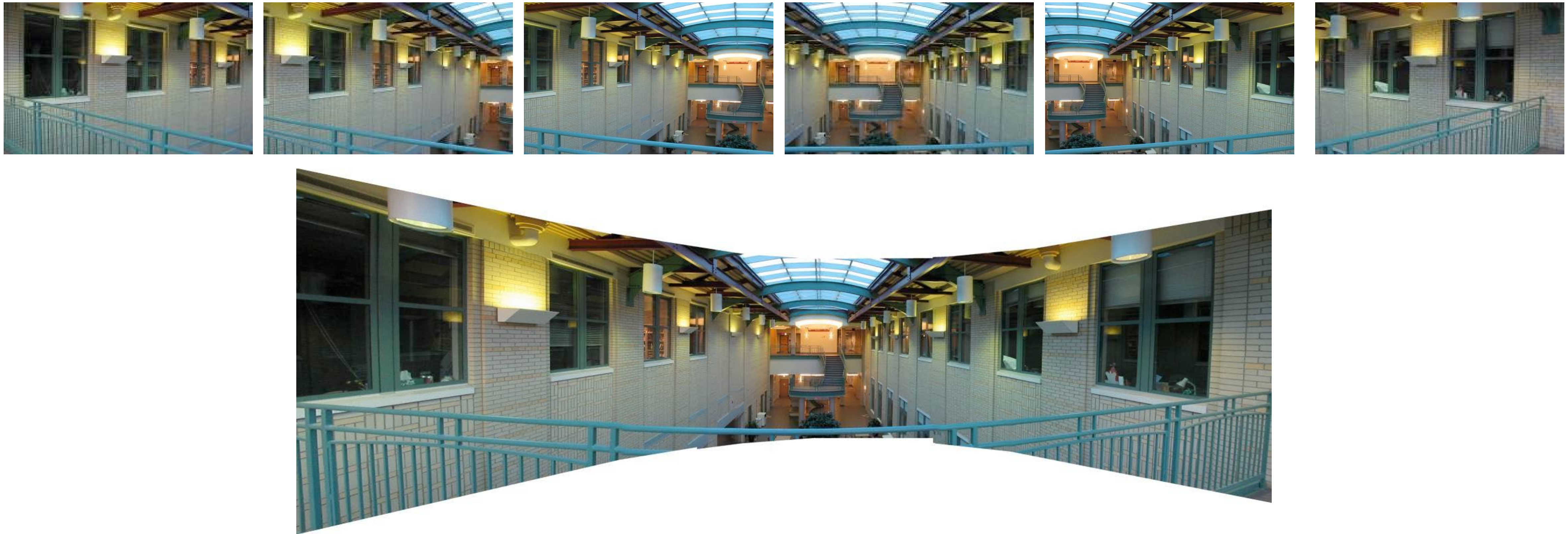
CMSC 491/691

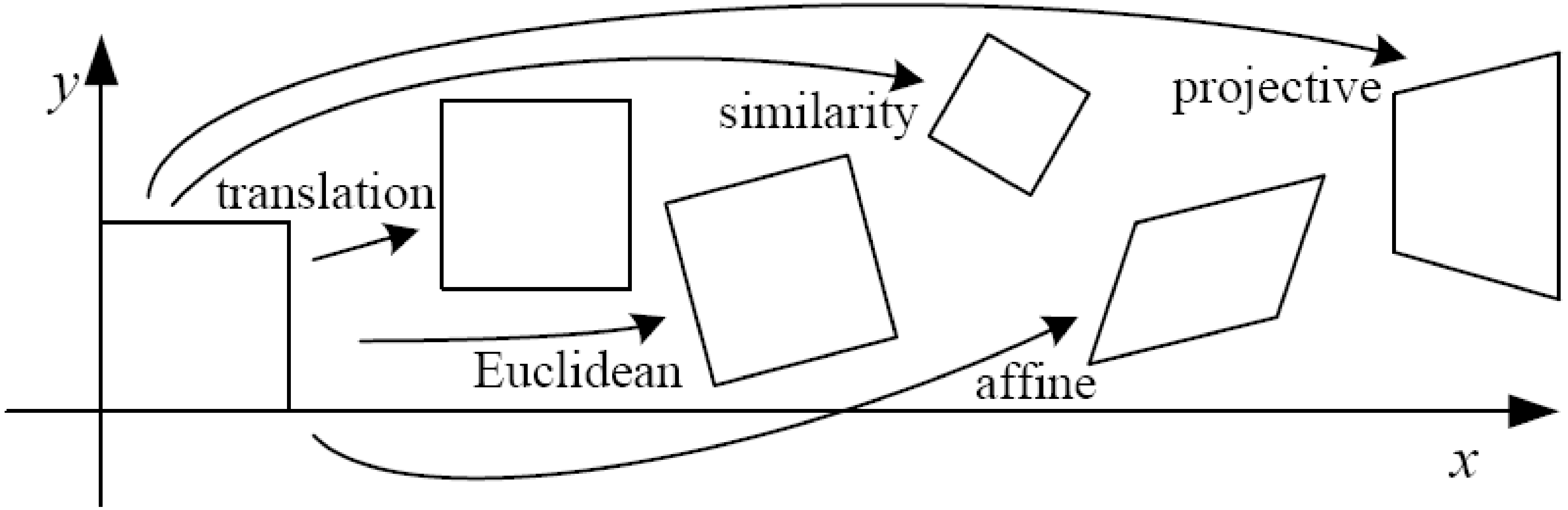
Lecture 12

Image Transformations II: Homographies



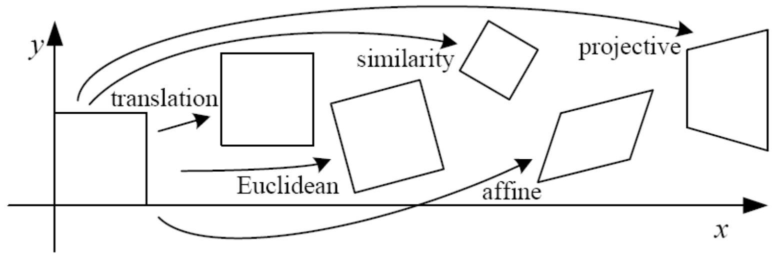
Welcome Back From Spring Break ...

Recap: Image Transformations



Classification of 2D transformations

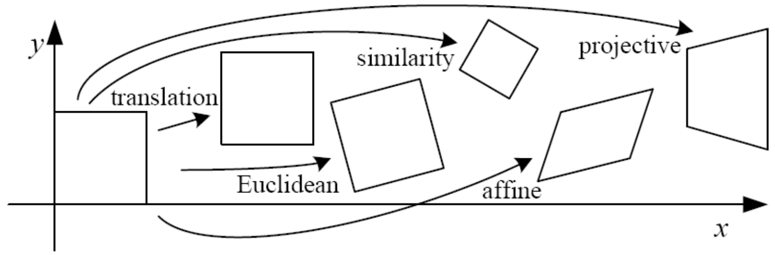
Translation: $$\begin{bmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{bmatrix}$$



Classification of 2D transformations

Euclidean (rigid):
rotation + translation

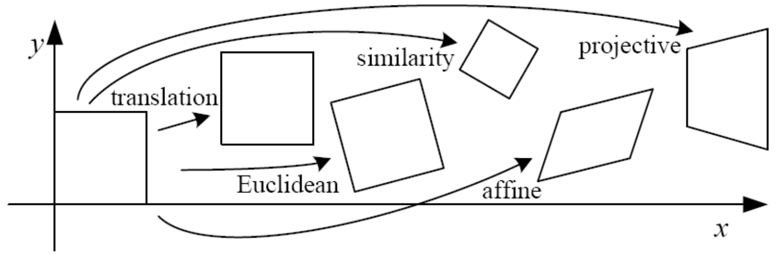
$$\begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$



Classification of 2D transformations

Euclidean (rigid):
rotation + translation

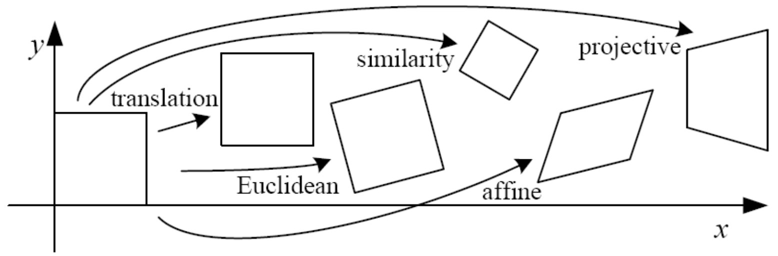
$$\begin{bmatrix} \cos\theta & -\sin\theta & r_3 \\ \sin\theta & \cos\theta & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$



Classification of 2D transformations

Similarity:
uniform scaling + rotation
+ translation

$$\begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

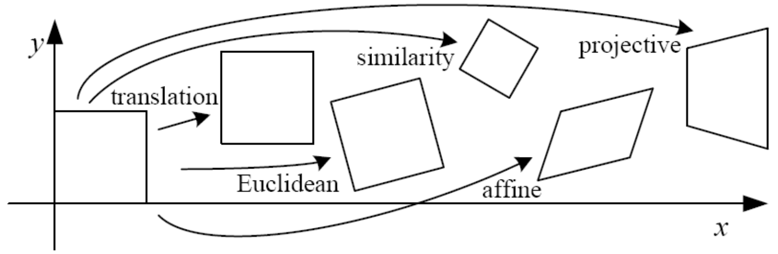


Classification of 2D transformations

multiply these four by scale **s**

Similarity:
uniform scaling + rotation
+ translation

$$\begin{bmatrix} \boxed{\begin{matrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{matrix}} & \begin{matrix} r_3 \\ r_6 \end{matrix} \\ \begin{matrix} 0 & 0 \end{matrix} & 1 \end{bmatrix}$$



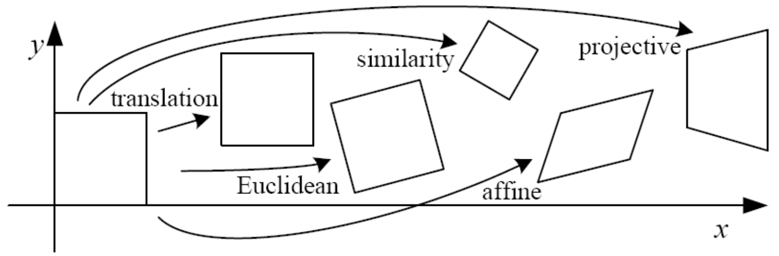
Classification of 2D transformations

Affine transform

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

$$A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \cos(-\Phi) & -\sin(-\Phi) \\ \sin(-\Phi) & \cos(-\Phi) \end{bmatrix} \cdots$$
$$\cdots \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \cos\Phi & -\sin\Phi \\ \sin\Phi & \cos\Phi \end{bmatrix}$$

Linear part can be decomposed



Affine transformations

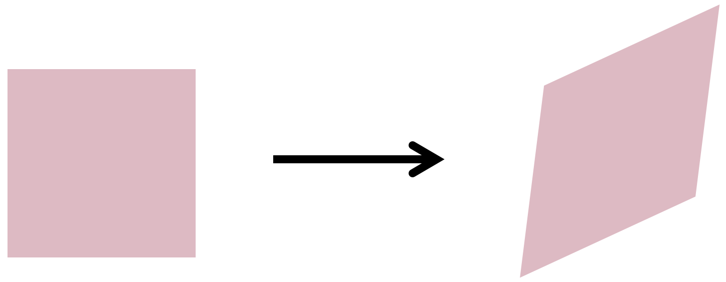
Affine transformations are combinations of

- arbitrary (4-DOF) linear transformations; and
- translations

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

Properties of affine transformations:

- origin does not necessarily map to origin
- lines map to lines
- parallel lines map to parallel lines
- ratios are preserved
- compositions of affine transforms are also affine transforms

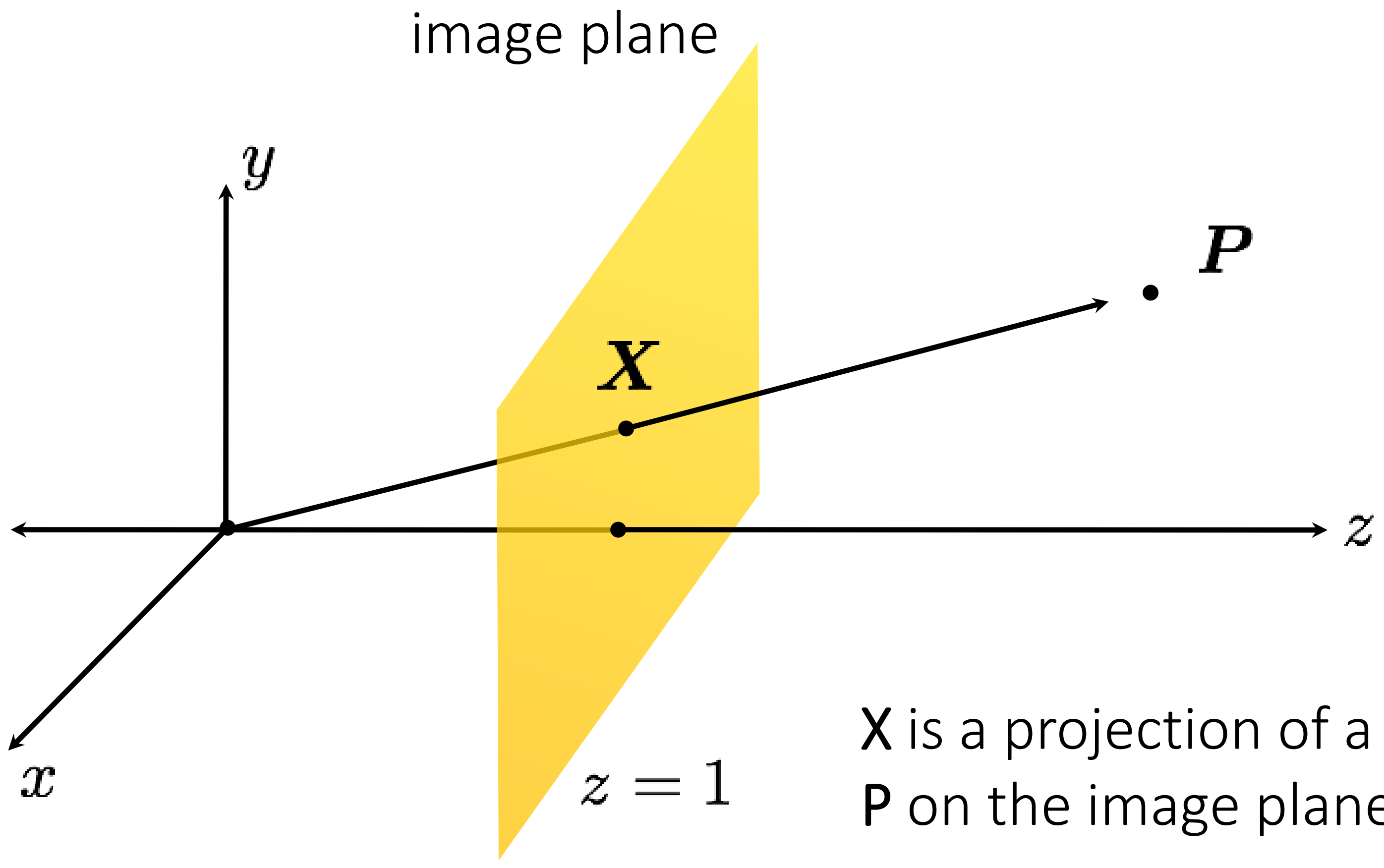


Projective transformations

image point in pixel coordinates $\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$\Updownarrow$$

image point in heterogeneous coordinates $\boldsymbol{X} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$



X is a projection of a point P on the image plane

Projective transformations

Projective transformations are combinations of

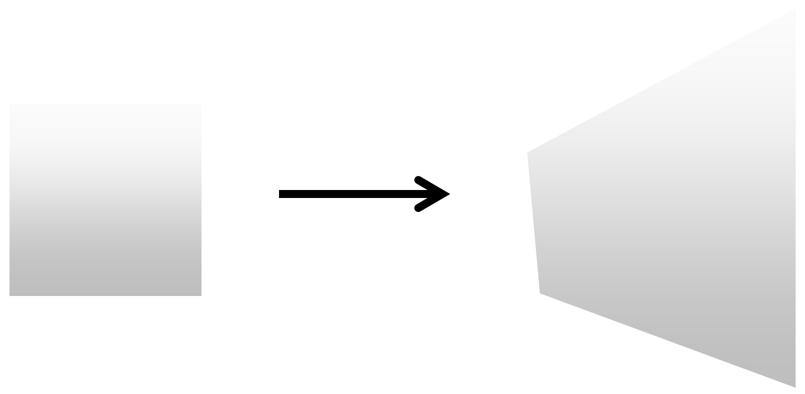
- affine transformations; and
- projective wraps

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

How many degrees of freedom?

Properties of projective transformations:

- origin does not necessarily map to origin
- lines map to lines
- parallel lines do not necessarily map to parallel lines
- ratios are not necessarily preserved
- compositions of projective transforms are also projective transforms



Projective transforms = 8Dof

$$k_{p2}\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} k_{p1}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\downarrow$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \frac{k_{p1}}{k_{p2}}\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\downarrow$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \frac{k_{p1}}{k_{p2}}\begin{bmatrix} a_{11}/a_{33} & a_{12}/a_{33} & a_{13}/a_{33} \\ a_{21}/a_{33} & a_{22}/a_{33} & a_{23}/a_{33} \\ a_{31}/a_{33} & a_{32}/a_{33} & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\nearrow$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = k\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Projective transformations

Projective transformations are combinations of

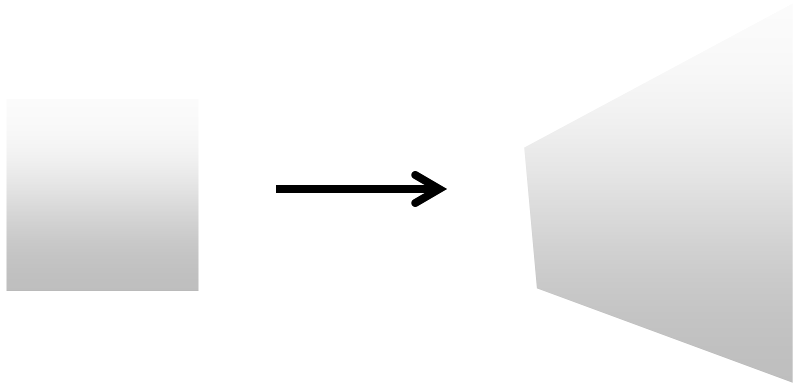
- affine transformations; and
- projective wraps

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

8 DOF: vectors (and therefore matrices) are defined up to scale)

Properties of projective transformations:

- origin does not necessarily map to origin
- lines map to lines
- parallel lines do not necessarily map to parallel lines
- ratios are not necessarily preserved
- compositions of projective transforms are also projective transforms



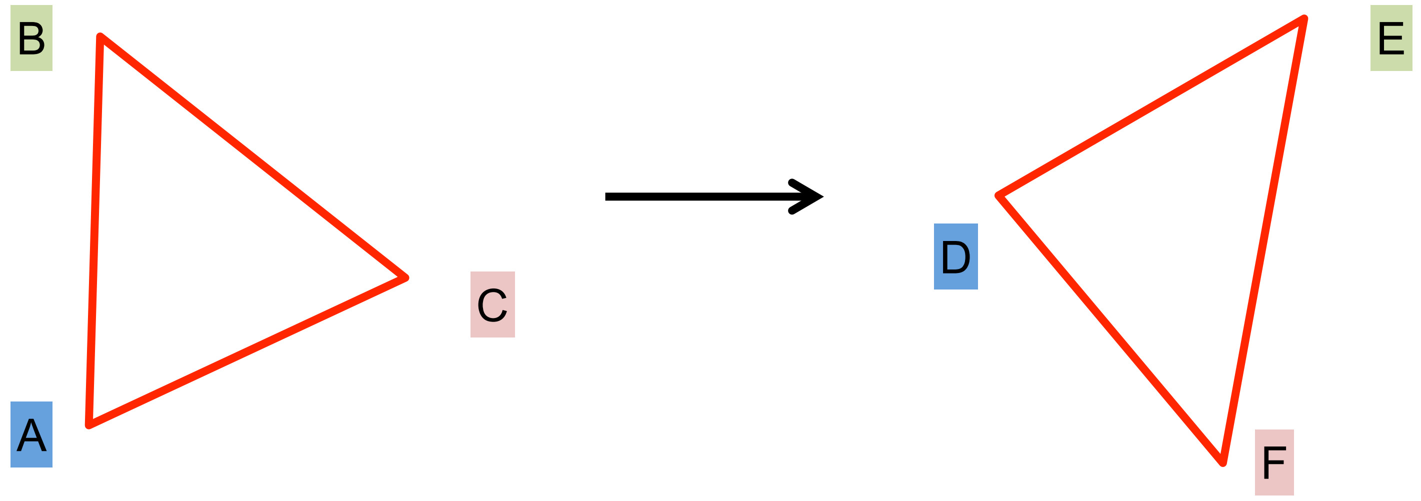
Classification of 2D transformations

Name	Matrix	# D.O.F.
translation	$\left[\begin{array}{c\|c} \boldsymbol{I} & \boldsymbol{t} \end{array}\right]$	2
rigid (Euclidean)	$\left[\begin{array}{c\|c} \boldsymbol{R} & \boldsymbol{t} \end{array}\right]$	3
similarity	$\left[\begin{array}{c\|c} s\boldsymbol{R} & \boldsymbol{t} \end{array}\right]$	3
affine	$\left[\begin{array}{c} \boldsymbol{A} \end{array}\right]$	6
projective	$\left[\begin{array}{c} \tilde{\boldsymbol{H}} \end{array}\right]$	8

Determining unknown 2D transformations

Determining unknown transformations

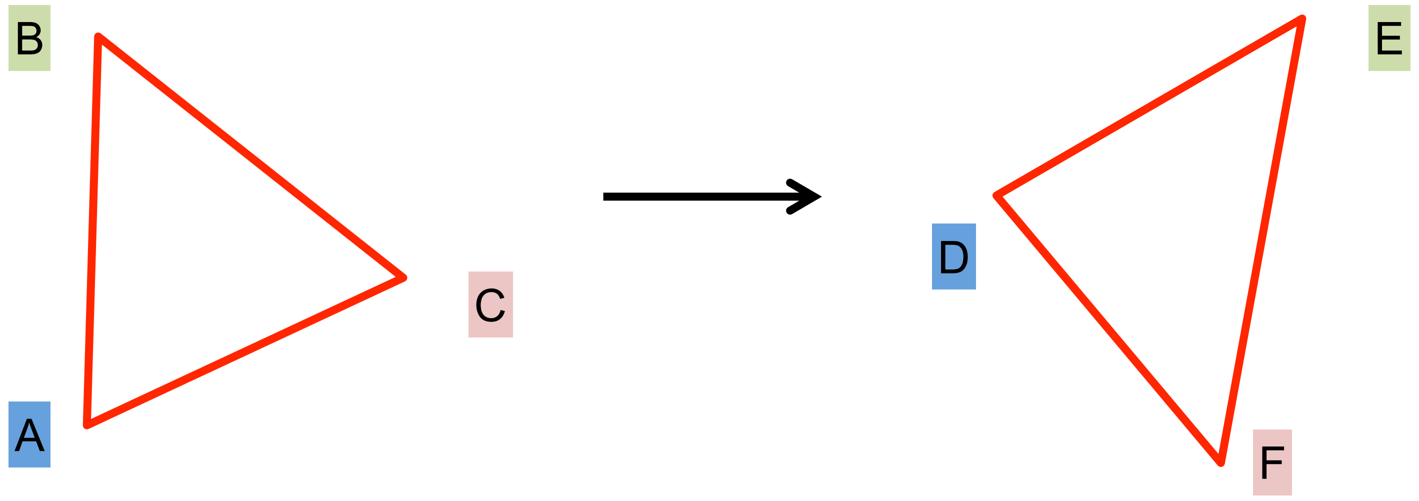
Suppose we have two triangles: ABC and DEF.

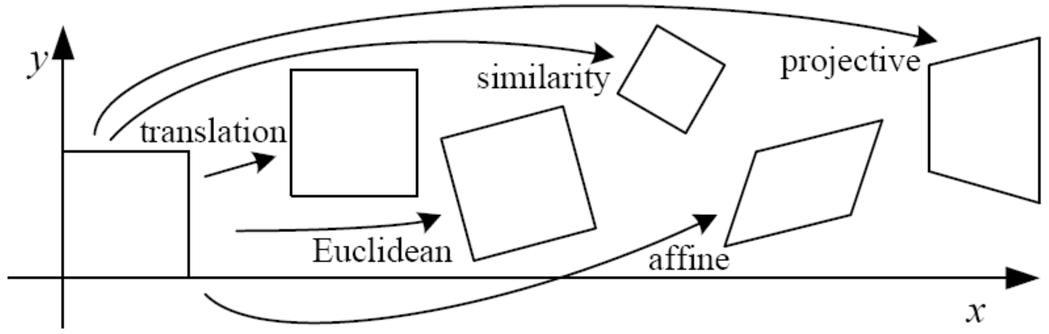


Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

- What type of transformation will map A to D, B to E, and C to F?

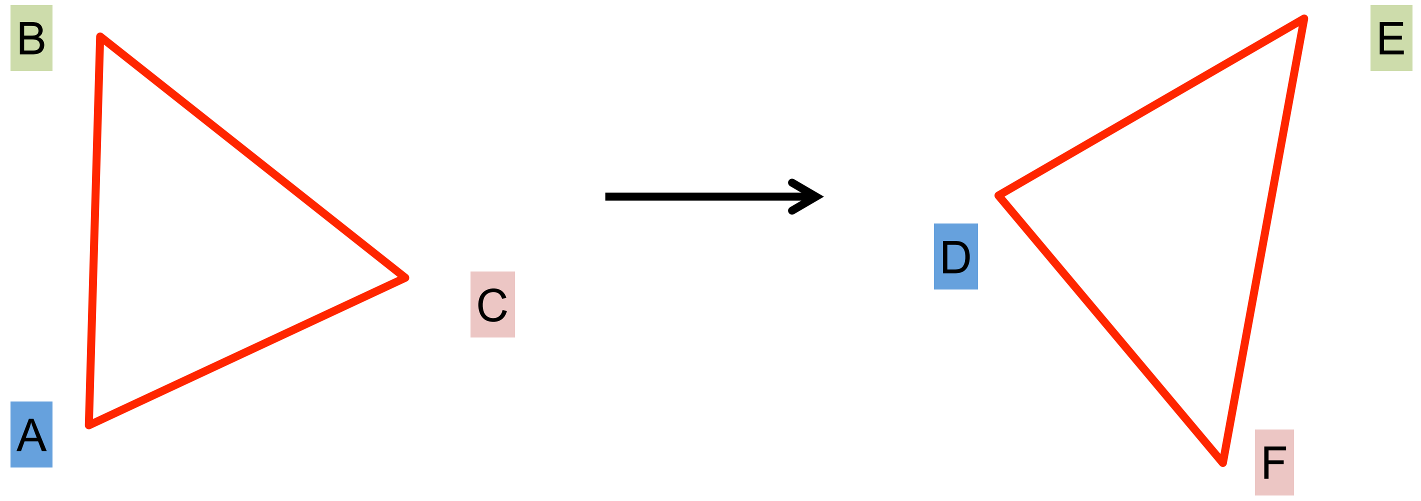




Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

- What type of transformation will map A to D, B to E, and C to F?
- How do we determine the unknown parameters?



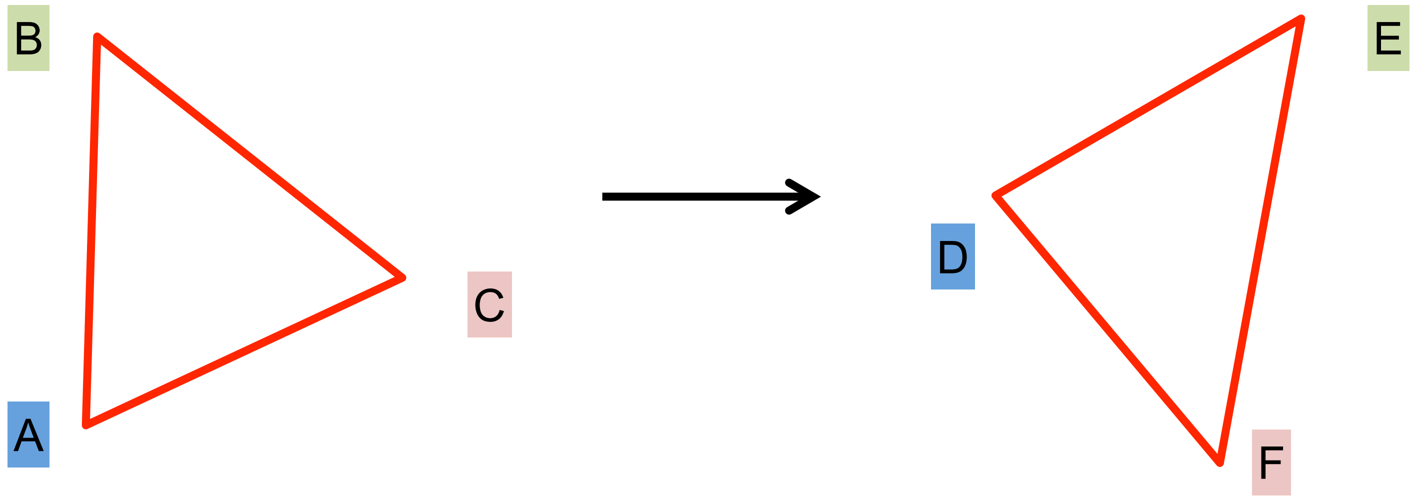
Affine transform:
uniform scaling + shearing
+ rotation + translation

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

- What type of transformation will map A to D, B to E, and C to F?
- How do we determine the unknown parameters?



unknowns

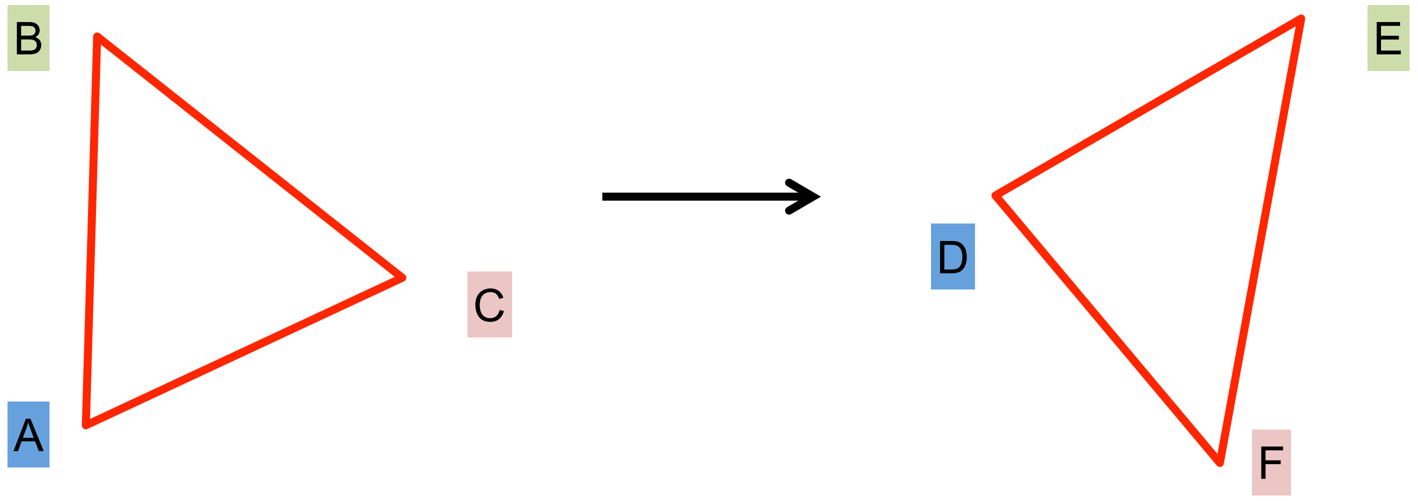
$$\boldsymbol{x}' = \mathbf{M}\boldsymbol{x}$$

point correspondences

Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

- What type of transformation will map A to D, B to E, and C to F?
- How do we determine the unknown parameters?

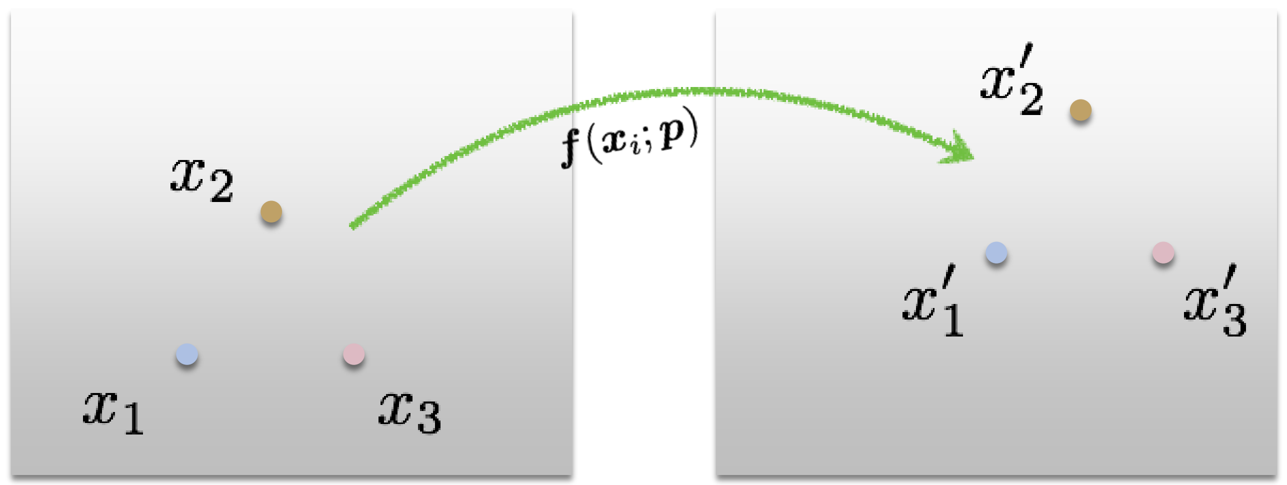


unknowns

$$\boldsymbol{x}' = \mathbf{M}\boldsymbol{x}$$

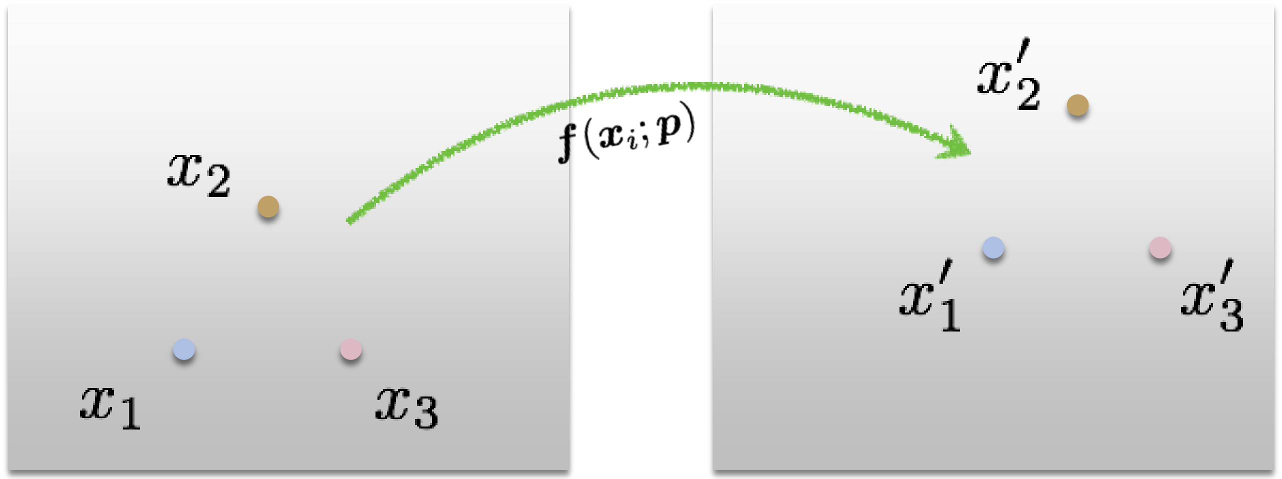
point correspondences

How do we solve this for **M**?



Least Squares Error

$$E_{\mathrm{LS}} = \sum_i \|\boldsymbol{f}(\boldsymbol{x}_i;\boldsymbol{p}) - \boldsymbol{x}'_i\|^2$$



Find parameters that minimize squared error

$$\hat{\boldsymbol{p}} = \arg\min_{\boldsymbol{p}} \sum_i \|\boldsymbol{f}(\boldsymbol{x}_i; \boldsymbol{p}) - \boldsymbol{x}'_i\|^2$$

Determining unknown transformations

Affine transformation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Why can we drop the last line?

Vectorize transformation parameters:

Stack equations from point correspondences:

Notation in system form:

$$\underbrace{\begin{bmatrix} x' \\ y' \\ x' \\ y' \\ \vdots \\ x' \\ y' \end{bmatrix}}_{\boldsymbol{b}} = \underbrace{\begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ & & & \vdots & & \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{bmatrix}}_{\boldsymbol{x}}$$

Solving the linear system

Convert the system to a linear least-squares problem:

$$E_{\mathrm{LLS}} = \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|^2$$

Solve this using least squares in Python: **numpy.linalg.lstsq**

Linear least squares estimation only works when the transform function is **linear!**

Also doesn’t deal well with outliers

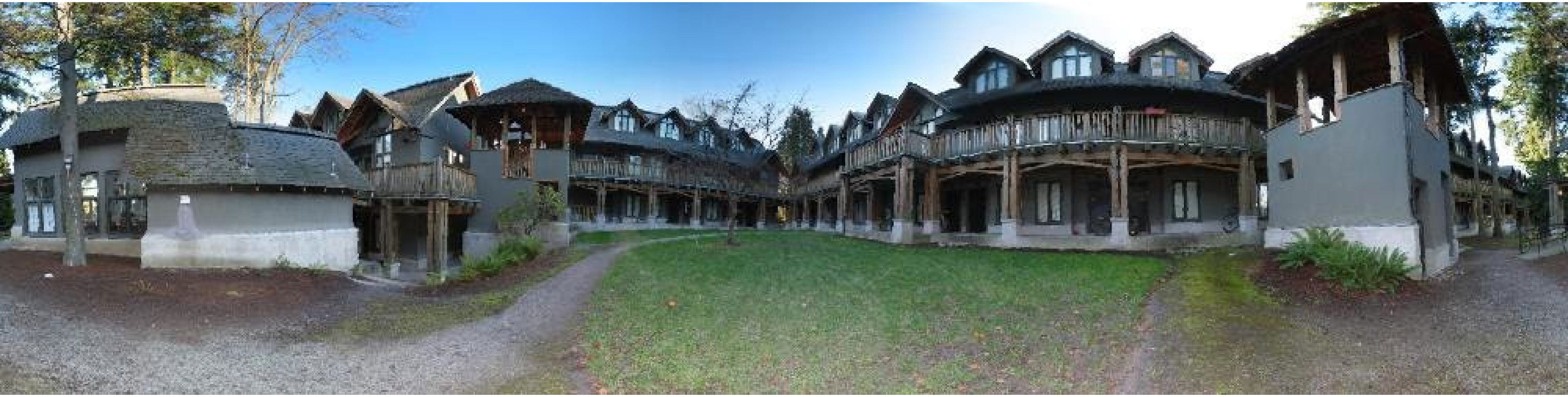
ME FINALLY UNDERSTANDING TRANSFORMATIONS
SUCCESS

How do you create a panorama?

Panorama: an image of (near) 360^{o} field of view.

How do you create a panorama?

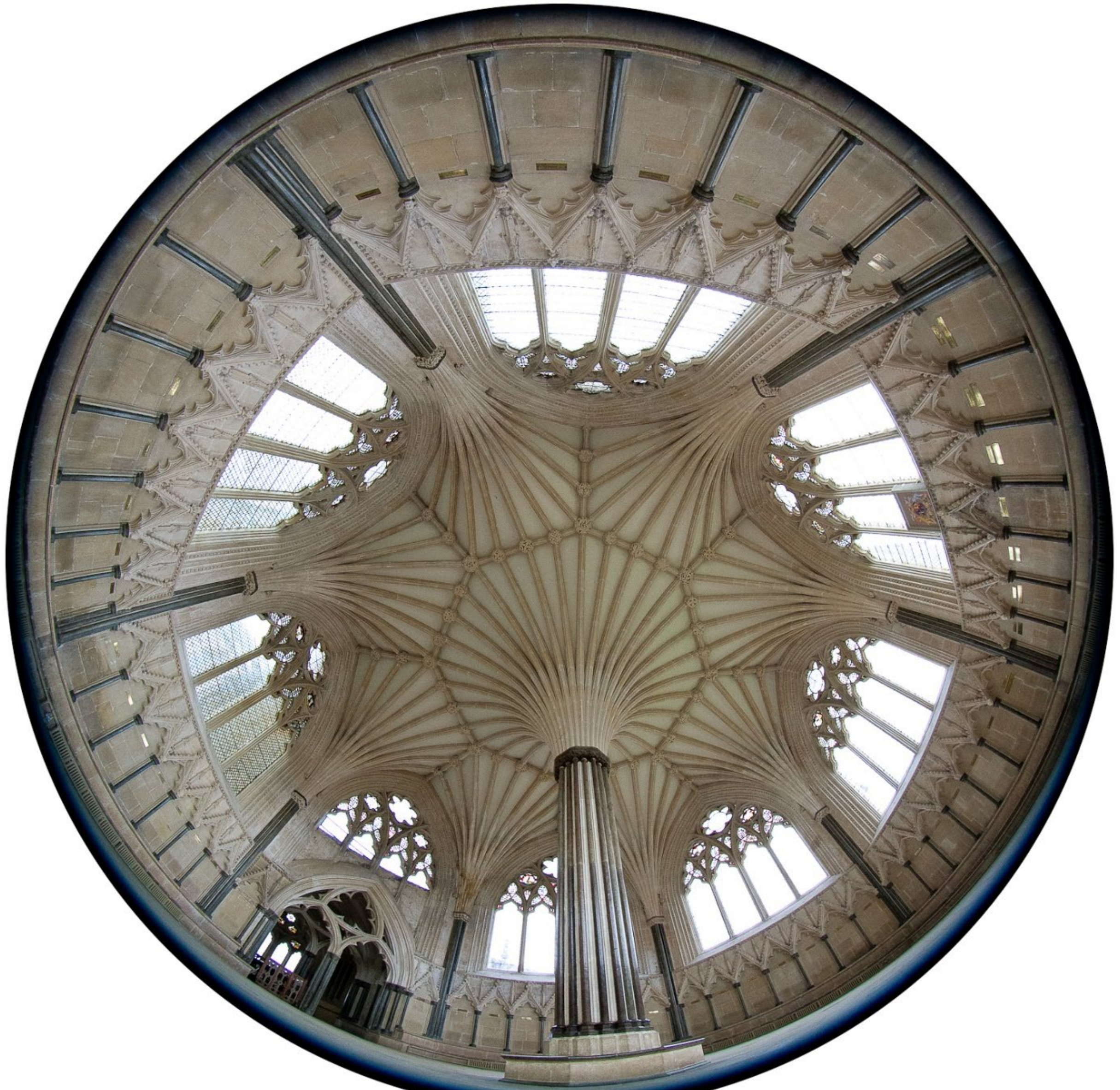
Panorama: an image of (near) 360^o field of view.



1. Use a very wide-angle lens.

Wide-angle lenses

Fish-eye lens: can produce (near) hemispherical field of view.



What are the pros and cons of this?

How do you create a panorama?

Panorama: an image of (near) 360^o field of view.

1. Use a very wide-angle lens.

- Pros: Everything is done optically, single capture.
- Cons: Lens is super expensive and bulky, lots of distortion (can be dealt-with in post). (is it still expensive? This slide is old … iPhone 0.5x mode?)

How do you create a panorama?

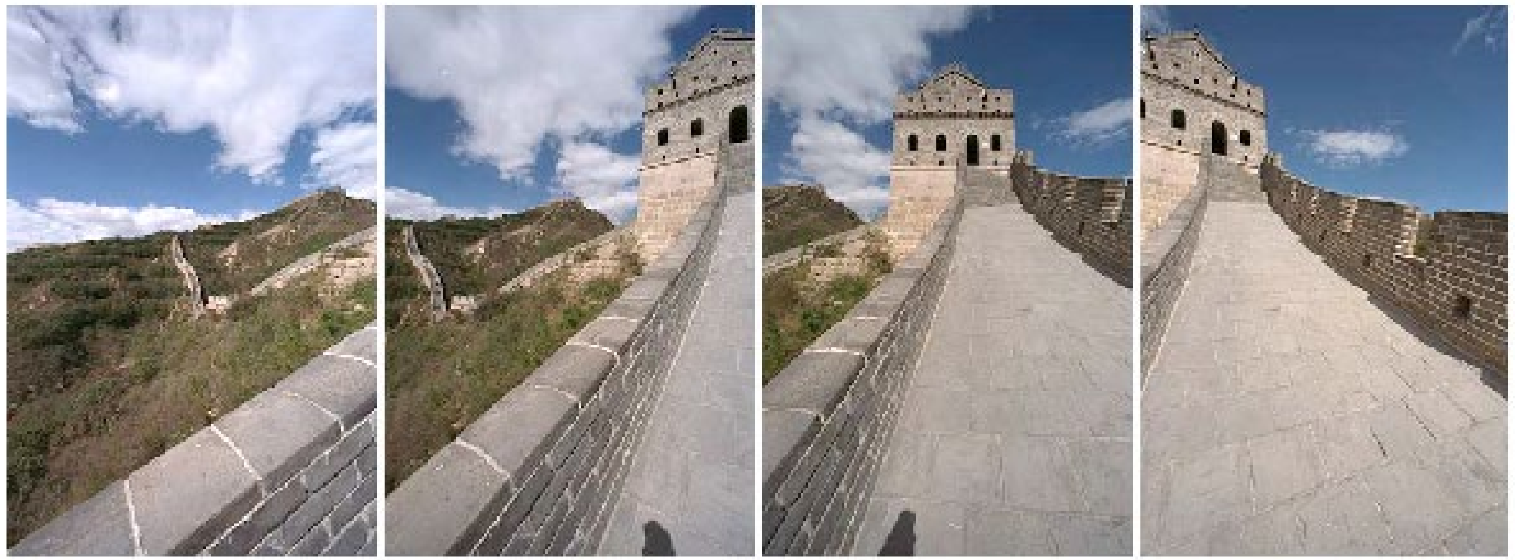
Panorama: an image of (near) 360^o field of view.

1. Use a very wide-angle lens.
- Pros: Everything is done optically, single capture.
- Cons: Lens is super expensive and bulky, lots of distortion (can be dealt-with in post).

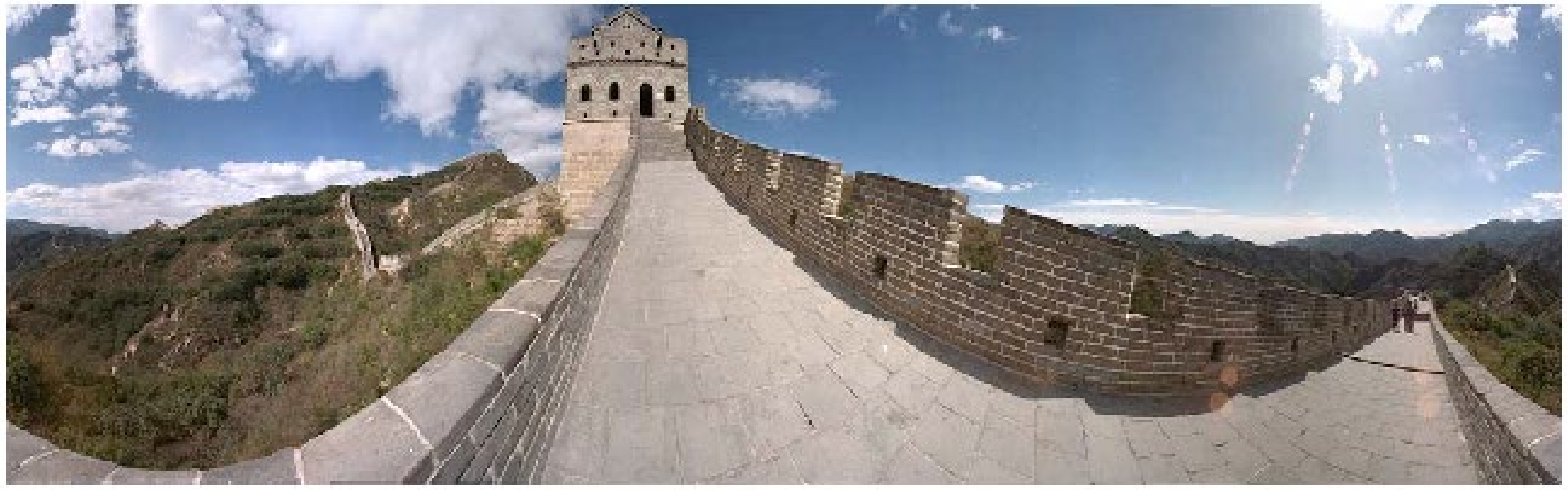
2. Capture multiple images and combine them.

Panoramas from image stitching

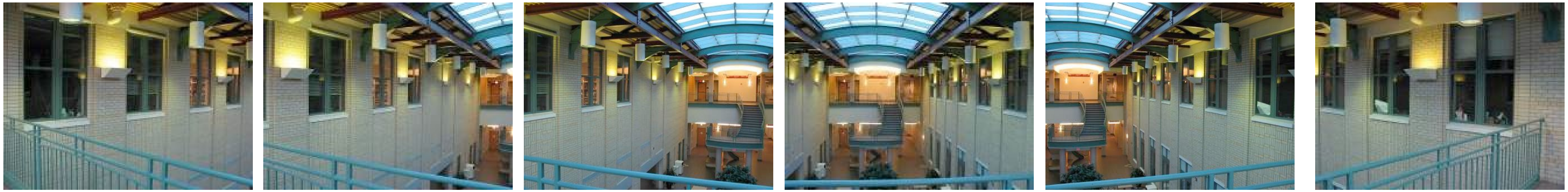
1. Capture multiple images from different viewpoints.



2. Stitch them together into a virtual wide-angle image.



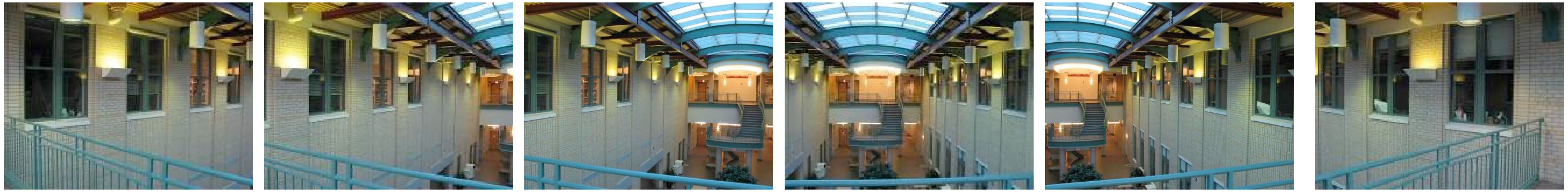
How do we stitch images from different viewpoints?



Will standard stitching work?

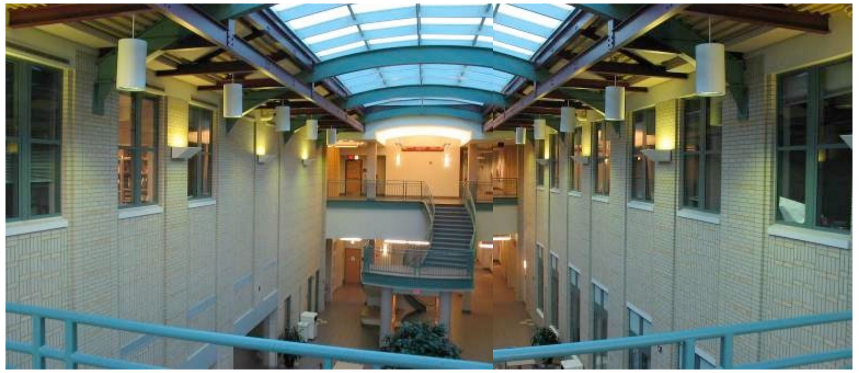
1. Translate one image relative to another.
2. (Optionally) find an optimal seam.

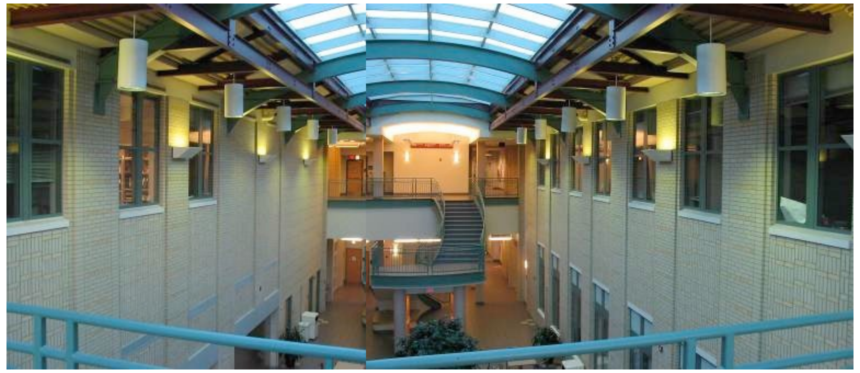
How do we stitch images from different viewpoints?



Will standard stitching work?

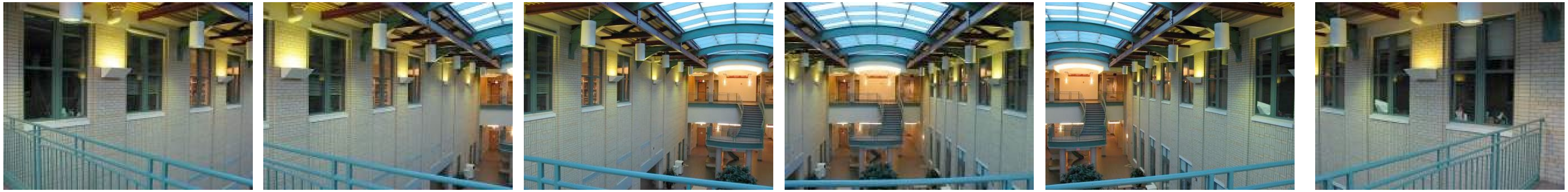
1. Translate one image relative to another.
2. (Optionally) find an optimal seam.

left on top 

 right on top

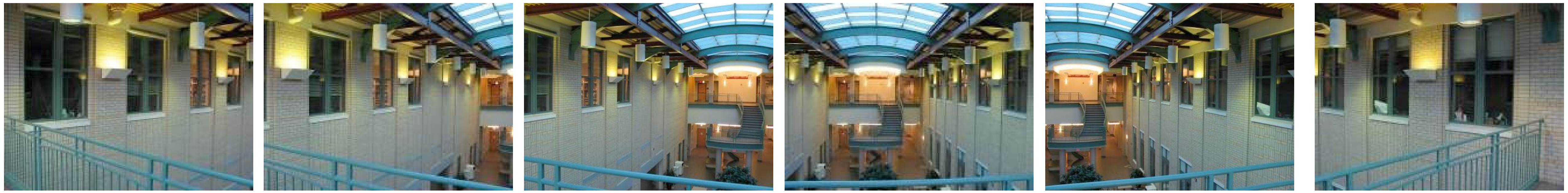
Translation-only stitching is not enough to mosaic these images.

How do we stitch images from different viewpoints?

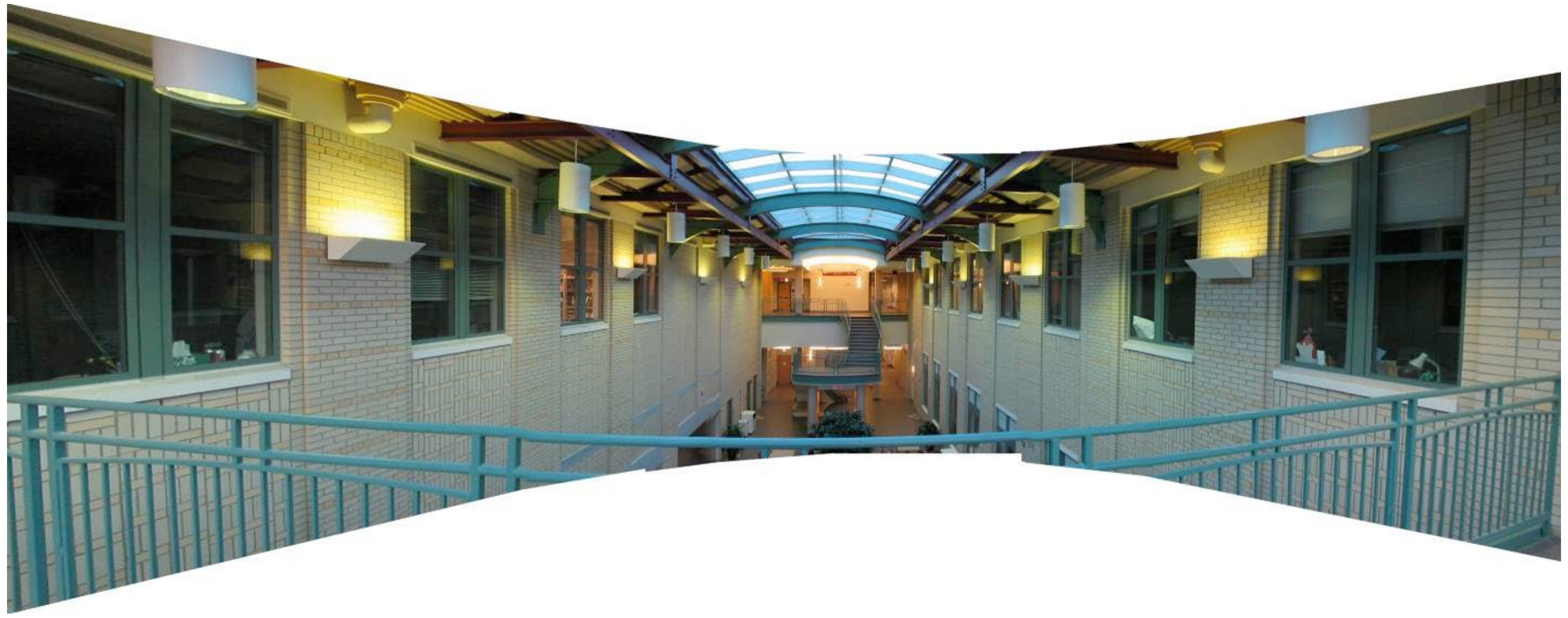


What else can we try?

How do we stitch images from different viewpoints?

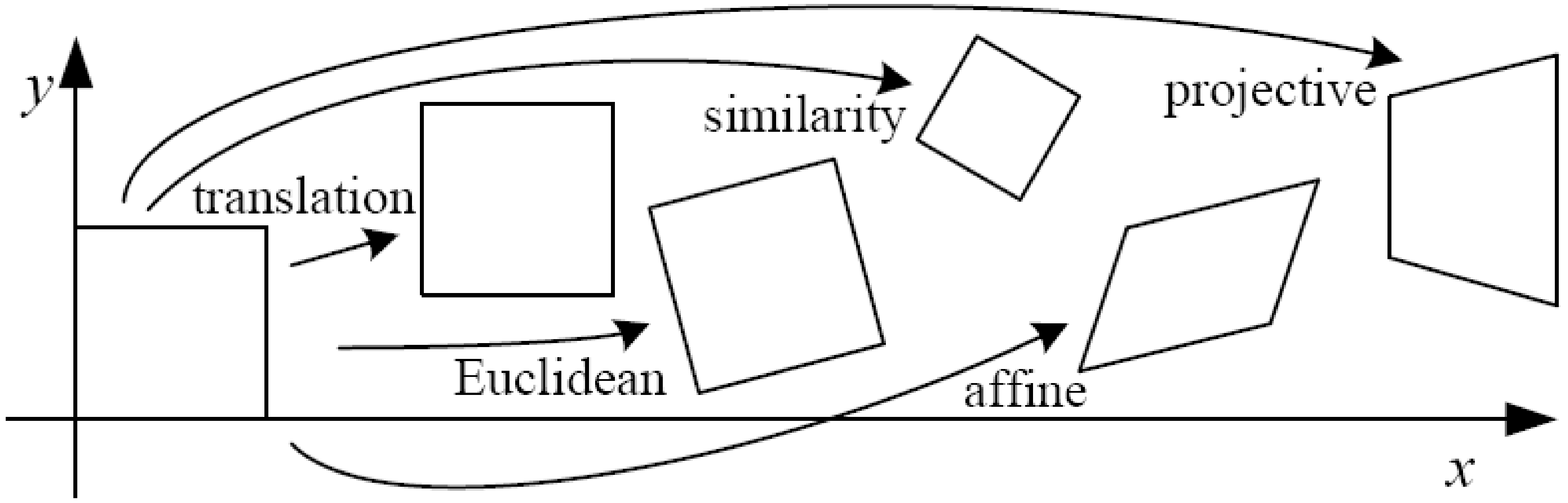


Use image homographies.



Back to warping: image homographies

Classification of 2D transformations

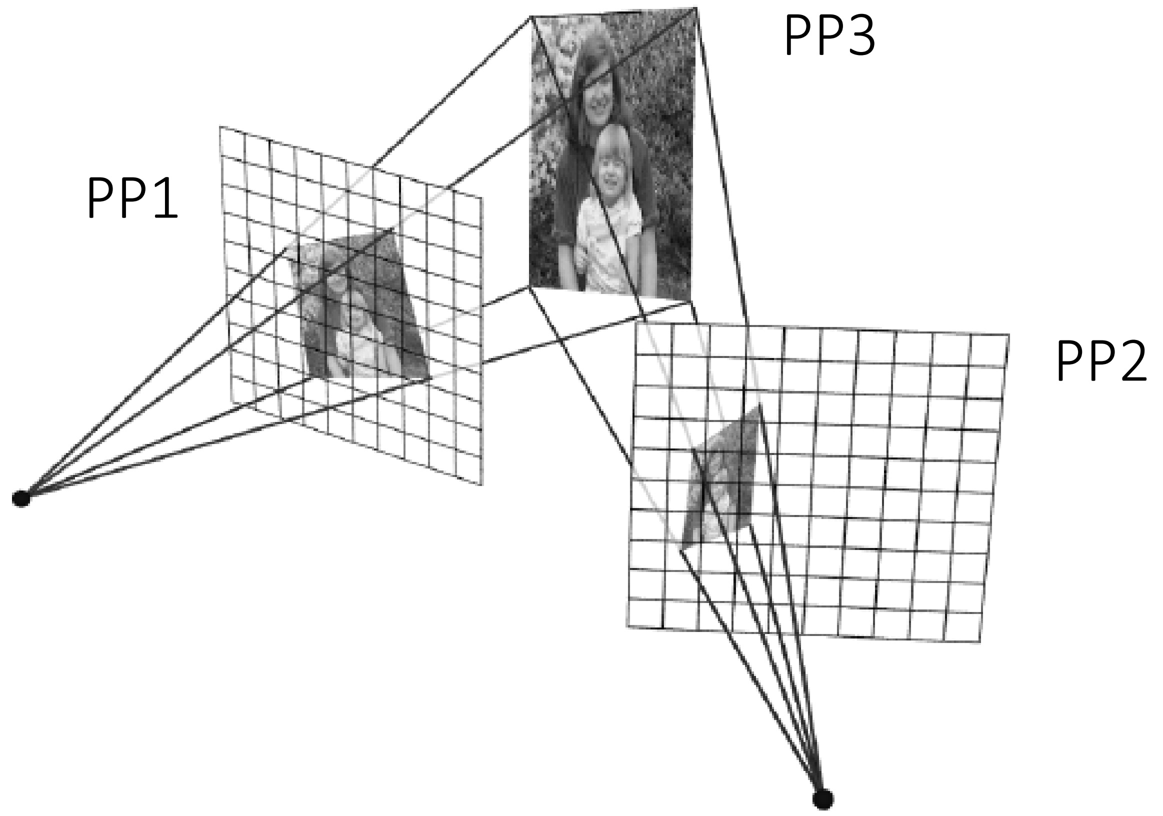


Name	Matrix	# D.O.F.
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8

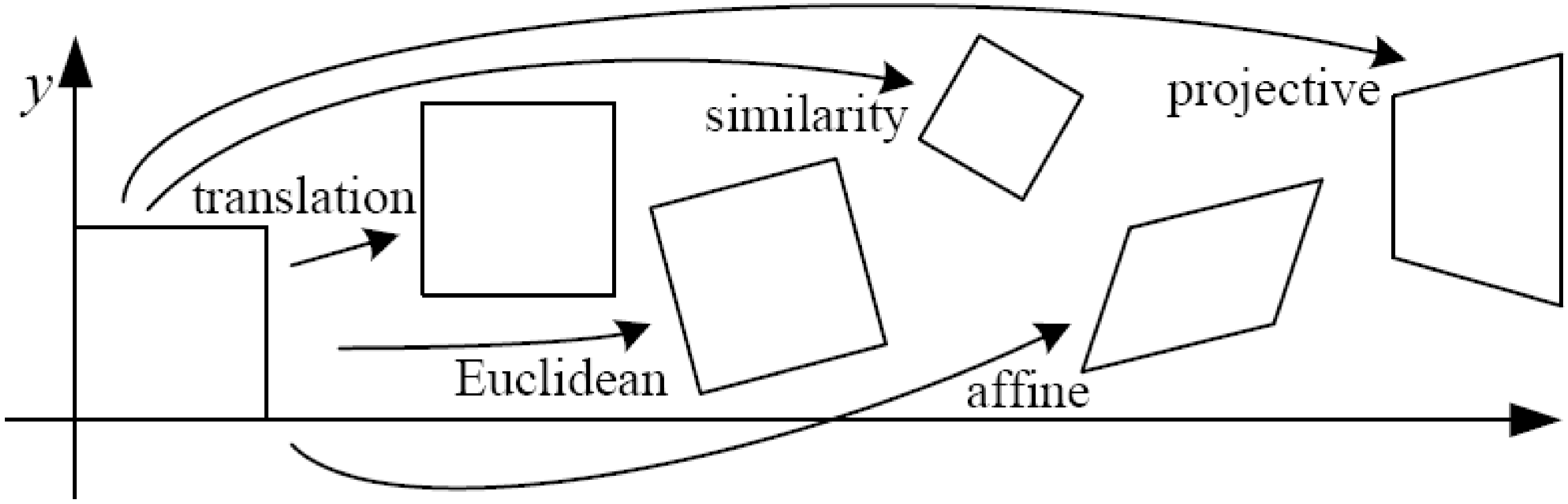
The Image Alignment Problem:

Input to a panorama-creating system:

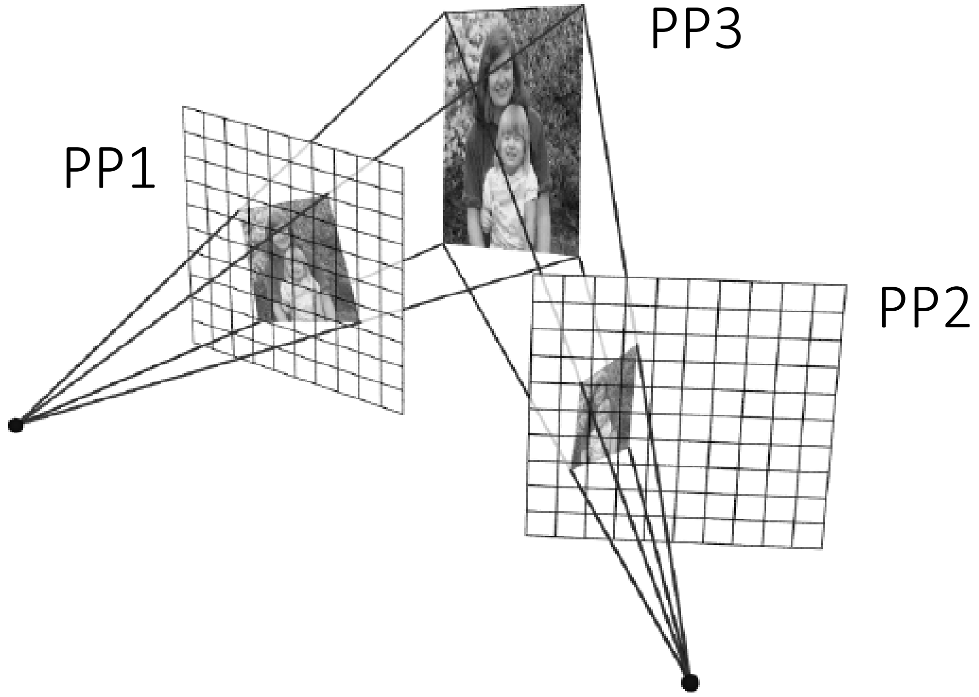
- Multiple Images of the same scene
- taken by a camera (or different cameras)
- from different locations $(x, y, z, \theta, \phi, \psi)$



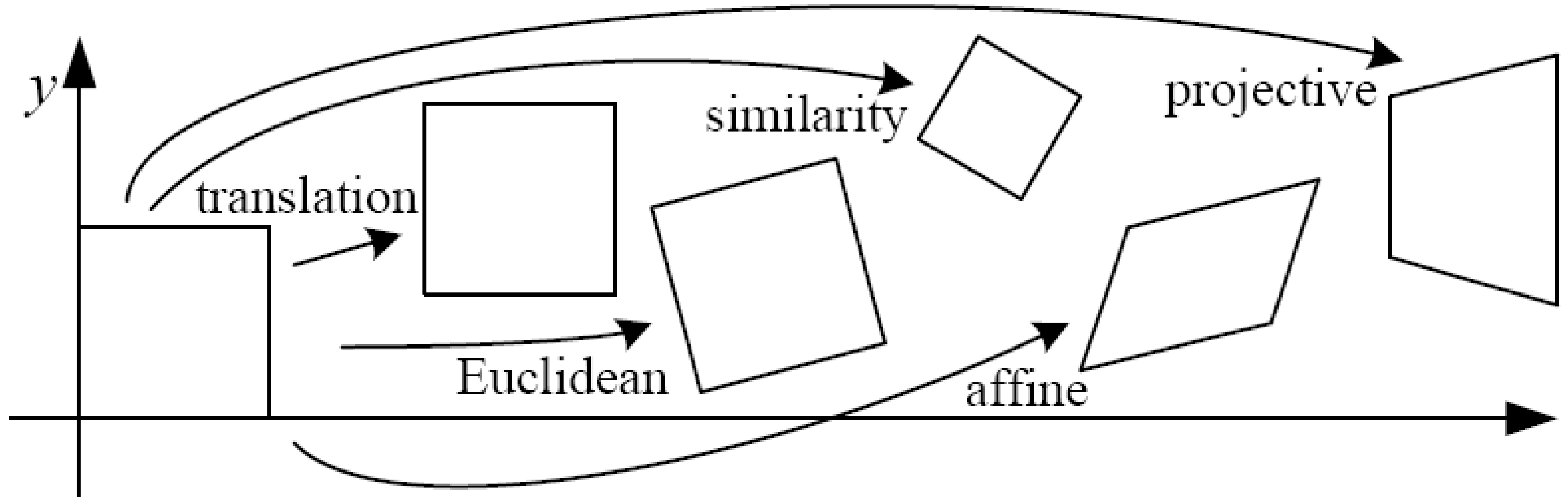
Classification of 2D transformations

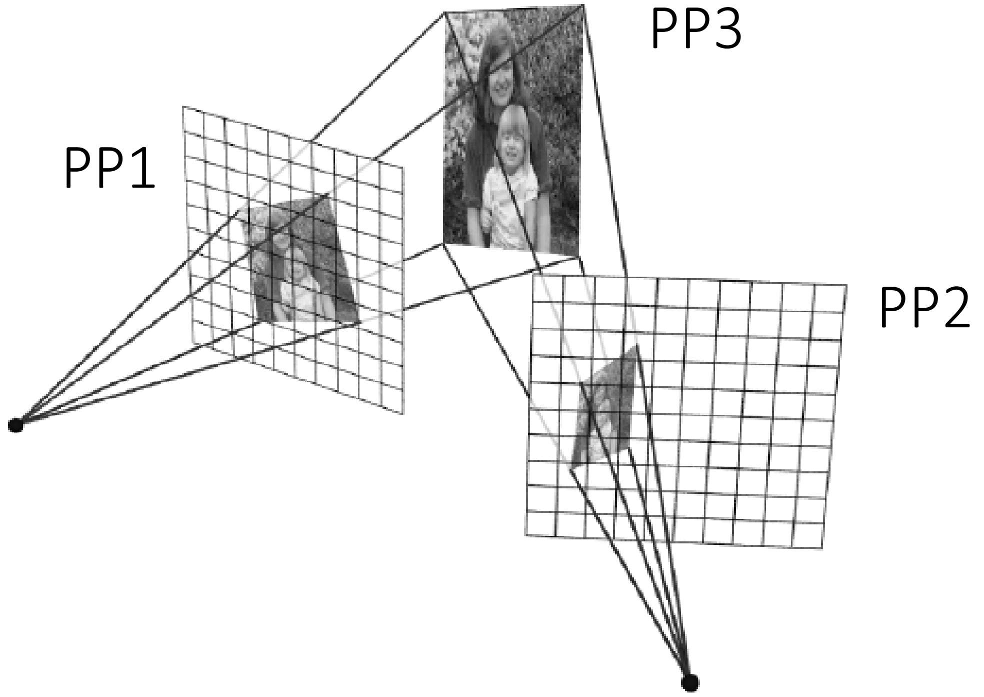


Which kind transformation is needed to warp projective plane 1 into projective plane 2?



Classification of 2D transformations





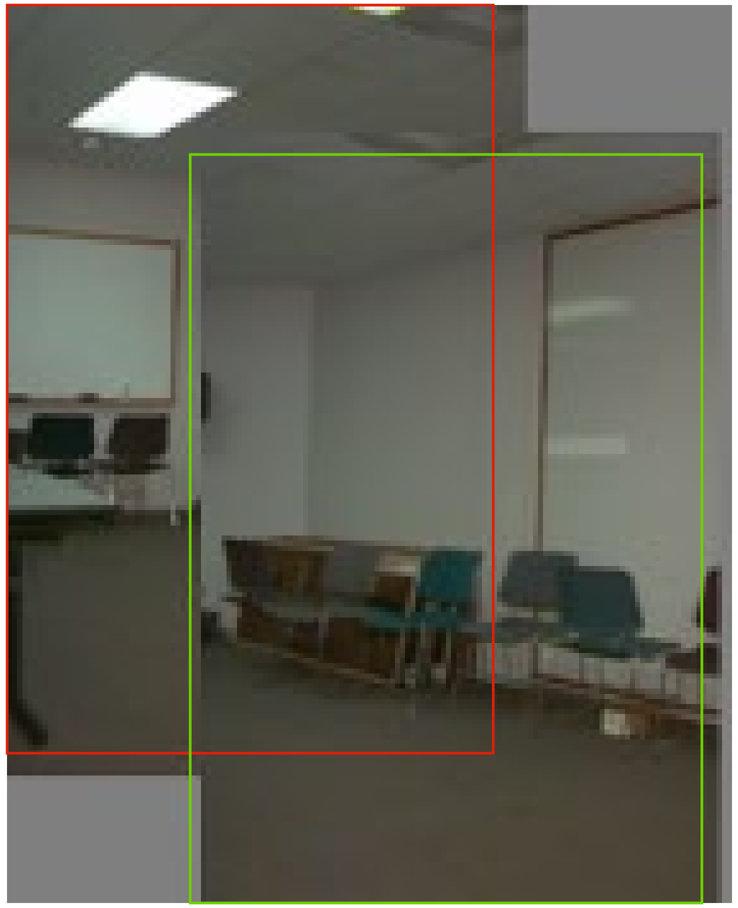
Which kind transformation is needed to warp projective plane 1 into projective plane 2?

- A projective transformation (a.k.a. a homography).

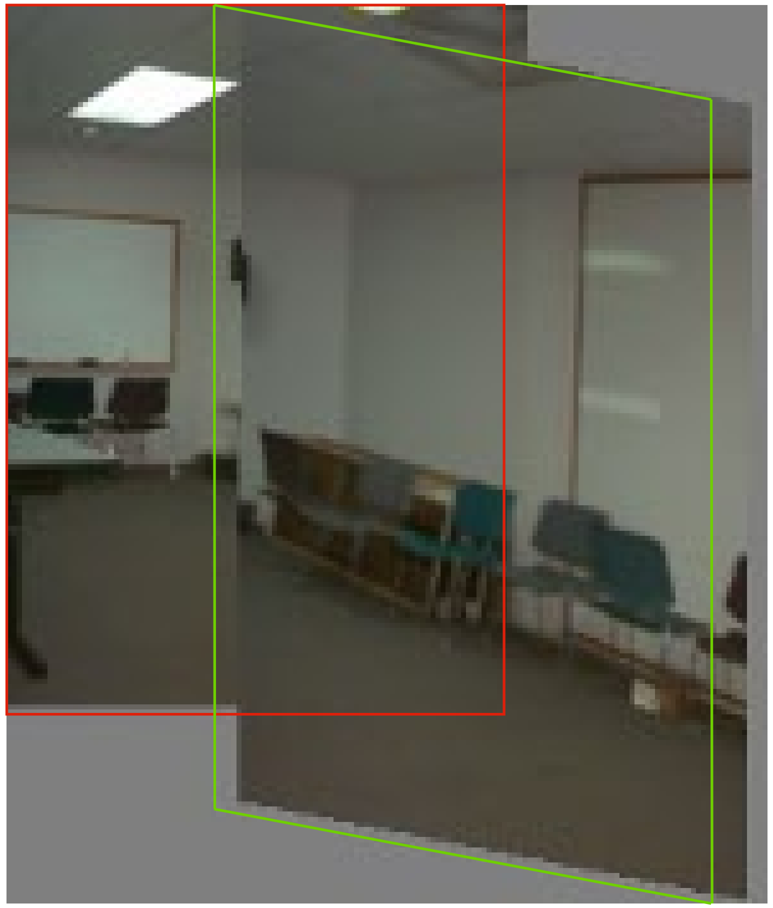
Applications

Warping with different transformations

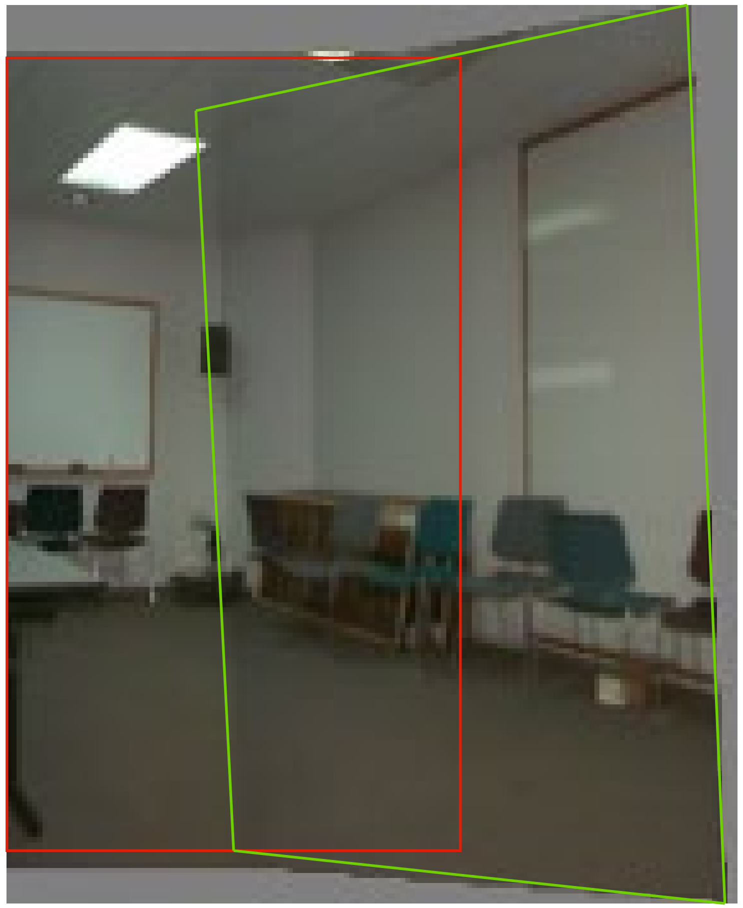
translation



affine

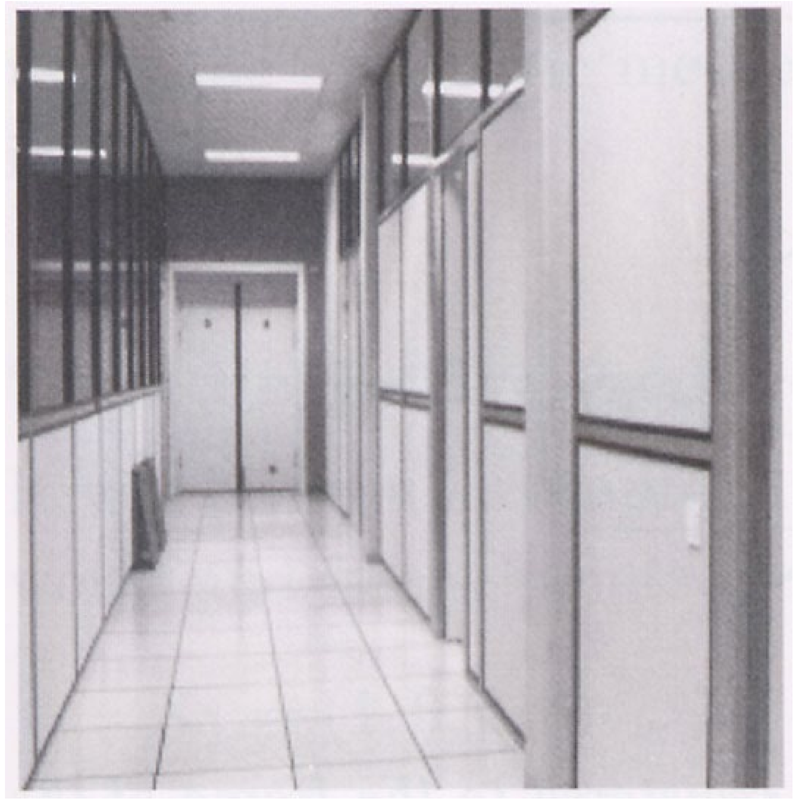


projective (homography)

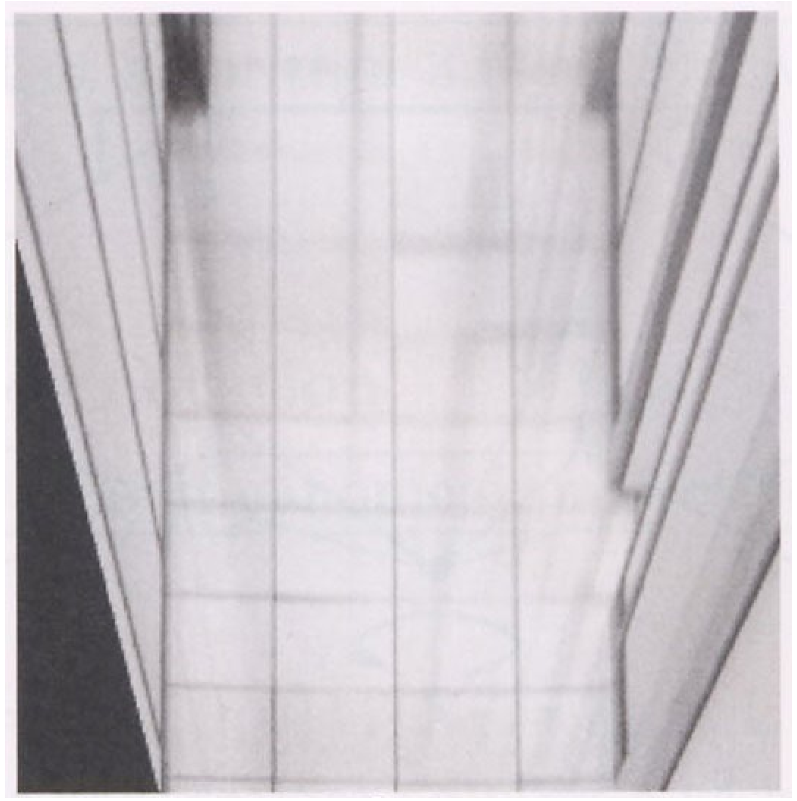


View warping

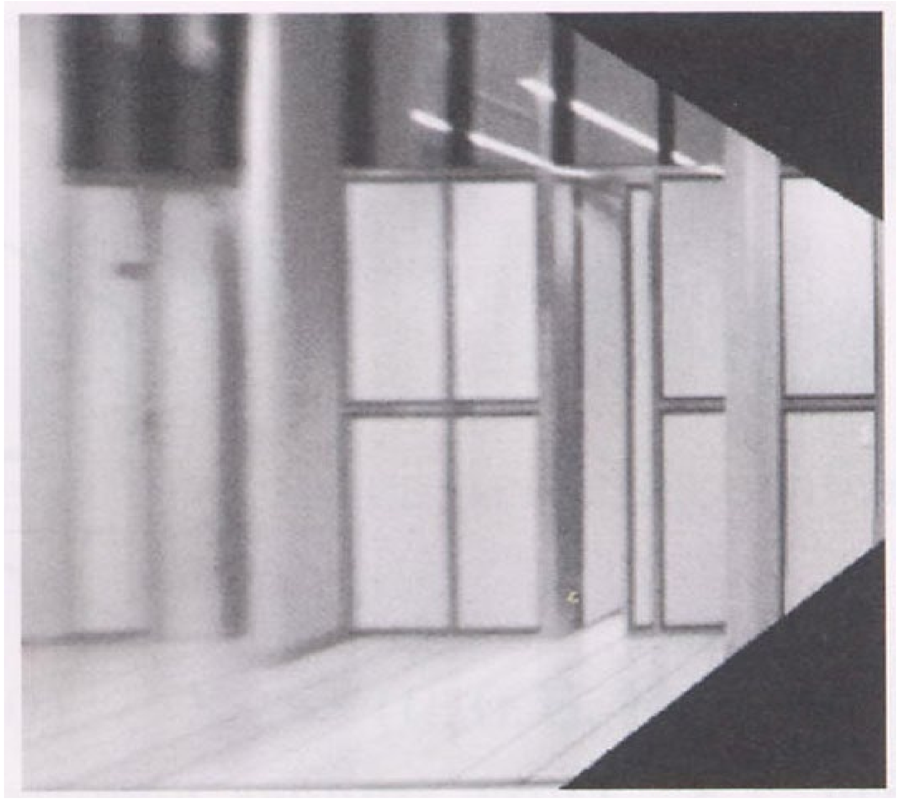
original view



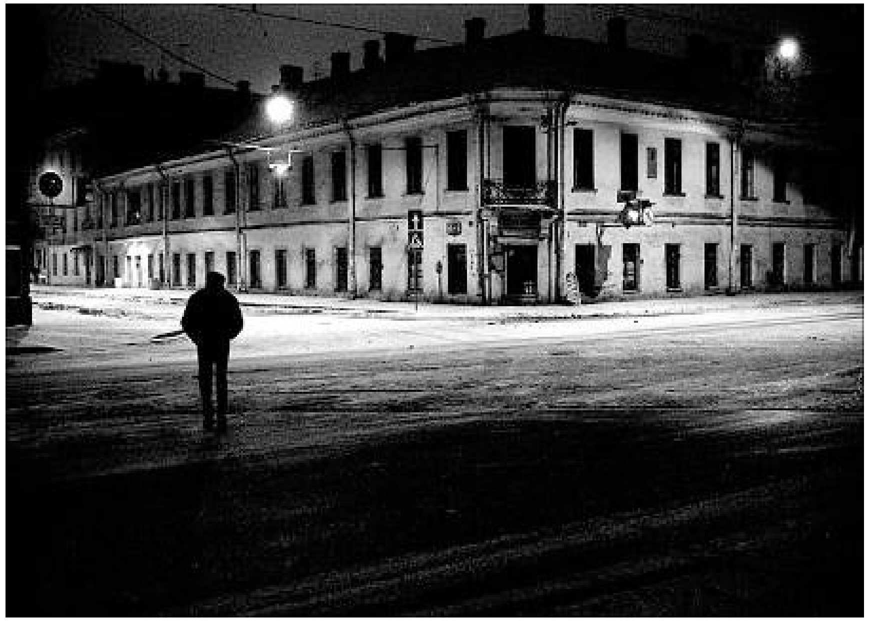
synthetic top view



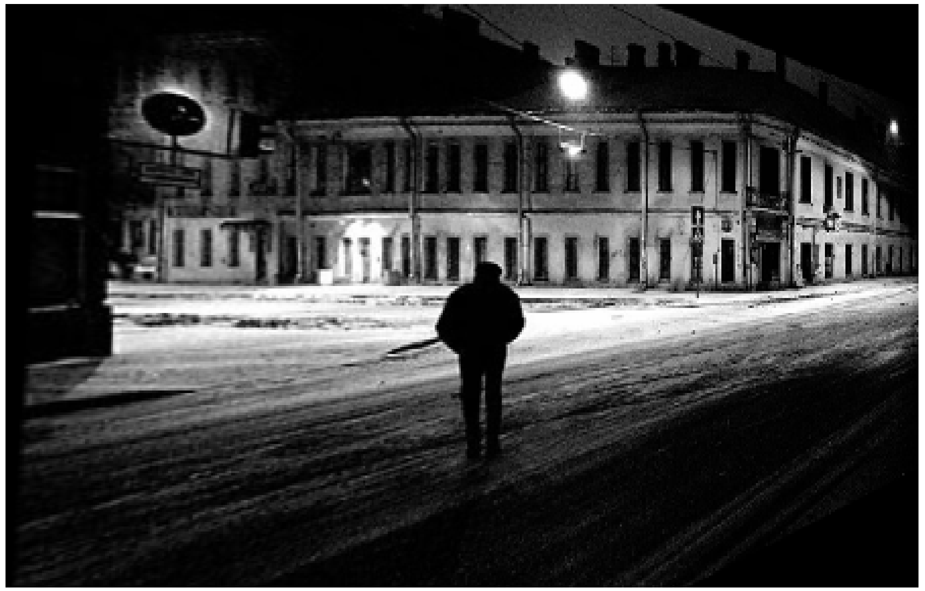
synthetic side view

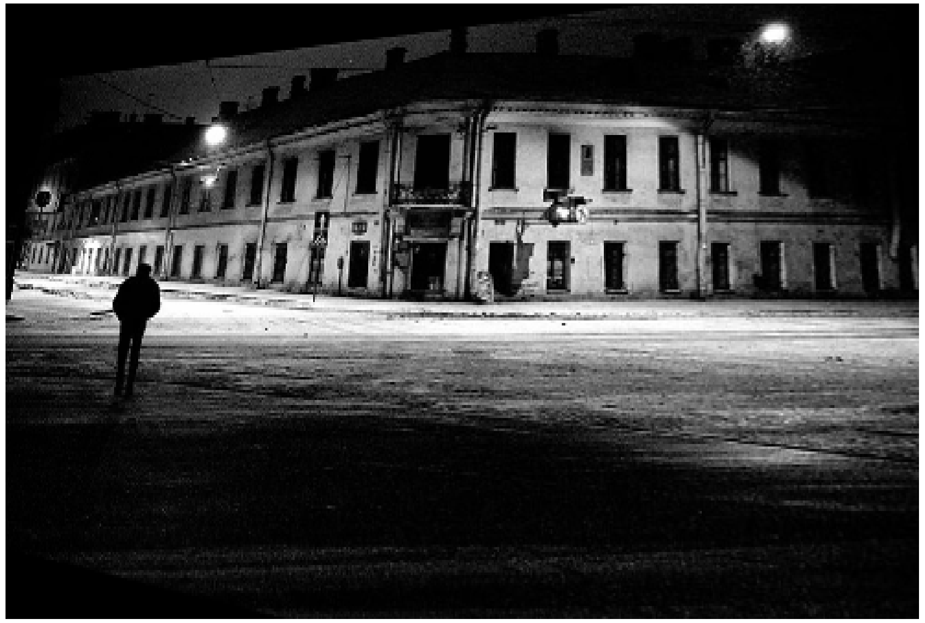


Virtual camera rotations



original view

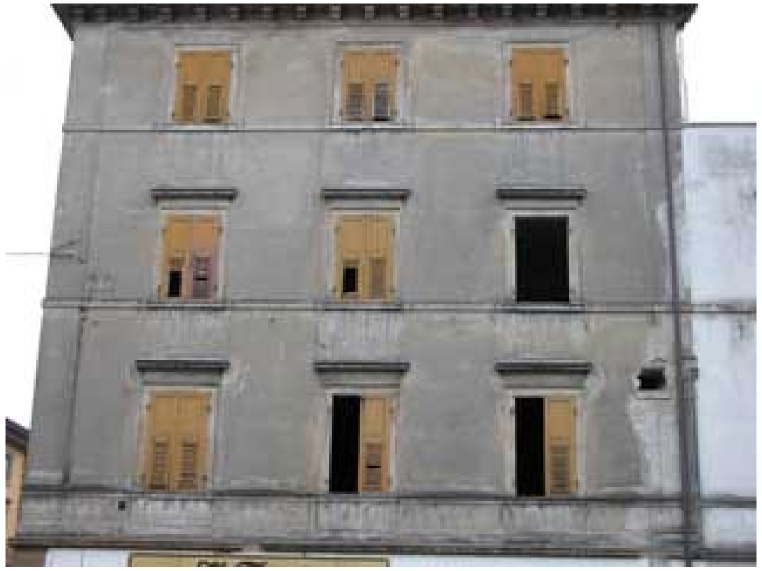


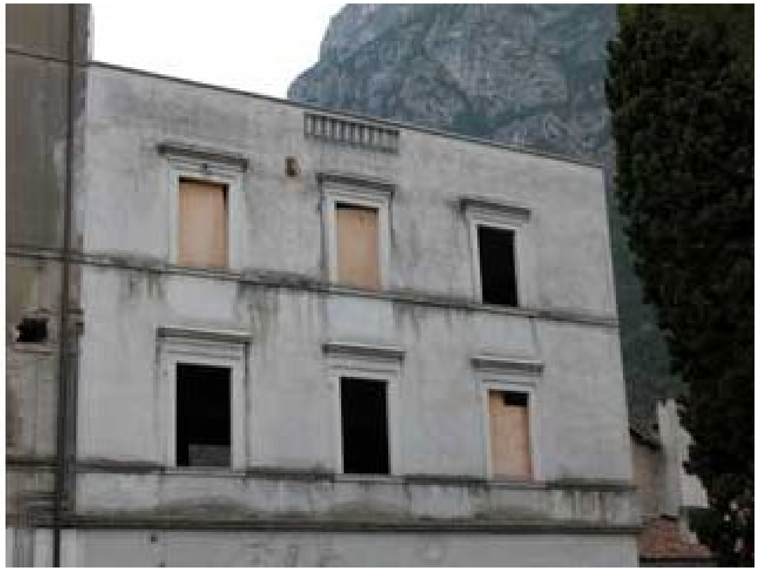


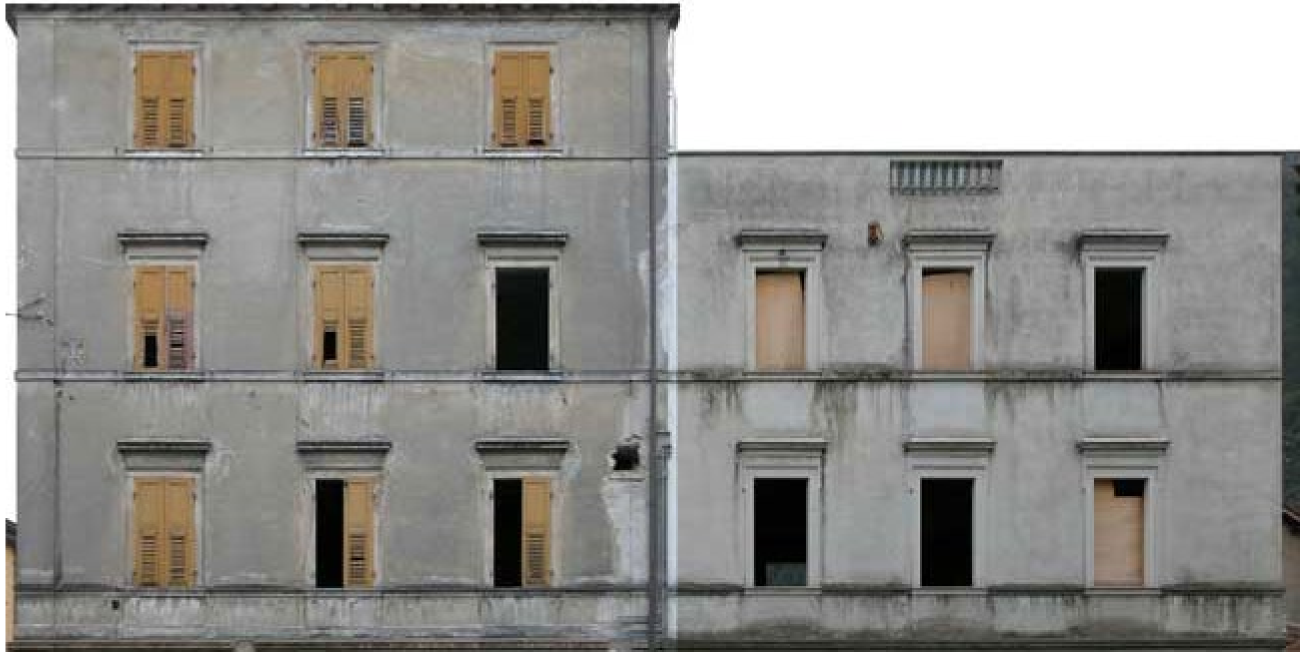
synthetic rotations

Image rectification

two original images





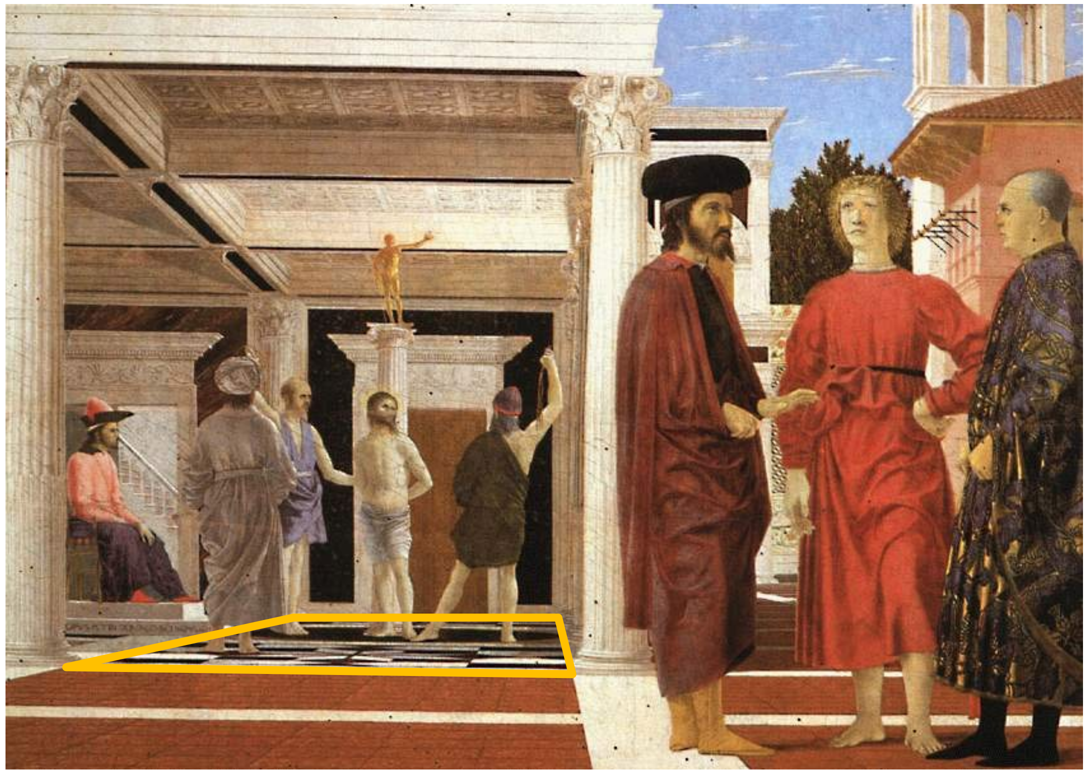


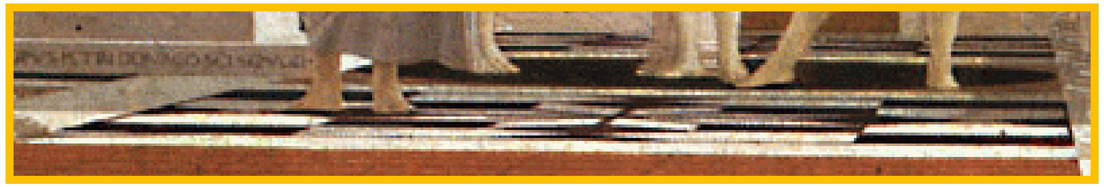
rectified and stitched

Street art

Understanding geometric patterns

What is the pattern on the floor?

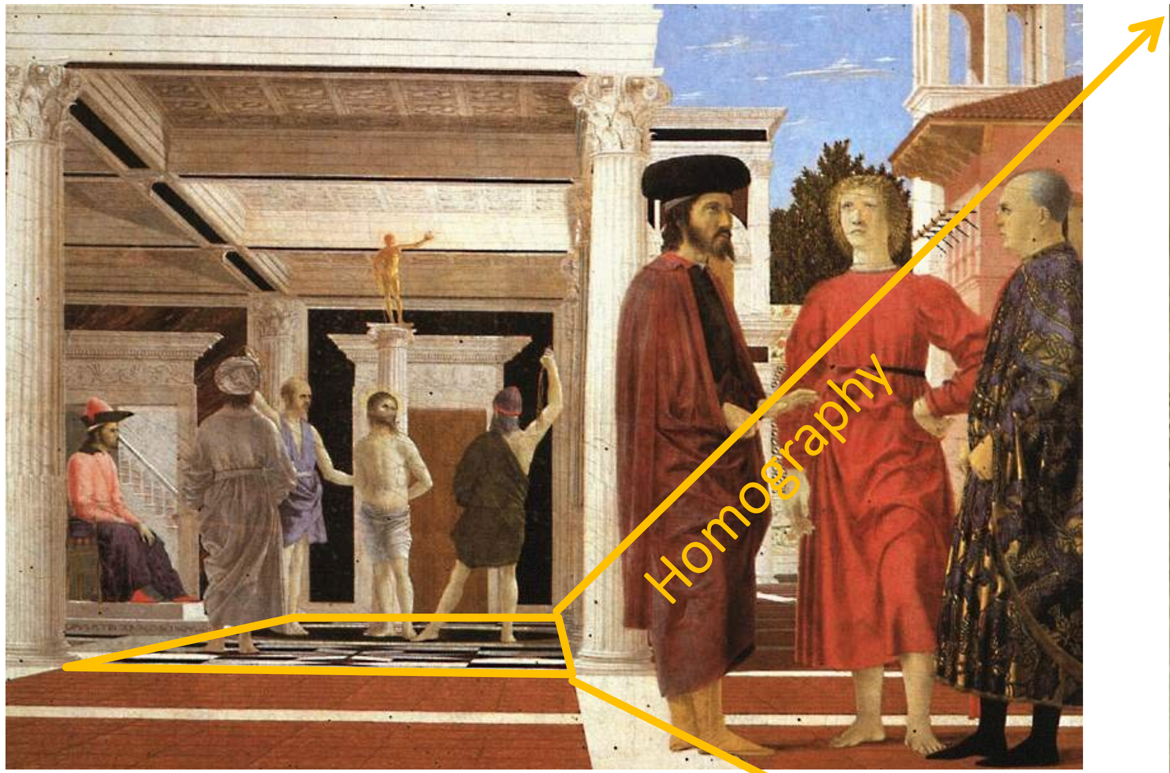




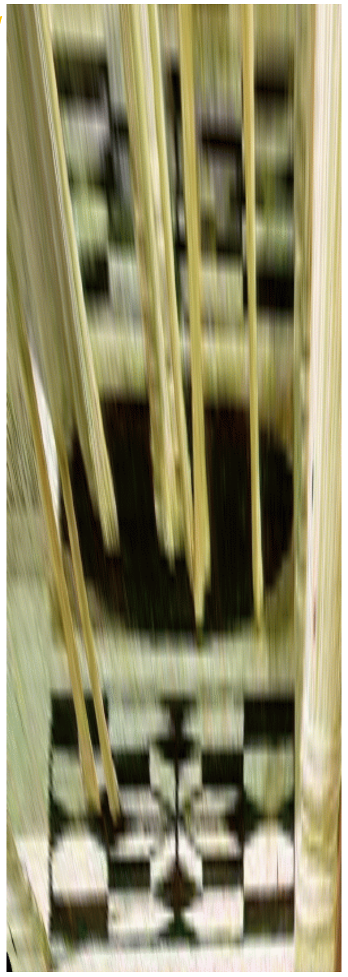
magnified view of floor

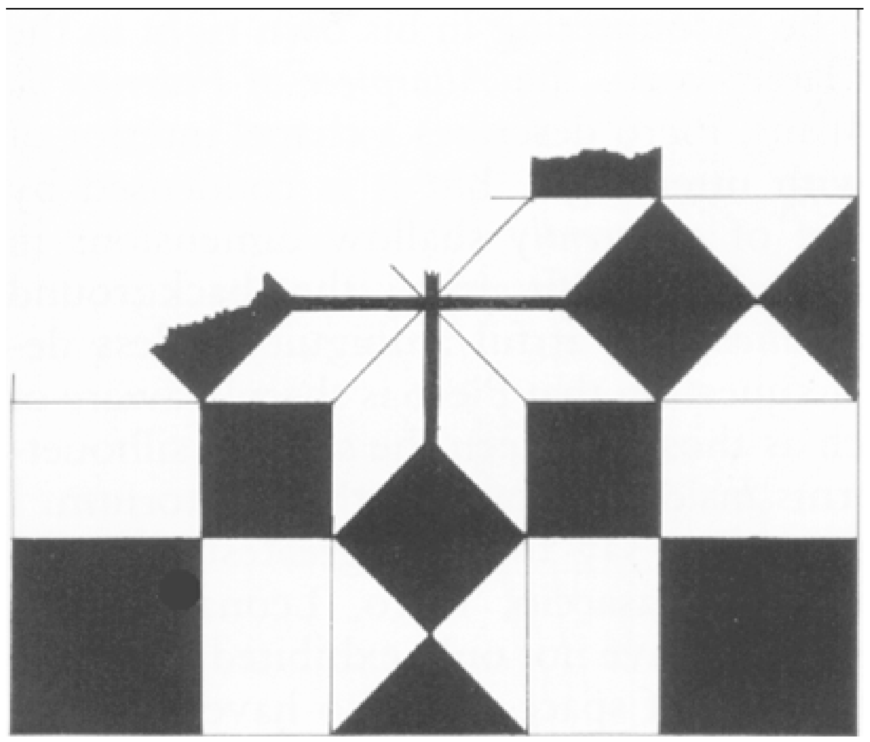
Understanding geometric patterns

What is the pattern on the floor?



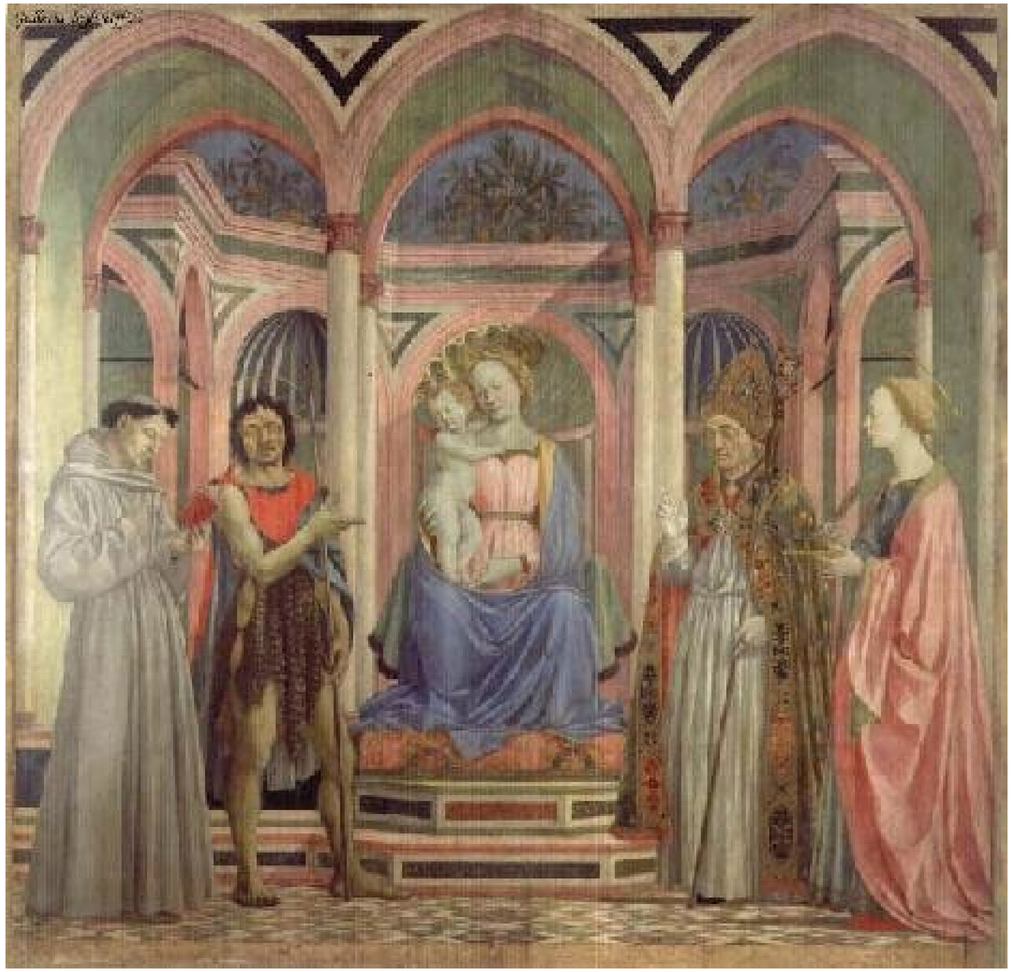
magnified view of floor

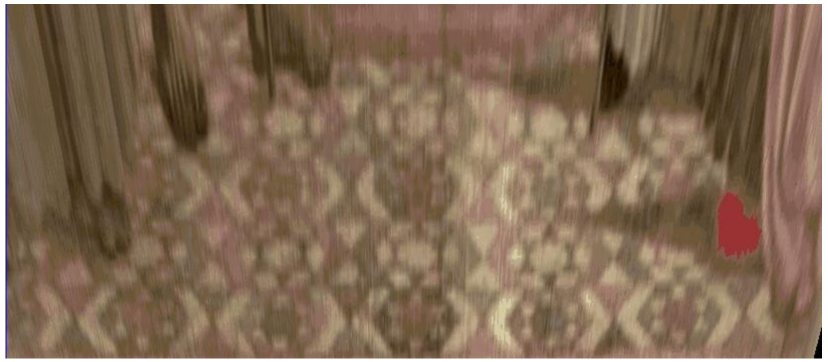

rectified view


reconstruction from rectified view

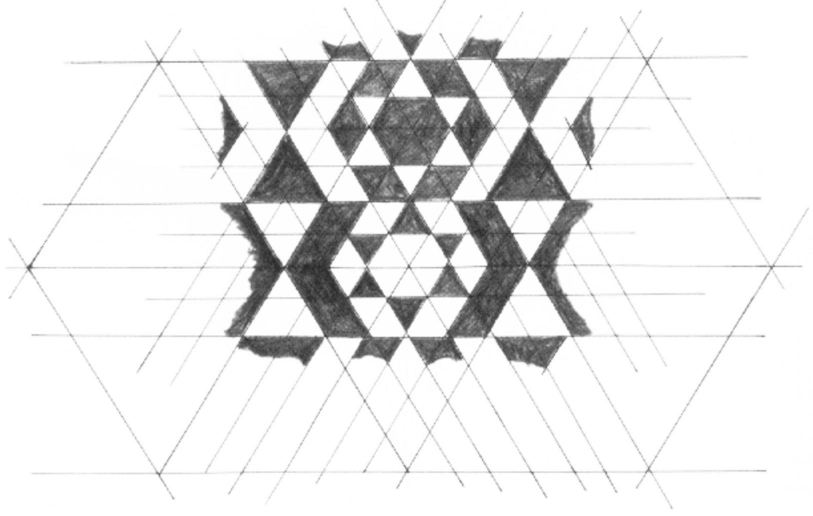
Understanding geometric patterns

Very popular in renaissance drawings
(when Europe first learnt about the concept of perspectives)





rectified view of floor



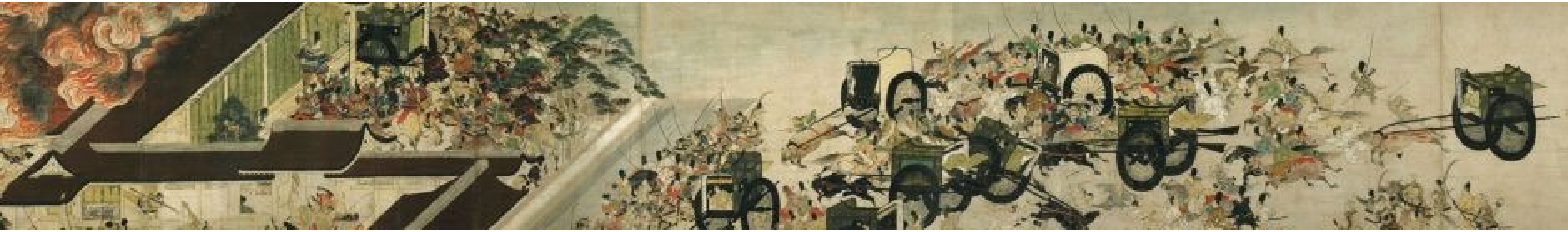
reconstruction

A Historical Note on Perspective

- The assertion in many books on art, optics, graphics, etc. that “perspective was discovered during the European Renaissance” is **questionable**
  - The Renaissance was the first time that European artists used perspective after the dark ages
- evidence from other older civilizations: China 1st century BC, Japan 8th century AD
  - Used “oblique” projection, but still looked realistic – why?
  - Because the observer (camera) was “very far from the scene”

$$\begin{bmatrix} 1 & 0 & \frac{1}{2}\cos\alpha \\ 0 & 1 & \frac{1}{2}\sin\alpha \\ 0 & 0 & 0 \end{bmatrix}$$

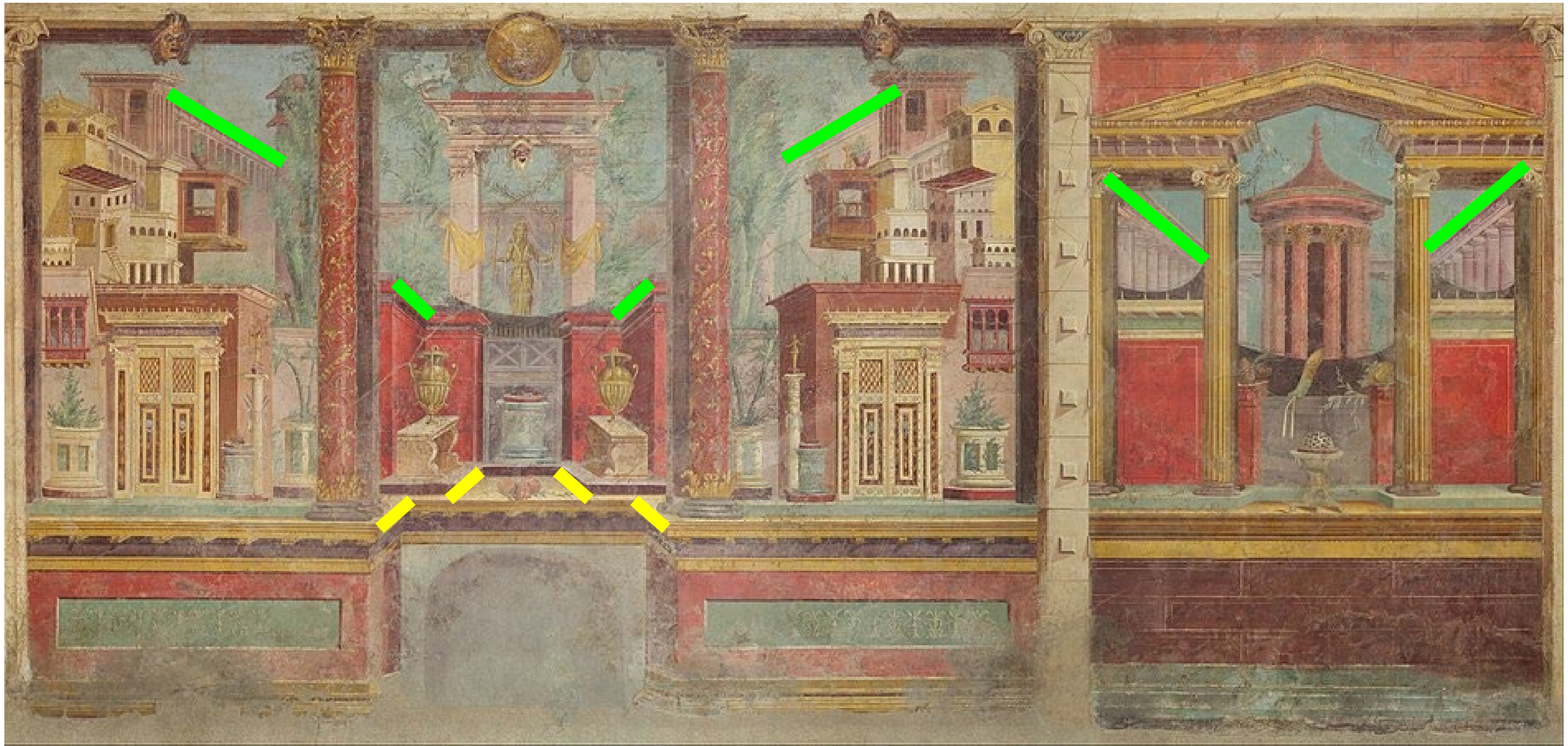
  - This looks like a panorama, right? ☺



Unknown, Heiji Monogatari Emaki (Sanjo Scroll) (平治物語絵巻 (三条殿焼討)) (Night Attack on the Sanjō Palace) (detail) (Kamakura, late 1200s), colour and ink on paper, 41.3 x 699.7 cm, Museum of Fine Arts, Boston, MA. Wikimedia Commons.

A Historical Note on Perspective

- The assertion in many books on art, optics, graphics, etc. that “perspective was discovered during the European Renaissance” is **questionable**
  - The Renaissance was the first time that European artists used perspective after the dark ages
- evidence from other older civilizations: Ancient Rome (1st centure BC)



A Historical Note on Perspective

- The assertion in many books that “perspective was discovered during the European Renaissance” is **questionable**
  - The Renaissance was the first time that European artists used perspective after the dark ages
- evidence from other older civilizations: (ruins of Pompeii)

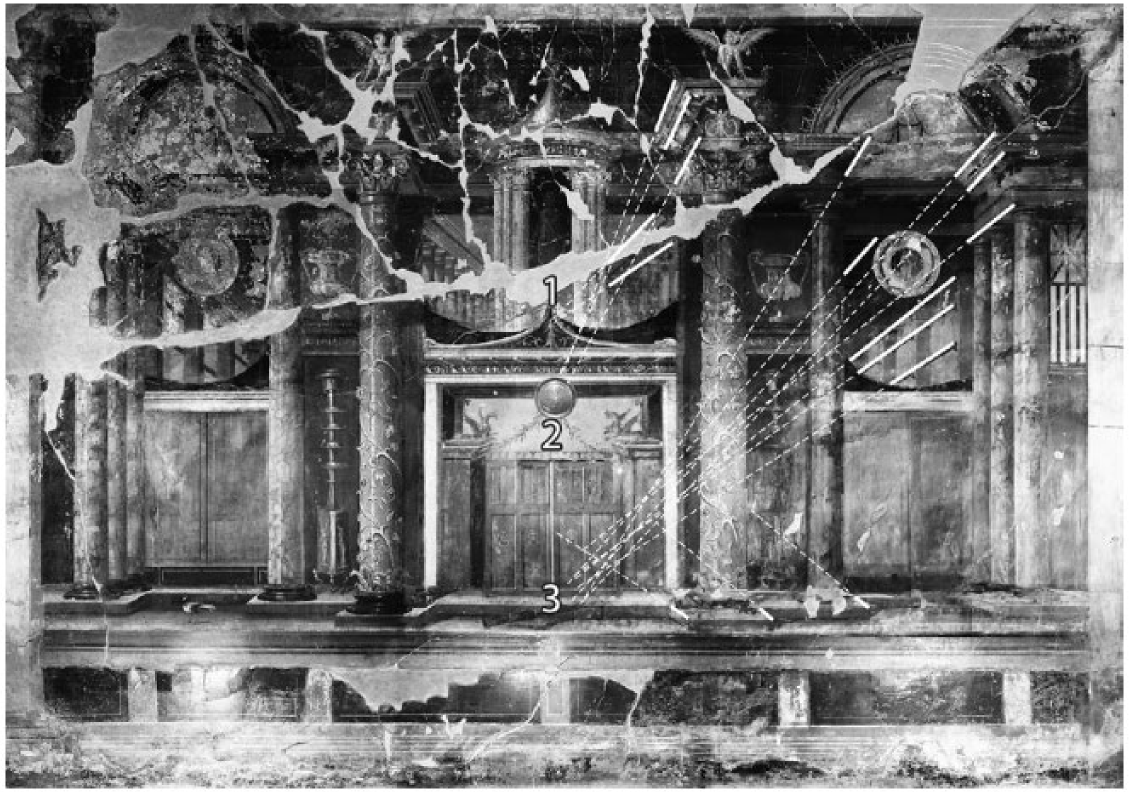


Fig. 7. Diagram showing perspective systems on the west wall of Room 14, Villa of Oplontis; nos. 1–3 represent three distinct convergence systems (courtesy Ministero per i Beni e le Attività Culturali, Soprintendenza Speciale per i Beni Archeologici di Napoli e Pompei).

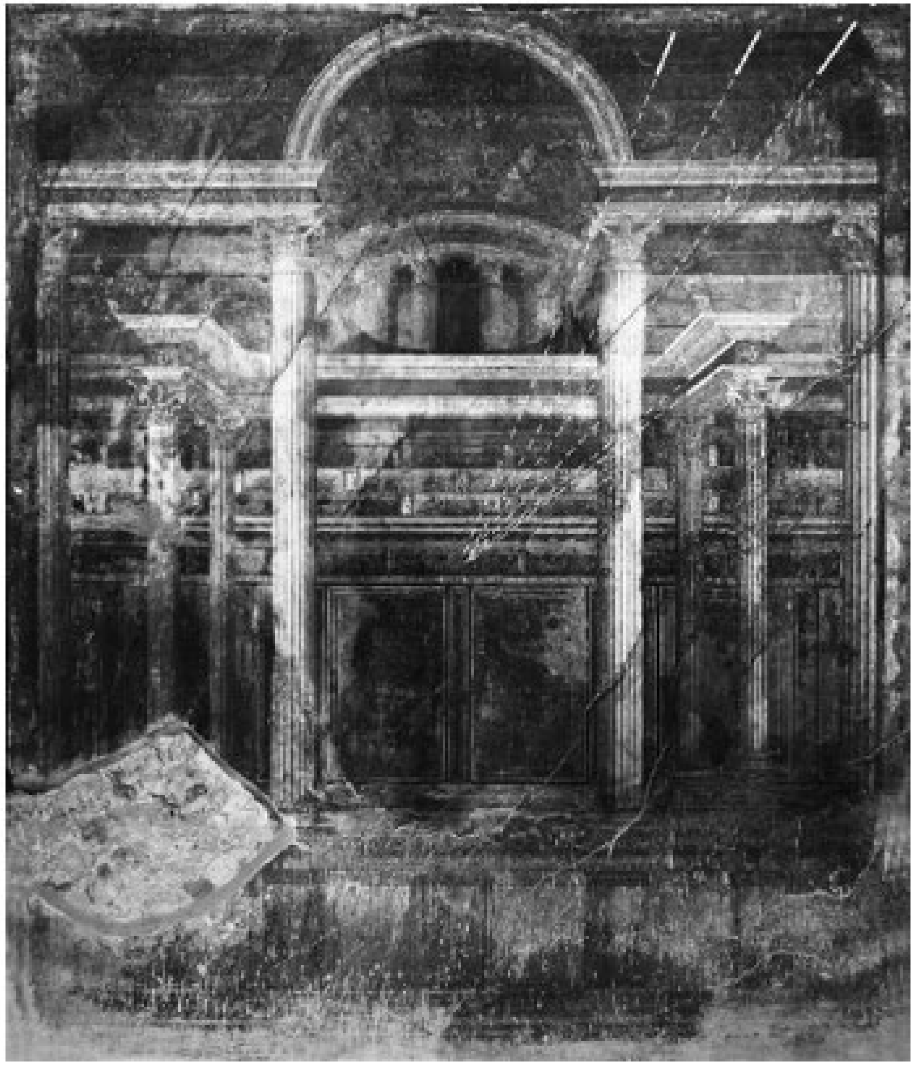
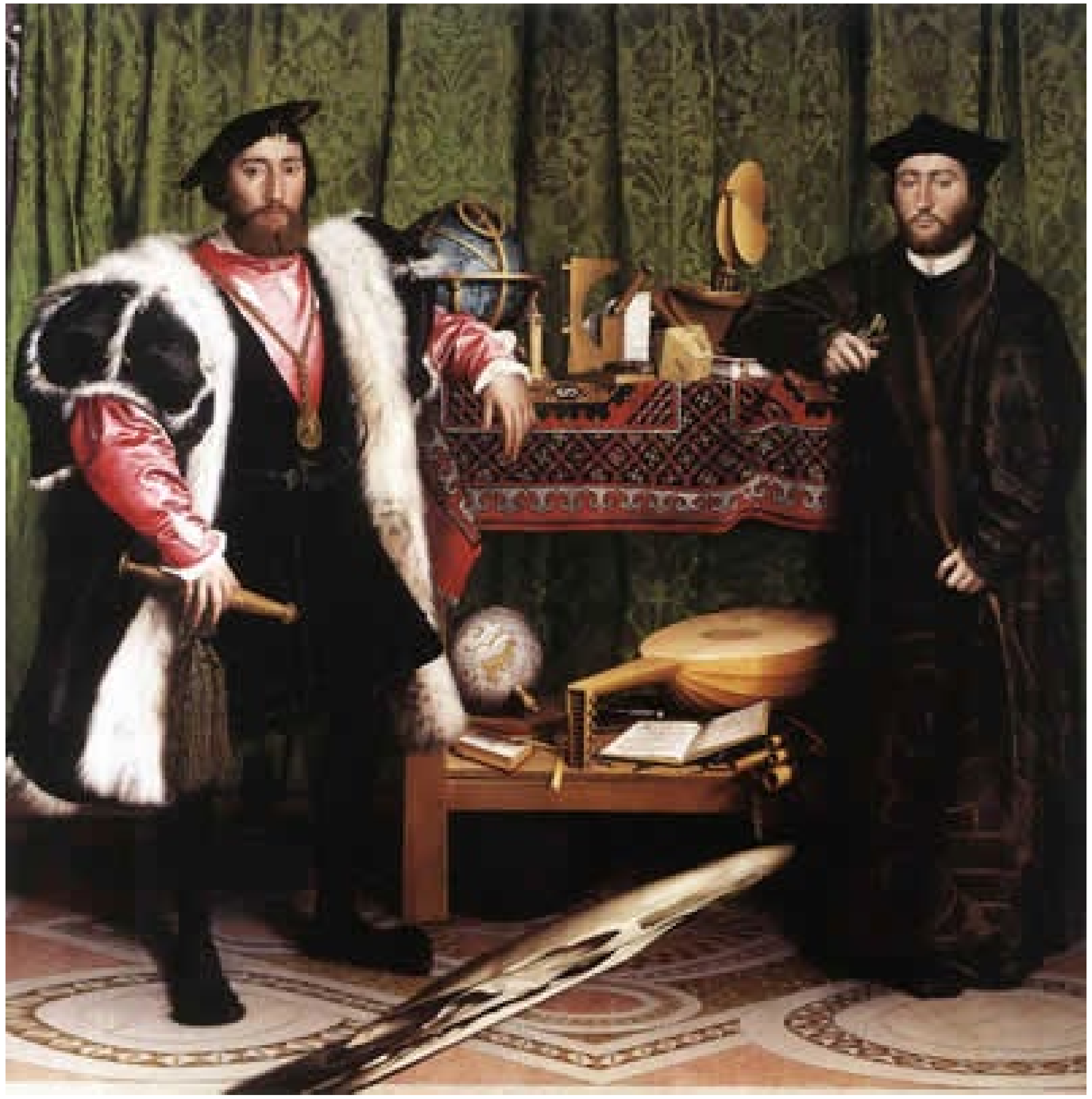


Fig. 5. Diagram showing perspective systems on the back wall of Alcove B, Room 16, Villa of the Mysteries, Pompeii (courtesy Ministero per i Beni e le Attività Culturali, Soprintendenza Speciale per i Beni Archeologici di Napoli e Pompei).

Philip Stinson. (2011). Perspective Systems in Roman Second Style Wall Painting. *American Journal of Archaeology*, *115*(3), 403–426. https://doi.org/10.3764/aja.115.3.0403

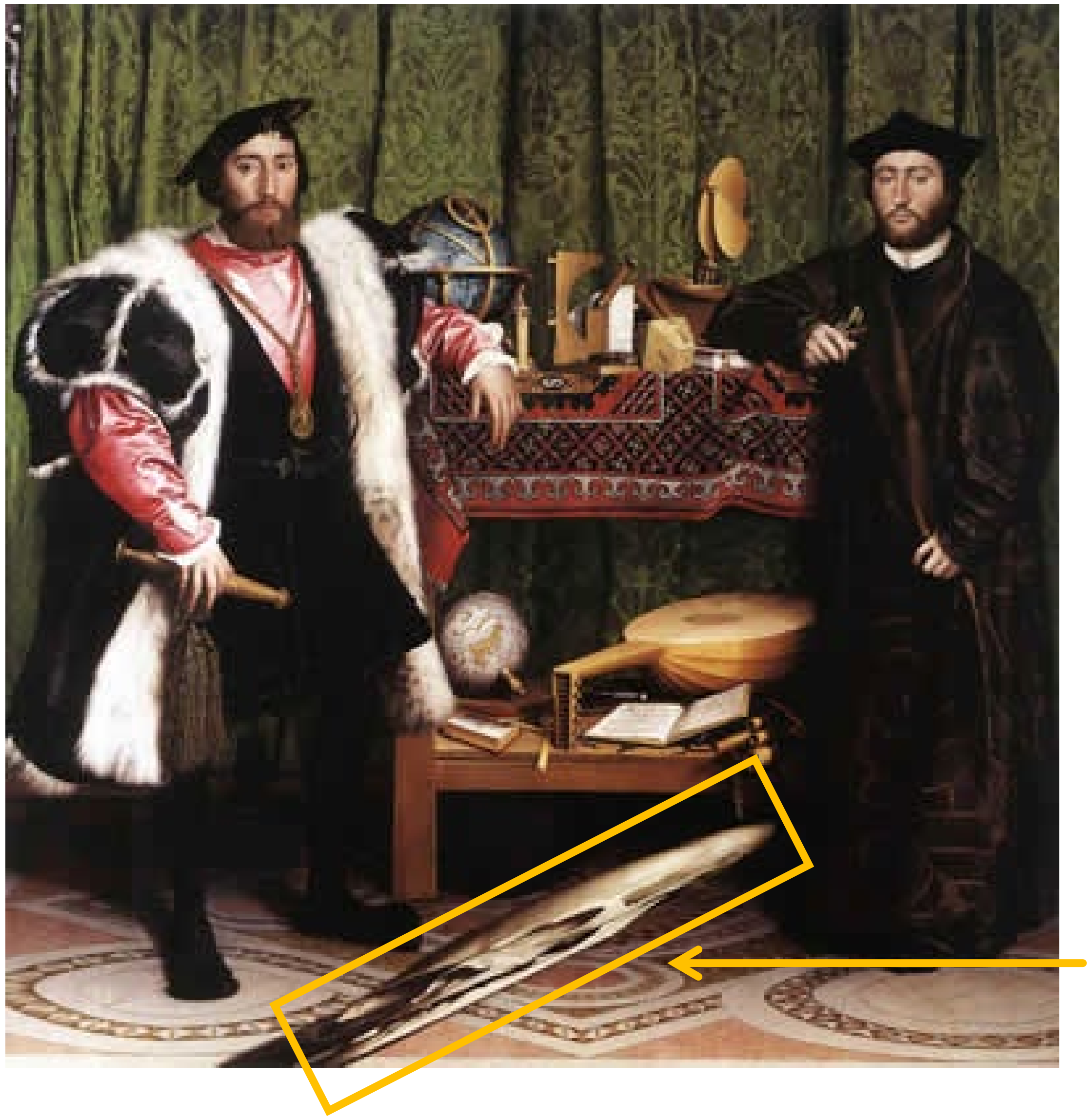
A weird drawing

Holbein, “The Ambassadors”



A weird drawing

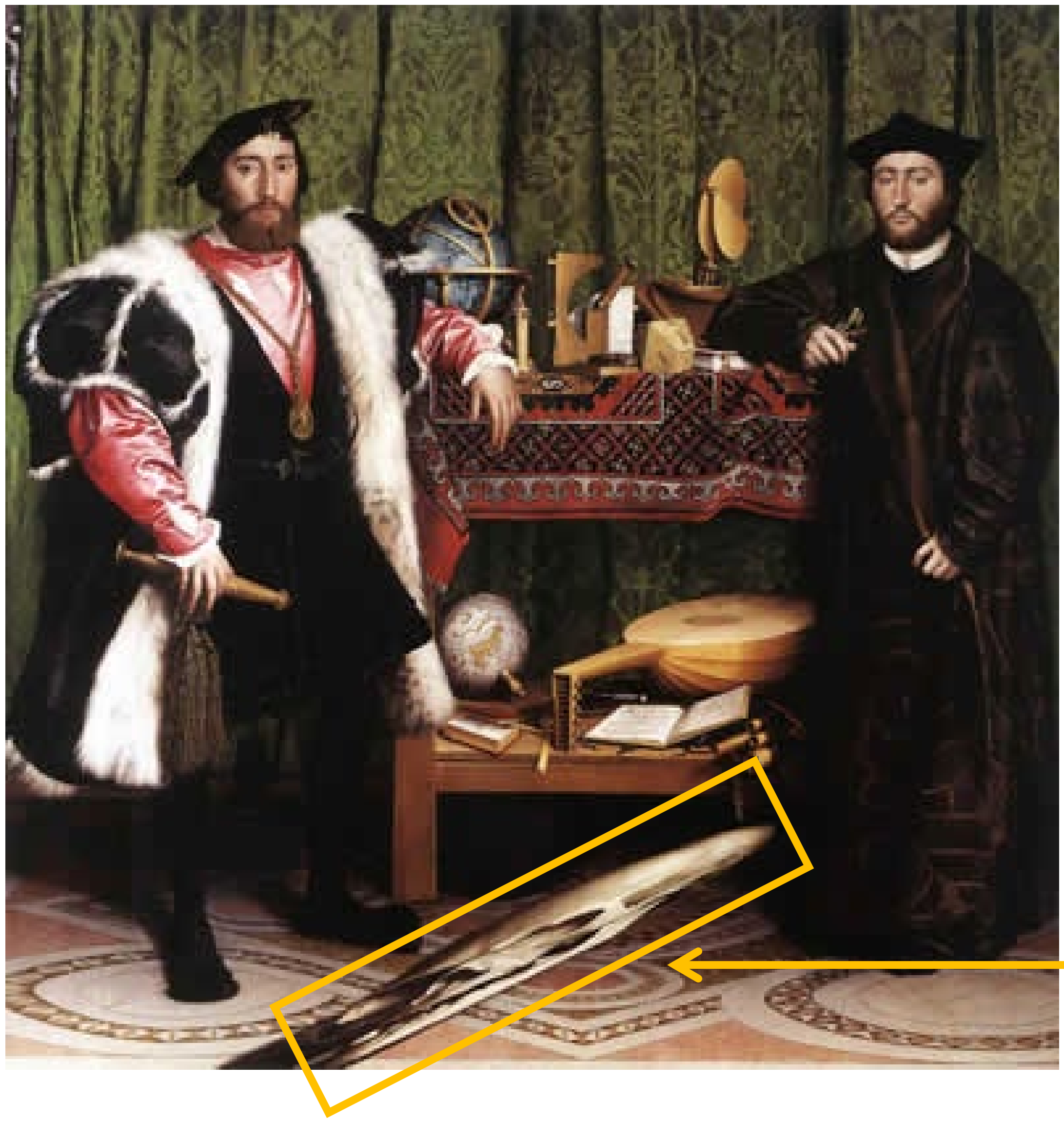
Holbein, “The Ambassadors”

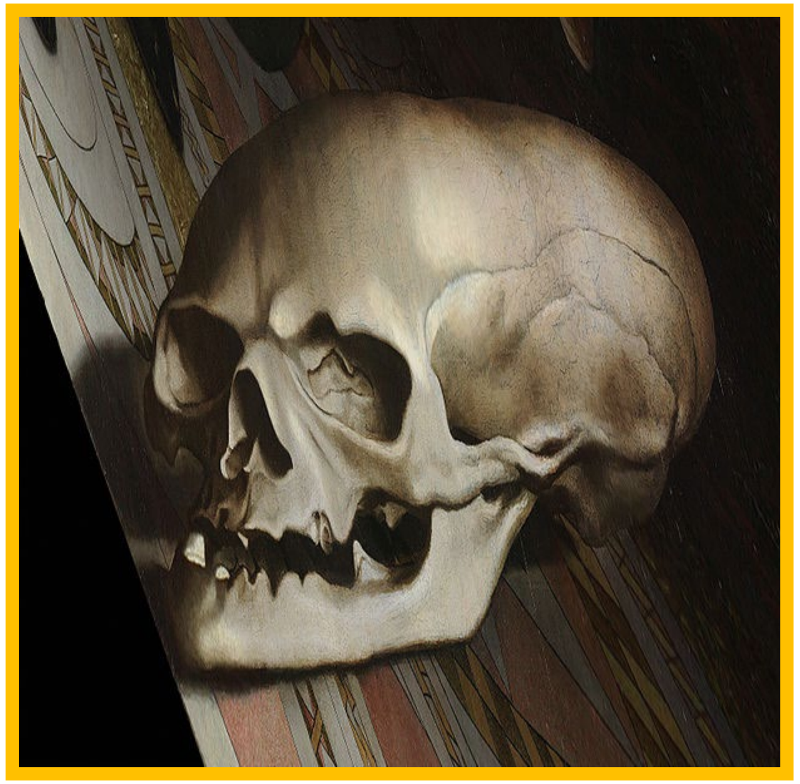


What’s this???

A weird drawing

Holbein, “The Ambassadors”



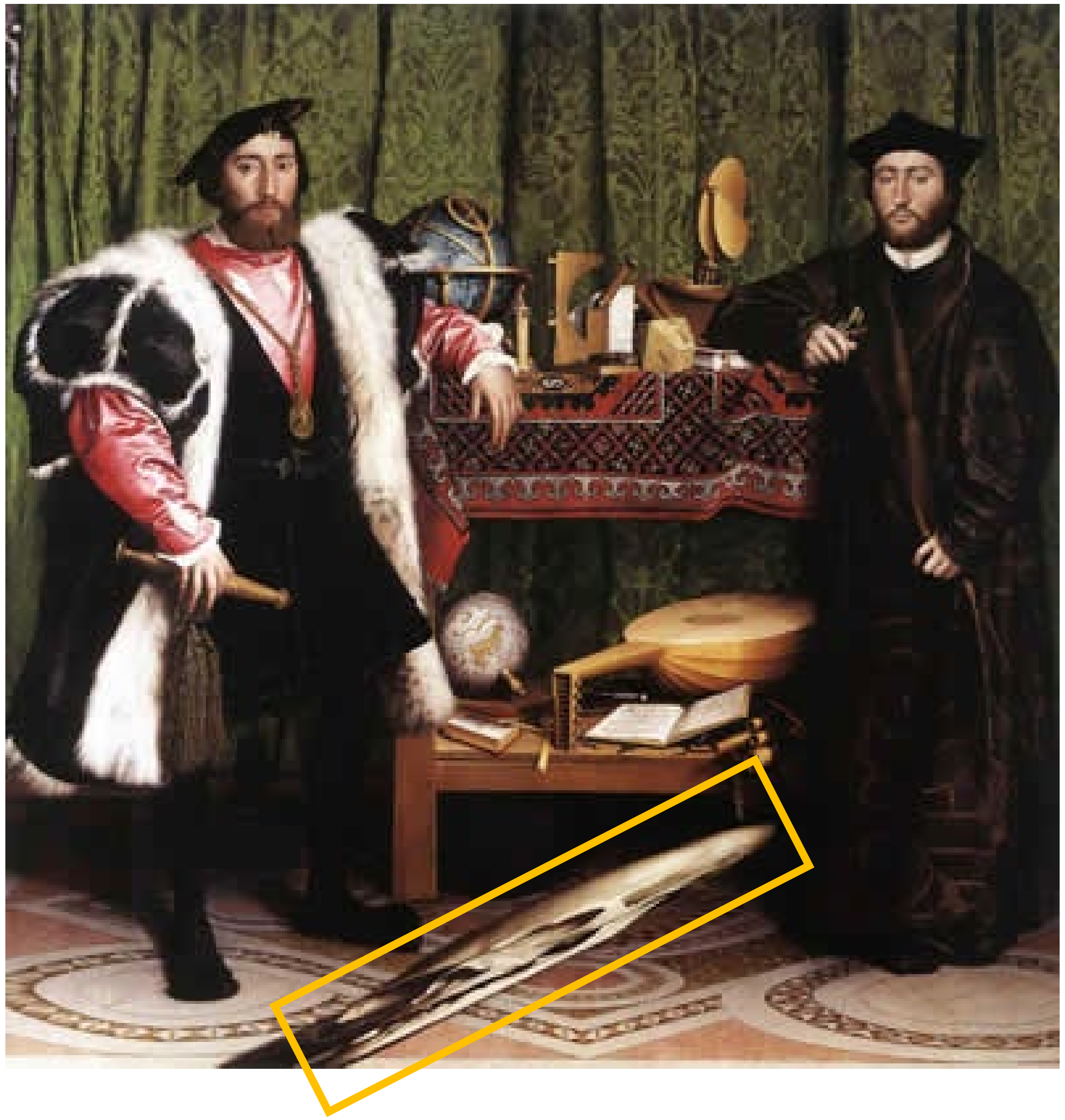


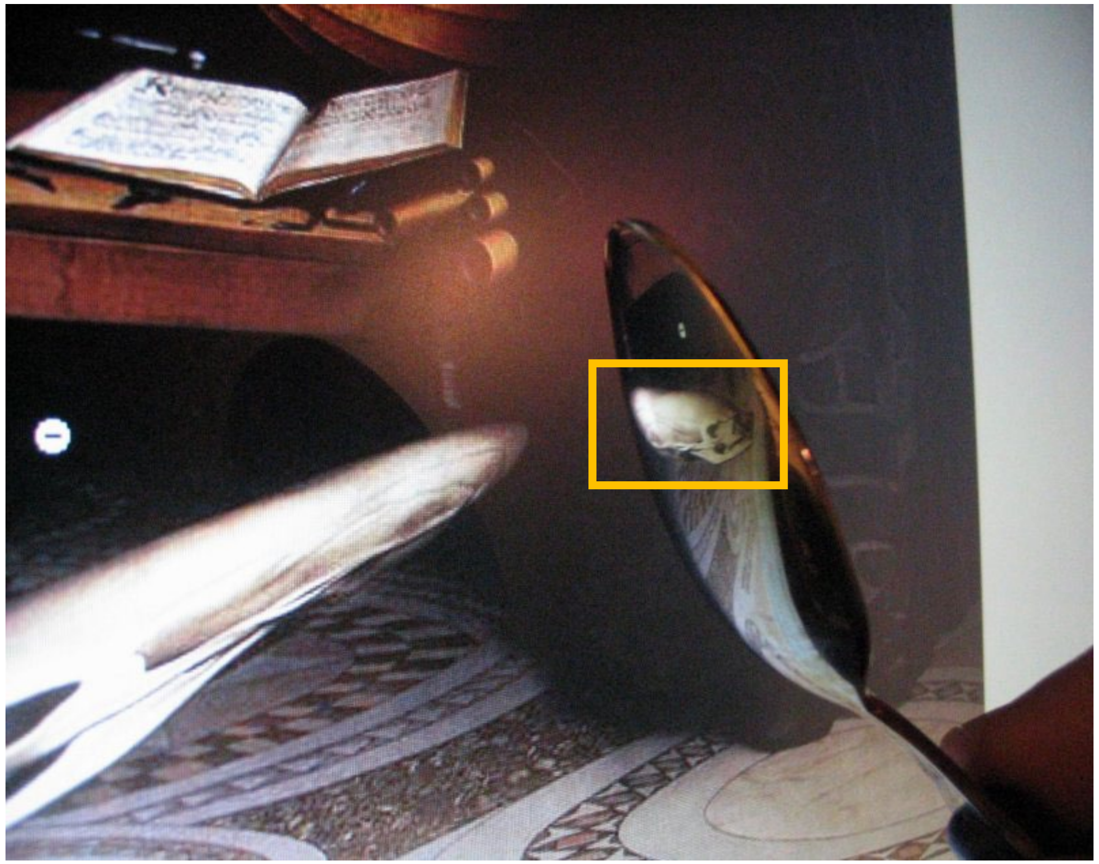
rectified view

skull under anamorphic perspective

A weird drawing

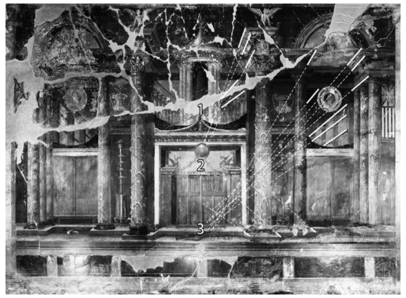
Holbein, “The Ambassadors”





DIY: use a polished spoon to see the skull

Extra Credit Assignment (no due date)



Go to these (or any other) art museums:
(both are FREE and open Wed-Sun 1000—1700)

Baltimore Museum of Art
10 Art Museum Dr, Baltimore, MD 21218

Walters Art Museum
600 N. Charles St, Baltimore, MD 21201
FREE

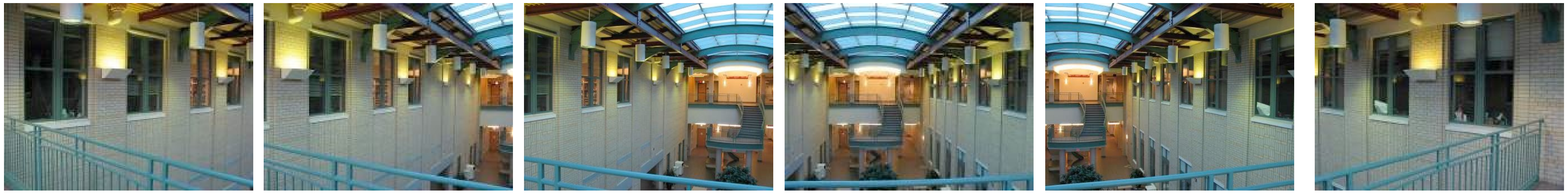
Find examples in which perspective projection is weird …

Take pictures, note down the region and year (e.g. “China 11th century A.D.”)

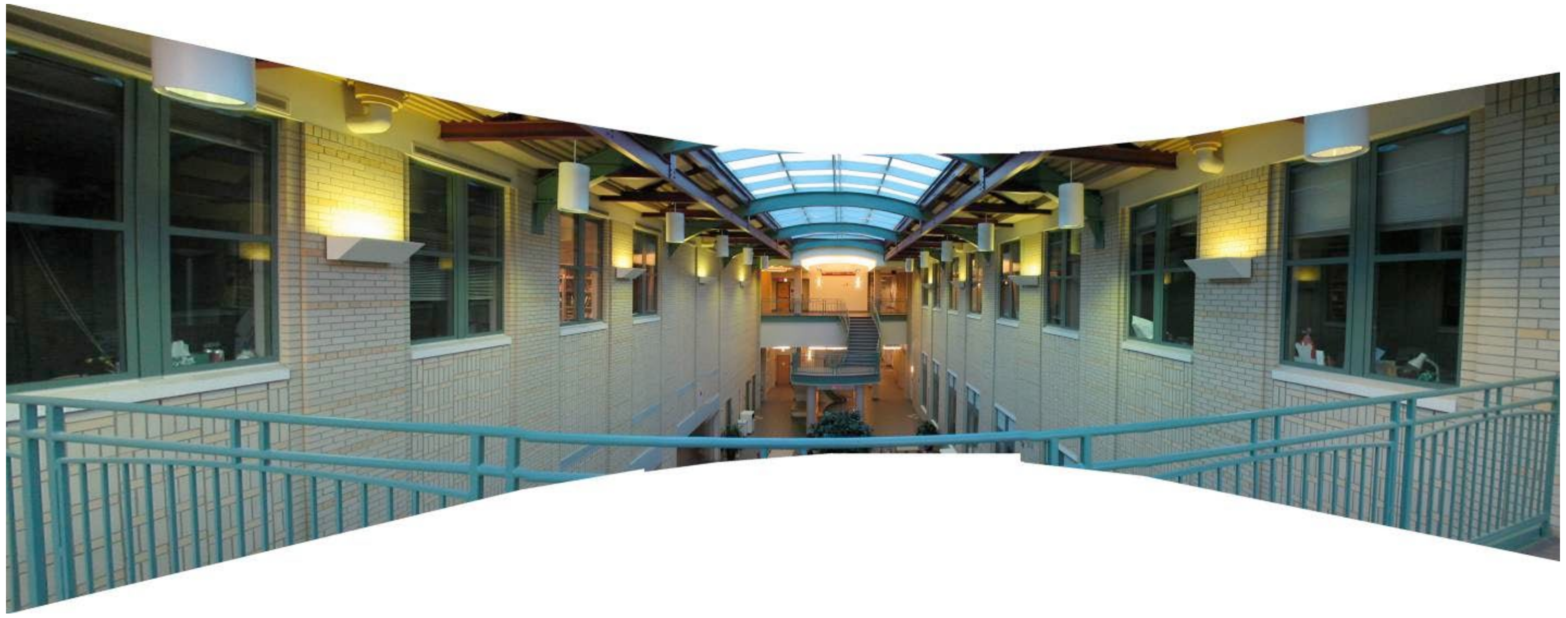
Tell us what is weird about the perspective ☺

We will send a combined report to the museums and publish our findings via the CSEE department!

How do we stitch images from different viewpoints?

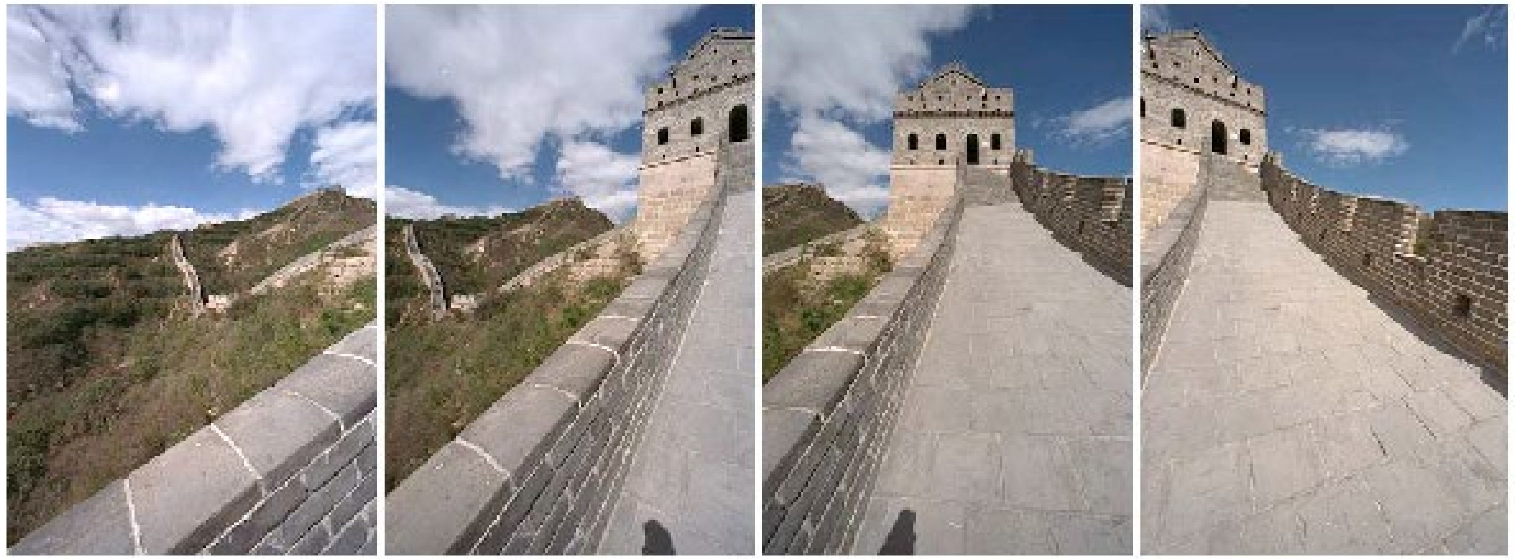


Use image homographies.

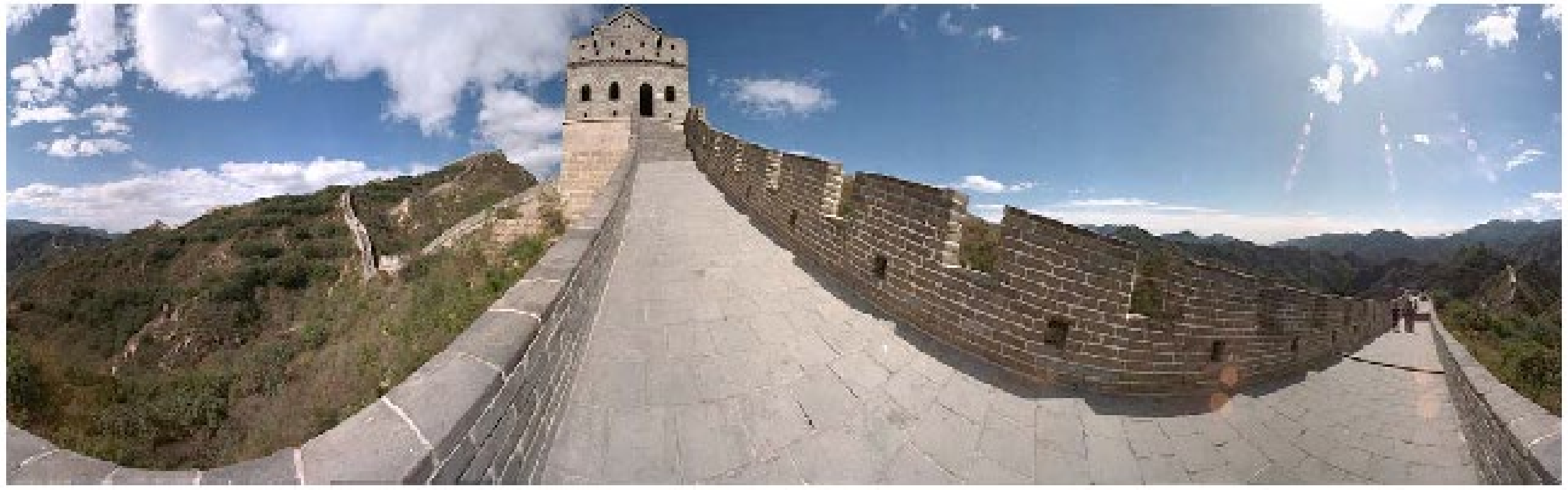


Panoramas from image stitching

1. Capture multiple images from different viewpoints.

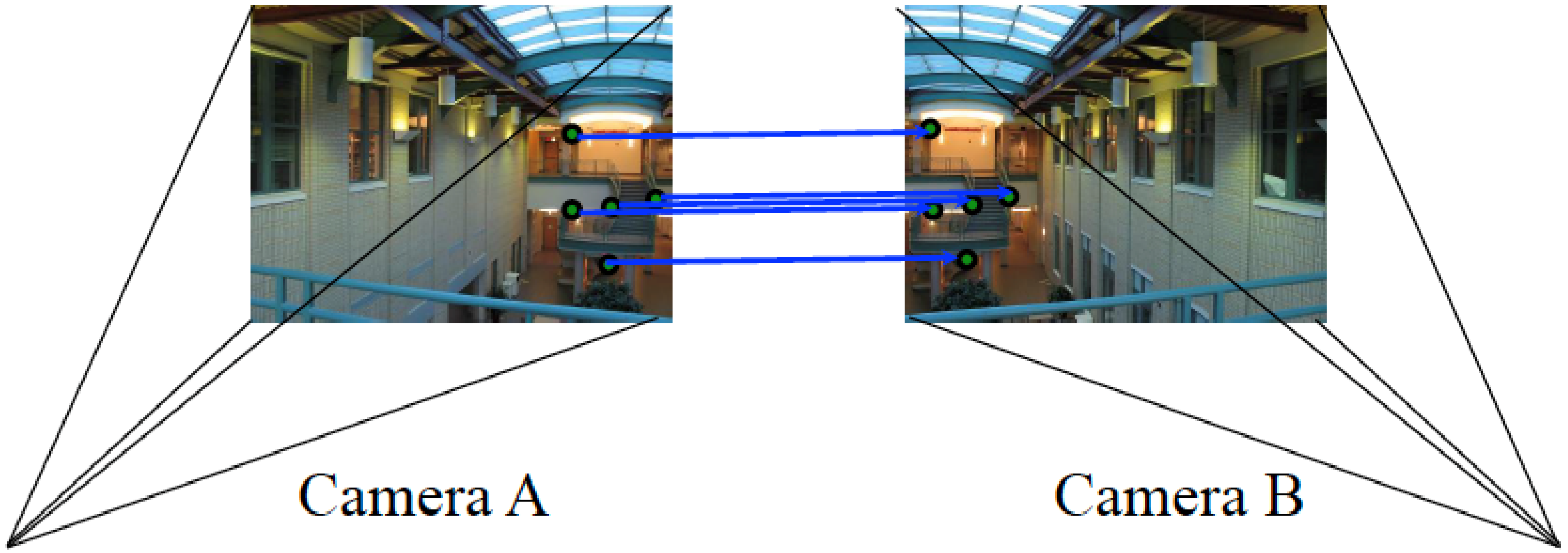


2. Stitch them together into a virtual wide-angle image.



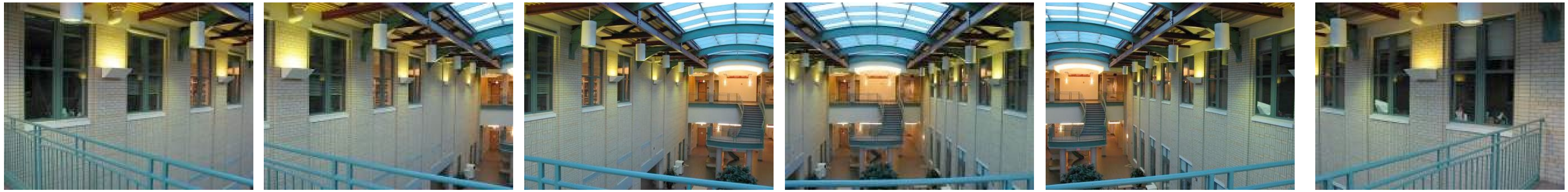
When can we use homographies?

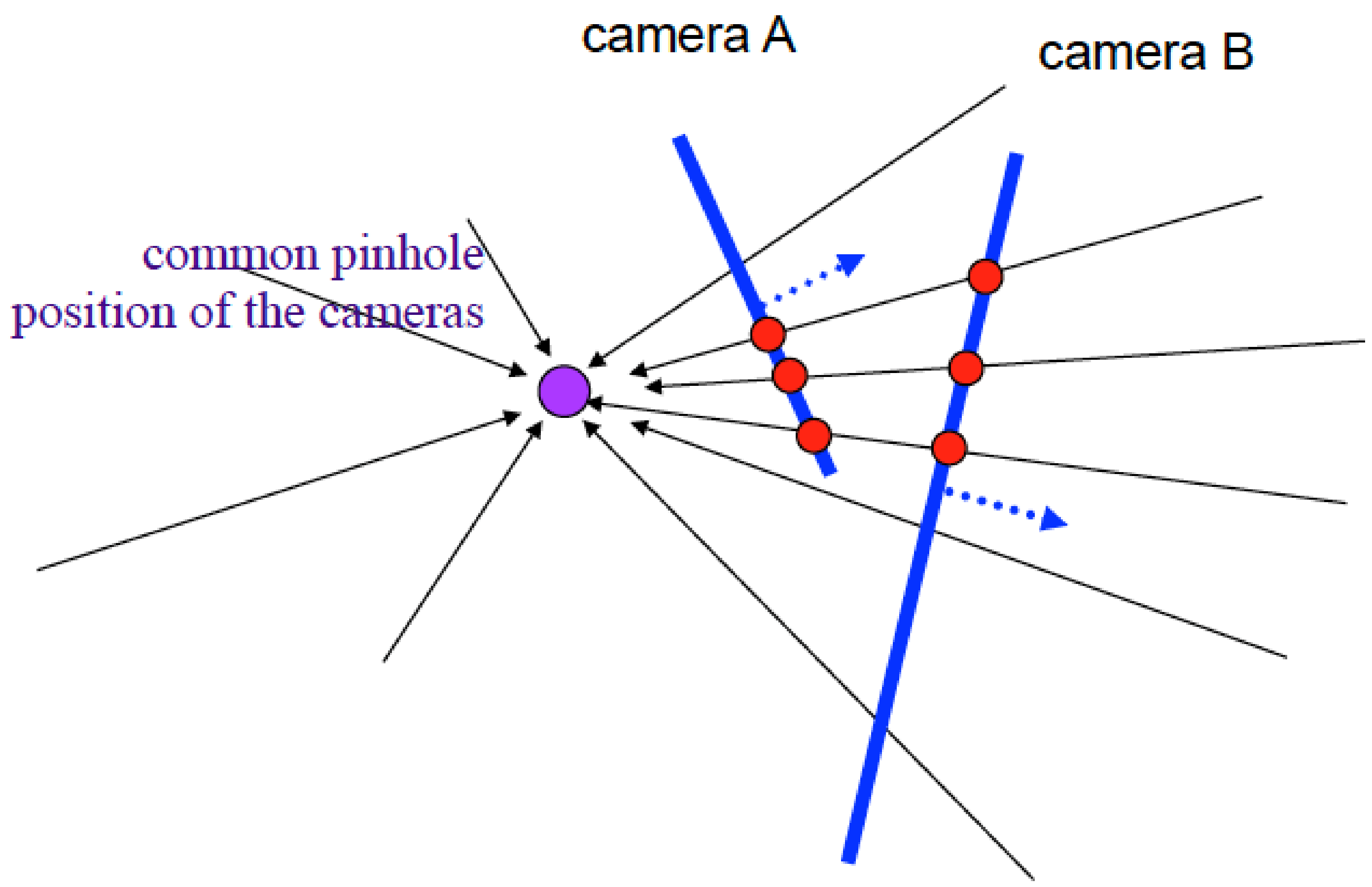
Under what conditions can you know where to translate each point of image A to where it would appear in camera B (with calibrated cameras), knowing nothing about image depths?



We can use homographies when…

1. The scene is captured under camera rotation only (no translation or pose change)





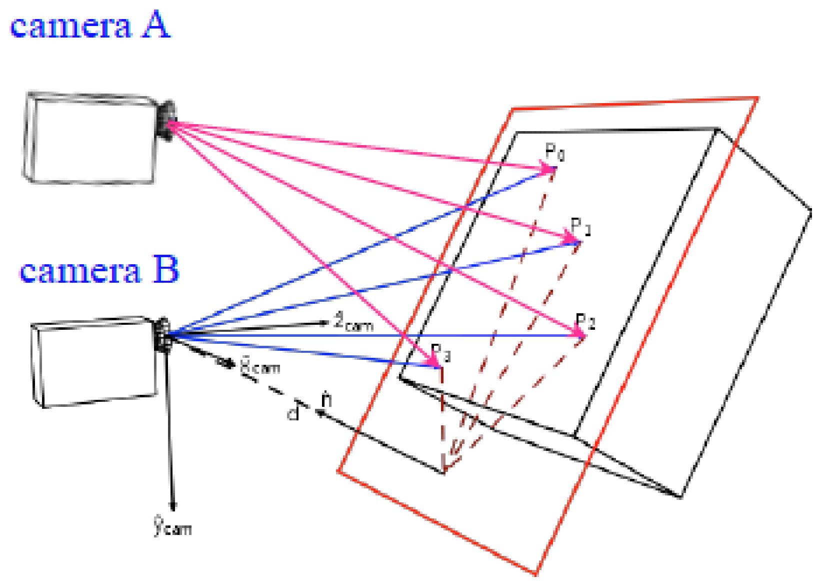
Can generate any synthetic camera view
as long as it has the same center of projection!

We can use homographies when…

2. The scene is planar; or

3. … the scene is very far or has small (relative) depth variation → scene is approximately planar

Images of planar objects, taken by generically offset cameras, are also related by a homography.



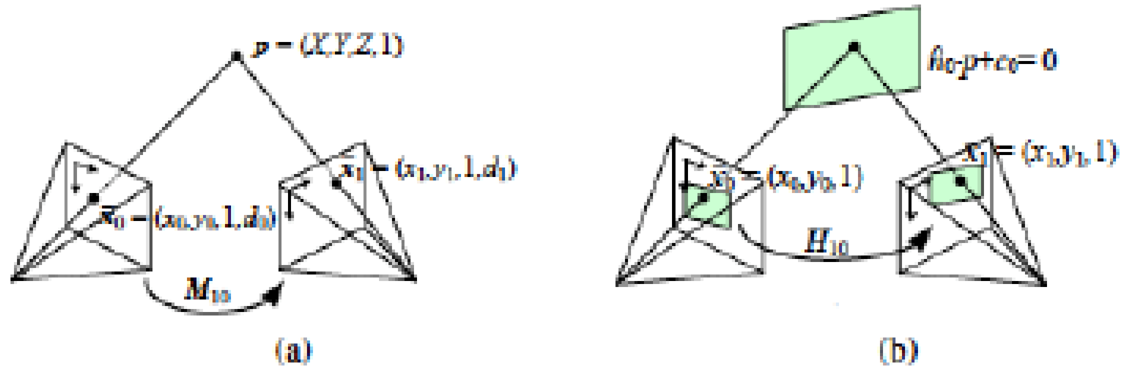
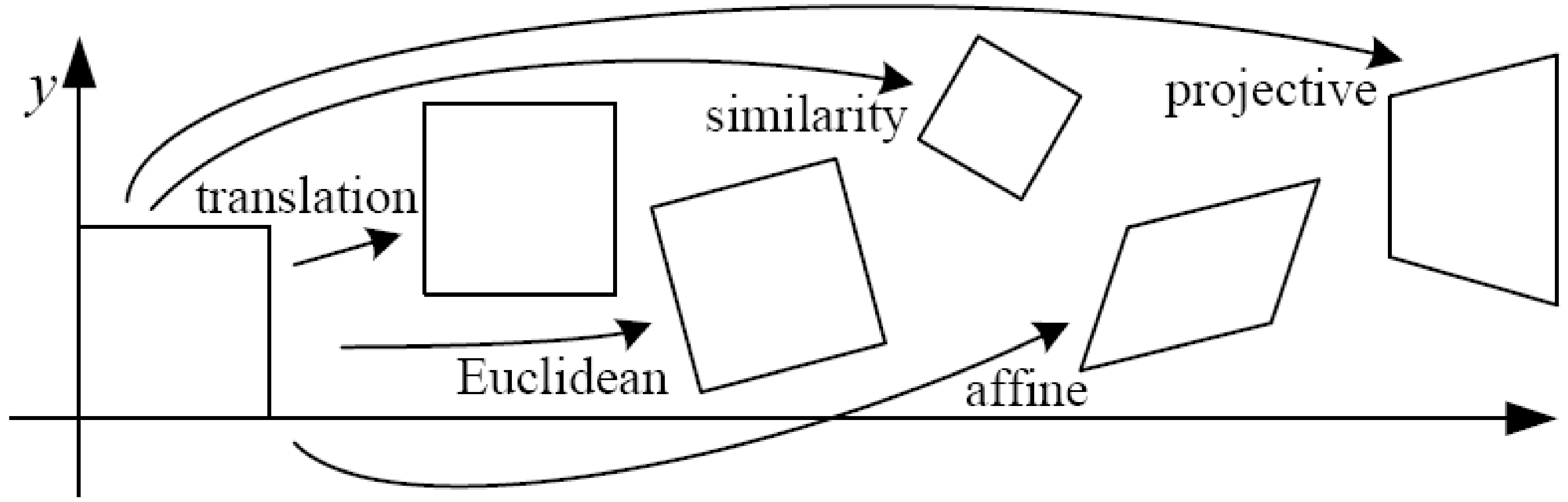
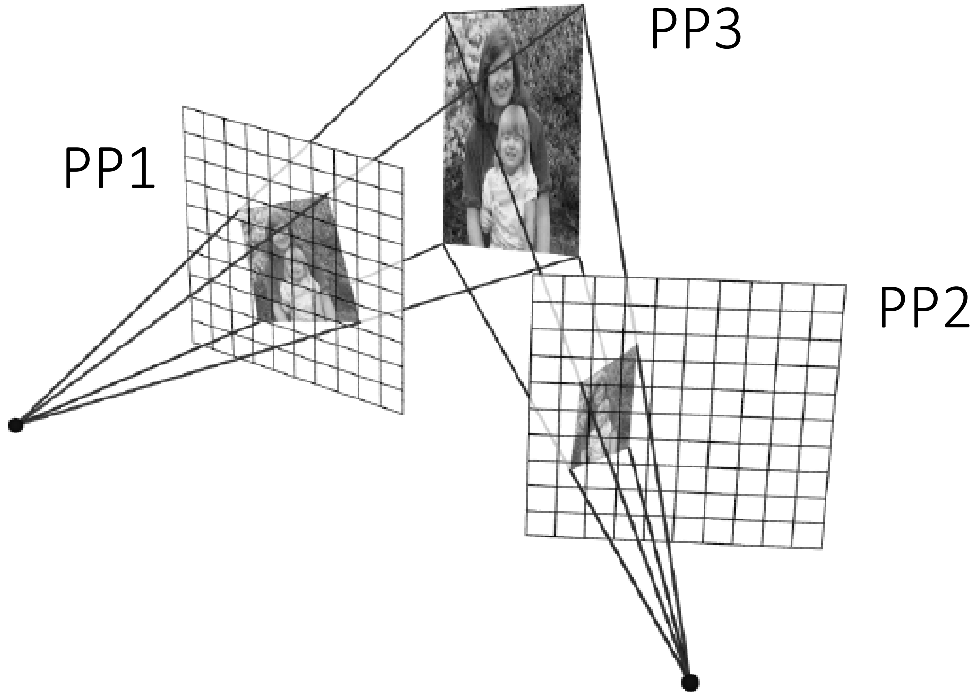

$p = (X,Y,Z,1)$
$\tilde{x}_0 = (x_0, y_0, 1, d_0)$
$\tilde{x}_1 = (x_1, y_1, 1, d_1)$
M_{10}
(a)
$\hat{n}_0 \cdot p + c_0 = 0$
$\tilde{x}_0 = (x_0, y_0, 1)$
$\tilde{x}_1 = (x_1, y_1, 1)$
H_{10}
(b)

Figure 2.12 A point is projected into two images: (a) relationship between the 3D point coordinate $(X, Y, Z, 1)$ and the 2D projected point $(x, y, 1, d)$; (b) planar homography induced by points all lying on a common plane $\hat{n}_0 \cdot p + c_0 = 0$.

Computing with homographies

Classification of 2D transformations





Which kind transformation is needed to warp projective plane 1 into projective plane 2?

- A projective transformation (a.k.a. a homography).

Applying a homography

1. Convert to homogeneous coordinates: $p = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

What is the size of the homography matrix? $\searrow$

2. Multiply by the homography matrix: $P' = H \cdot P$

3. Convert back to heterogeneous coordinates: $P' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \Rightarrow p' = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$

Applying a homography

1. Convert to homogeneous coordinates: $p = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

What is the size of the homography matrix? ↘ Answer: 3 x 3

2. Multiply by the homography matrix: $P' = H \cdot P$

How many degrees of freedom does the homography matrix have? ↗

3. Convert back to heterogeneous coordinates: $P' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \Rightarrow p' = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$

Applying a homography

1. Convert to homogeneous coordinates: $p = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

What is the size of the homography matrix? ↘ Answer: 3 x 3

2. Multiply by the homography matrix: $P' = H \cdot P$

How many degrees of freedom does the homography matrix have? ↗ Answer: 8

3. Convert back to heterogeneous coordinates: $P' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \Rightarrow p' = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$

Applying a homography

What is the size of the homography matrix? $\searrow$ Answer: 3 x 3

$$P' = H \cdot P$$

How many degrees of freedom does the homography matrix have? $\nearrow$ Answer: 8

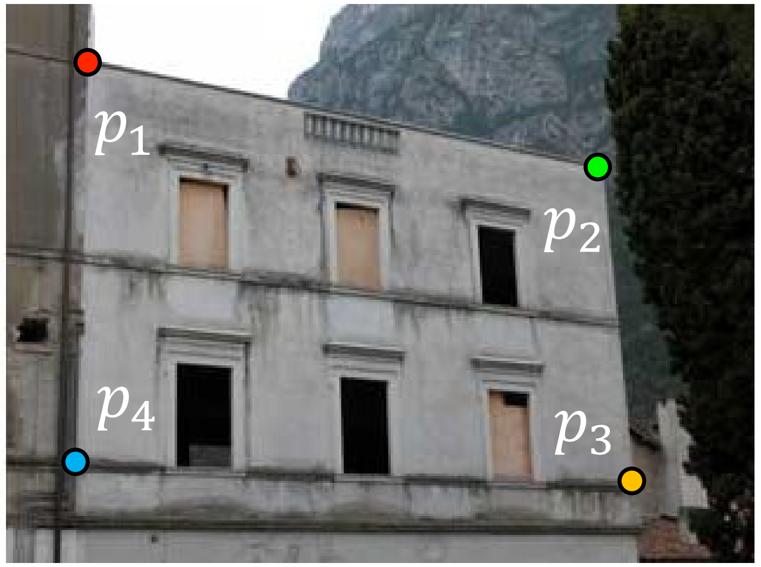
How do we compute the homography matrix?

The direct linear transform (DLT) using the SVD

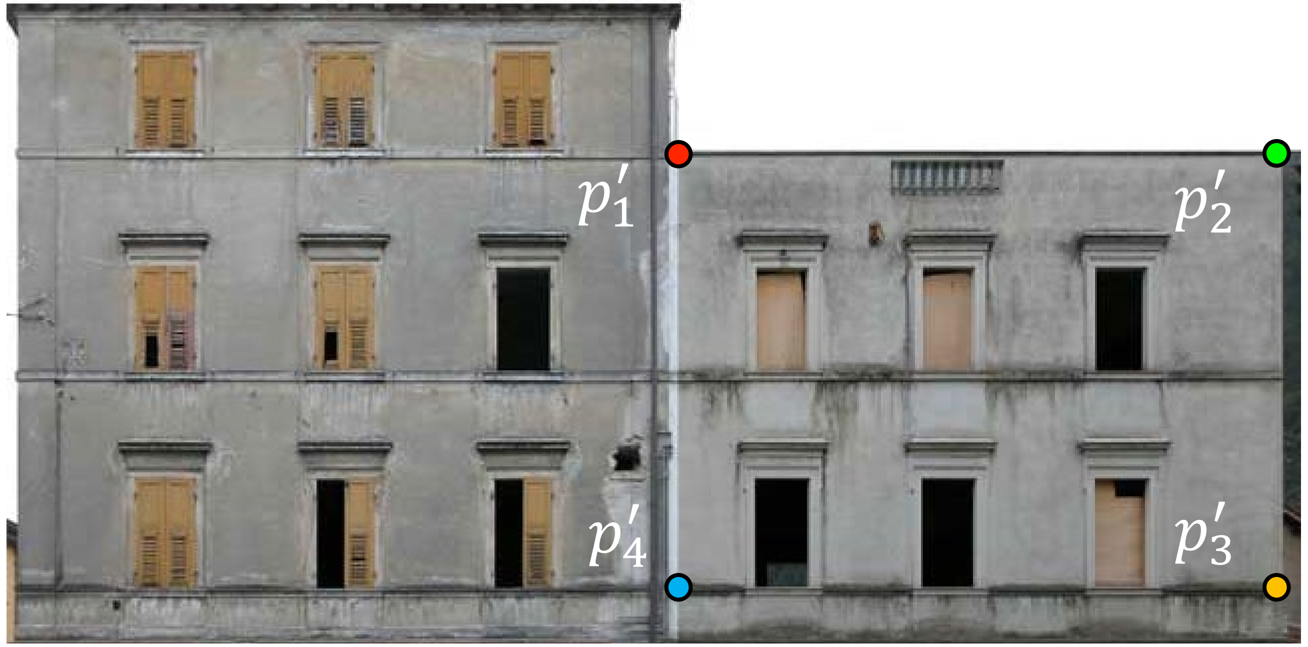
Create point correspondences

Given a set of matched feature points $\{p_i, p_i'\}$ find the best estimate of H such that

$$P' = H \cdot P$$



original image



target image

How many correspondences do we need?

Determining the homography matrix

Write out linear equation for each correspondence:

$$P' = H \cdot P \quad \text{or} \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Determining the homography matrix

Write out linear equation for each correspondence:

$$P' = H \cdot P \quad \text{or} \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Expand matrix multiplication:

$$x' = \alpha(h_1x + h_2y + h_3)$$
$$y' = \alpha(h_4x + h_5y + h_6)$$
$$1 = \alpha(h_7x + h_8y + h_9)$$

Determining the homography matrix

Write out linear equation for each correspondence:

$$P' = H \cdot P \quad \text{or} \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Expand matrix multiplication:

$$x' = \alpha(h_1x + h_2y + h_3)$$
$$y' = \alpha(h_4x + h_5y + h_6)$$
$$1 = \alpha(h_7x + h_8y + h_9)$$

Divide out unknown scale factor:

$$x'(h_7x + h_8y + h_9) = (h_1x + h_2y + h_3)$$
$$y'(h_7x + h_8y + h_9) = (h_4x + h_5y + h_6)$$

How do you rearrange terms to make it a linear system?

$$x'(h_7x + h_8y + h_9) = (h_1x + h_2y + h_3)$$
$$y'(h_7x + h_8y + h_9) = (h_4x + h_5y + h_6)$$

Just rearrange the terms

$$\boxed{\begin{aligned} h_7xx' + h_8yx' + h_9x' - h_1x - h_2y - h_3 = 0 \\ h_7xy' + h_8yy' + h_9y' - h_4x - h_5y - h_6 = 0 \end{aligned}}$$

Determining the homography matrix

Re-arrange terms:

$$h_7 x x' + h_8 y x' + h_9 x' - h_1 x - h_2 y - h_3 = 0$$
$$h_7 x y' + h_8 y y' + h_9 y' - h_4 x - h_5 y - h_6 = 0$$

Re-write in matrix form:

How many equations from one point correspondence?

$$\mathbf{A}_i \boldsymbol{h} = \mathbf{0}$$

$$\mathbf{A}_i = \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix}$$

$$\boldsymbol{h} = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 & h_7 & h_8 & h_9 \end{bmatrix}^\top$$

Determining the homography matrix

Stack together constraints from multiple point correspondences:

$$\mathbf{A}\boldsymbol{h} = \mathbf{0}$$

$$\begin{bmatrix} \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \\ \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \\ \vdots \\ \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Homogeneous linear least squares problem

Reminder: Determining unknown transformations

Affine transformation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Why can we drop the last line?

Vectorize transformation parameters:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{bmatrix}$$

Stack equations from point correspondences:

$$\underbrace{\begin{bmatrix} x' \\ y' \\ x' \\ y' \\ \vdots \\ x' \\ y' \end{bmatrix}}_{\boldsymbol{b}} = \underbrace{\begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ & & & \vdots & & \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{bmatrix}}_{\boldsymbol{x}}$$

Notation in system form:

$$\boxed{Ax = b}$$

Reminder: Determining unknown transformations

Convert the system to a linear least-squares problem:

$$E_{\mathrm{LLS}} = ||\mathbf{A}\boldsymbol{x} - \boldsymbol{b}||^2$$

$\boxed{\text{Solve using linear least squares solver in Python}}$

Determining the homography matrix

Stack together constraints from multiple point correspondences:

$$\mathbf{A}\boldsymbol{h} = \mathbf{0}$$

$$\begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix}$$
$$\begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix}$$
$$\vdots$$
$$\begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Homogeneous linear least squares problem

- How do we solve this?

Determining the homography matrix

Stack together constraints from multiple point correspondences:

$$\mathbf{A}\boldsymbol{h} = \mathbf{0}$$

$$\begin{bmatrix} \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \\ \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \\ \vdots \\ \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Homogeneous linear least squares problem

- Solve with SVD

Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD)

- SVD is a matrix technique that has some important uses in computer vision
- These include:
  - Solving a set of homogeneous linear equations
    - Namely we solve for the vector **x** in the equation **Ax** = **0**
  - Guaranteeing that the entries of a matrix estimated numerically satisfy some given constraints (e.g., orthogonality)
    - For example, we have computed **R** and now want to make sure that it is a valid rotation matrix

Singular Value Decomposition (SVD)

- Any (real) *mxn* matrix **A** can be written as the product of three matrices

  $$\mathbf{A} = \mathbf{U}\,\mathbf{D}\,\mathbf{V}^T$$

  - **U** (*mxm*) and **V**(*nxn*) have columns that are mutually orthogonal unit vectors
  - **D** (*mxn*) is diagonal; its diagonal elements σ_i are called singular values, and $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_n \geq 0$

Some properties of SVD

- We can represent A in terms of the vectors u and v

  $\mathbf{A}\,\mathbf{v}_j = \sigma_j\,\mathbf{u}_j$
- or

$$\mathbf{A} = \sum_{j=0}^{p-1} \sigma_j \mathbf{u}_j \mathbf{v}_j^T$$

- The vectors $\mathbf{u}_j$ are called the "principal components" of **A**

- Sometimes we want to compute an approximation to **A** using fewer principal components
- If we truncate the expansion, we obtain the best possible least squares approximation[1] to the original matrix **A**

$$\mathbf{A} \approx \sum_{j=0}^{t} \sigma_j \mathbf{u}_j \mathbf{v}_j^T$$

[1]In terms of the Frobenius norm, defined as

$$\|\mathbf{A}\|_F = \sum_{i,j} a_{i,j}^2$$

Singular Value Decomposition

$$\mathbf{A} = \underset{n \times m}{} \; \overset{\text{ortho-normal}}{\underset{n \times n}{\mathbf{U}}} \; \overset{\text{diagonal}}{\underset{n \times m}{\mathbf{\Sigma}}} \; \overset{\text{ortho-normal}}{\underset{m \times m}{\mathbf{V}}}^\top$$

unit norm constraint

n x m

$$= \sum_{i=1}^{9} \sigma_i \underset{n \times 1}{\boldsymbol{u}_i} \underset{1 \times m}{\boldsymbol{v}_i^\top}$$

Each column of V represents a solution for $\mathbf{A}\boldsymbol{h} = \mathbf{0}$

where the singular value represents the reprojection error

Solving for H using DLT

Given $\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$ solve for H such that $\boldsymbol{x}' = \mathbf{H}\boldsymbol{x}$

1. For each correspondence, create 2x9 matrix $\mathbf{A}_i$

2. Concatenate into single 2n x 9 matrix $\mathbf{A}$

3. Compute SVD of $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$

4. Store singular vector of the smallest singular value $\boldsymbol{h} = \boldsymbol{v}_{\hat{i}}$

5. Reshape to get $\mathbf{H}$

General form of total least squares

(**Warning:** change of notation. x is a vector of parameters!)

$$E_{\text{TLS}} = \sum_i (\boldsymbol{a}_i \boldsymbol{x})^2$$

$$= \|\mathbf{A}\boldsymbol{x}\|^2$$ (matrix form)

$$\|\boldsymbol{x}\|^2 = 1$$ constraint

minimize $\|\mathbf{A}\boldsymbol{x}\|^2$

subject to $\|\boldsymbol{x}\|^2 = 1$

$\Rightarrow$

(Rayleigh quotient)

minimize $\dfrac{\|\mathbf{A}\boldsymbol{x}\|^2}{\|\boldsymbol{x}\|^2}$

Solution is the eigenvector corresponding to smallest eigenvalue of

$$\mathbf{A}^\top \mathbf{A}$$

(equivalent)

Solution is the column of **V** corresponding to smallest singular value

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

Application: Solving a System of Homogeneous Equations (continued)

- The solution **x** is the eigenvector corresponding to the only zero eigenvalue of $\mathbf{A}^T\mathbf{A}$
  - Proof: We want to minimize

$$\|\mathbf{Ax}\|^2 = (\mathbf{Ax})^T\mathbf{Ax} = \mathbf{x}^T\mathbf{A}^T\mathbf{Ax} \qquad \text{subject to} \quad \mathbf{x}^T\mathbf{x} = 1$$

  - Introducing a Lagrange multiplier λ, this is equivalent to minimizing

$$L(\mathbf{x}) = \mathbf{x}^T\mathbf{A}^T\mathbf{Ax} - \lambda\left(\mathbf{x}^T\mathbf{x} - 1\right)$$

  - Take derivative wrt **x** and set to zero

$$\mathbf{A}^T\mathbf{Ax} - \lambda\mathbf{x} = 0$$

  - Thus, λ is an eigenvalue of $\mathbf{A}^T\mathbf{A}$, and $\mathbf{x} = \mathbf{e}_\lambda$ is the corresponding eigenvector. $L(\mathbf{e}_\lambda) = \lambda$ is minimized at $\lambda=0$, so $\mathbf{x} = \mathbf{e}_0$ is the eigenvector corresponding to the zero eigenvalue.

Solving Homogeneous Equations with SVD

- Given a system of linear equations **Ax** = 0
- Then the solution **x** is the eigenvector corresponding to the only zero eigenvalue of $\mathbf{A}^T\mathbf{A}$
- Equivalently, we can take the SVD of **A**; ie., $\mathbf{A} = \mathbf{U}\,\mathbf{D}\,\mathbf{V}^T$
  - And **x** is the column of **V** corresponding to the zero singular value of **A**
  - (Since the columns are ordered, this is the rightmost column of **V**)

- Example

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\text{Svd}: \quad \mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

So the last column of V is indeed the solution **x**

Solving for H using DLT

Given $\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$ solve for H such that $\boldsymbol{x}' = \mathbf{H}\boldsymbol{x}$

1. For each correspondence, create 2x9 matrix $\mathbf{A}_i$

2. Concatenate into single 2n x 9 matrix $\mathbf{A}$

3. Compute SVD of $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$

4. Store singular vector of the smallest singular value $\boldsymbol{h} = \boldsymbol{v}_{\hat{i}}$

5. Reshape to get $\mathbf{H}$

Solving the homogenous linear system

Fitting homographies

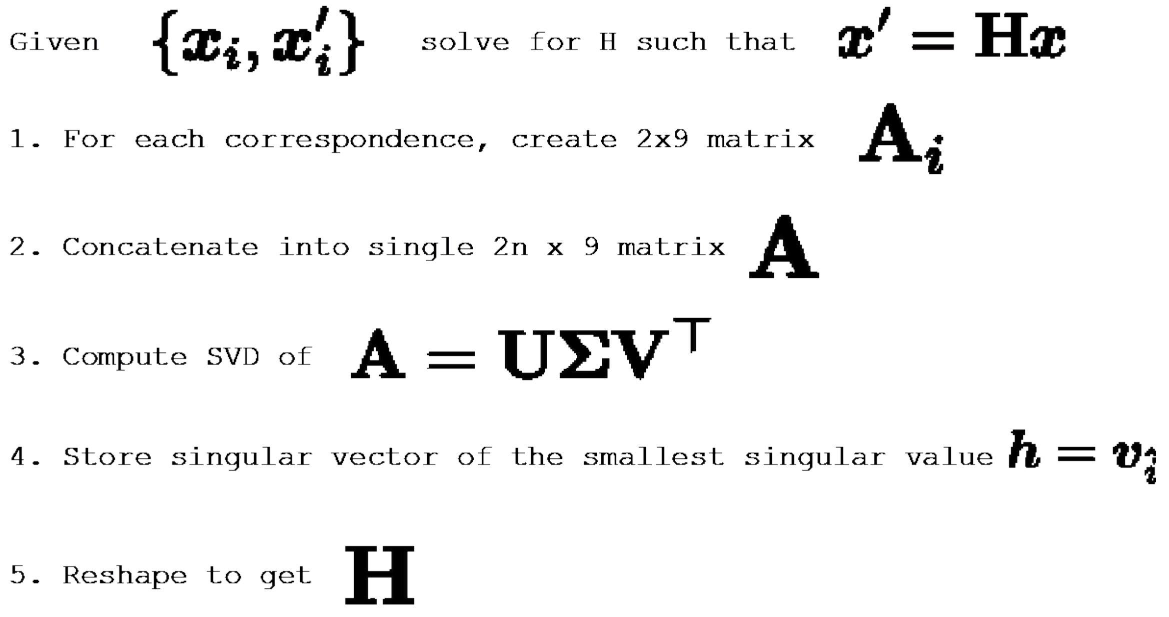
- Homography has 9 parameters
- But can’t determine scale factor, so only 8: 4 points!

$$Ah = 0 \text{ s.t } \|\mathbf{h}\| = 1$$

- Or because we will have noise:

$$\min_{\mathbf{h}} \|A\mathbf{h}\|^2 \text{ s.t } \|\mathbf{h}\| = 1$$

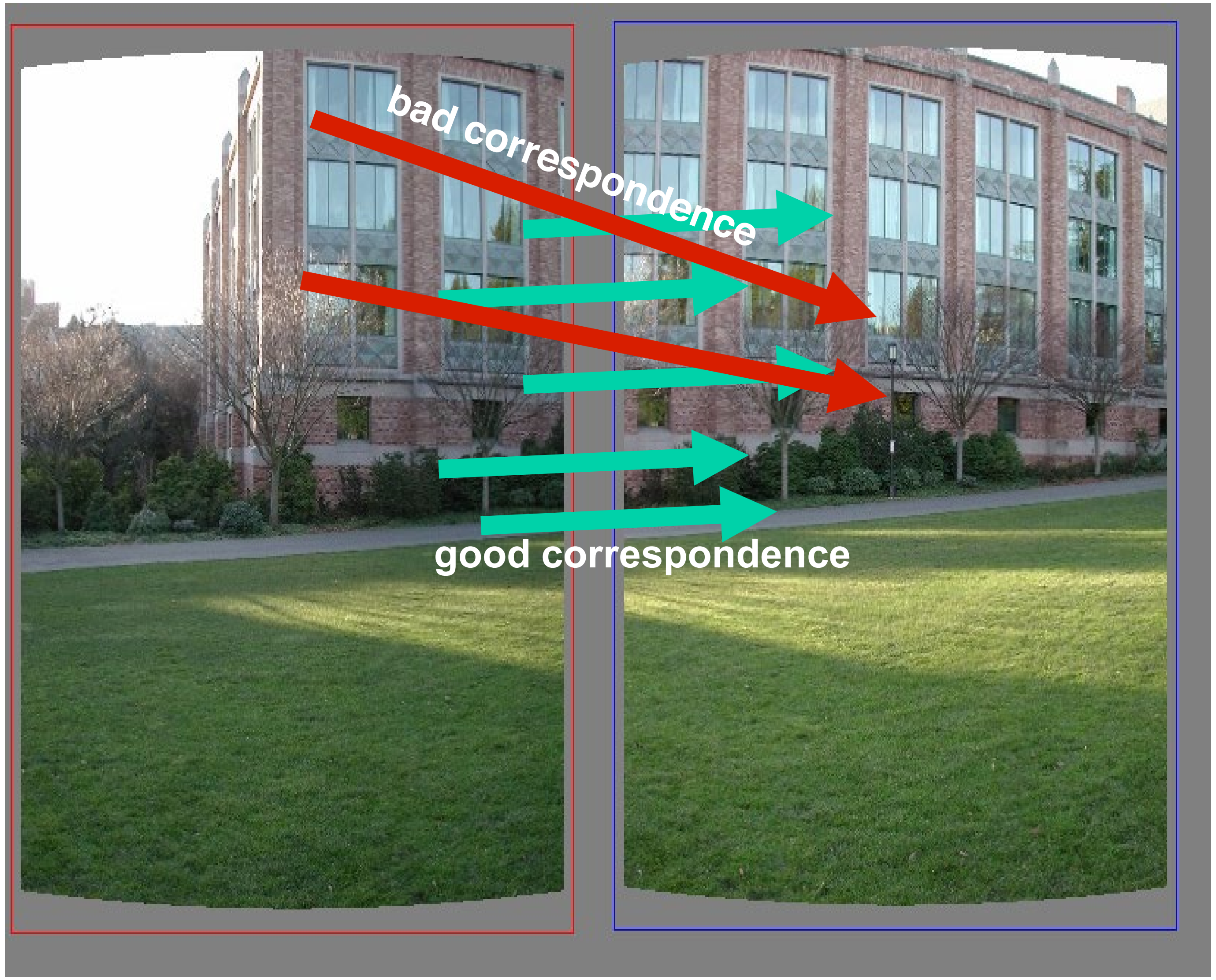
Solving for H using DLT



Given $\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$ solve for H such that $\boldsymbol{x}' = \mathbf{H}\boldsymbol{x}$

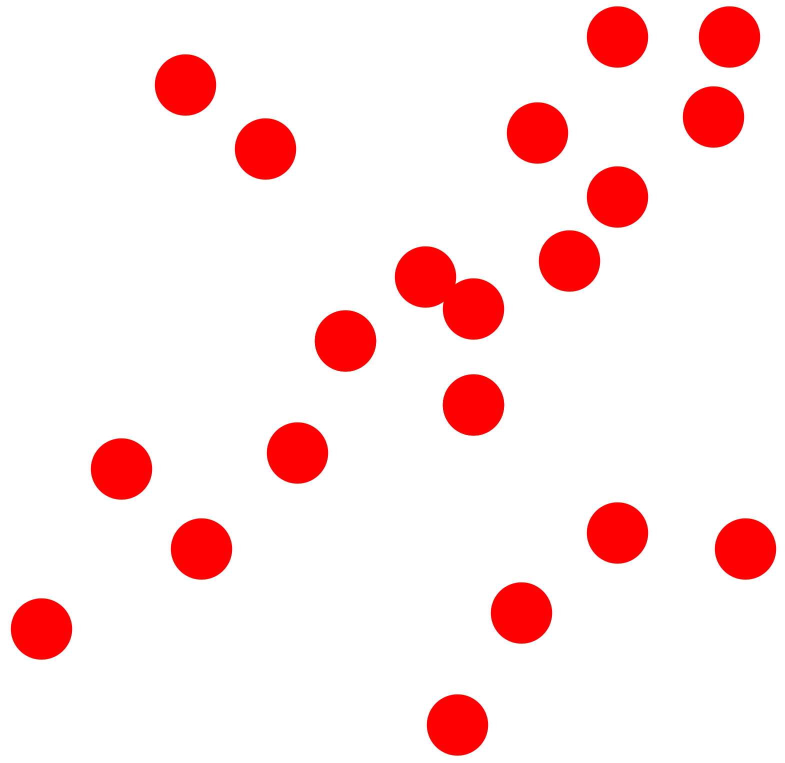
1. For each correspondence, create 2x9 matrix $\mathbf{A}_i$
2. Concatenate into single 2n x 9 matrix $\mathbf{A}$
3. Compute SVD of $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$
4. Store singular vector of the smallest singular value $\boldsymbol{h} = \boldsymbol{v}_{\hat{i}}$
5. Reshape to get $\mathbf{H}$

How to deal with outliers?



Random Sample Consensus (RANSAC)

Fitting lines
(with outliers)

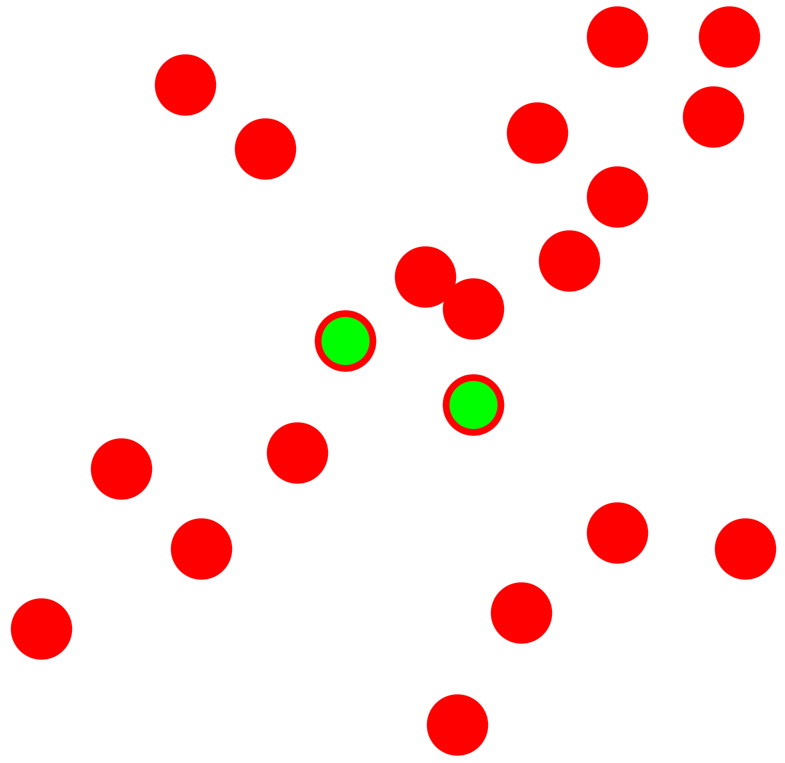


Algorithm:

1. Sample (randomly) the number of points required to fit the model
2. Solve for model parameters using samples
3. Score by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Fitting lines
(with outliers)

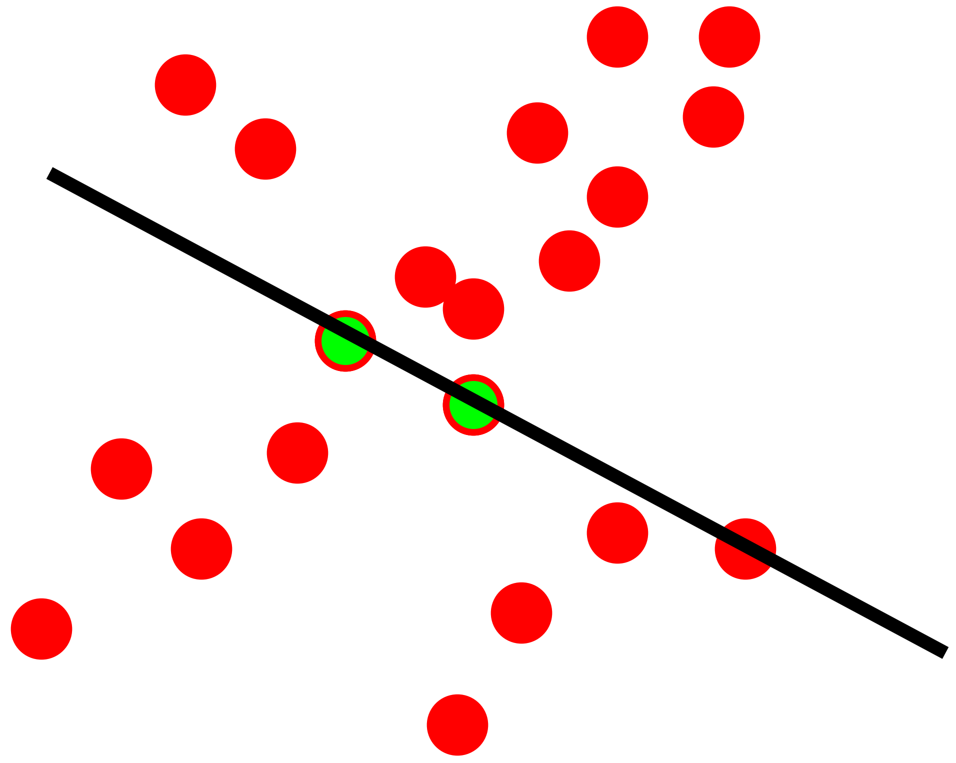


Algorithm:

1. **Sample (randomly) the number of points required to fit the model**
2. Solve for model parameters using samples
3. Score by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Fitting lines
(with outliers)



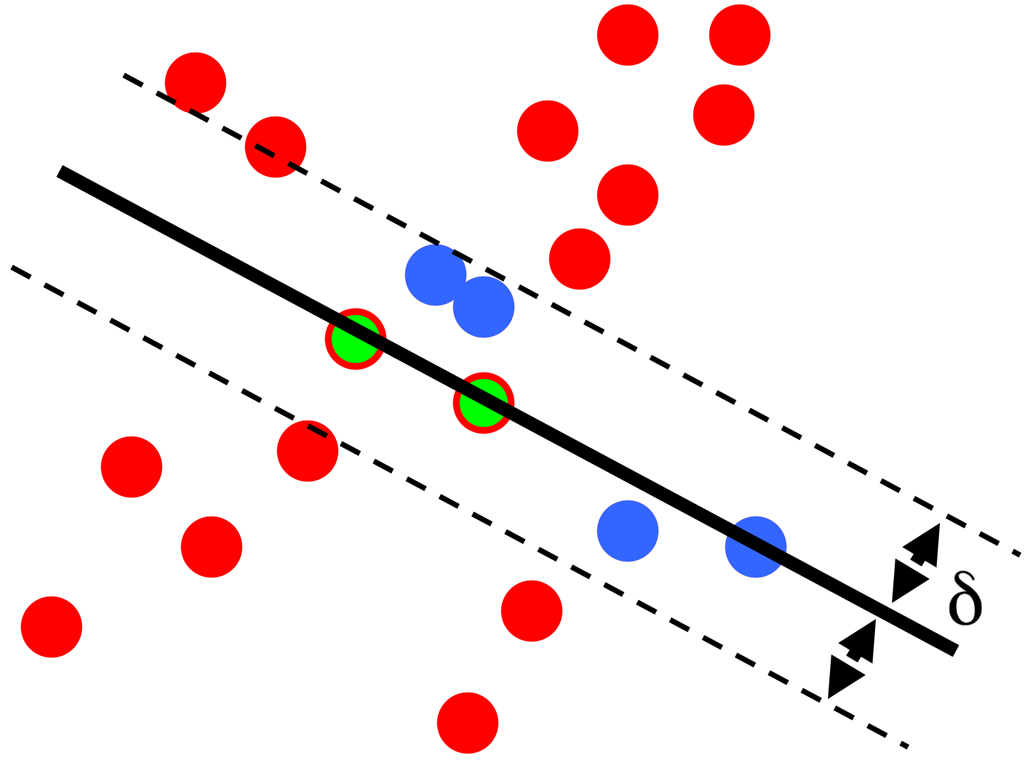
Algorithm:

1. Sample (randomly) the number of points required to fit the model
2. **Solve for model parameters using samples**
3. Score by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Fitting lines
(with outliers)

$N_I = 6$

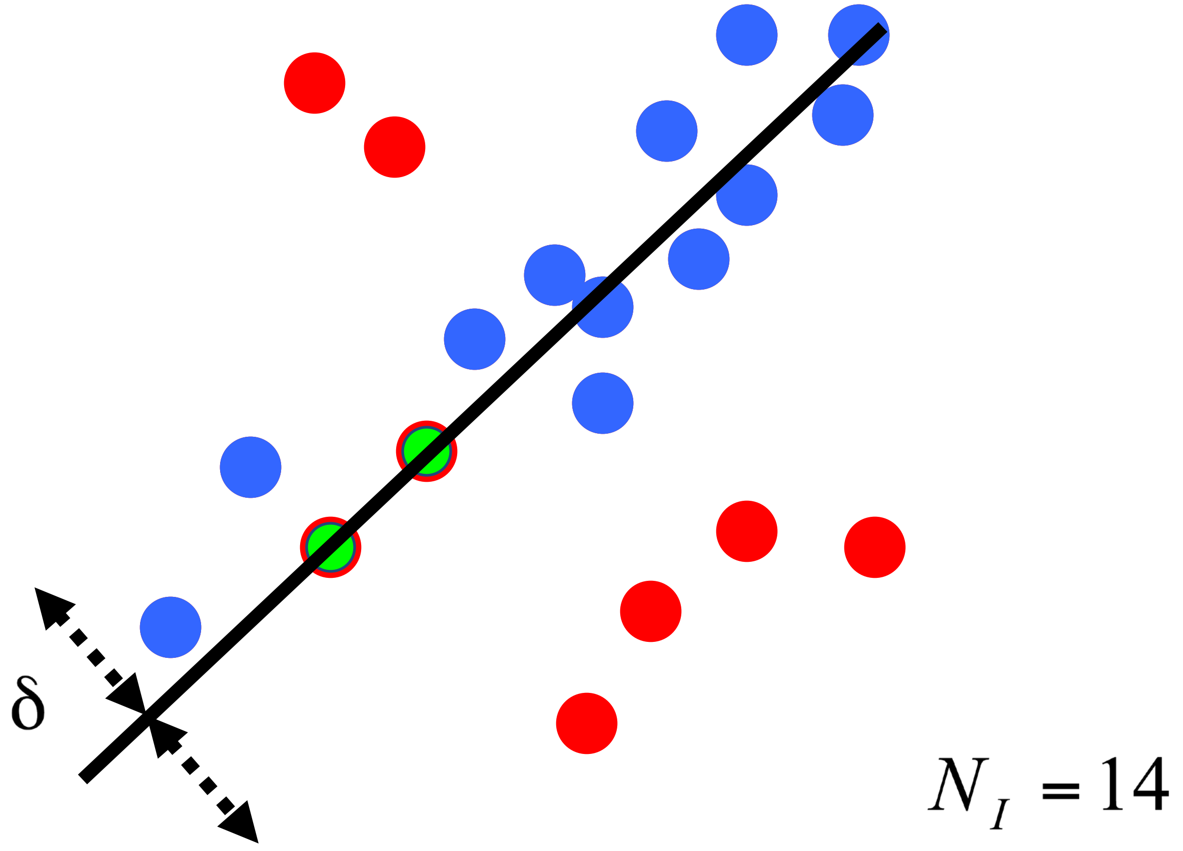


Algorithm:

1. Sample (randomly) the number of points required to fit the model
2. Solve for model parameters using samples
3. **Score by the fraction of inliers within a preset threshold of the model**

Repeat 1-3 until the best model is found with high confidence

Fitting lines
(with outliers)



Algorithm:

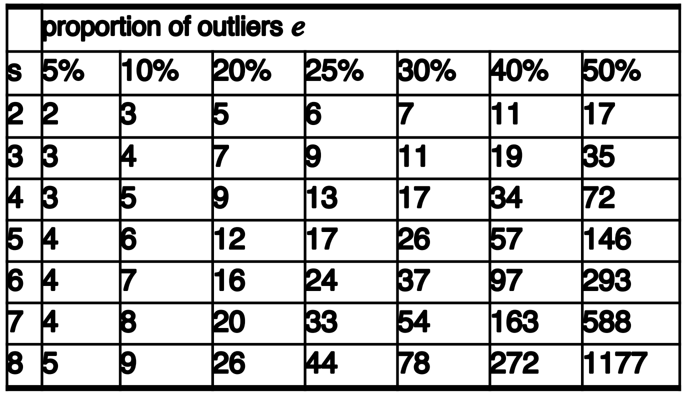
1. Sample (randomly) the number of points required to fit the model
2. Solve for model parameters using samples
3. Score by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

How to choose parameters?

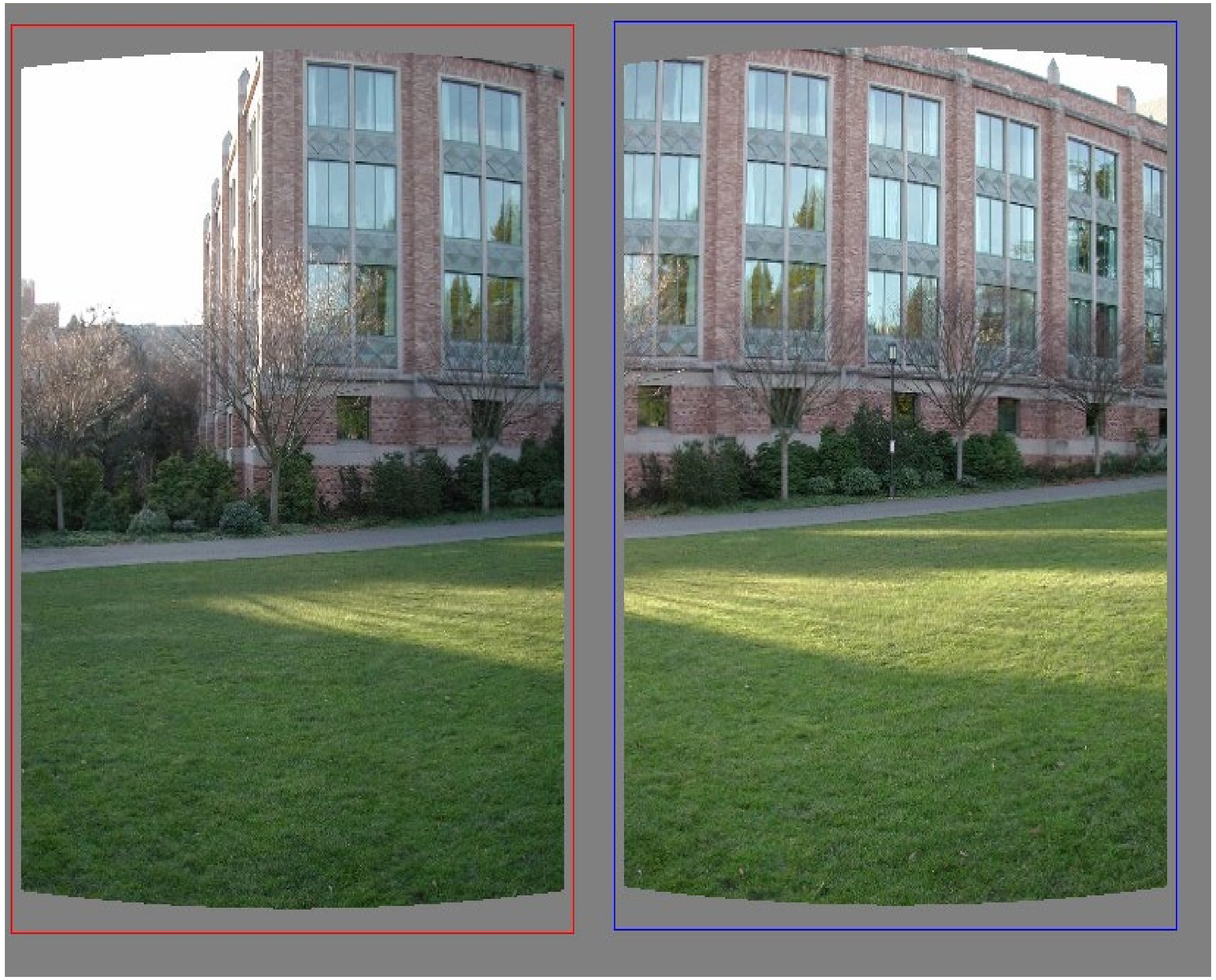
- Number of samples N
  - Choose N so that, with probability p, at least one random sample is free from outliers (e.g. p=0.99) (outlier ratio: e)
- Number of sampled points s
  - Minimum number needed to fit the model
- Distance threshold δ
  - Choose δ so that a good point with noise is likely (e.g., prob=0.95) within threshold
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$

$$N = \frac{\log(1-p)}{\log\left(1-(1-e)^s\right)}$$



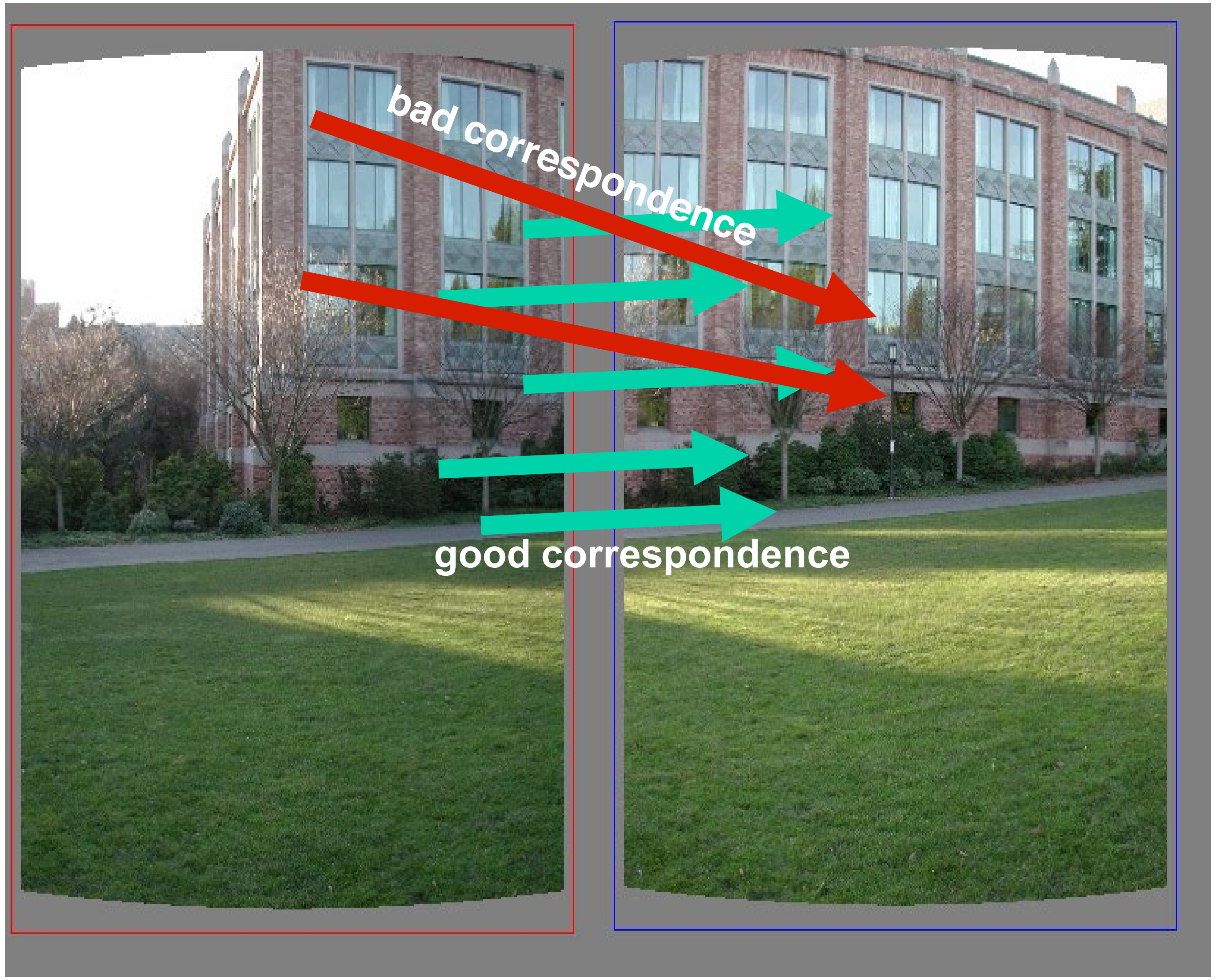
	proportion of outliers *e*						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Given two images…



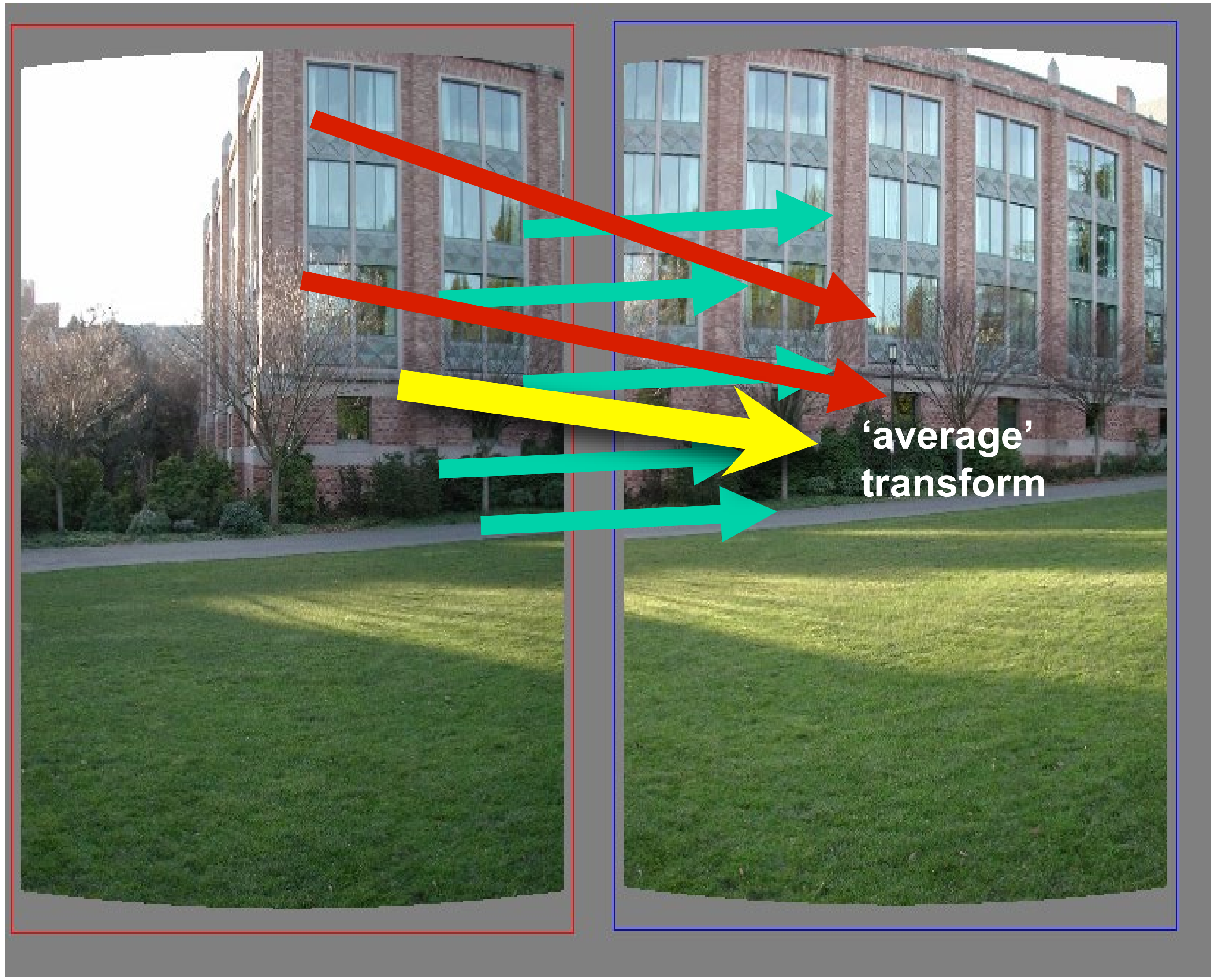
find matching features (e.g., SIFT) and a translation transform

Matched points will usually contain bad correspondences



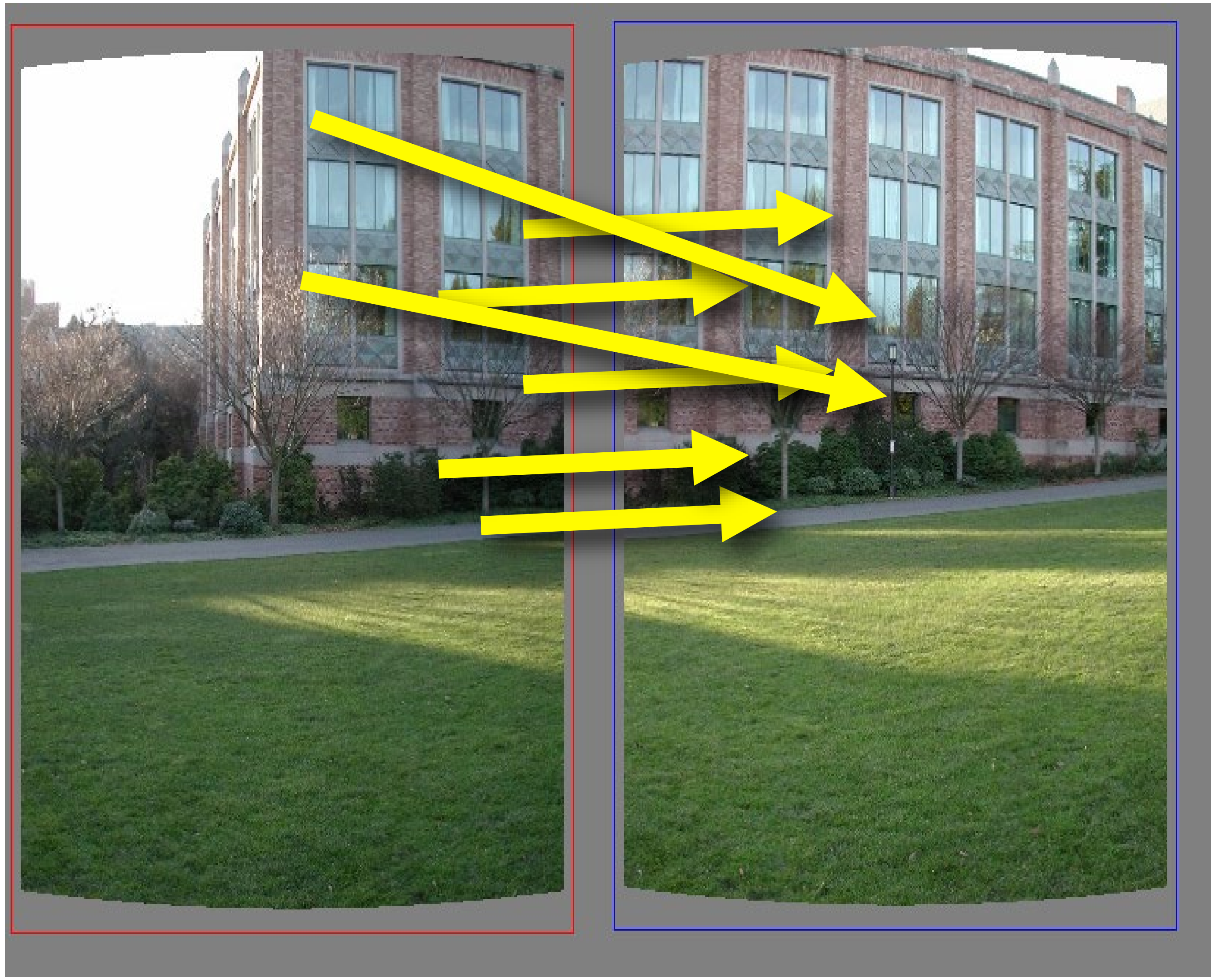
how should we estimate the transform?

LLS will find the 'average' transform

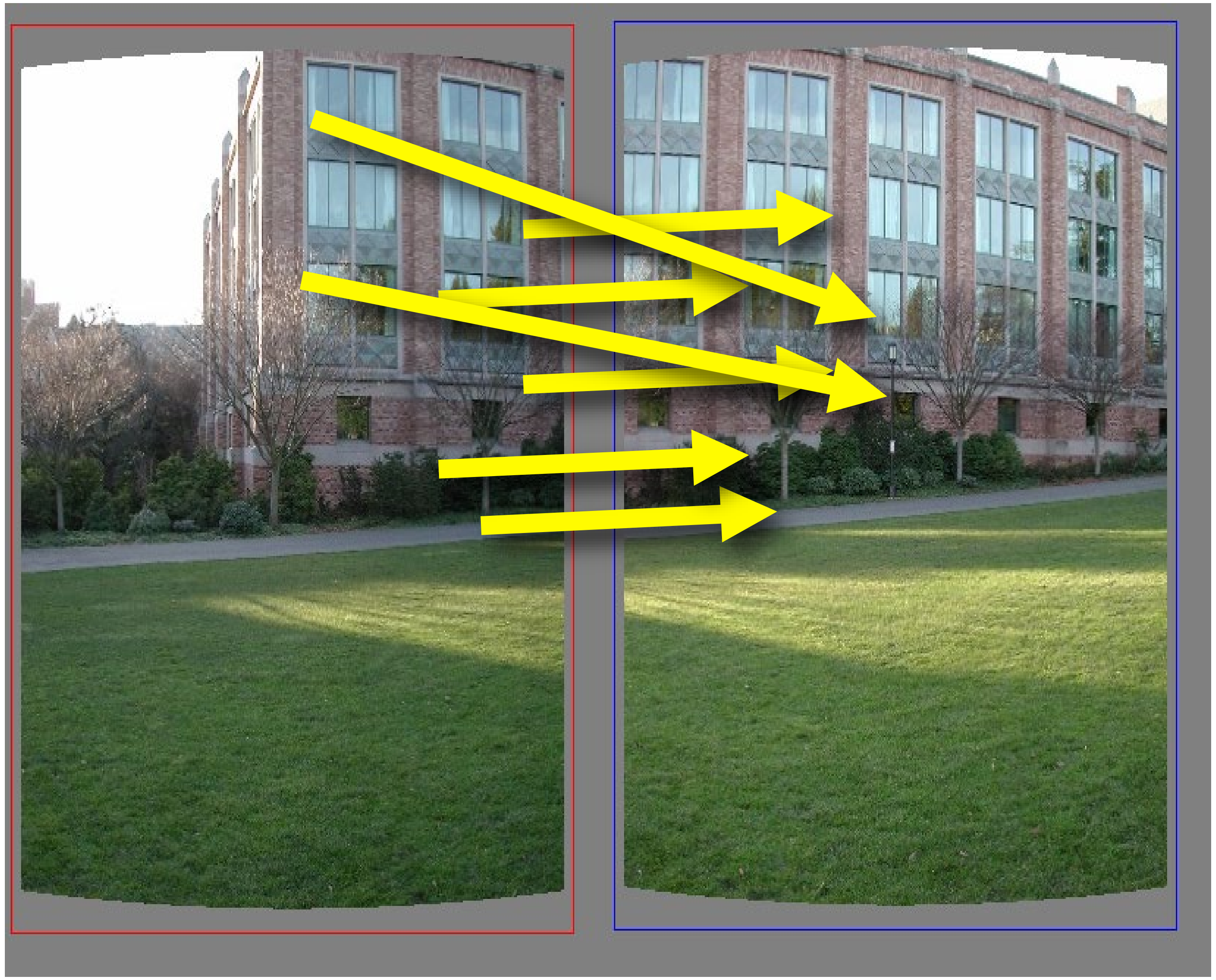


solution is corrupted by bad correspondences

Use RANSAC

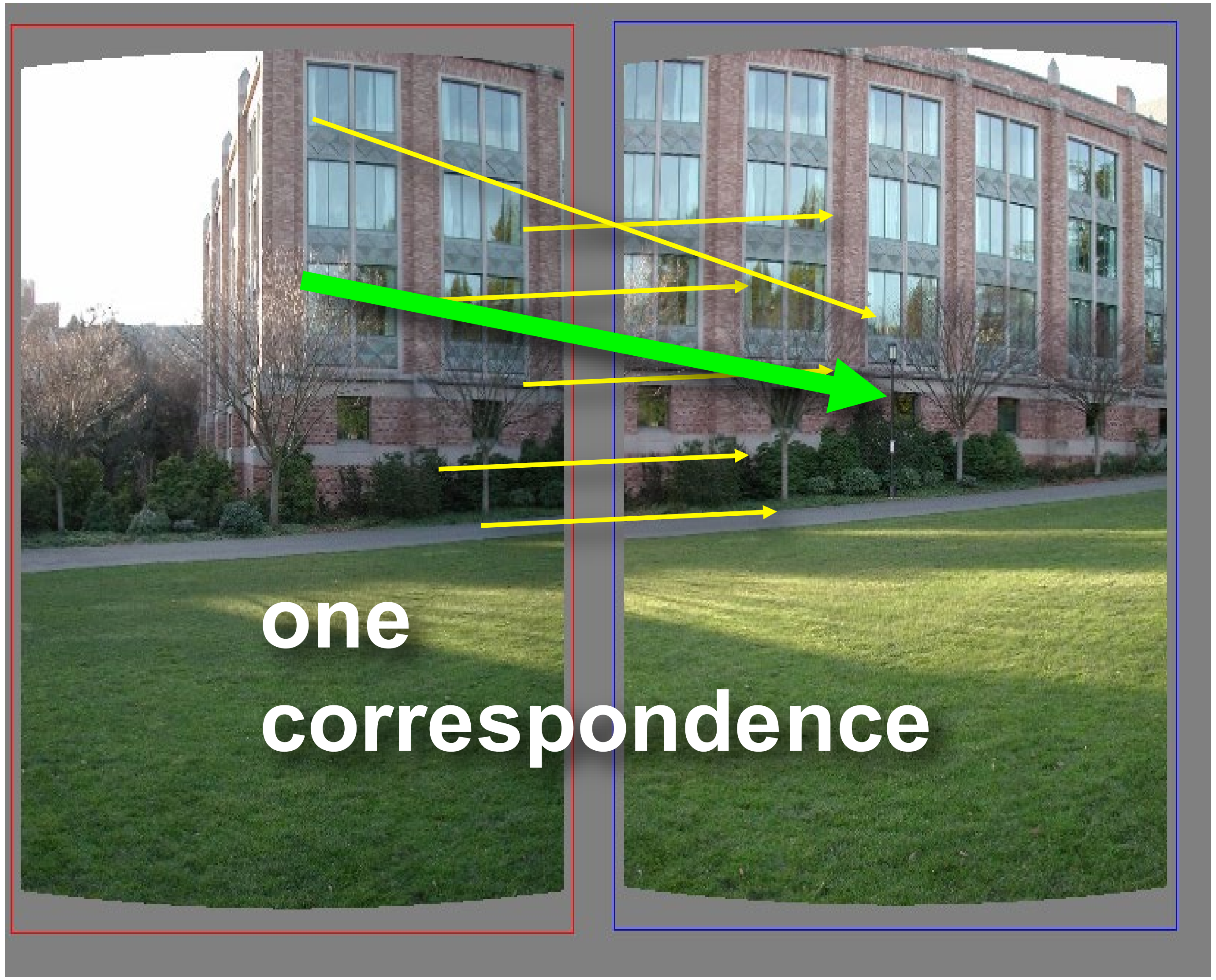


How many correspondences to compute translation transform?

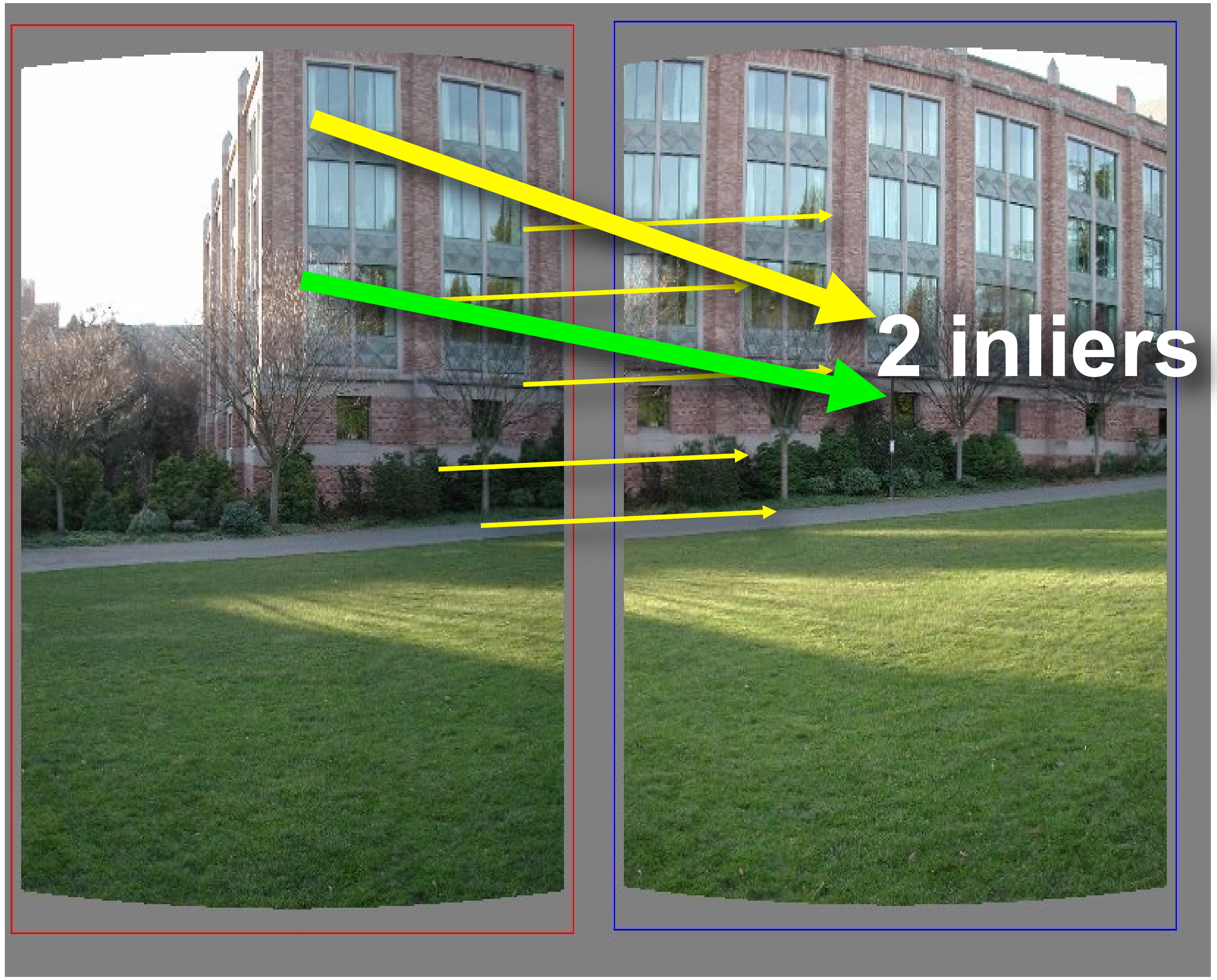


Need only **one correspondence**, to find translation model

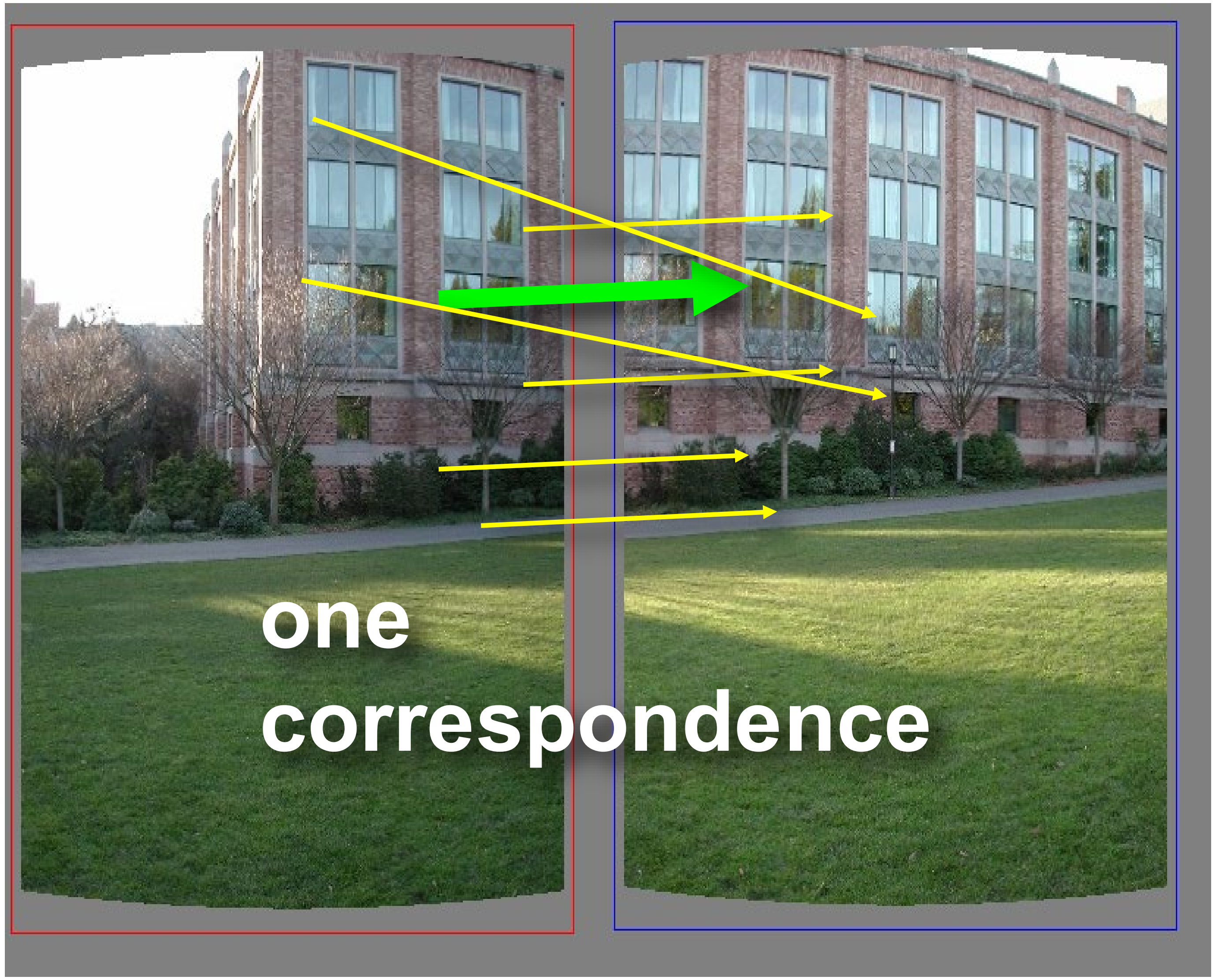
Pick one correspondence, count inliers



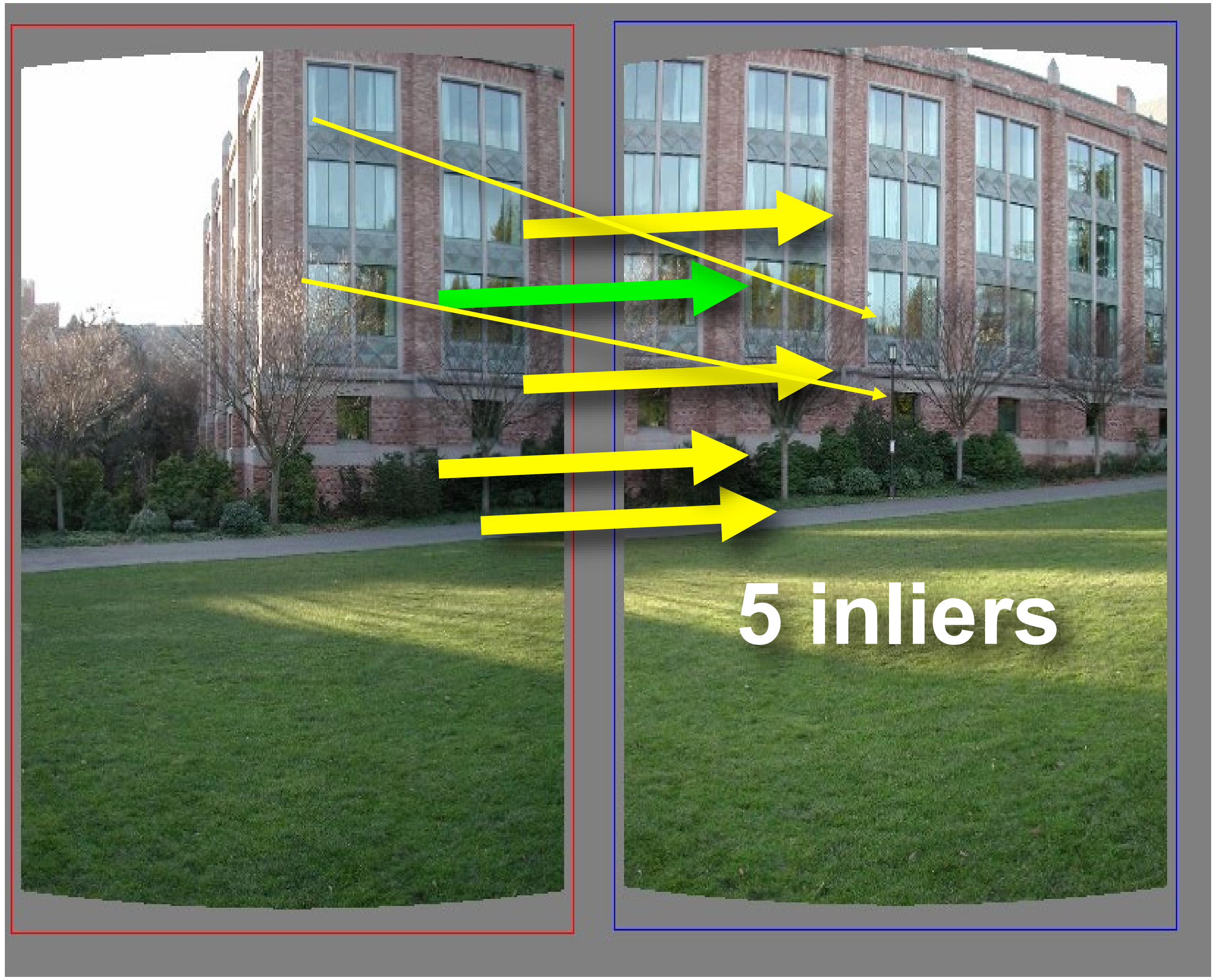
Pick one correspondence, count inliers



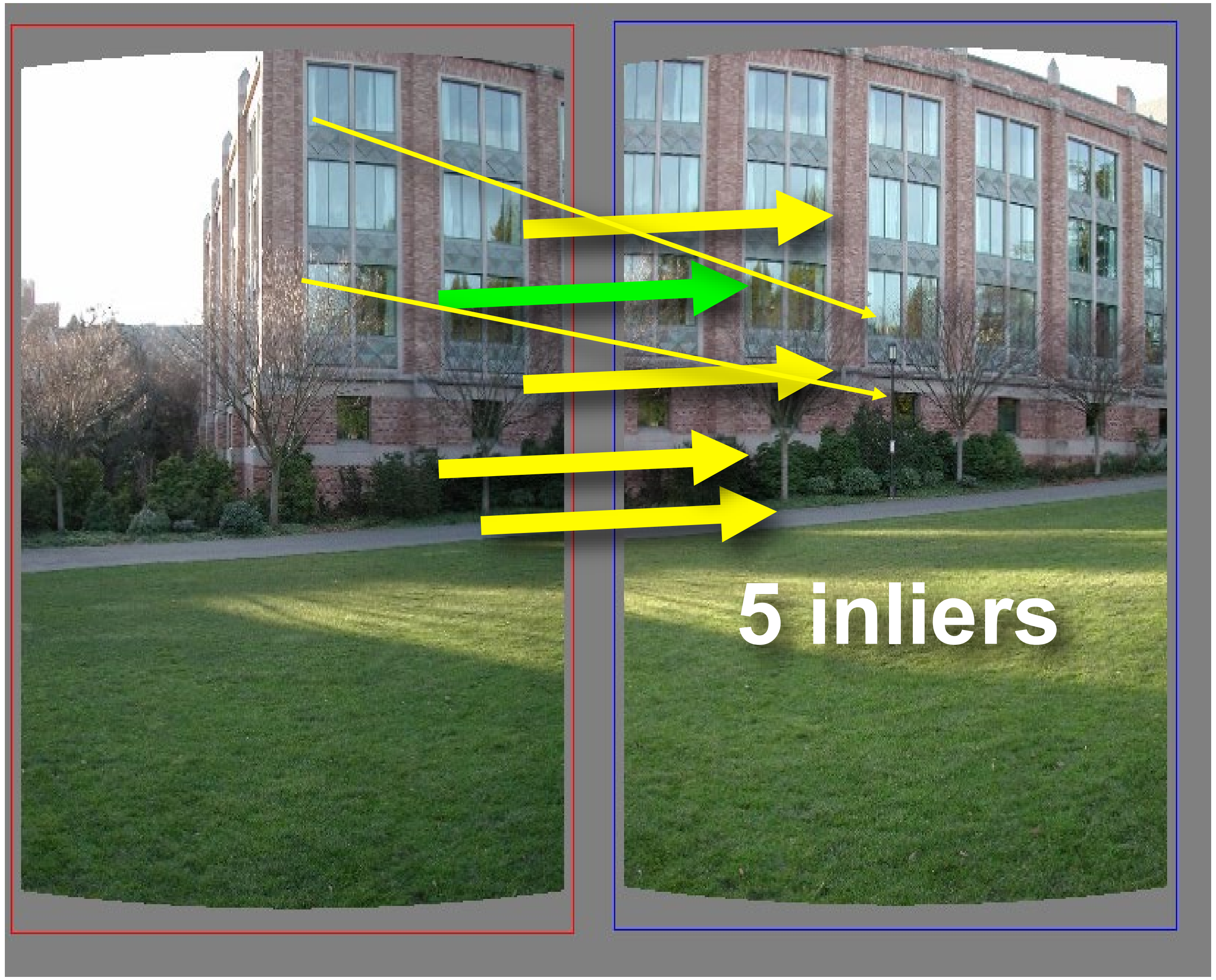
Pick one correspondence, count inliers



Pick one correspondence, count inliers



Pick one correspondence, count inliers



Pick the model with the highest number of inliers!

Estimating homography using RANSAC

- RANSAC loop
    1. Get four point correspondences (randomly)
    2. Compute H using DLT
    3. Count inliers
    4. Keep H if largest number of inliers
- Recompute H using all inliers

RANSAC

- An example of a “voting”-based fitting scheme
- Each hypothesis gets voted on by each data point, best hypothesis wins

- There are many other types of voting schemes
  - E.g., Hough transforms…