolation:

Weighted average

$$\begin{aligned} I(x,y) = {} & A3*I(i,j) \\ & + A4*I(i+1,j) \\ & + A2*I(i+1,j+1) \\ & + A1*I(i,j+1) \end{aligned}$$

Image Warping in Matlab

interp2 is Matlab's built-in function for image warping
Usage: interp2(X,Y,Z,XI,YI)
Simpler Usage: interp2(Z,XI,YI)



Interp2 can do bilinear interpolation.

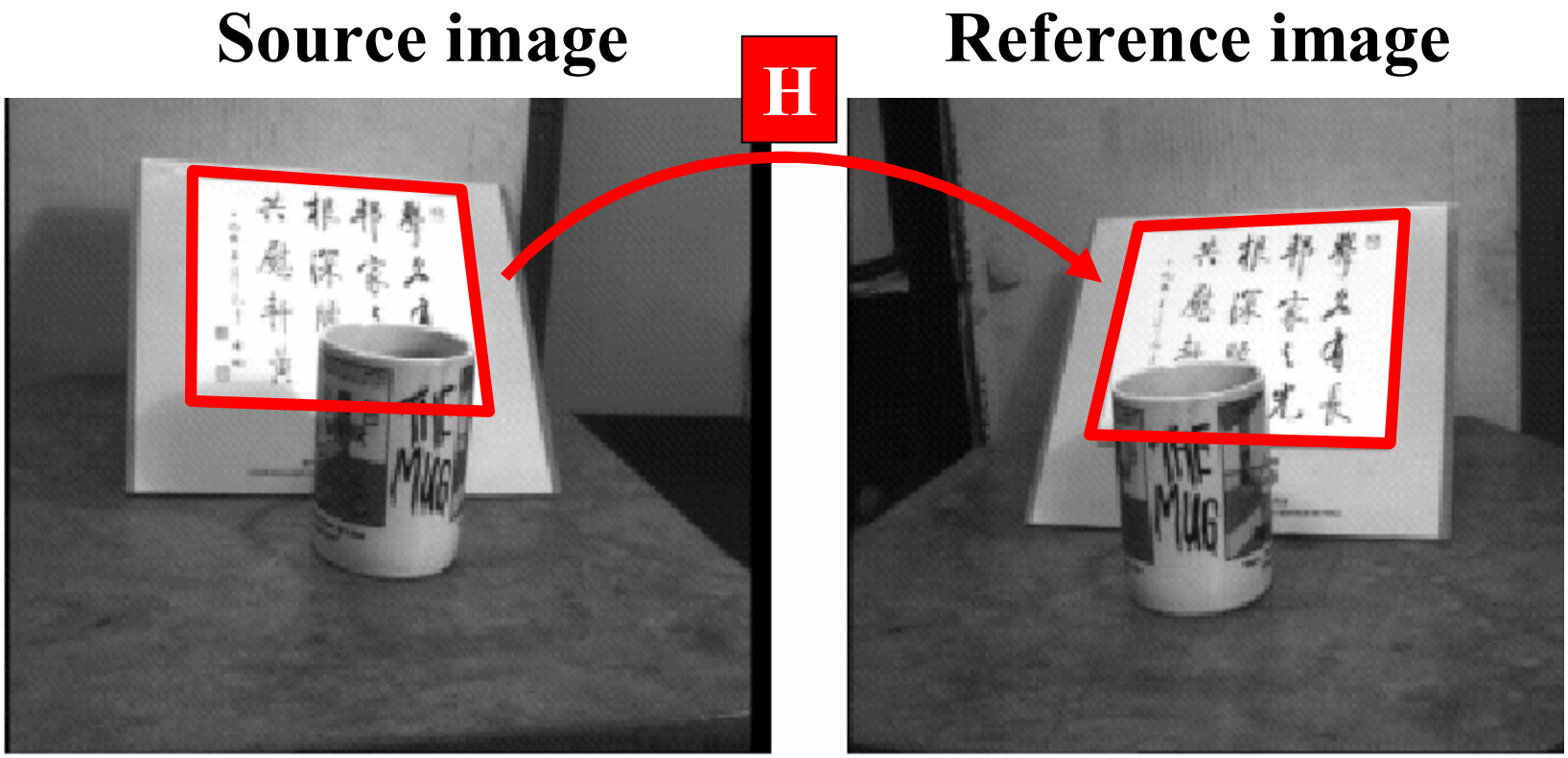
Note on Planar Mosaicing

Assumes scene is roughly planar.

What if scene isn't planar? Alignment will not be good if significant 3D relief

→ "Ghosting"

Ghosting Example



Ghosting Example (cont)

Mosaic

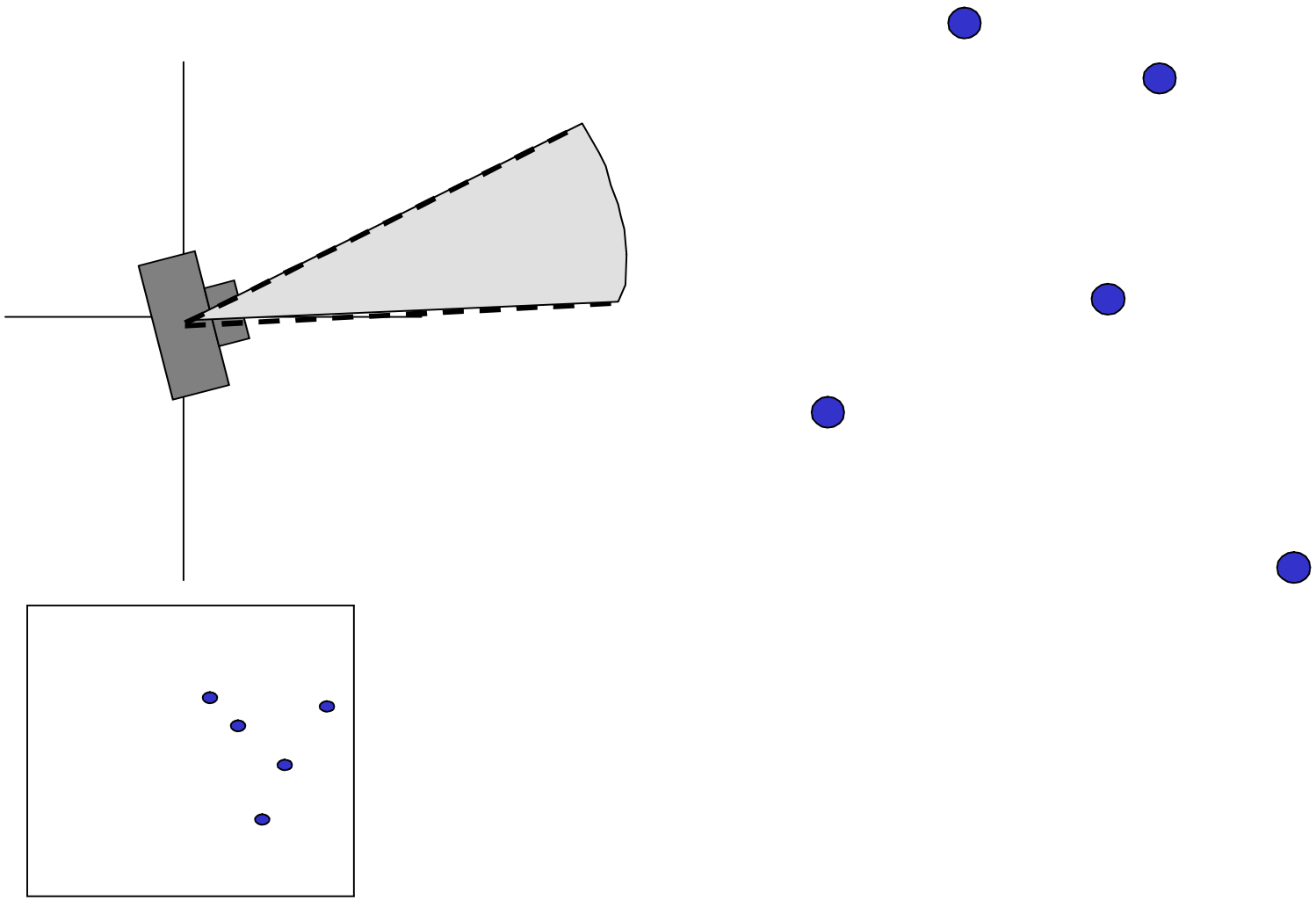
Mosaics from Rotating Cameras

However, there is a mitigating factor in regards to ghosting…

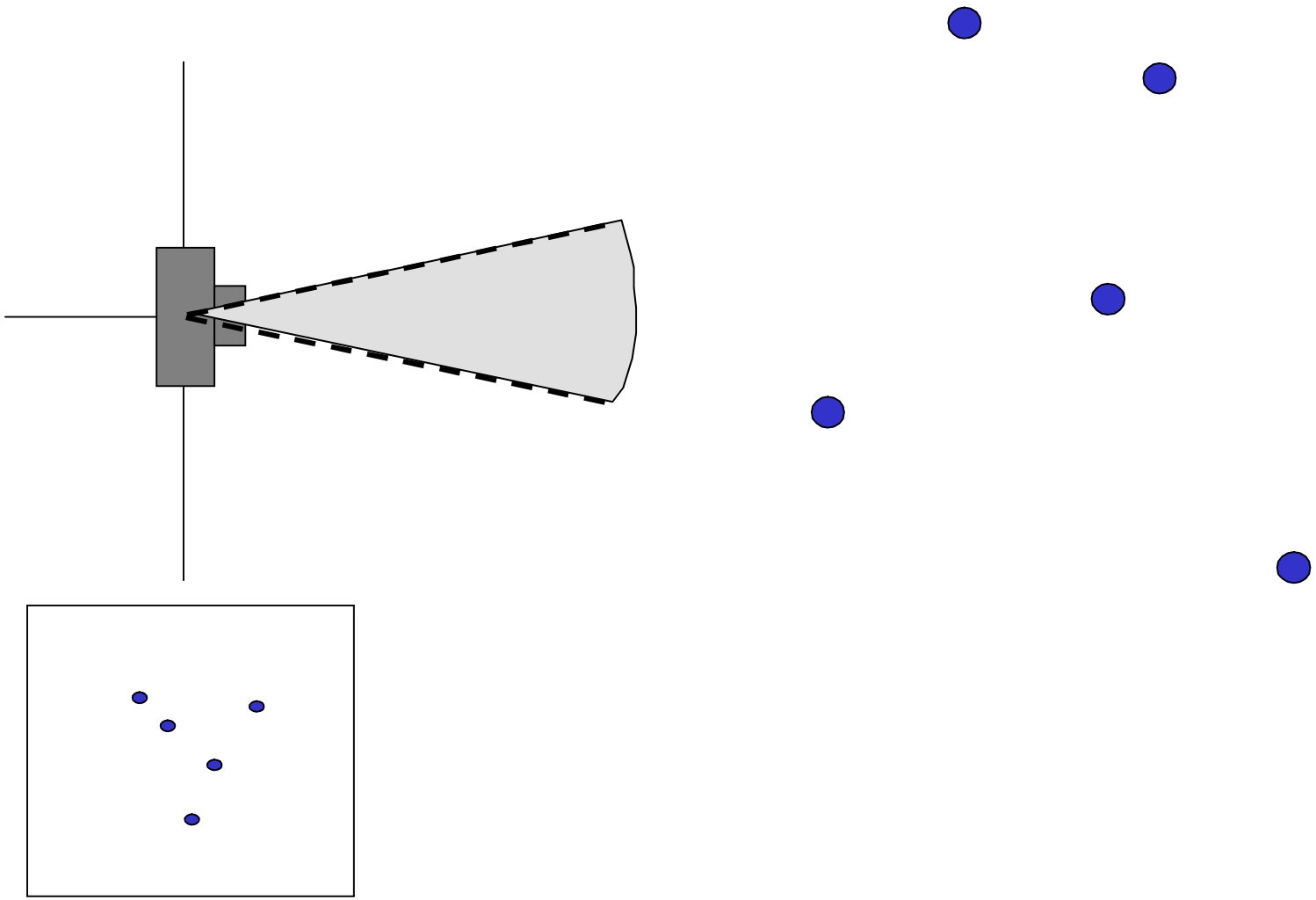
Images taken from a rotating camera are related by a 2D homography…

<u>regardless of scene structure!</u>

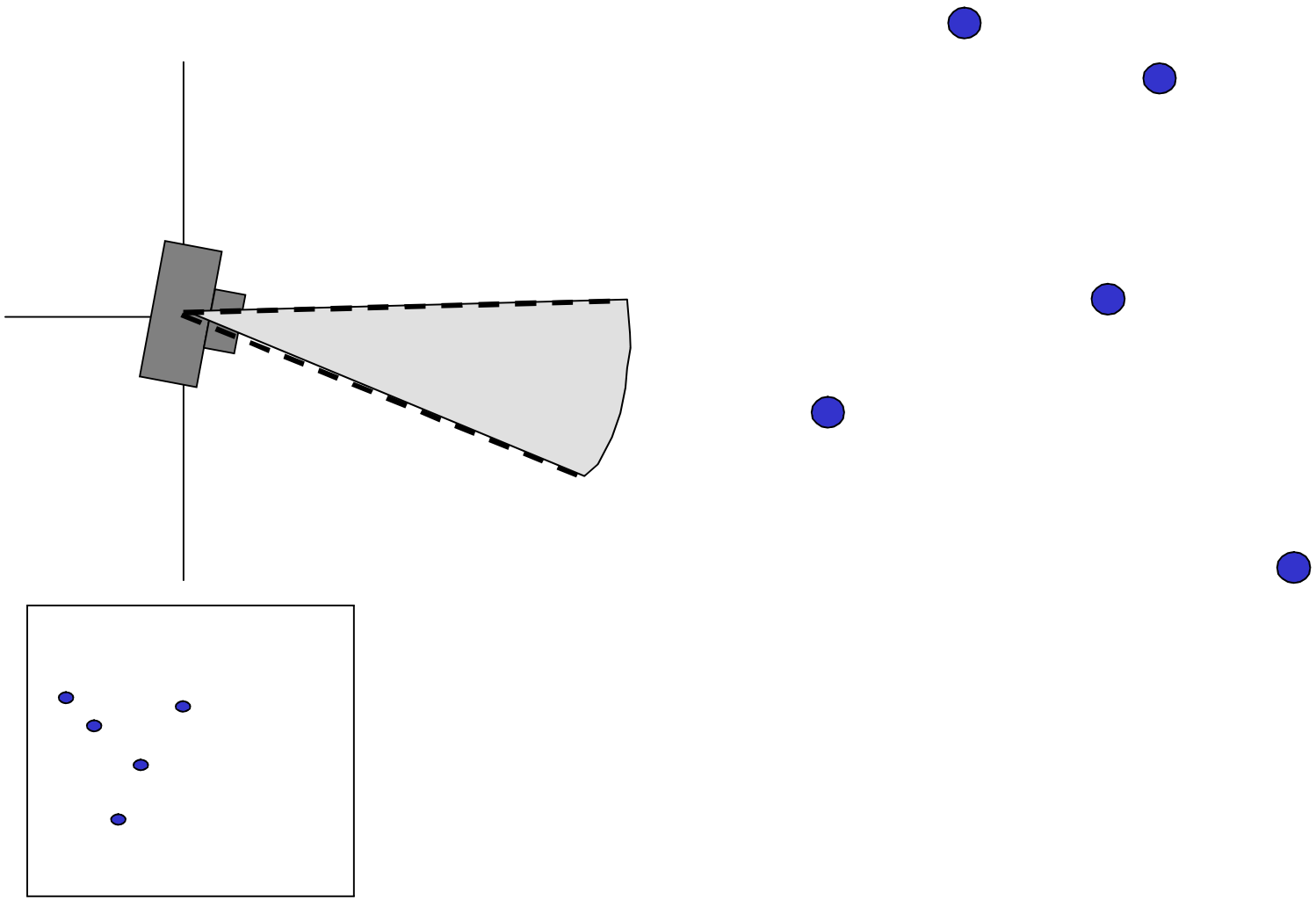
Rotating Camera (top-down view)



Rotating Camera (top-down view)

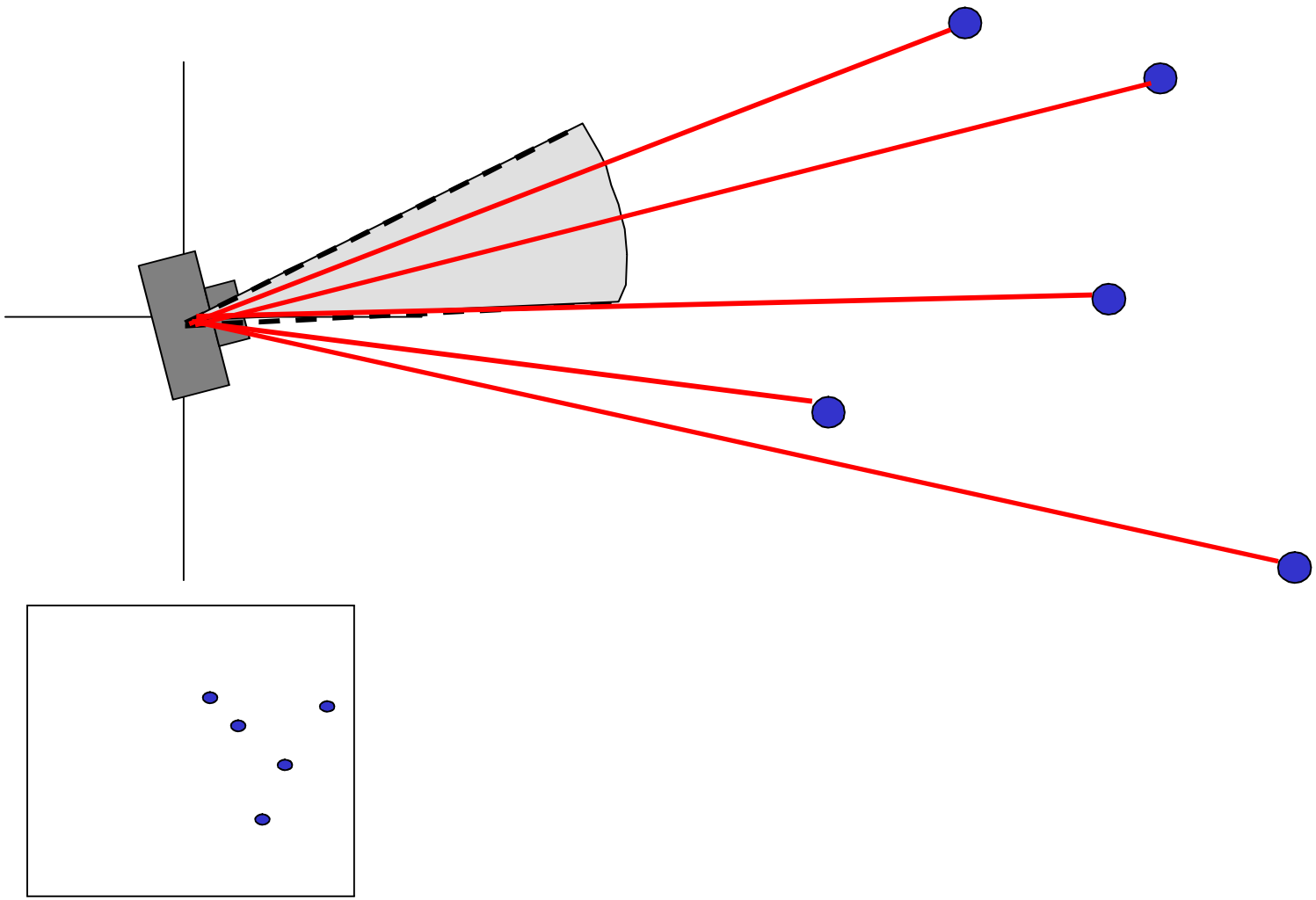


Rotating Camera (top-down view)



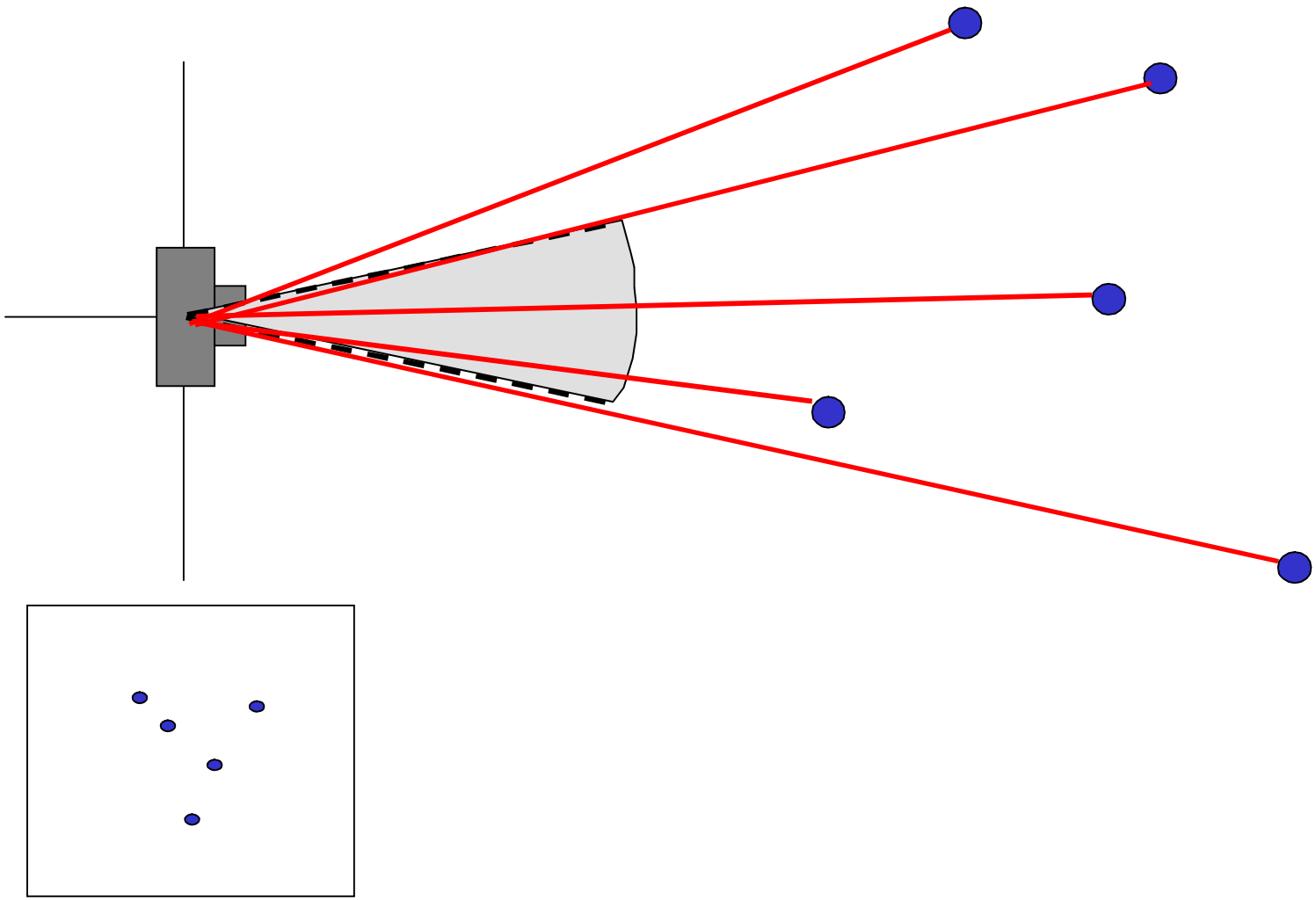
Rotating Camera (top-down view)

Rays in camera coord system are invariant!



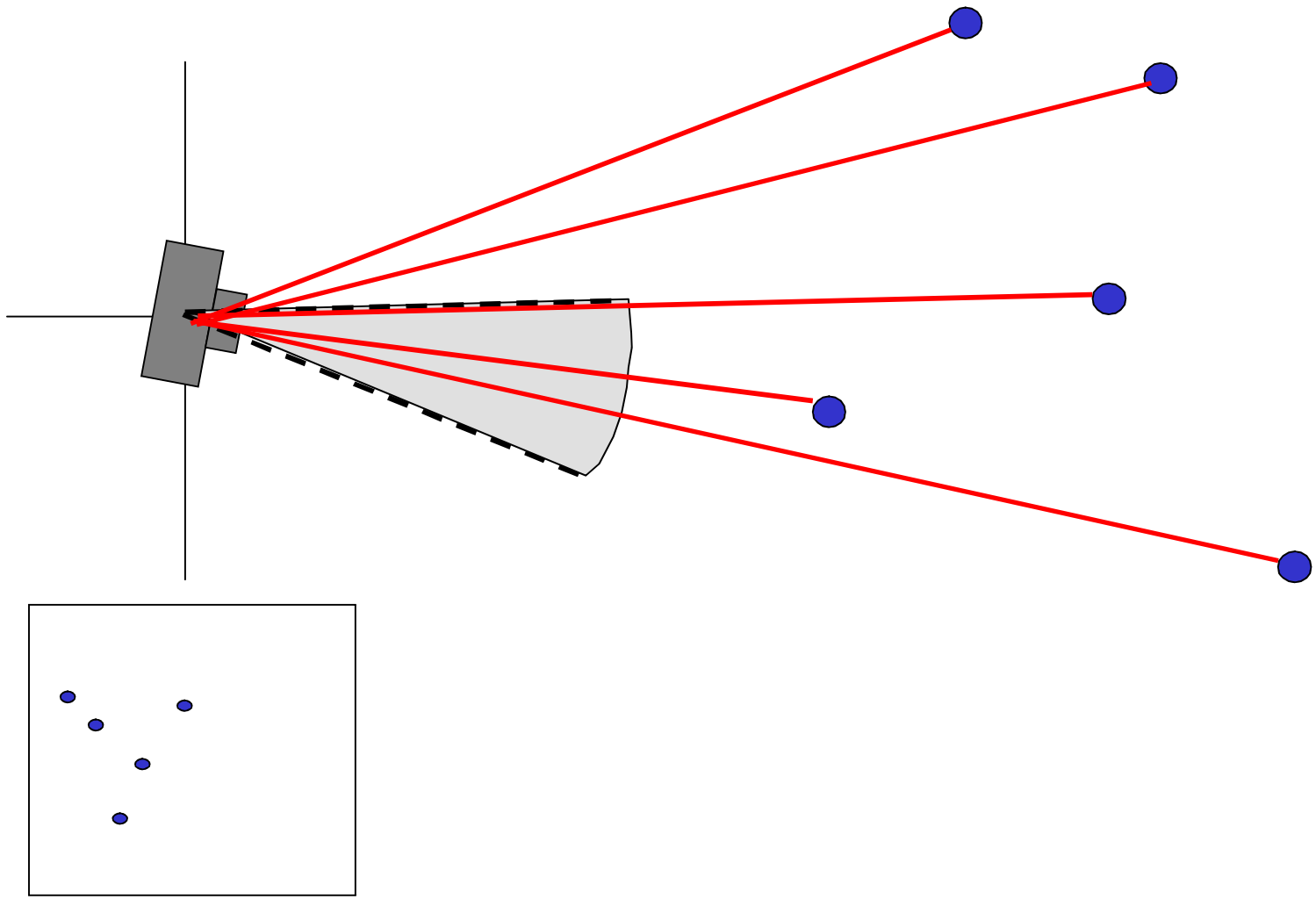
Rotating Camera (top-down view)

Rays in camera coord system are invariant!



Rotating Camera (top-down view)

Rays in camera coord system are invariant!



Special Case : Rotating Camera

Internal params **Projection** **Relative R,T**

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Relative Rotation of camera

Translation is 0
This is important!

Special Case : Rotating Camera

Internal params **Projection** **Relative R,T**

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

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

Relations among Images Taken by Rotating Camera

Image 1

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

Same ray!

Image 2

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

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

Mosaicing Example

Original Images (from a pan/tilt camera)

Panoramic (Mosaic) View

One more detail: Blending!

Approaches to Blending

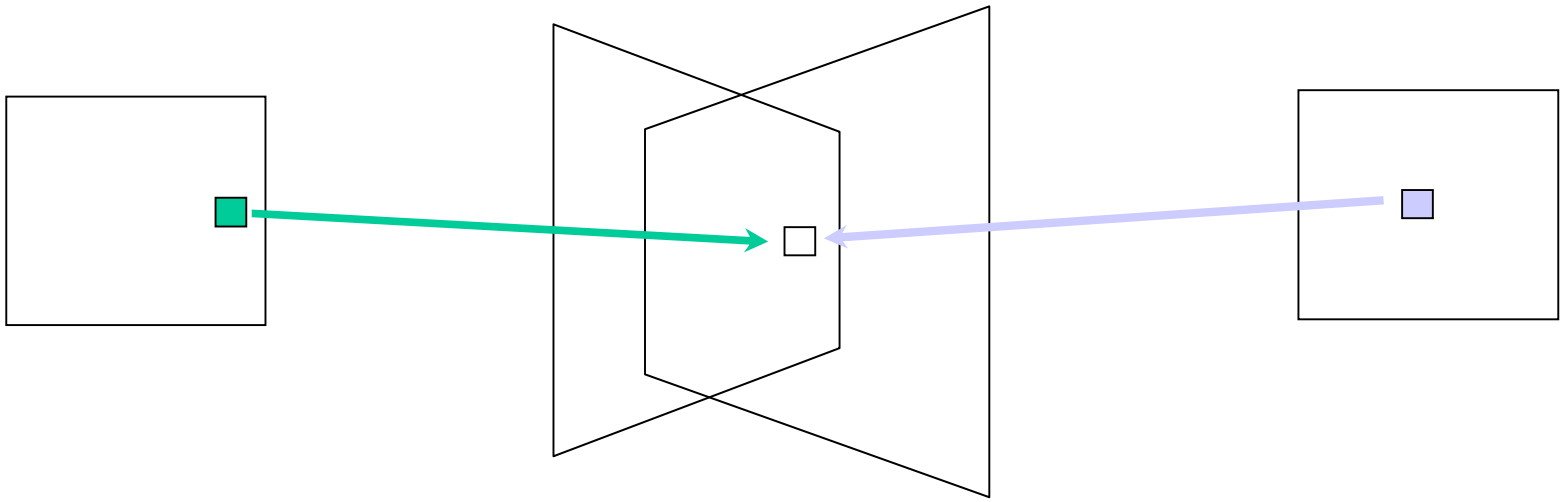
How to combine colors in area of overlap?

1) Straight averaging $\mathbf{P = (P_1 + P_2) / 2}$

2) Feathering $\mathbf{P = (w_1 * P_1 + w_2 * P_2) / (w_1 + w_2)}$

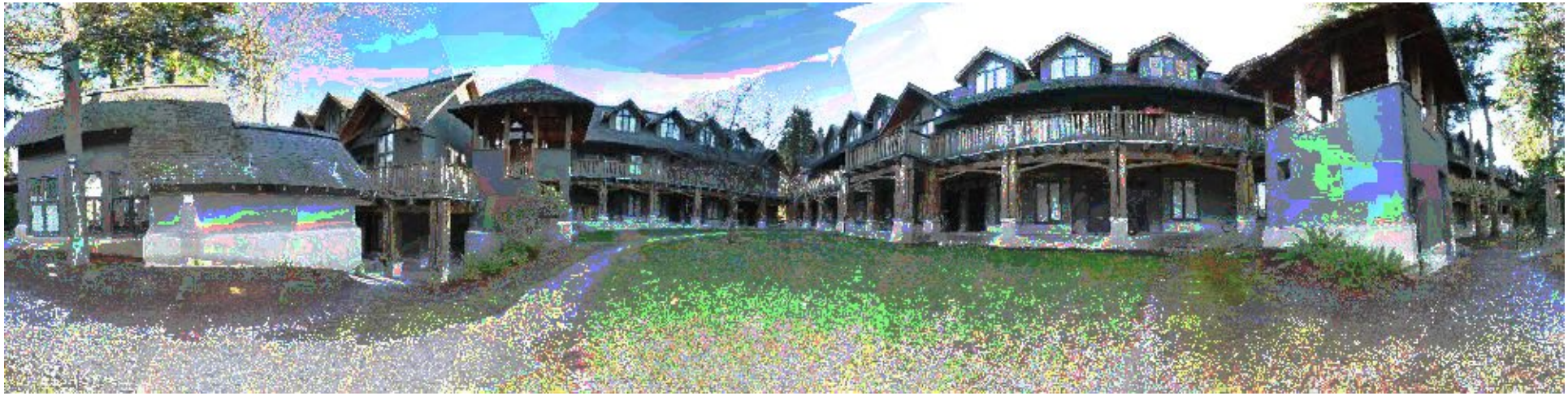
With w_i being distance from image border

3) Equalize intensity statistics (gain, offset)



Multi-band Blending

- Burt & Adelson 1983
  - Blend frequency bands over range $\propto \lambda$



Brown and Lowe: 2-band Blending

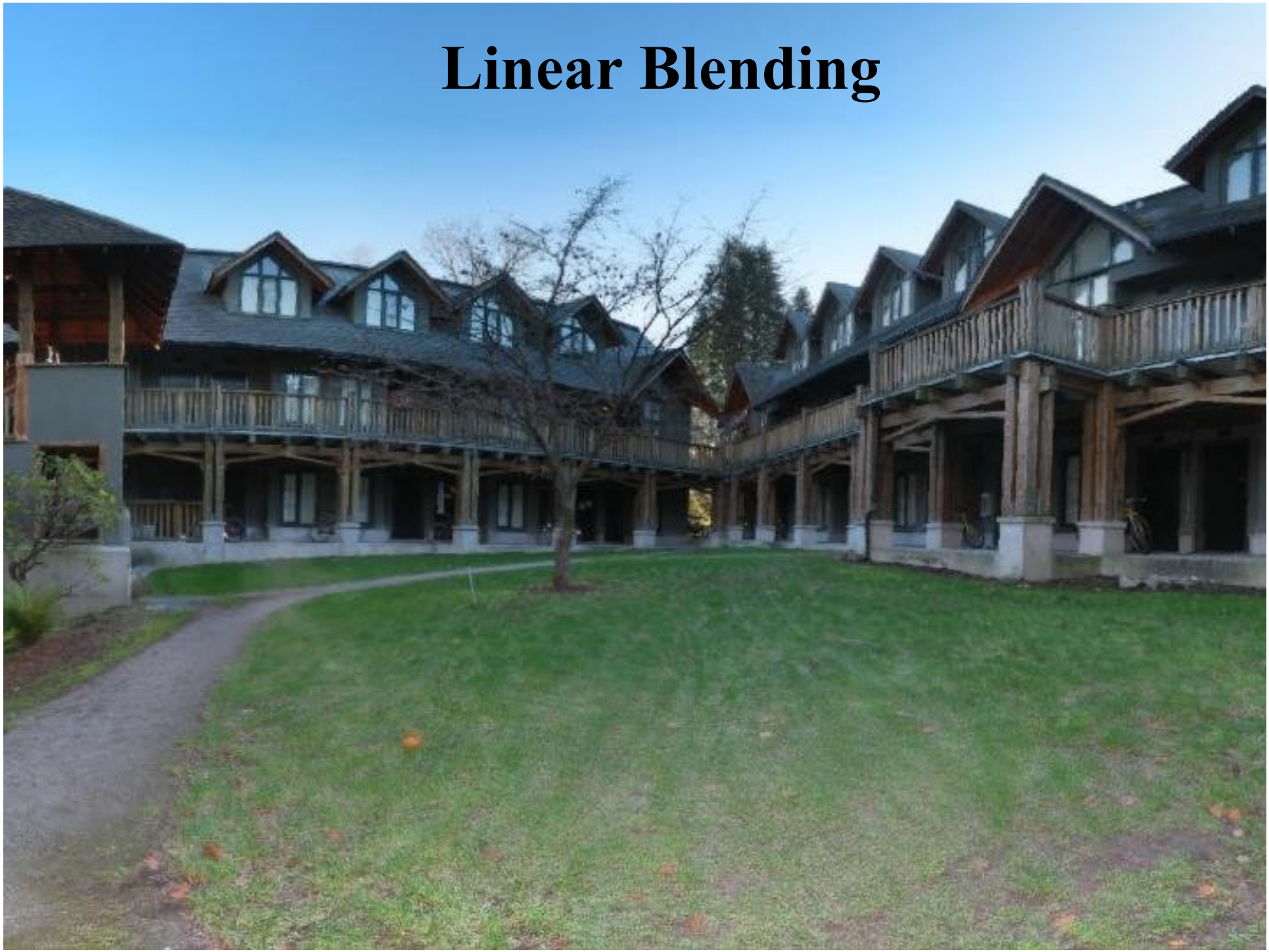
Low frequency (λ > 2 pixels)

High frequency (λ < 2 pixels)

Brown and Lowe: 2-band Blending

- Linearly combine low frequency bands (pixel values weighted relative to distance from center of image they come from)
- Nonlinearly combine high-frequency bands (high freq pixel corrections taken from image the pixel is closest to the center of – basically binary 0 or 1 weights).

Linear Blending



2-band Blending

360 Degree Panoramas?

Problem: Can’t just choose a reference image to map all other images to.

Solution: Use cylindrical or spherical mosaic surface rather than a plane.

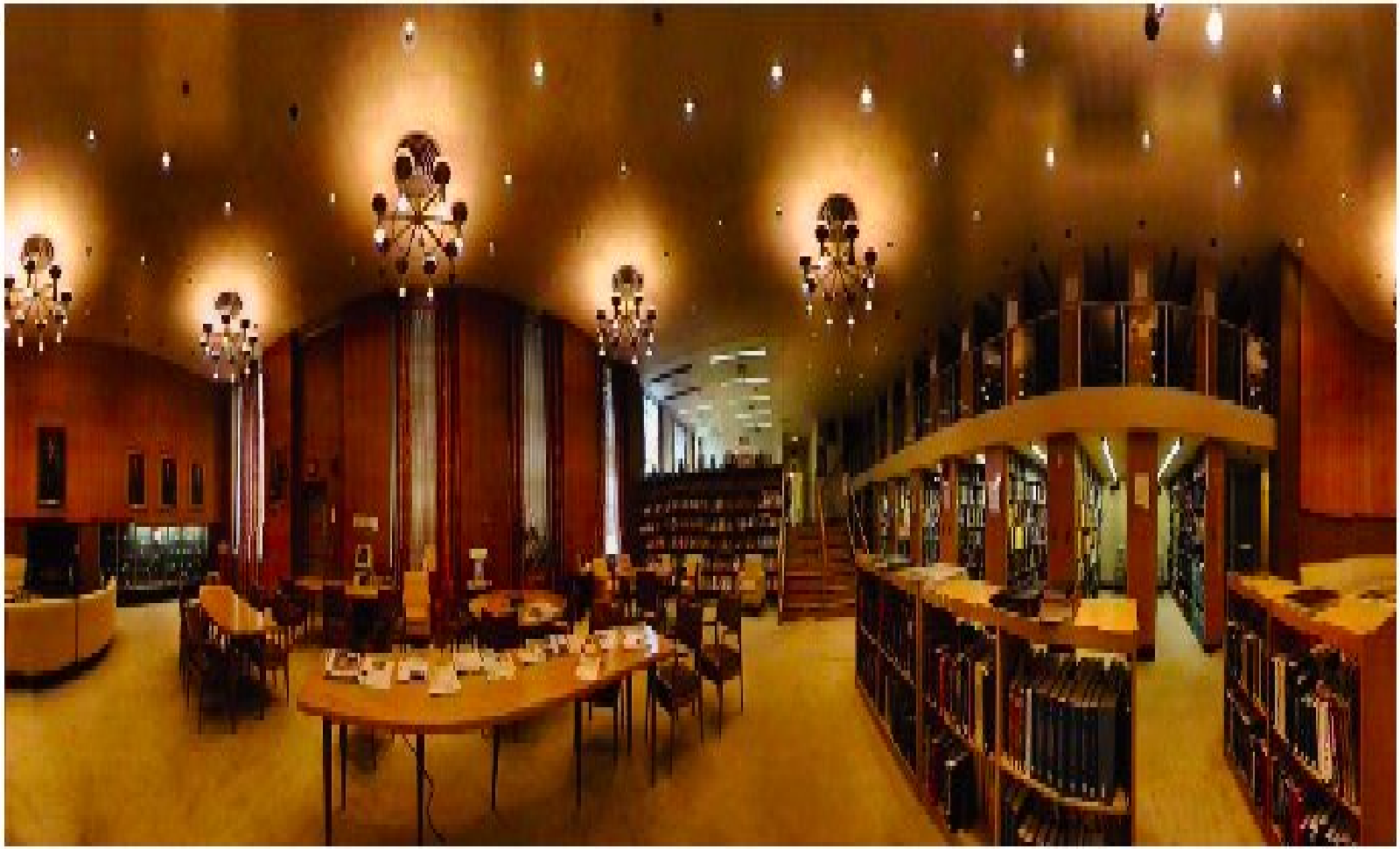
Panorama Input images

Sarnoff

Spherical Panorama Result

Sarnoff

Panorama Unwarped



Sarnoff

Quicktime VR

This example from www.panoguide.com/gallery/

Also big list at http://www.multimedialibrary.com/diana/qtvr_sites.asp

Quicktime VR

Quicktime VR

This example from www.ems.psu.edu/~fraser/qtvr/

Overview

what images are related by a homography:

- images of a planar surface (arbitrary perspective cameras)
- images taken from same optical center (rotating, zooming)

given matches:

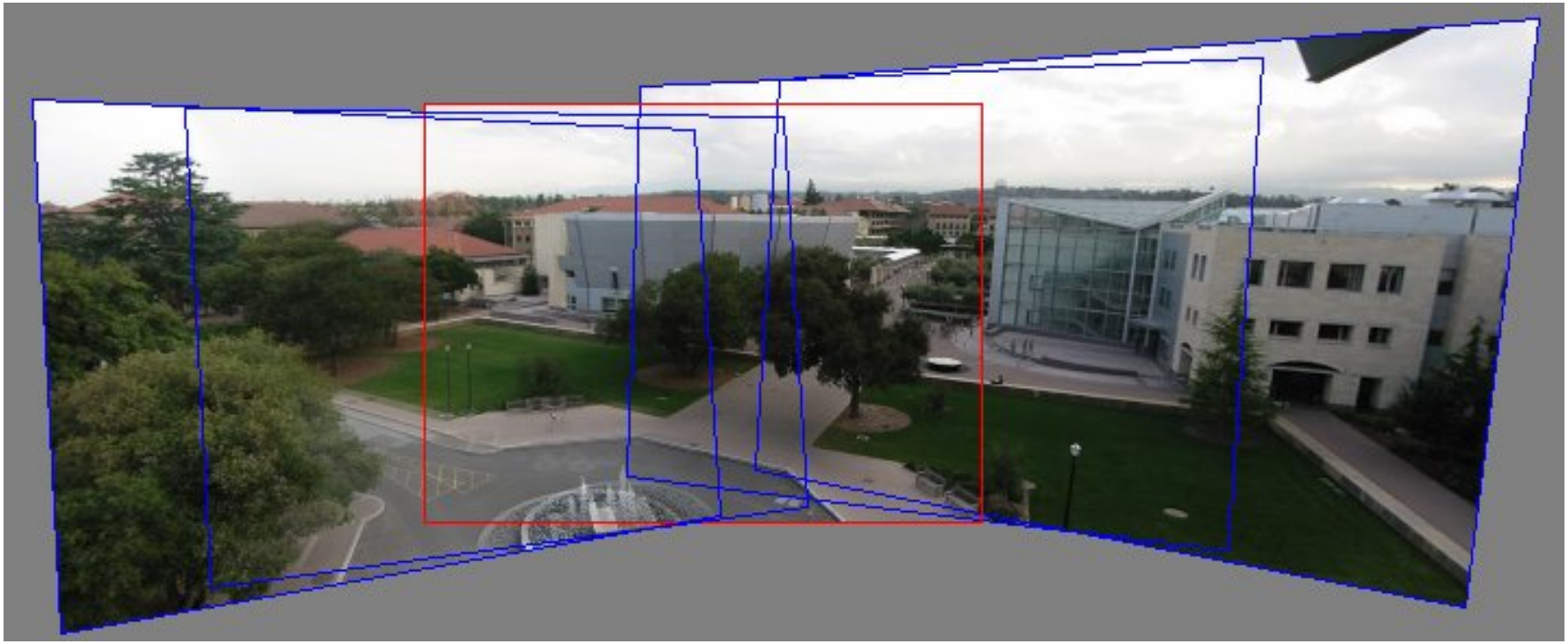
- estimating homography, warping, blending

computing matches

- Harris corners + RANSAC
- SIFT keys / Autostitch

You are here

Panoramas



1. Pick one image (red)
2. Warp the other images towards it (usually, one by one)
3. blend

Practical Application

- Stabilizing aerial imagery using RANSAC
  - find corners in two images
  - hypothesize matches using NCC
  - do RANSAC to find matches consistent with an affine transformation
  - take the inlier set found and estimate a full projective transformation (homography)

Why Stabilization and not Mosaicing

- Short answer: because I have the slides already

- Longer answer: because the steps are exactly the same, except, at the very end, after warping one image into the coordinate system of the other, you don’t just crop it at the image borders but instead make a larger image that encloses the union of all the valid warped pixel values.

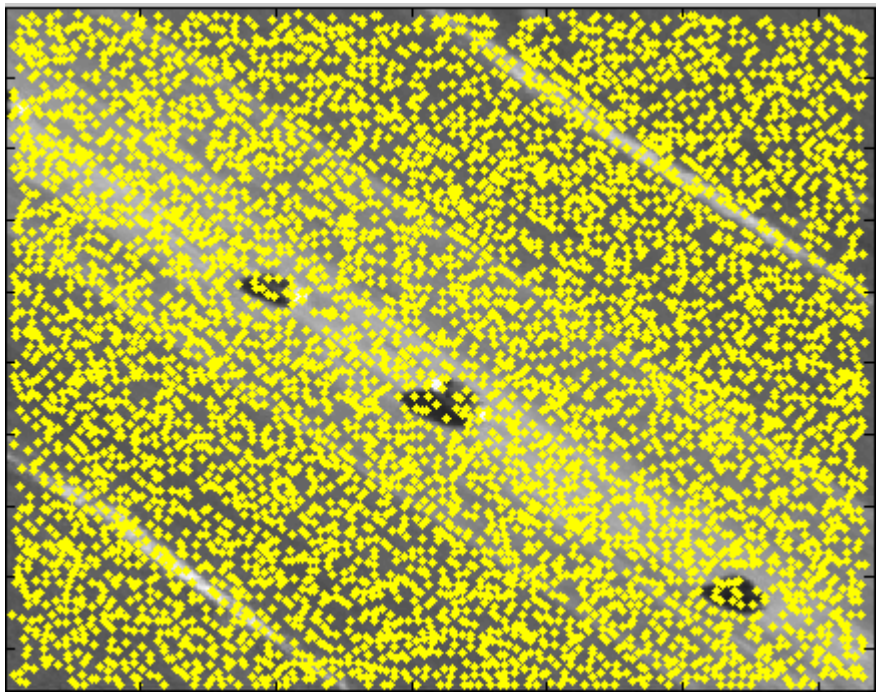
Stabilization Application

Input: two images from an aerial video sequence.

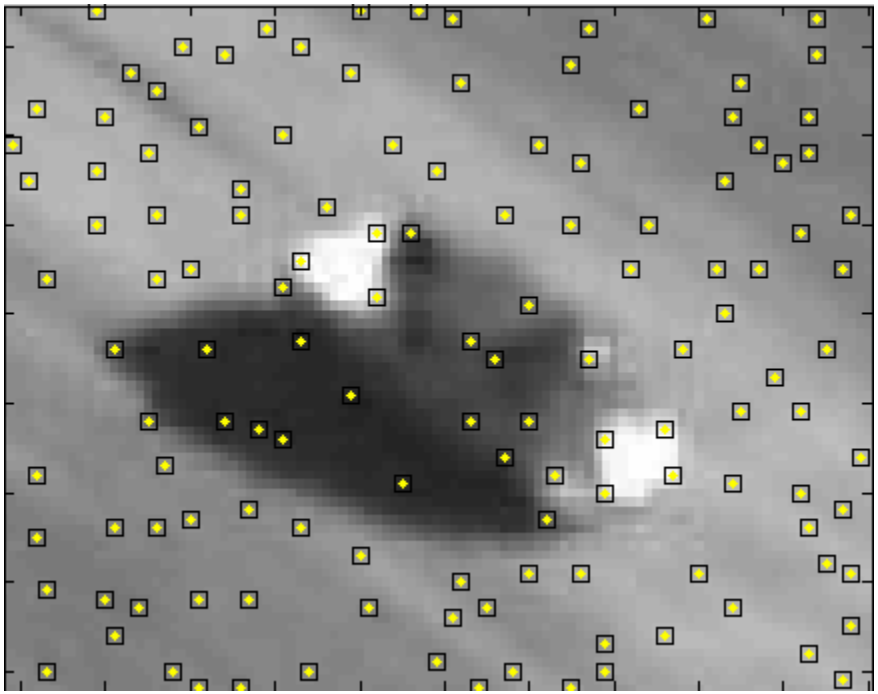
Note that the motion of the camera is “disturbing”

Stabilization Application

Step1: extract Harris corners from both frames. We use a small threshold for R because we want LOTS of corners (fodder for our next step, which is matching).



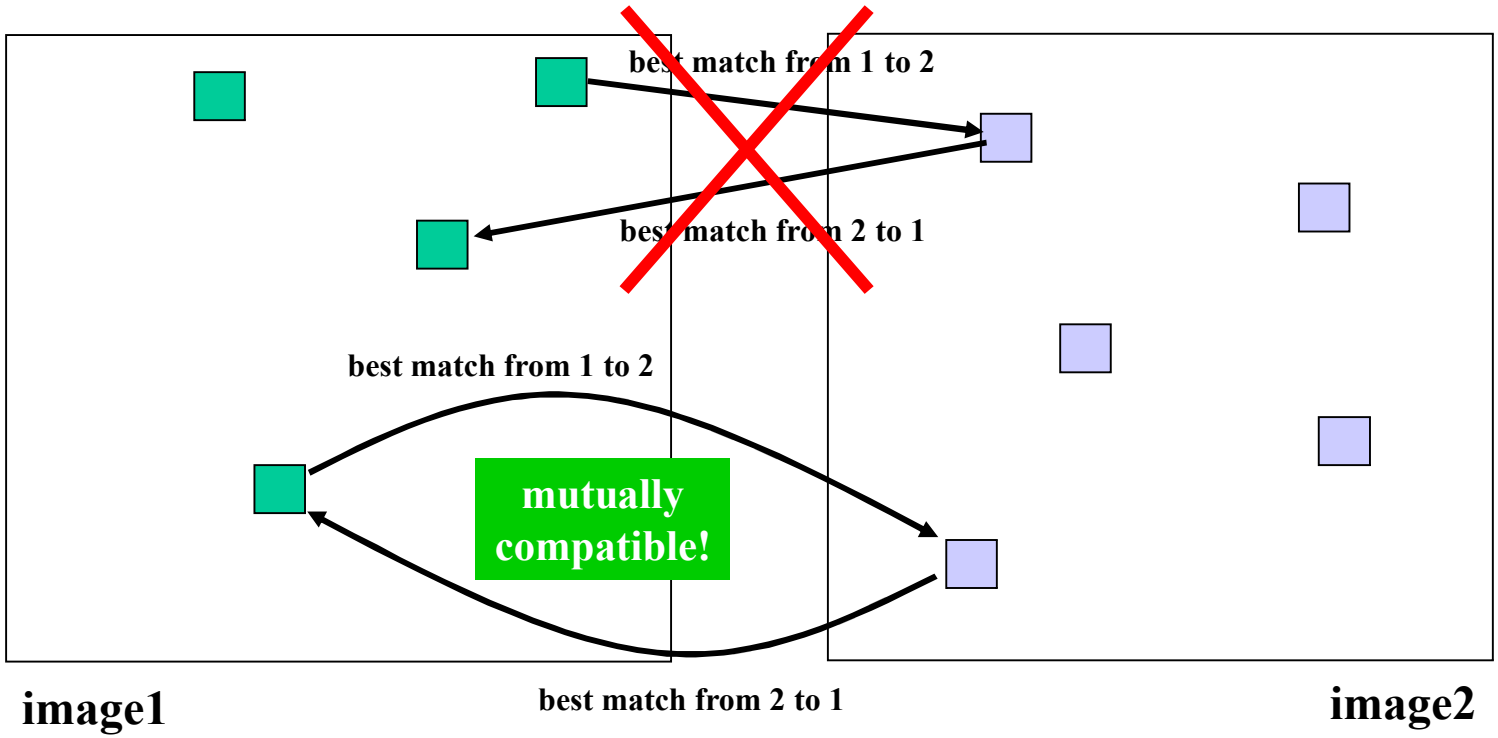
Harris corners for first frame



Detailed view

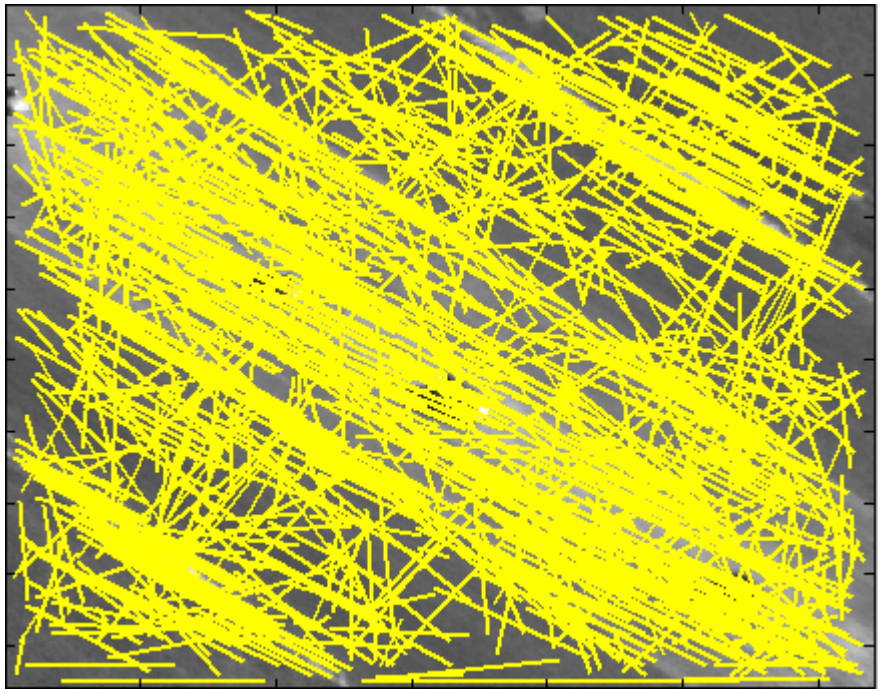
Stabilization Application

Step2: hypothesize matches. For each corner in image 1, look for matching intensity patch in image2 using NCC. Make sure matching pairs have highest NCC match scores in BOTH directions.



Stabilization Application

Step2: hypothesize matches.

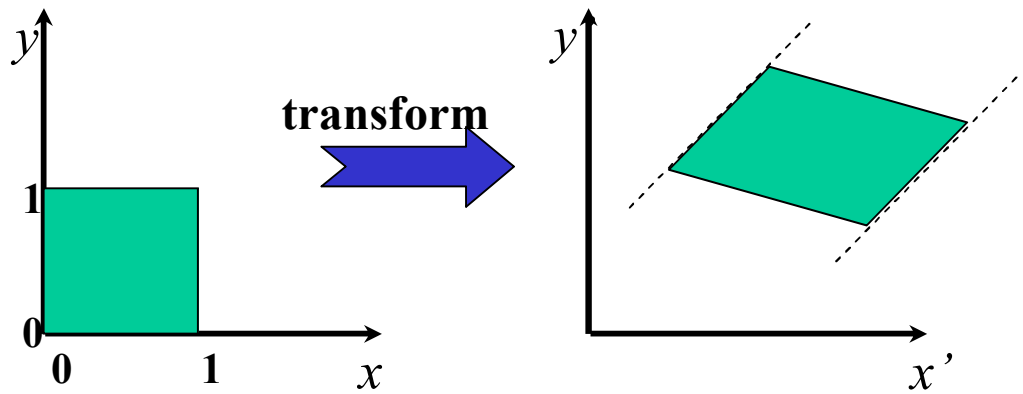


yikes!

As you can see, a lot of false matches get hypothesized. The job of RANSAC will be to clean this mess up.

Stabilization Application

Step3: Use RANSAC to robustly fit best affine transformation to the set of point matches.



$$x_i' = a\, x_i + c\, y_i + c$$
$$y_i' = d\, x_i + e\, y_i + f$$

How many unknowns?
How many point matches are needed?

Robust Estimation

- View estimation as a two-stage process:
  - Classify data points as outliers or inliers
  - Fit model to inliers while ignoring outliers

- Example: RANSAC
  (RANdom **SA**mple **C**onsensus**)**

M. A. Fischler and R. C. Bolles (June 1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". *Comm. of the ACM* **24**: 381--395.

Stabilization Application

Step3: Use RANSAC to robustly fit best affine transformation to the set of point matches.

Affine transformation has 6 degrees of freedom.
We therefore need 3 point matches [each gives 2 equations]

Randomly sample sets of 3 point matches. For each, compute the unique affine transformation they define. How?

note: we could go for projective directly, using subsets of 4 point matches. However, it is more difficult (= more iterations) to sample an inlier subset of size 4 rather than size 3.

Stabilization Application

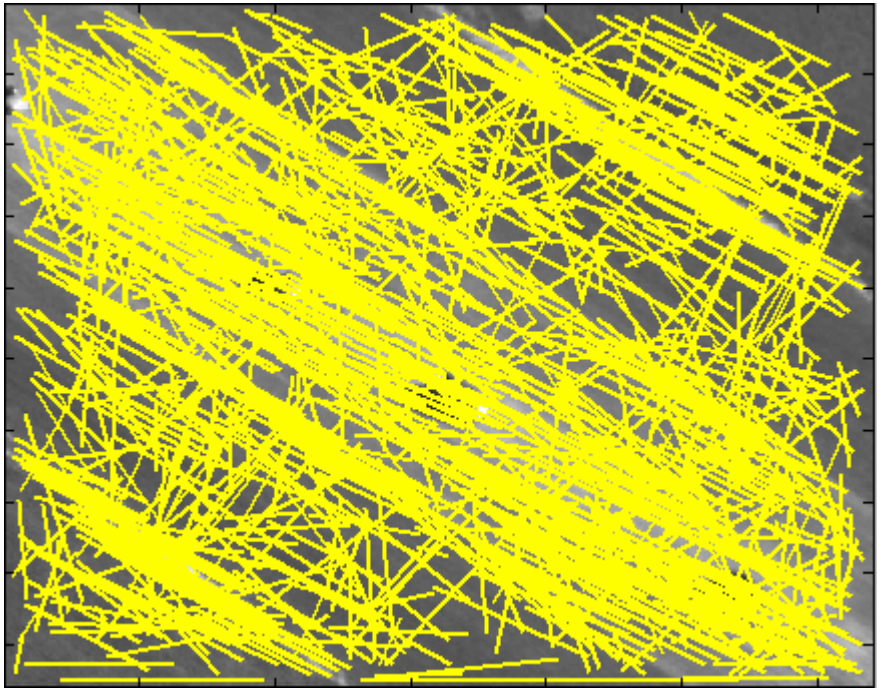
How to compute affine transformation from 3 point matches?
Use Least Squares! (renewed life for a nonrobust approach)

$$\begin{bmatrix} \Sigma x_i^2 & \Sigma x_i y_i & \Sigma x_i & 0 & 0 & 0 \\ \Sigma x_i y_i & \Sigma y_i^2 & \Sigma y_i & 0 & 0 & 0 \\ \Sigma x_i & \Sigma y_i & \Sigma 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Sigma x_i^2 & \Sigma x_i y_i & \Sigma x_i \\ 0 & 0 & 0 & \Sigma x_i y_i & \Sigma y_i^2 & \Sigma y_i \\ 0 & 0 & 0 & \Sigma x_i & \Sigma y_i & \Sigma 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} \Sigma x_i x_i' \\ \Sigma y_i x_i' \\ \Sigma x_i' \\ \Sigma x_i y_i' \\ \Sigma y_i y_i' \\ \Sigma y_i' \end{bmatrix}$$

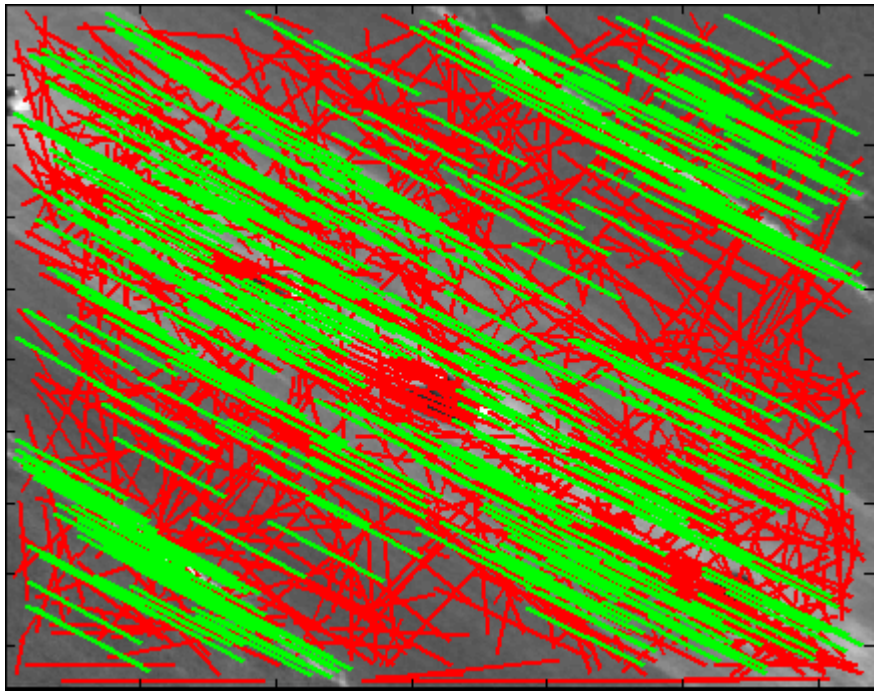
Then transform all points from image1 to image2 using that computed transformation, and see how many other matches confirm the hypothesis (fall within some threshold distance)

Repeat N times.

Stabilization Application



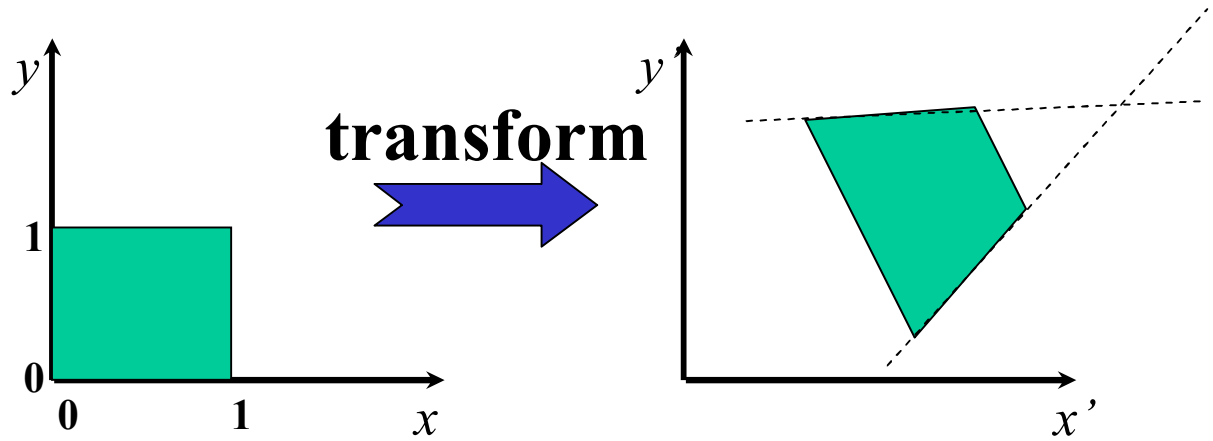
original point matches



labels from RANSAC
green: inliers
red: outliers

Stabilization Example

Step4: Take inlier set labeled by RANSAC, and now use least squares to estimate a projective transformation that aligns the images. (we will discuss this ad nauseum in a later lecture).



Projective Transformation

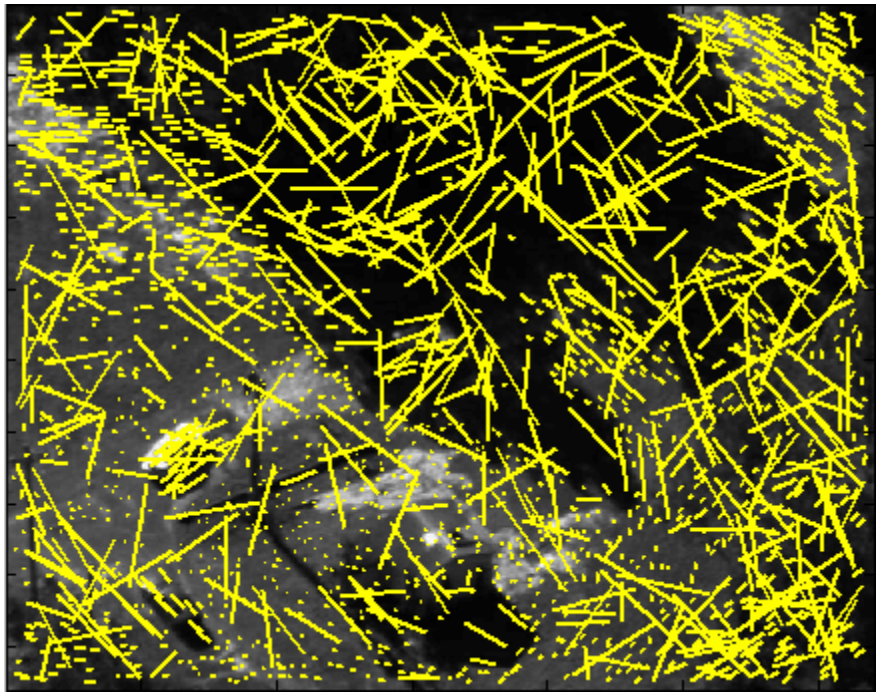
Stabilization Example

Step4: estimate projective transformation that aligns the images.

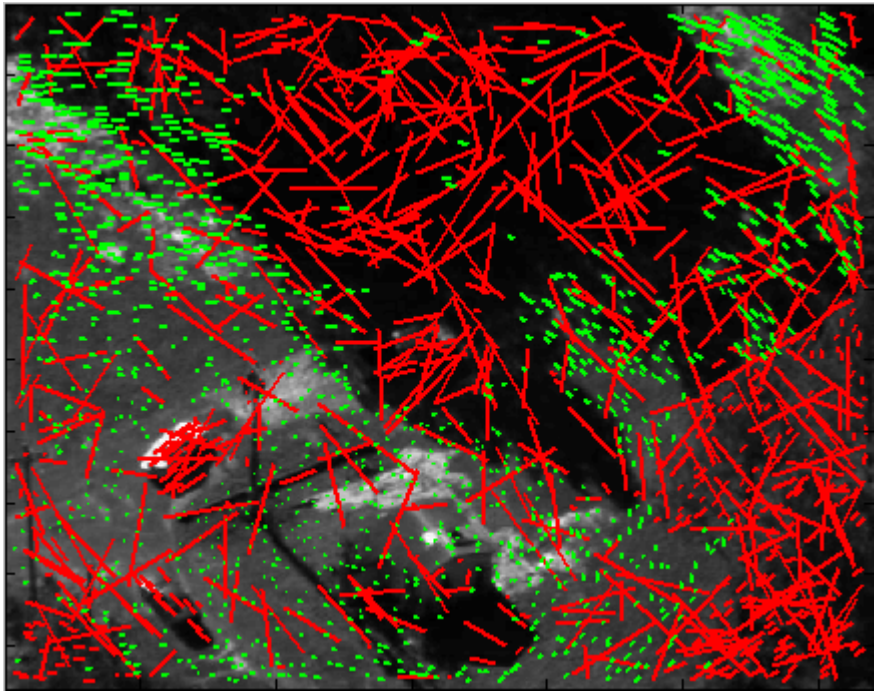
Now it is easier for people (and computers) to see the moving objects.

Stabilization Examples

Stabilization Examples



original point matches



labels from RANSAC

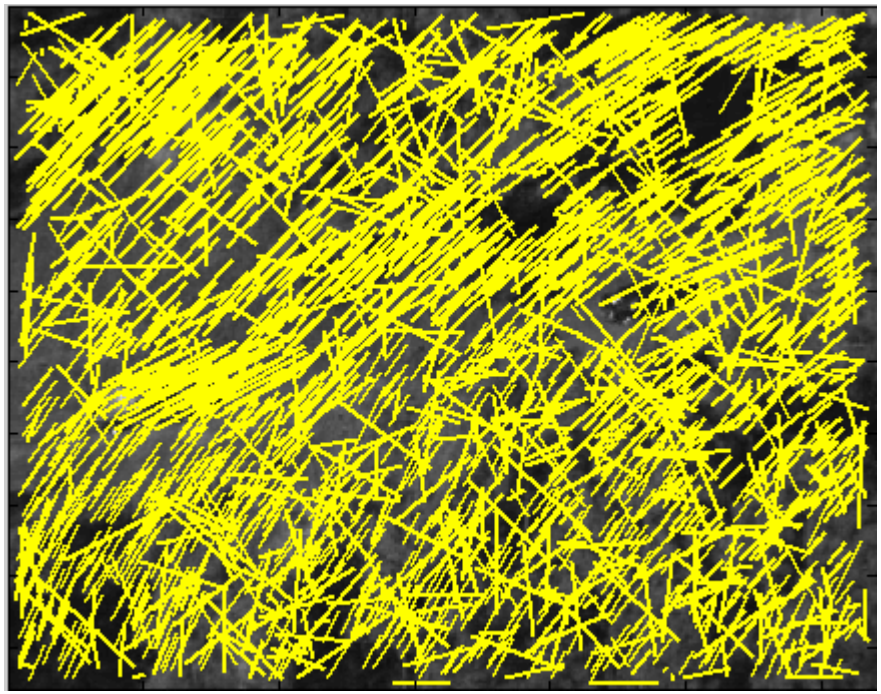
green: inliers

red: outliers

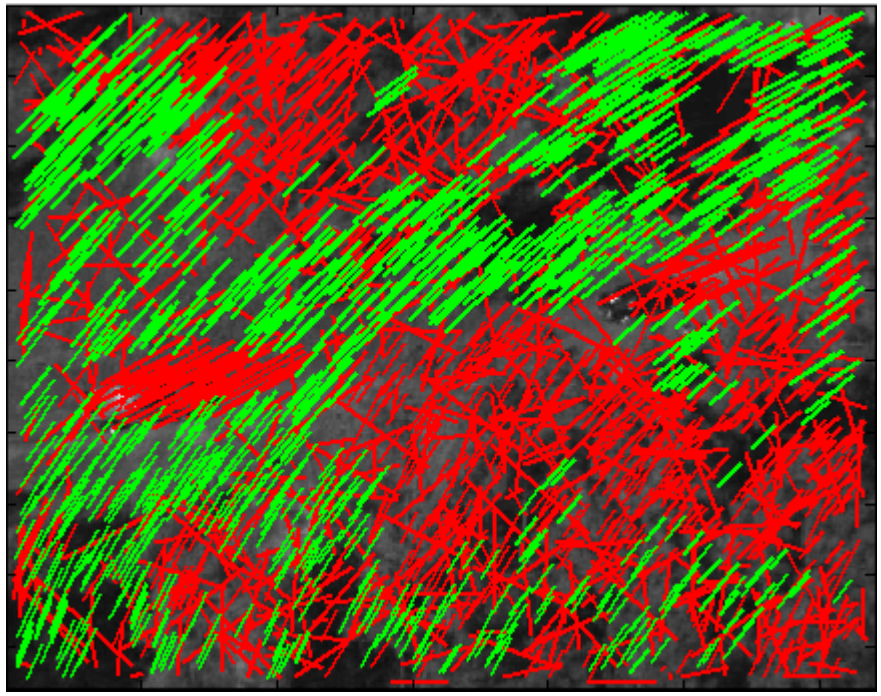
Stabilization Examples

Stabilization Examples

Stabilization Examples



original point matches

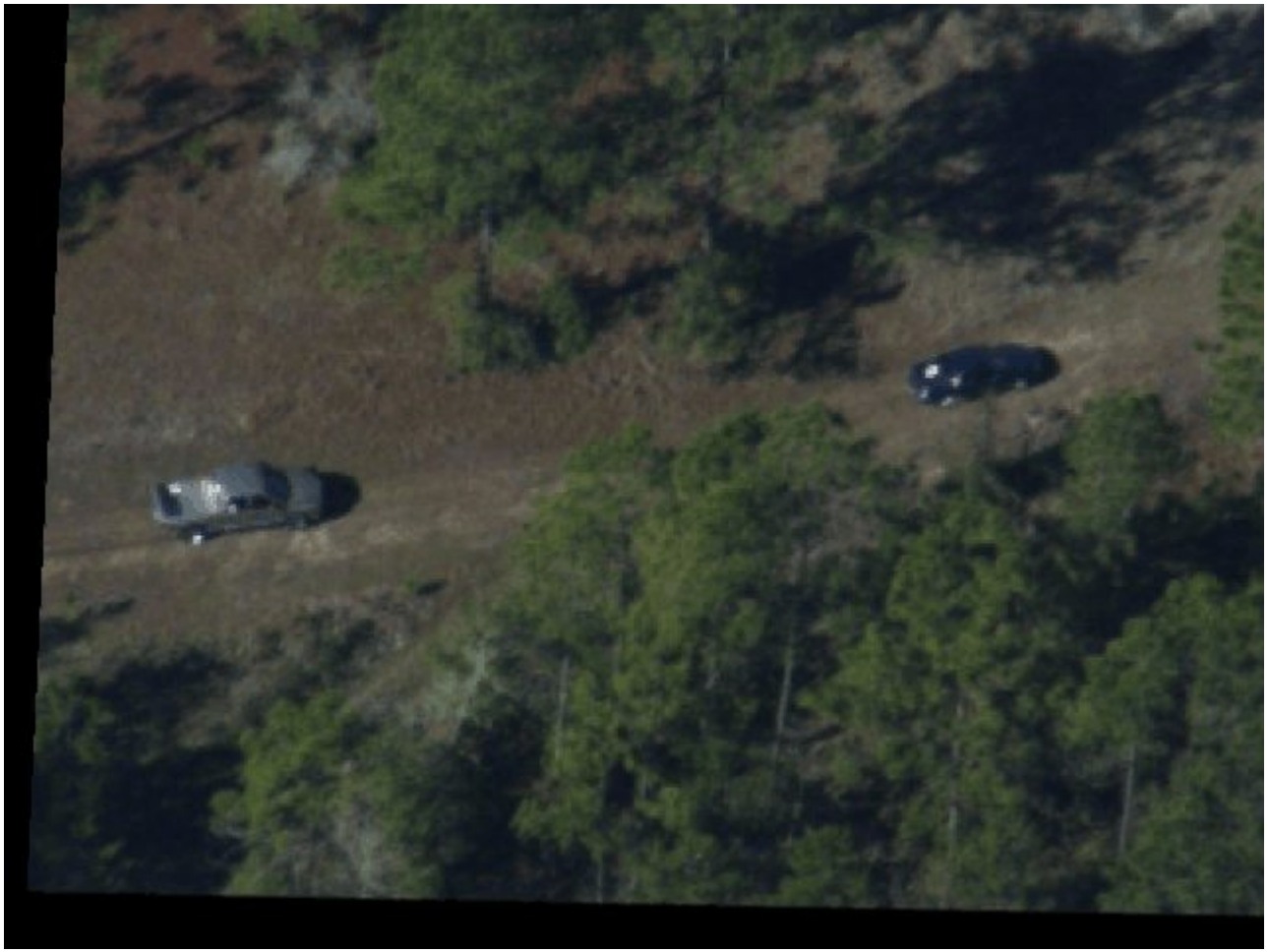


labels from RANSAC

green: inliers

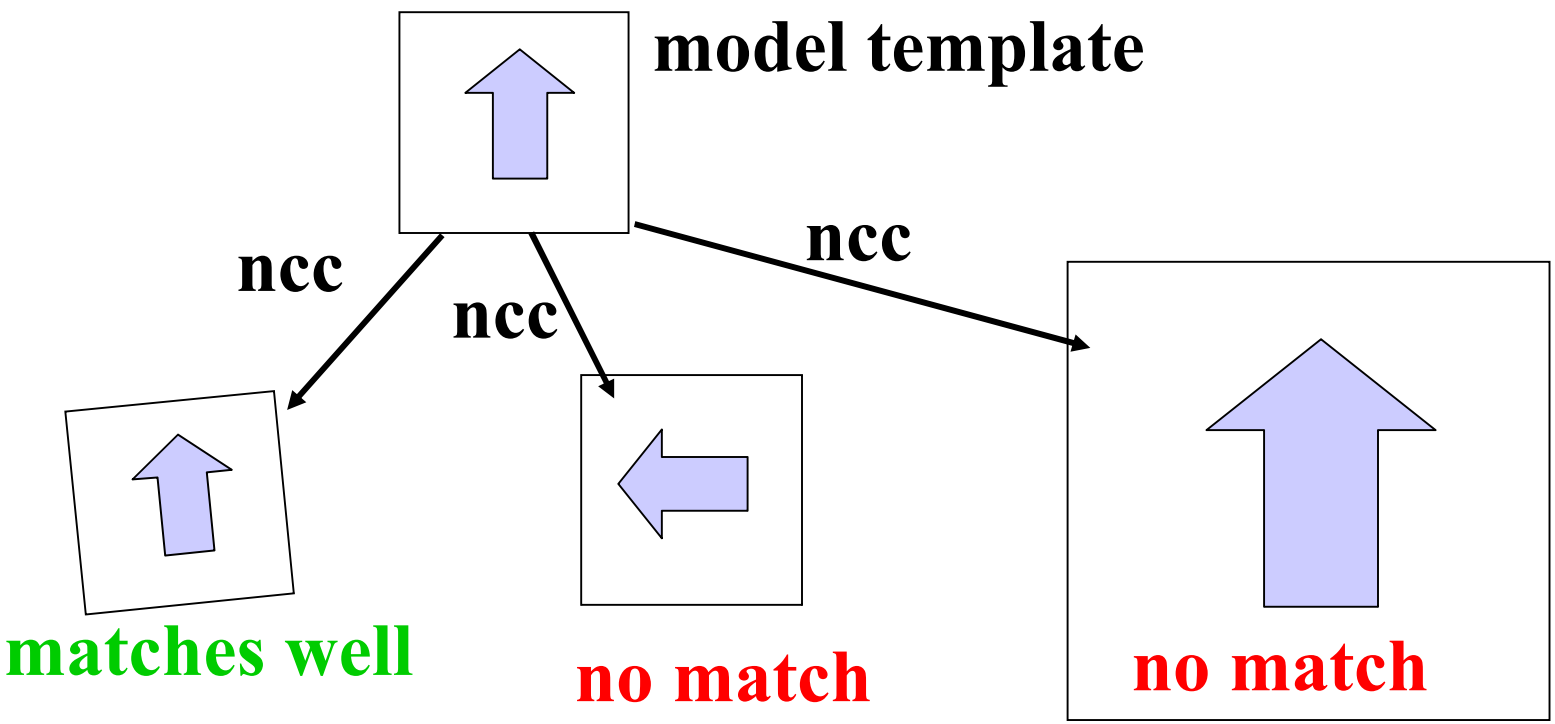
red: outliers

Stabilization Examples



Problem with Corners and NCC

Using NCC to match intensity patches puts restrictions on the amount of overall rotation and scaling allowed between the model and the image appearance.

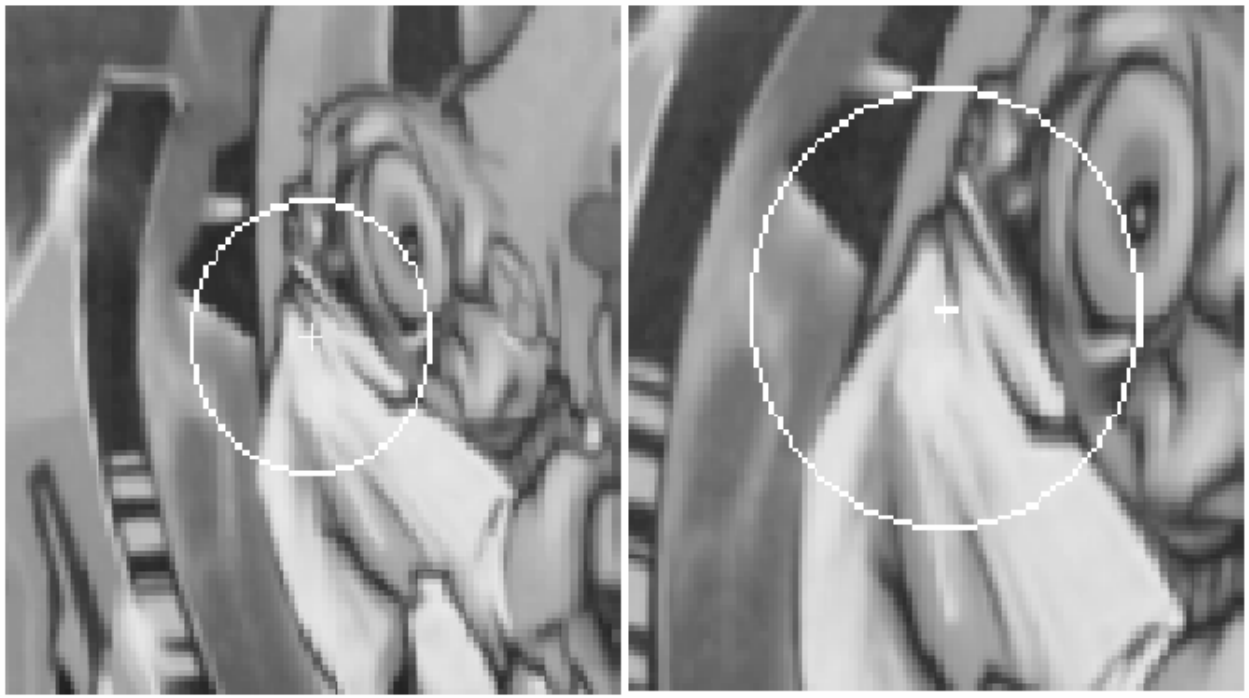


More General: Invariant Features

- Schmid & Mohr 1997, Lowe 1999, Baumberg 2000, Tuytelaars & Van Gool 2000, Mikolajczyk & Schmid 2001, Brown & Lowe 2002, Matas et. al. 2002, Schaffalitzky & Zisserman 2002

SIFT Keys: General Idea

- Reliably extract same image points regardless of new magnification and rotation of the image.
- Normalize image patches, extract feature vector
- Match feature vectors using correlation



SIFT Keys: General Idea

Want to detect/match same features regardless of

Translation : easy, almost every feature extraction and correlation matching algorithm in vision is translation invariant

Rotation : harder. Guess a canonical orientation for each patch from local gradients

Scaling : hardest of all. Create a multi-scale representation of the image and appeal to scale space theory to determine correct scale at each point.

Recall: Scale Space

Basic idea: different scales are appropriate for describing different objects in the image, and we may not know the correct scale/size ahead of time.

512 256 128 64 32 16 8

Scale Selection

Scale Selection Principle (T. Lindeberg):

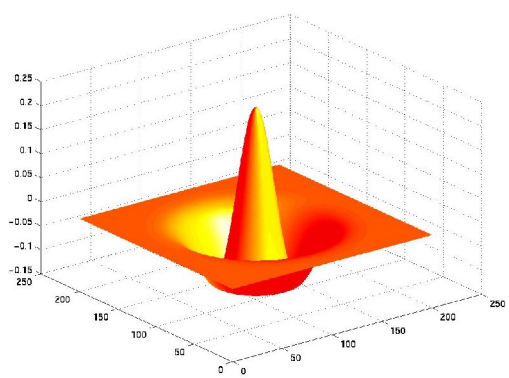
In the absence of other evidence, assume that a scale level, at which some (possibly non-linear) combination of normalized derivatives assumes a local maximum over scales, can be treated as reflecting a characteristic length of a corresponding structure in the data.

What are normalized derivatives?:

$$\sigma^{n+m} \frac{\partial^{(n+m)} f}{\partial x^n \partial x^m}$$

Example (using 2nd order derivatives):

$$\sigma^2 \nabla^2 f = \sigma^2 \underbrace{\left(\left(\frac{\partial^2 f}{\partial x^2} \right) + \left(\frac{\partial^2 f}{\partial y^2} \right) \right)}_{\text{“Laplacian” operator.}}$$



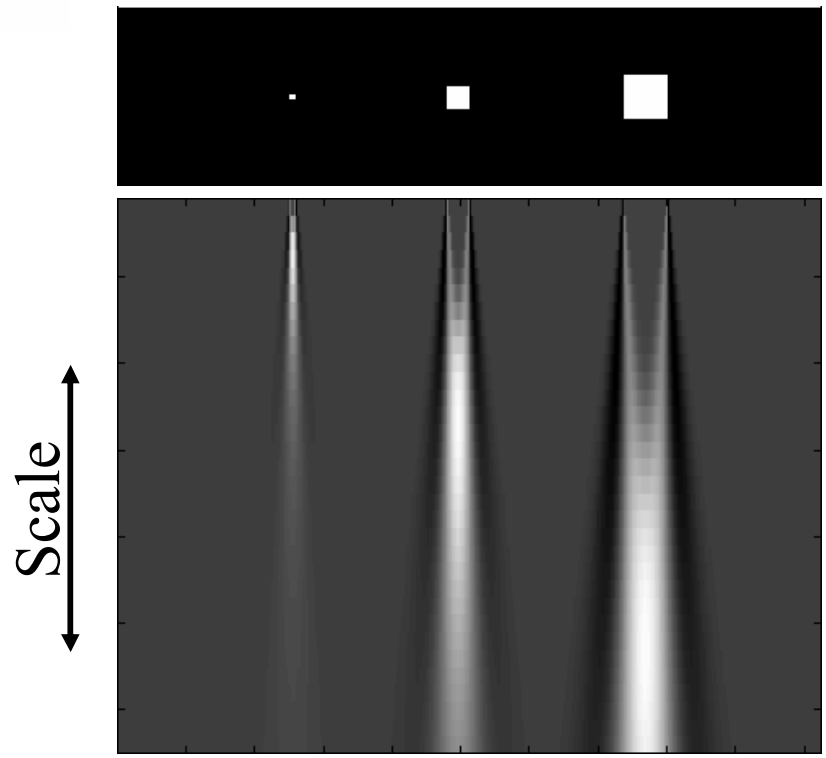
Local Scale Space Maxima

Lindeberg proposes that the natural scale for describing a feature is the scale at which a normalized derivative for detecting that feature achieves a local maximum both spatially and in scale.

$$\begin{cases} (\nabla(\mathcal{D}_{norm}L))(x_0;\ t_0) = 0, \\ (\partial_t(\mathcal{D}_{norm}L))(x_0;\ t_0) = 0. \end{cases}$$

DnormL is the DoG operator, in this case.

Example for blob detection



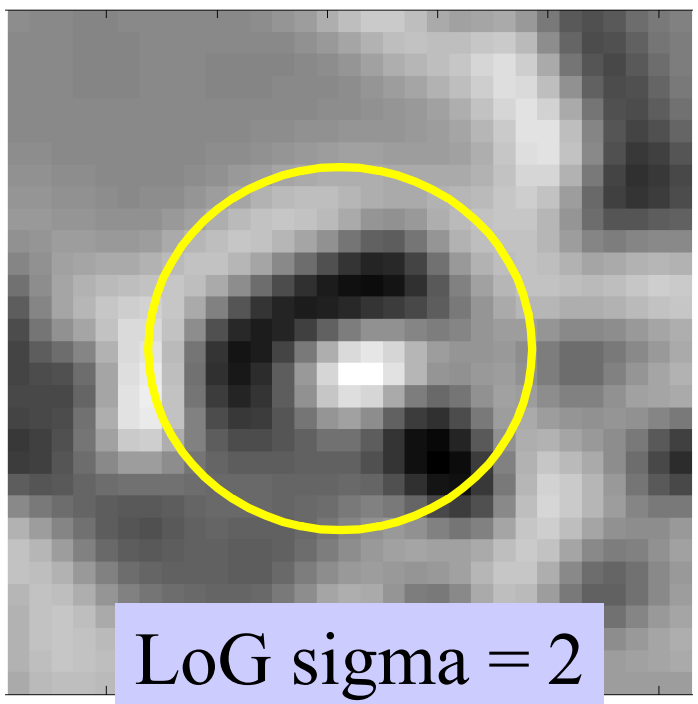
Recall: LoG Blob Finding

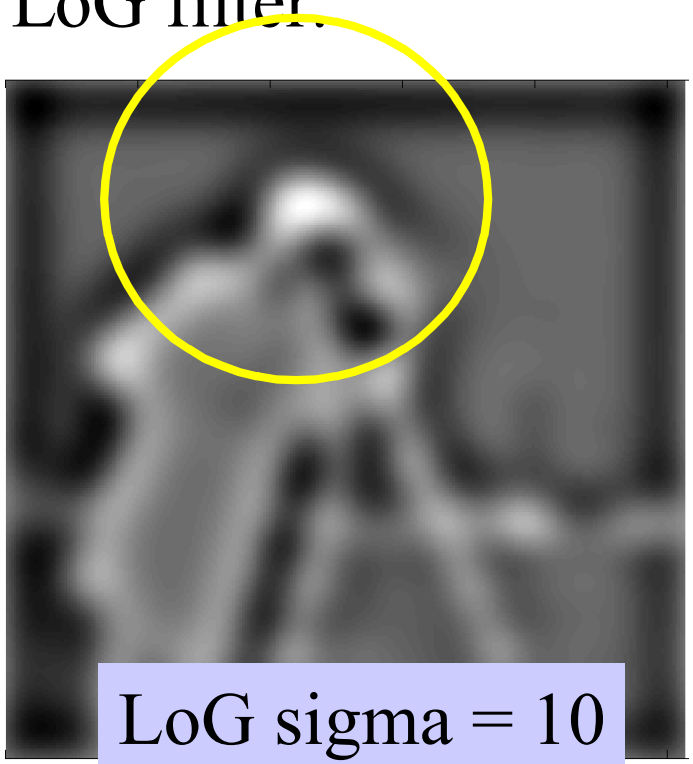
LoG filter extrema locates "blobs"

maxima = dark blobs on light background

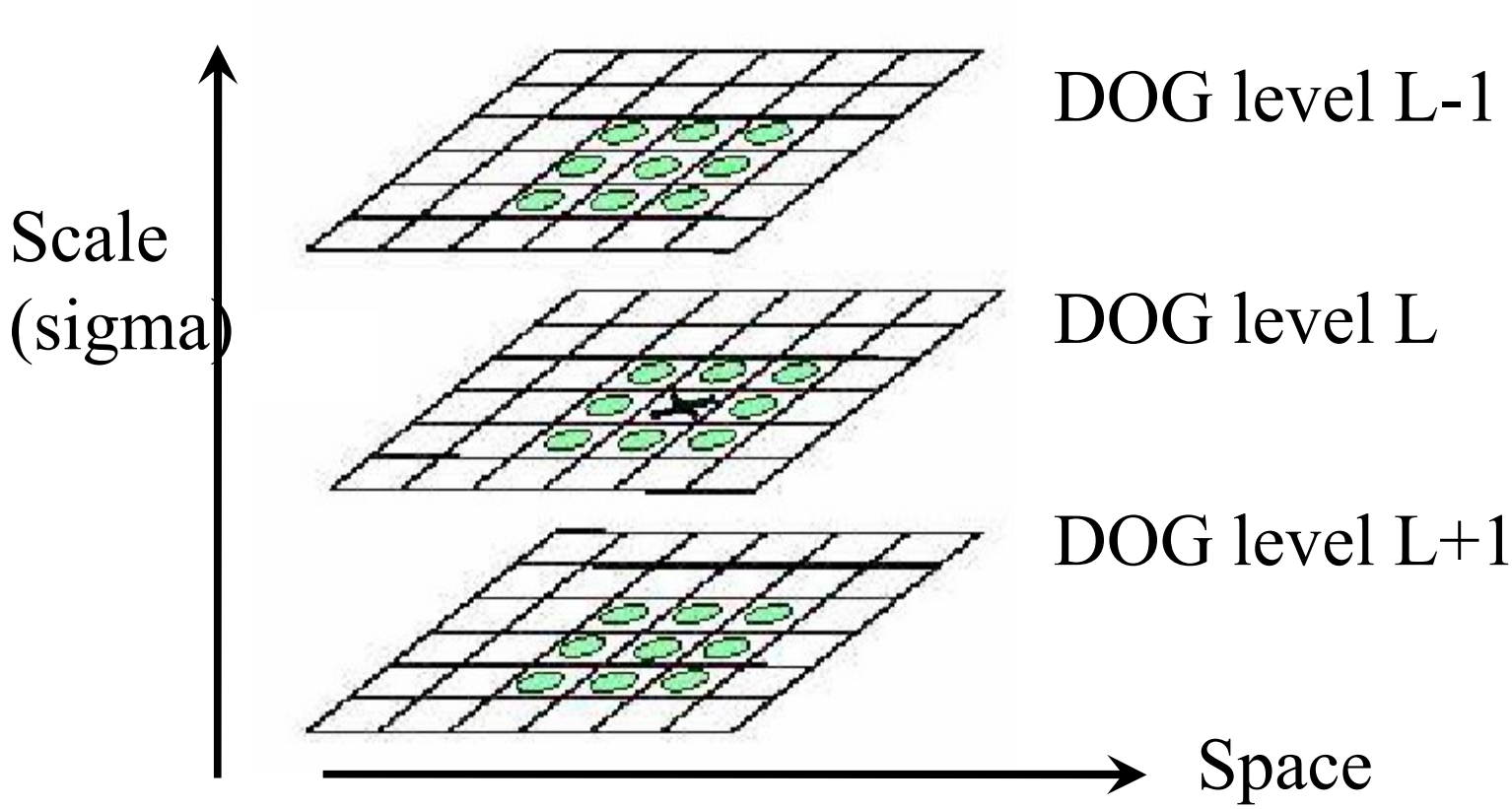
minima = light blobs on dark background

Scale of blob (size ; radius in pixels) is determined by the sigma parameter of the LoG filter.





Extrema in Space and Scale



Hint: when finding maxima or minima at level L, use DownSample or UpSample as necessary to make DOG images at level L-1 and L+1 the same size as L.

SIFT Keys: General Idea

Want to detect/match same features regardless of

Translation : easy, almost every feature extraction and correlation matching algorithm in vision is translation invariant

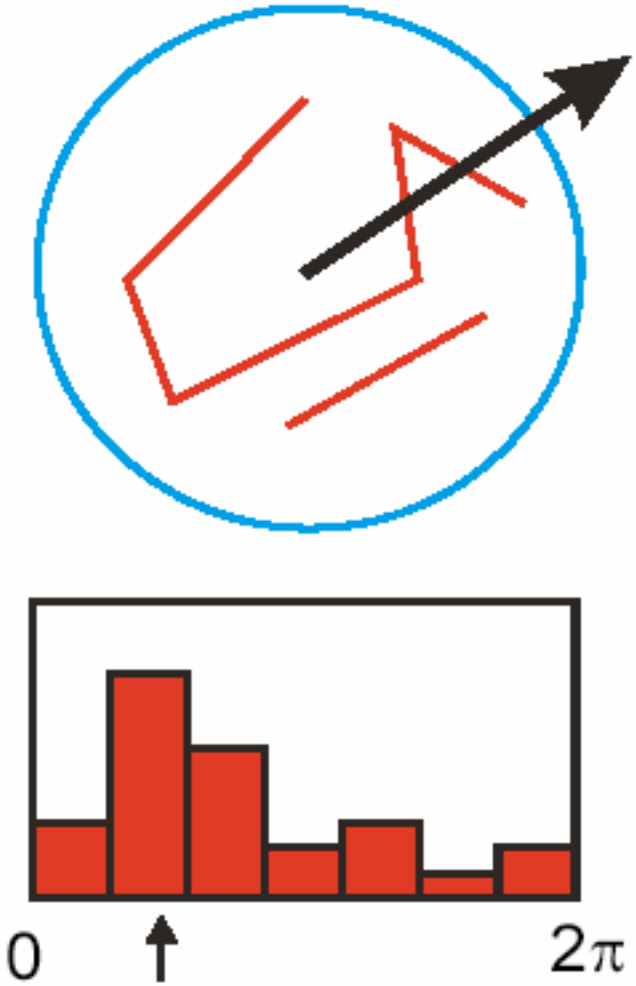
Rotation : harder. Guess a canonical orientation for each patch from local gradients

Scaling : hardest of all. Create a multi-scale representation of the image and appeal to scale space theory to determine correct scale at each point.

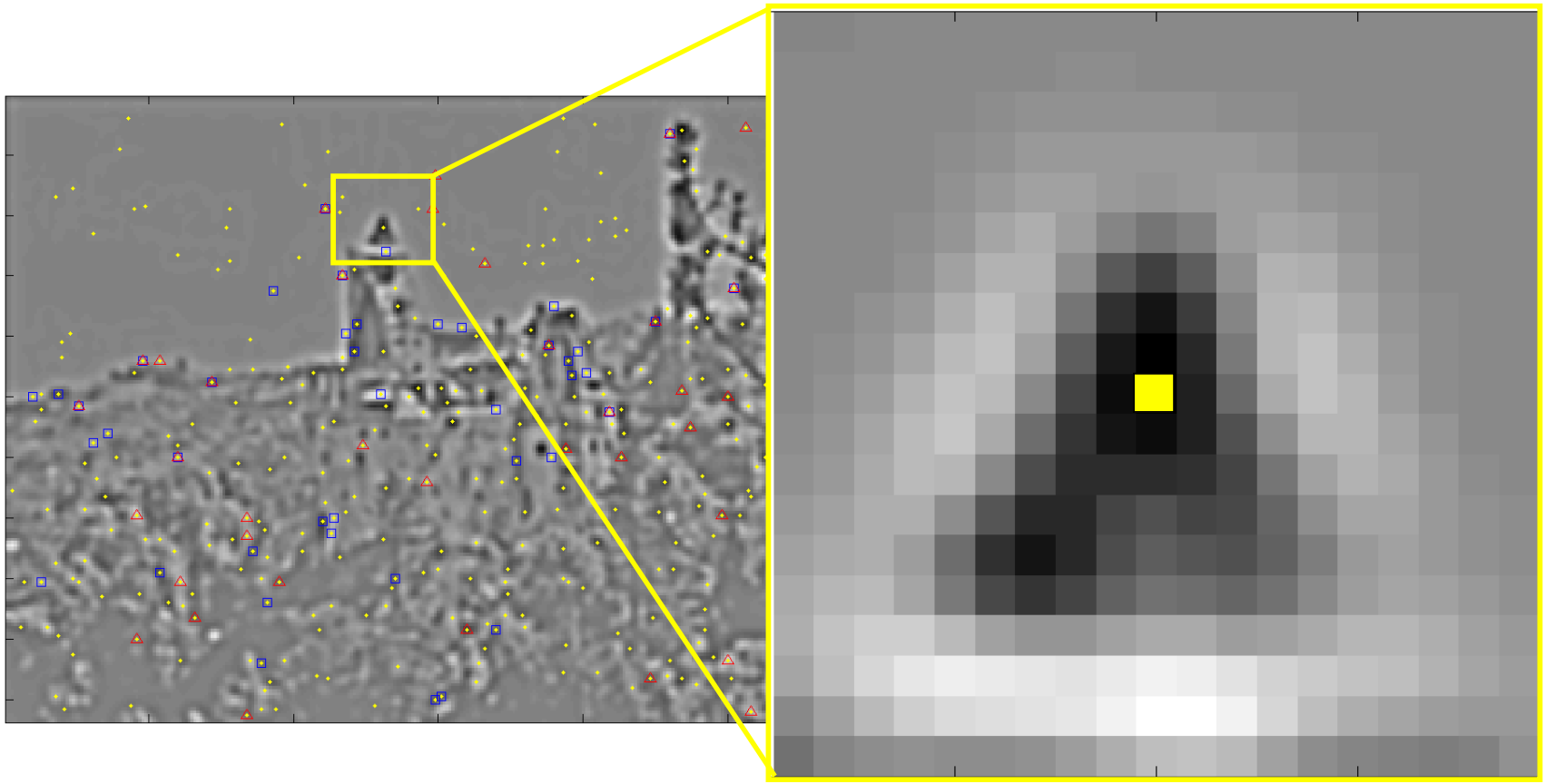
Sift Key Steps

Select canonical orientation

- Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)

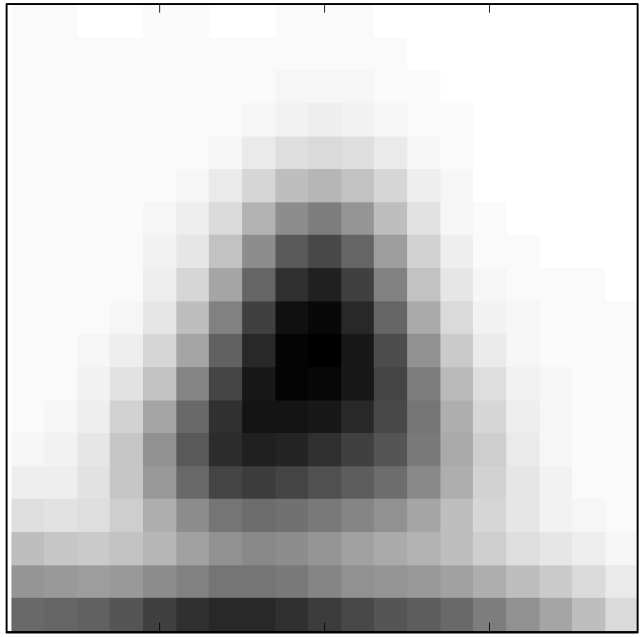


Example



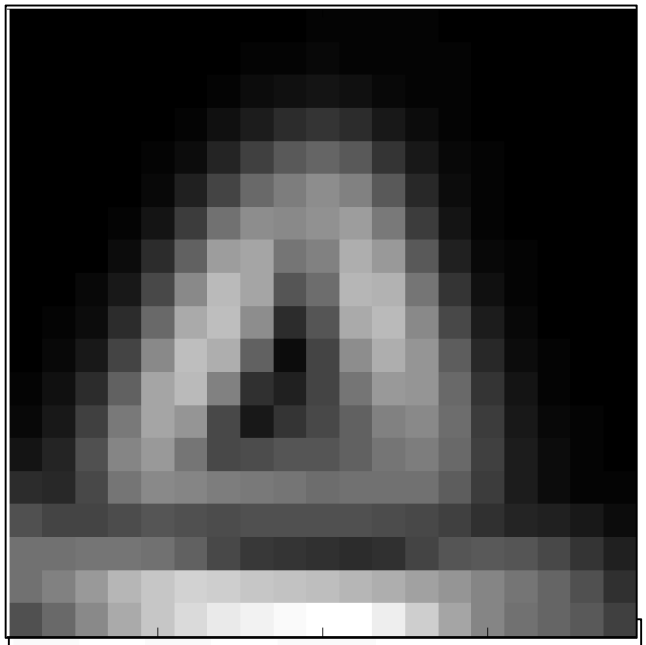
- Keypoint location = extrema location
- Keypoint scale is scale of the DOG image

Example (continued)

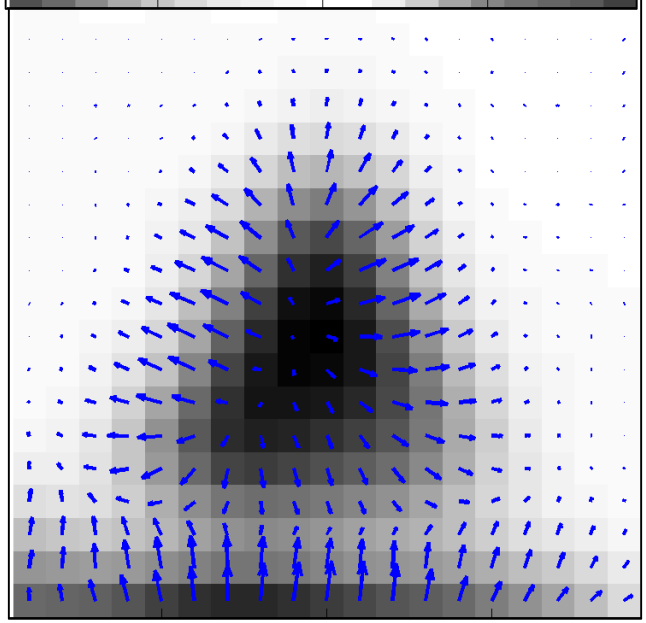


gaussian image
(at closest scale,
from pyramid)

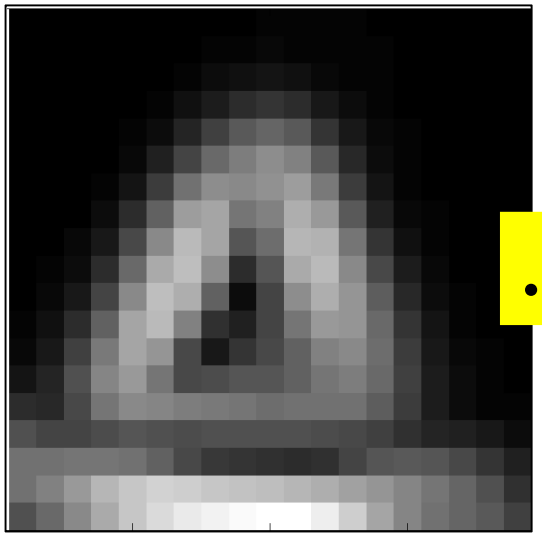
gradient
magnitude



gradient
orientation

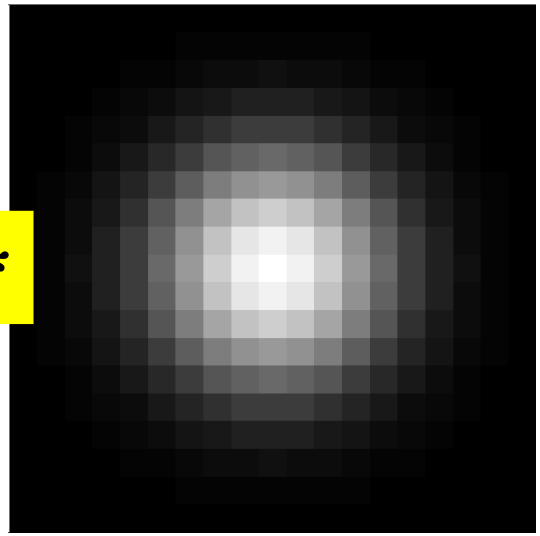


Example (continued)



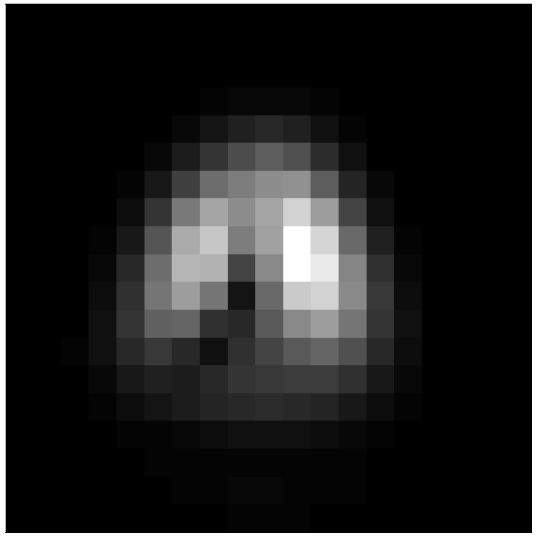
gradient magnitude

.*



weighted by 2D gaussian kernel

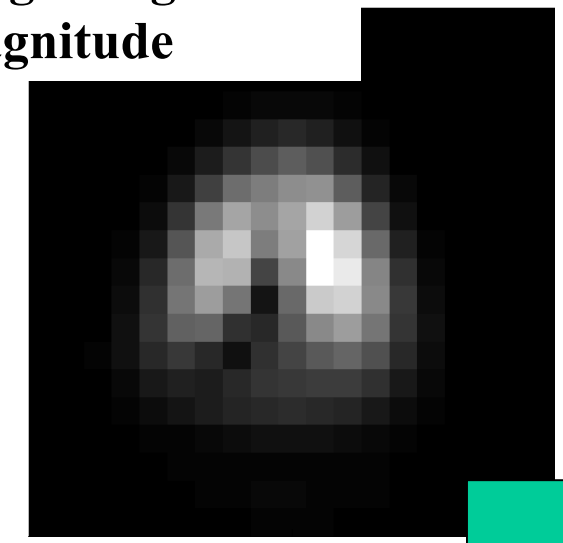
=



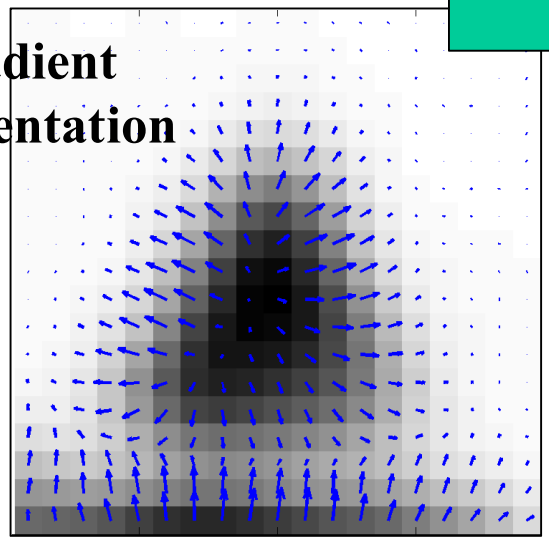
weighted gradient magnitude

Example (continued)

weighted gradient magnitude

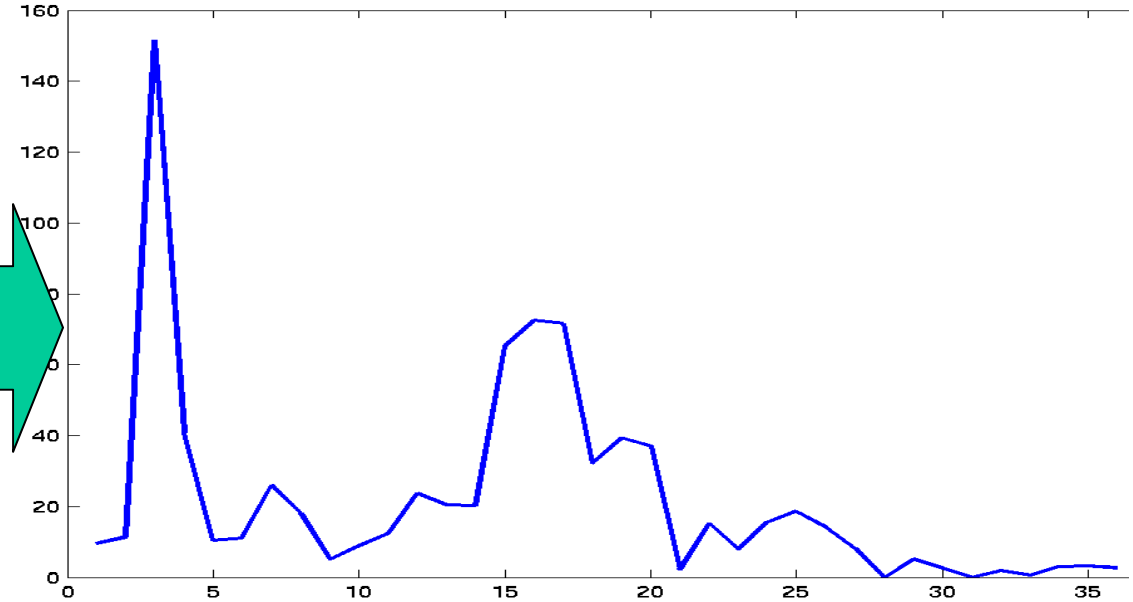


gradient orientation



weighted orientation histogram.

Each bucket contains sum of weighted gradient magnitudes corresponding to angles that fall within that bucket.



36 buckets

10 degree range of angles in each bucket, i.e.

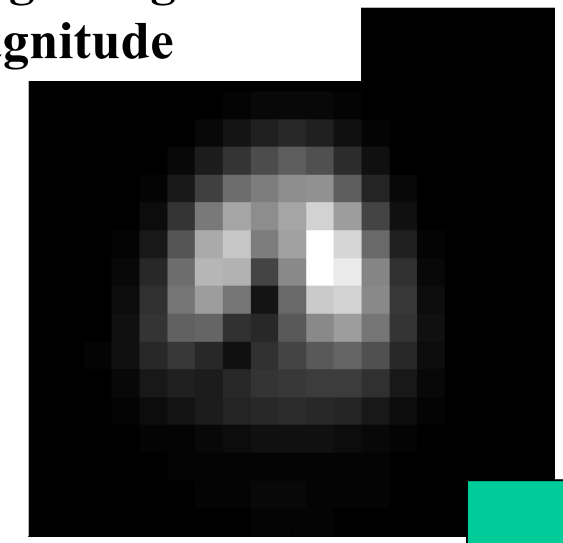
0 <=ang<10 : bucket 1

10<=ang<20 : bucket 2

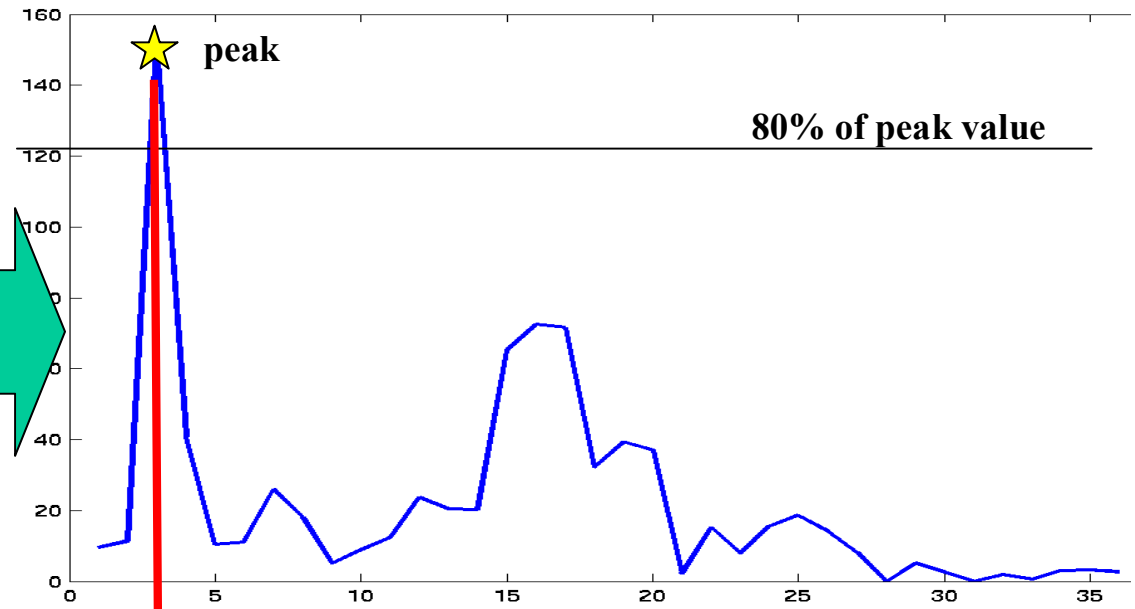
20<=ang<30 : bucket 3 …

Example (continued)

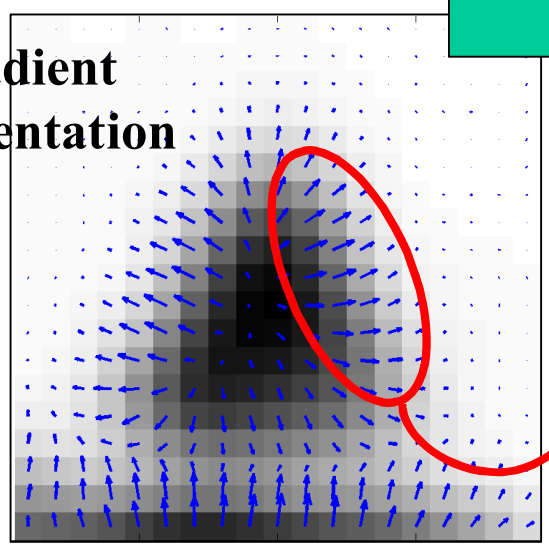
weighted gradient magnitude



weighted orientation histogram.



gradient orientation

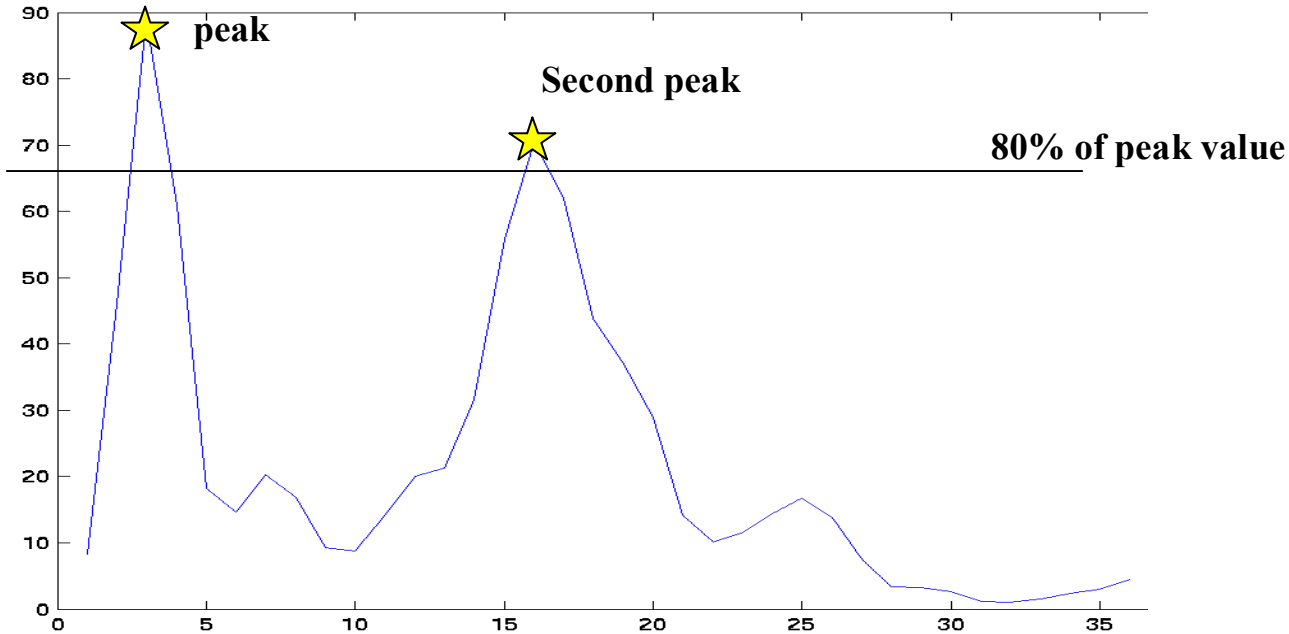


20-30 degrees

Orientation of keypoint is approximately 25 degrees

Example (continued)

There may be multiple orientations.



In this case, generate duplicate keypoints, one with orientation at 25 degrees, one at 155 degrees.

Design decision: you may want to limit number of possible multiple peaks to two.

Example of KeyPoint Detection

Each keypoint has a center point (location), an orientation (rotation) and a radius (scale).

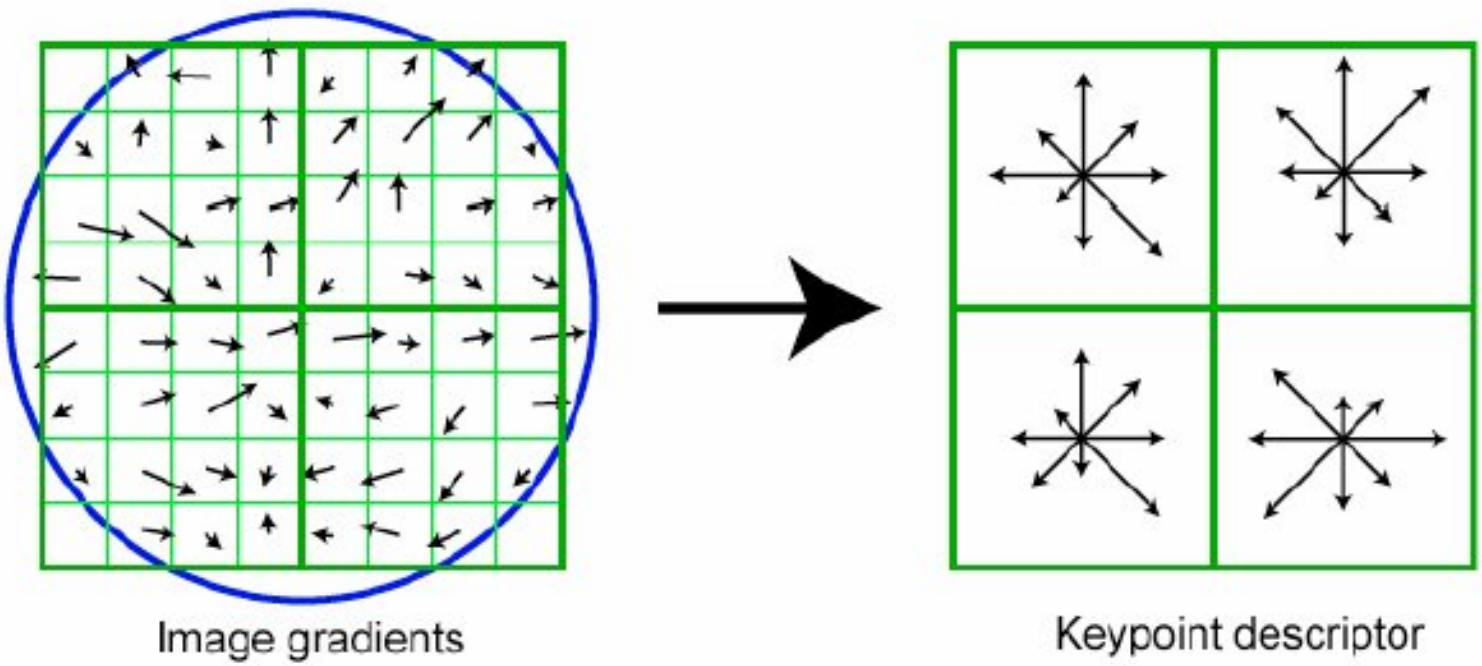
At this point, we could try to correlate patches (after first normalizing to a canonical orientation and scale).

SIFT Vector

to make things more insensitive to changes in lighting or small changes in geometry, Lowe constructs feature vector from image gradients.

SIFT Vector

- Thresholded image gradients are sampled over 16x16 array of locations in scale space
- Create array of orientation histograms
- 8 orientations x 4x4 histogram array = 128 dimensions



Sift Key Matching

- Sift keys are matches by nearest neighbor (shortest distance between SIFT key vectors)

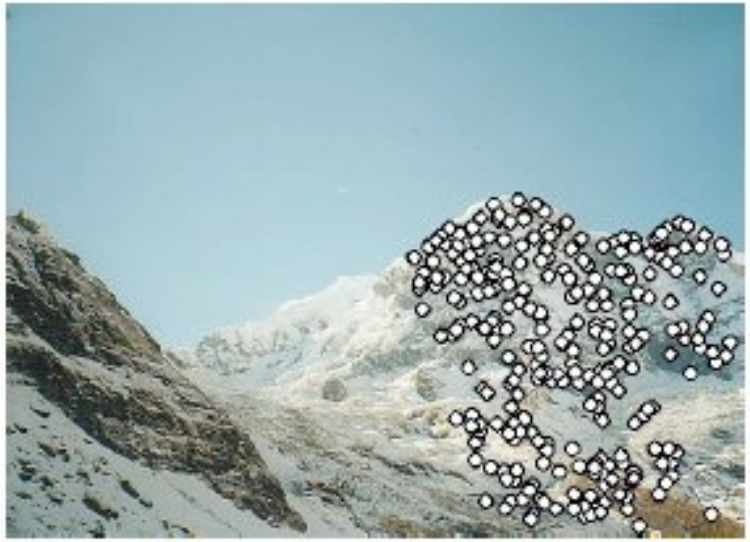
- Can use k-nearest neighbor to generate more potential matches.

- Find consistent (inlier) matches using RANSAC

Generating Panoramas

(a) Image 1

(b) Image 2



(c) SIFT matches 1

(d) SIFT matches 2

Brown and Lowe, ICCV03

Generating Panoramas

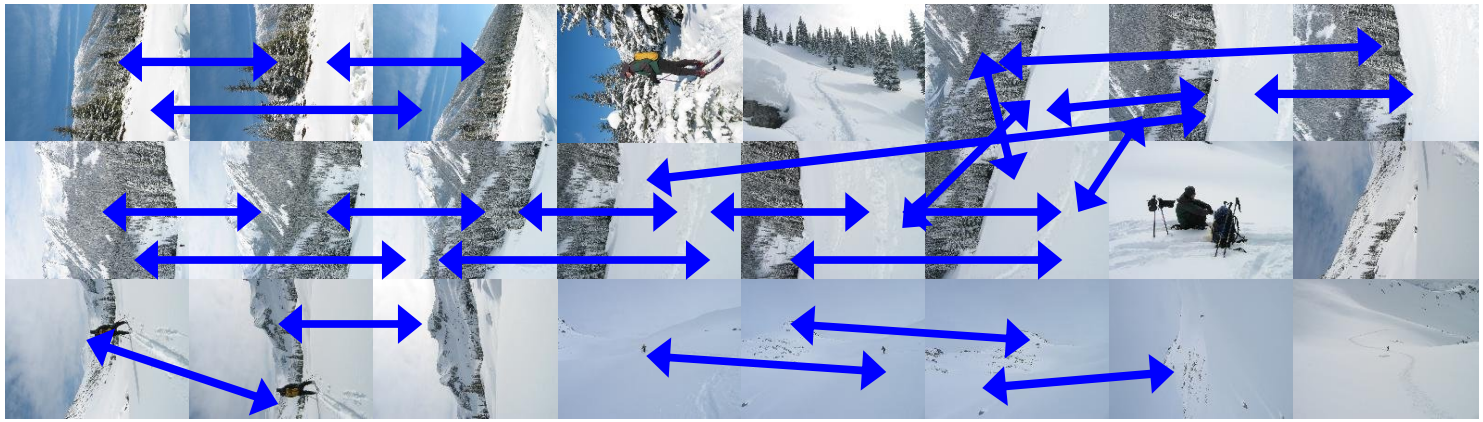
(e) Images aligned according to a homography

Brown and Lowe, ICCV03

Brown and Lowe Autostitch (ICCV03)

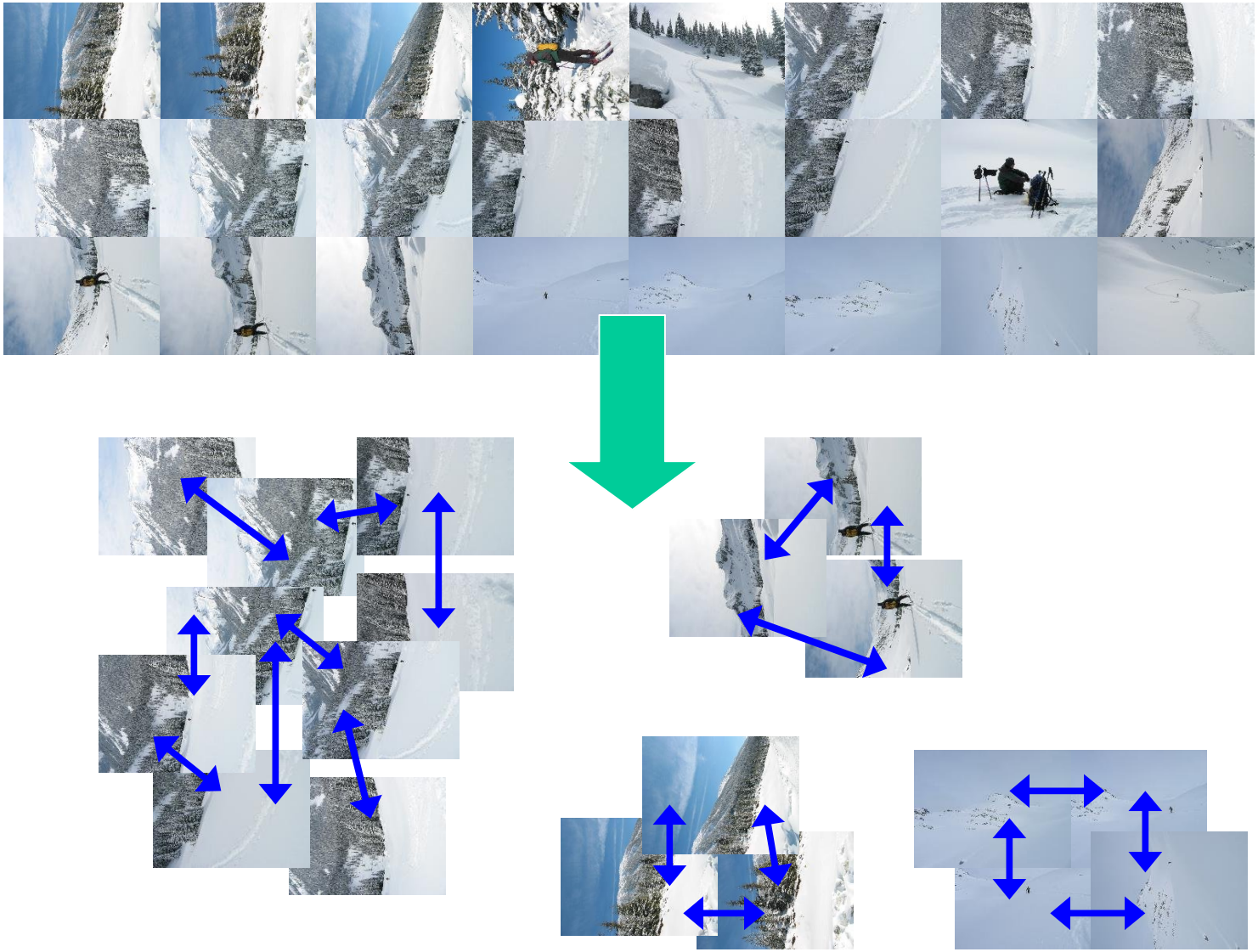
Given unsorted images downloaded from camera:

Hypothesis pairs of matching images with high overlap

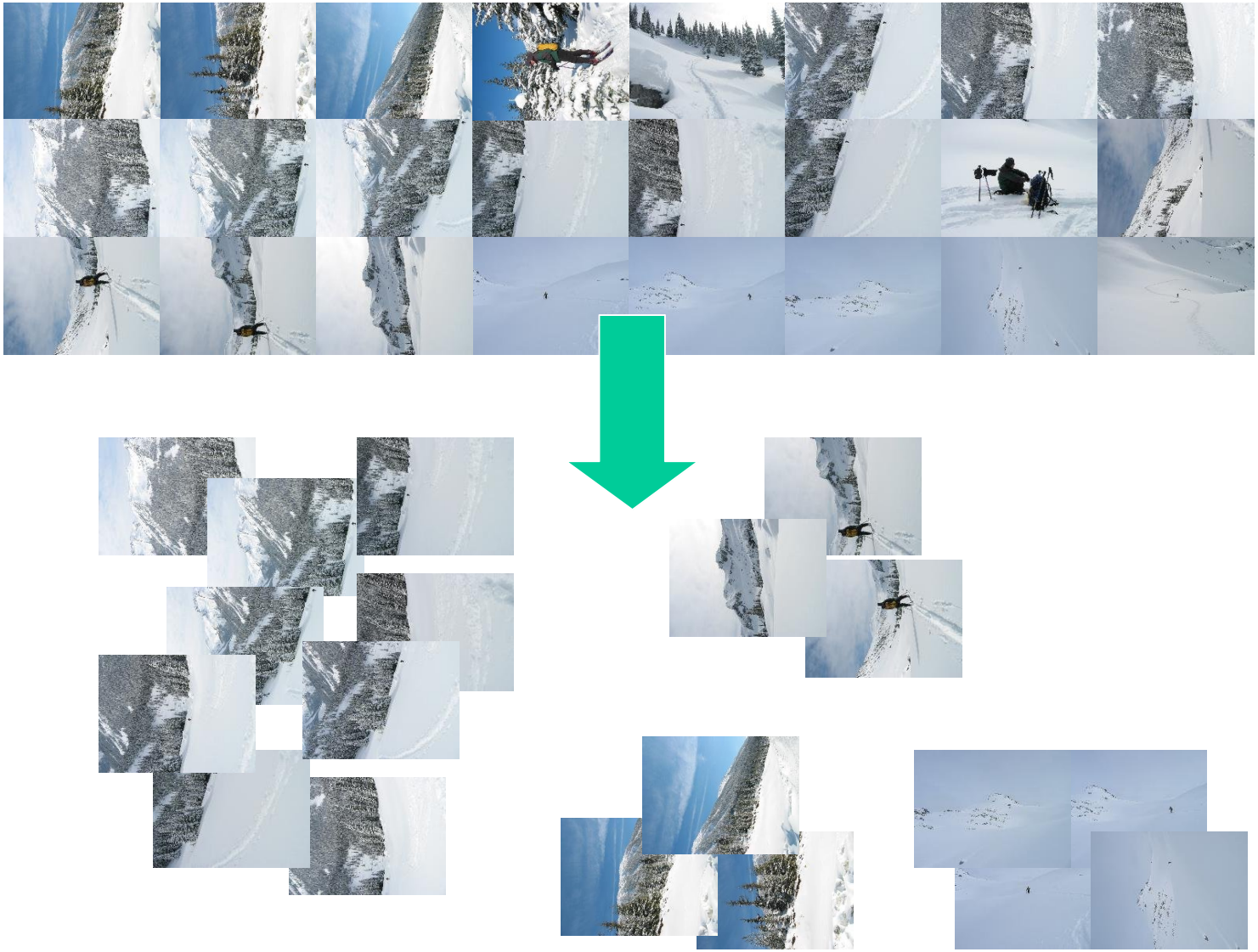


Finding the panoramas

Find connected components of images with high overlap

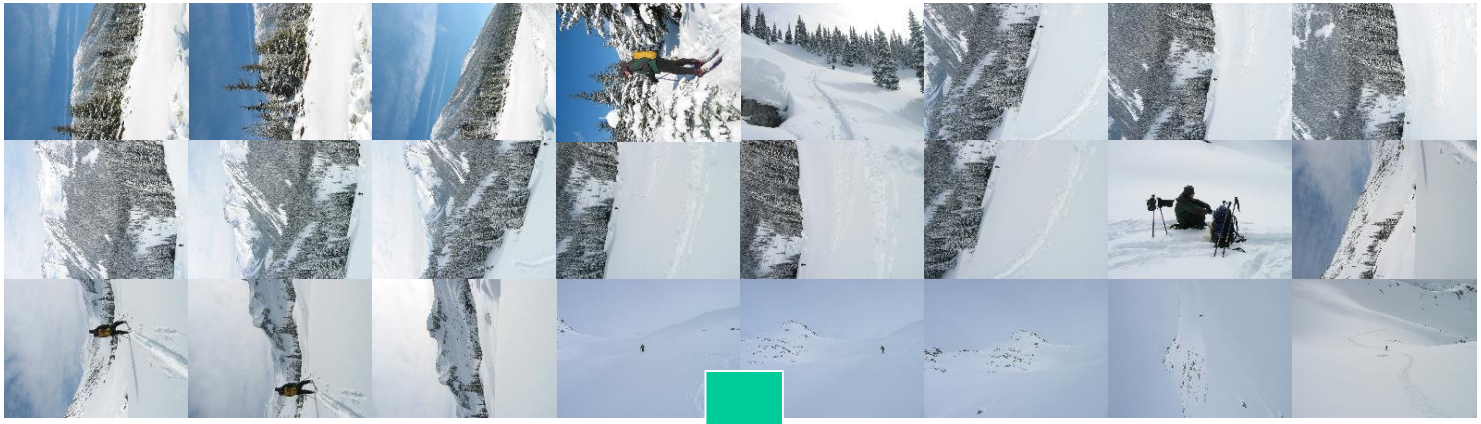


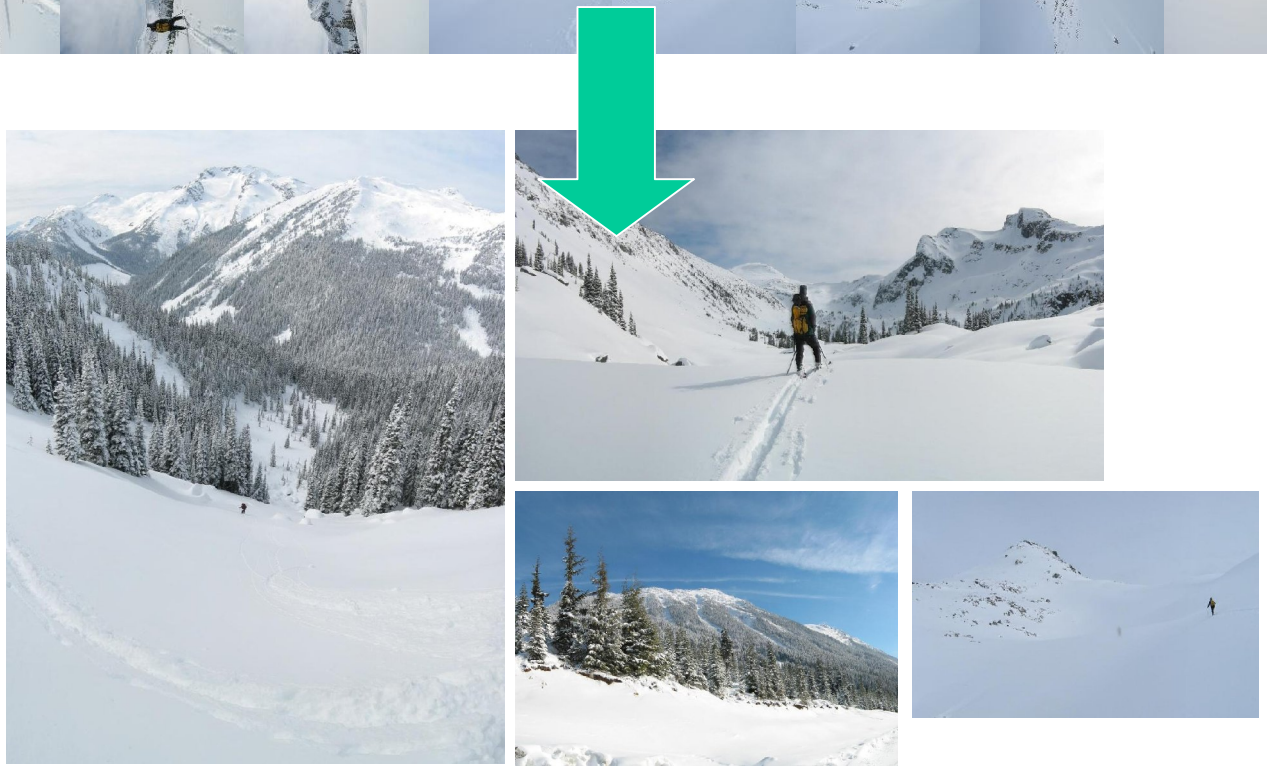
Finding the panoramas



Finding the panoramas

make a mosaic out of each connected component





Open-Source Software

- Hugin

  http://hugin.sourceforge.net/

http://www.flickriver.com/photos/tags/hugin/interesting/

Assignment

- **Produce one (or more) mosaics and post them to our flickr course site.**