Lecture 10:
2D Transformation & Alignment

COMP 590/776: Computer Vision

Instructor: Soumyadip (Roni) Sengupta

TA: Mykhailo (Misha) Shvets

Course Website:
Scan Me!

Jia-Bin Huang
@jbhuang0604

The ControlNet illusion art is FUN!

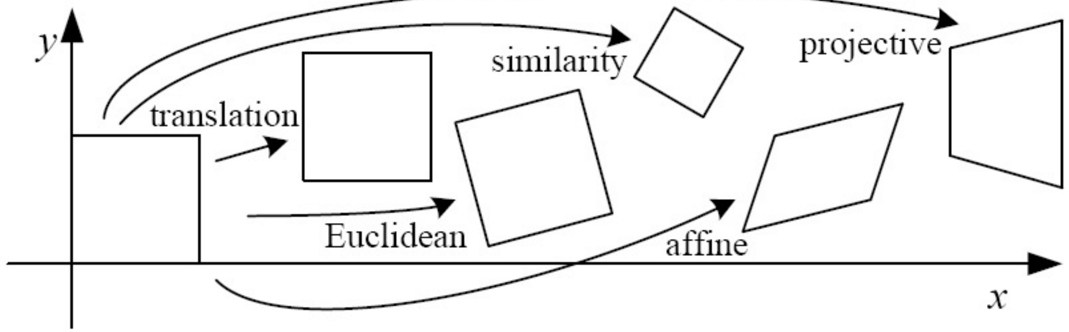
In some sense, it's a *Hybrid Image* (17-year-old method). It was my first homework in the computer vision class. I remember that it takes time to properly align the two images to get good results.

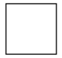
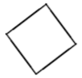
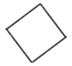
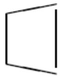
Now it's only one sentence and one click!

1:46 PM · Sep 27, 2023 · **821** Views

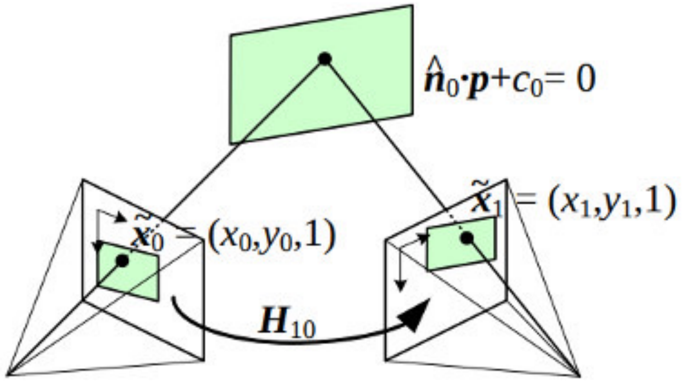
Recap

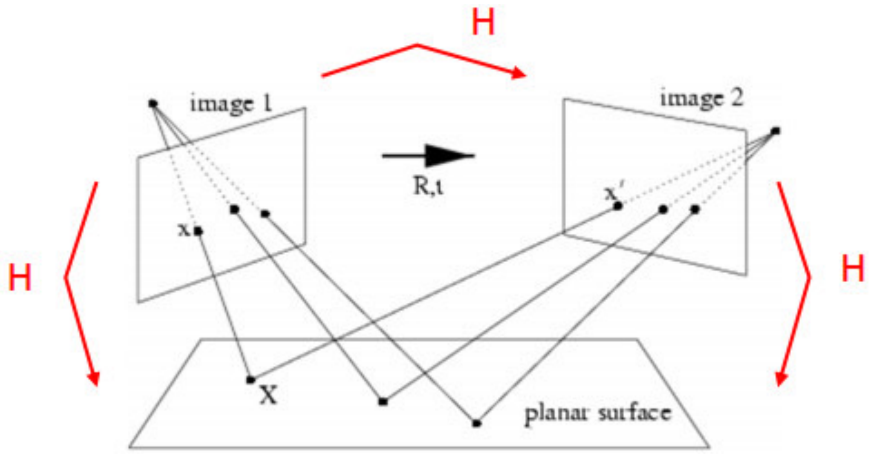
2D image transformations



Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\left[\begin{array}{c\|c} \boldsymbol{I} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	2	orientation + $\cdots$	
rigid (Euclidean)	$\left[\begin{array}{c\|c} \boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	3	lengths + $\cdots$	
similarity	$\left[\begin{array}{c\|c} s\boldsymbol{R} & \boldsymbol{t} \end{array}\right]_{2\times 3}$	4	angles + $\cdots$	
affine	$\left[\begin{array}{c} \boldsymbol{A} \end{array}\right]_{2\times 3}$	6	parallelism + $\cdots$	
projective	$\left[\begin{array}{c} \tilde{\boldsymbol{H}} \end{array}\right]_{3\times 3}$	8	straight lines	

Projective Transformations *aka* Homographies *aka* Planar Perspective Maps





$$\mathbf{H} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}$$

Called a *homography*
(or *planar perspective map*)

Any two images of the same planar surface in 3D space are related by a **homography** (assuming a pinhole camera model).

Affine transformations

- Affine transformations are combinations of …
  - 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 remain parallel
  - Ratios are preserved
  - Closed under composition

Homographies (Projective Transformation)

- Homographies …
  - Affine transformations, and
  - Projective warps

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

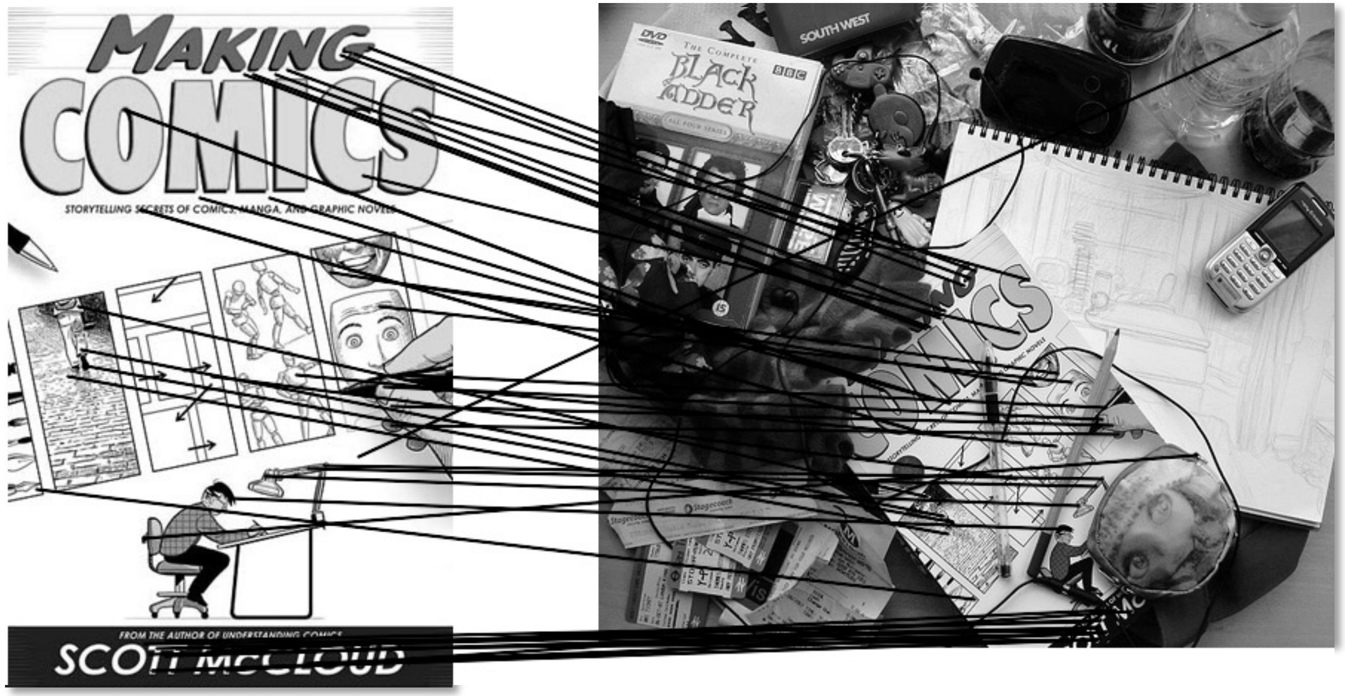
$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

where the length of the vector [h_{00} h_{01} … h_{22}] is 1.

- Properties of projective transformations:
  - Origin does not necessarily map to origin
  - Lines map to lines
  - Parallel lines do not necessarily remain parallel
  - Ratios are not preserved
  - Closed under composition

Computing transformations

- Given a set of matches between images A and B
  - How can we compute the transform T from A to B?



  - Find transform T that best “agrees” with the matches

6 unknowns, atleast 3 point matches required

Affine transformations

- Matrix form

$$
\begin{bmatrix}
x_1 & y_1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & x_1 & y_1 & 1 \\
x_2 & y_2 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & x_2 & y_2 & 1 \\
 & & & \vdots & & \\
x_n & y_n & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & x_n & y_n & 1
\end{bmatrix}
\begin{bmatrix}
a \\ b \\ c \\ d \\ e \\ f
\end{bmatrix}
=
\begin{bmatrix}
x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ \vdots \\ x'_n \\ y'_n
\end{bmatrix}
$$

$$\mathbf{A} \quad \mathbf{t} = \mathbf{b}$$

$\mathbf{A}$: $2n$ x 6 ; $\mathbf{t}$: 6 x 1 ; $\mathbf{b}$: $2n$ x 1

Solving for homographies

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

Rank(A) = 8

$$\mathbf{A} \qquad \mathbf{h} \qquad \mathbf{0}$$

2n × 9 **9** **2n**

- Smallest singular value of A also indicates how well the homography can be estimated.
- Optimal h = Singular vector corresponding to smallest singular value

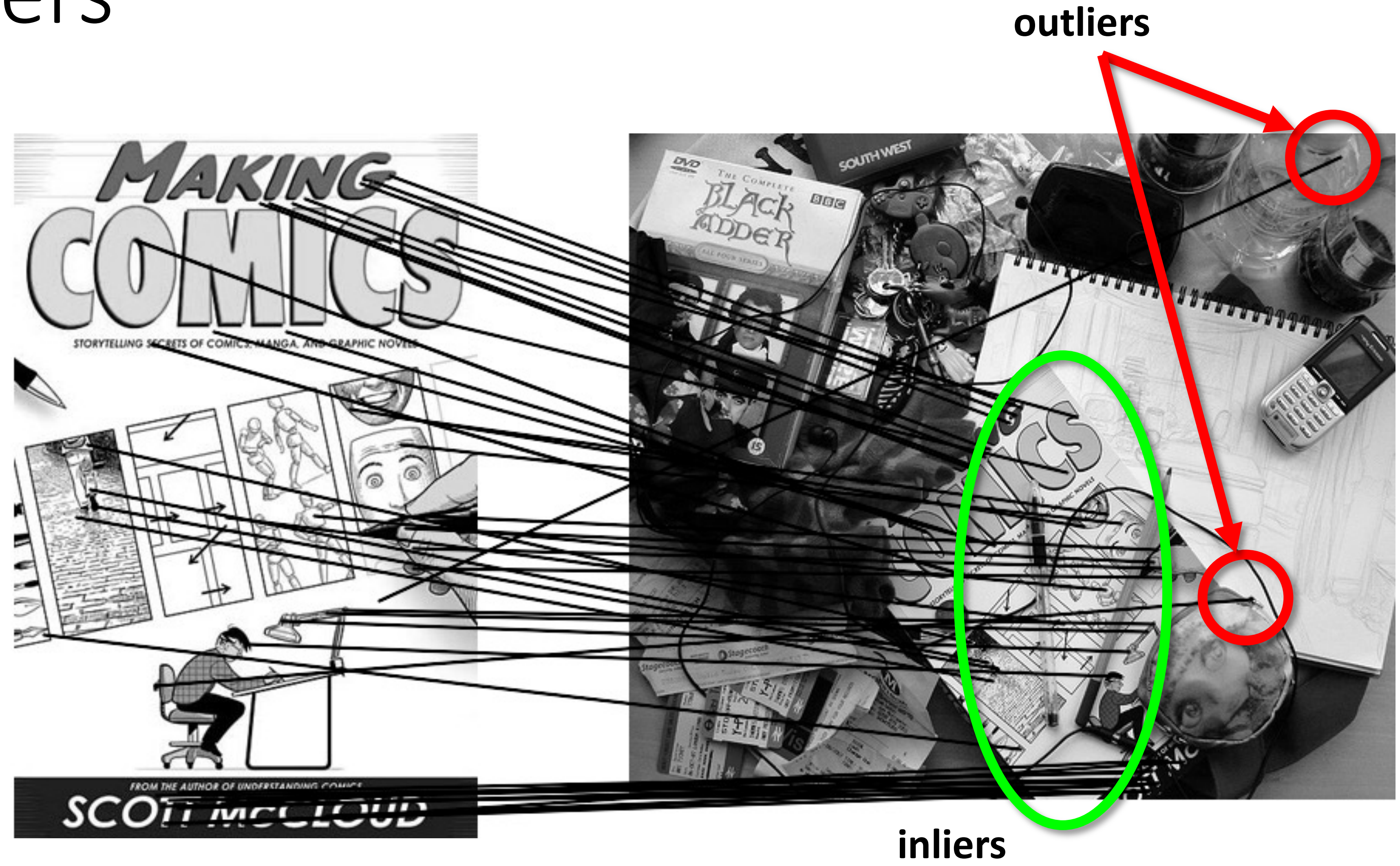
Today’s class

- Fitting with outliers – RANSAC
- Warping
- Blending
- HW3 Motivation

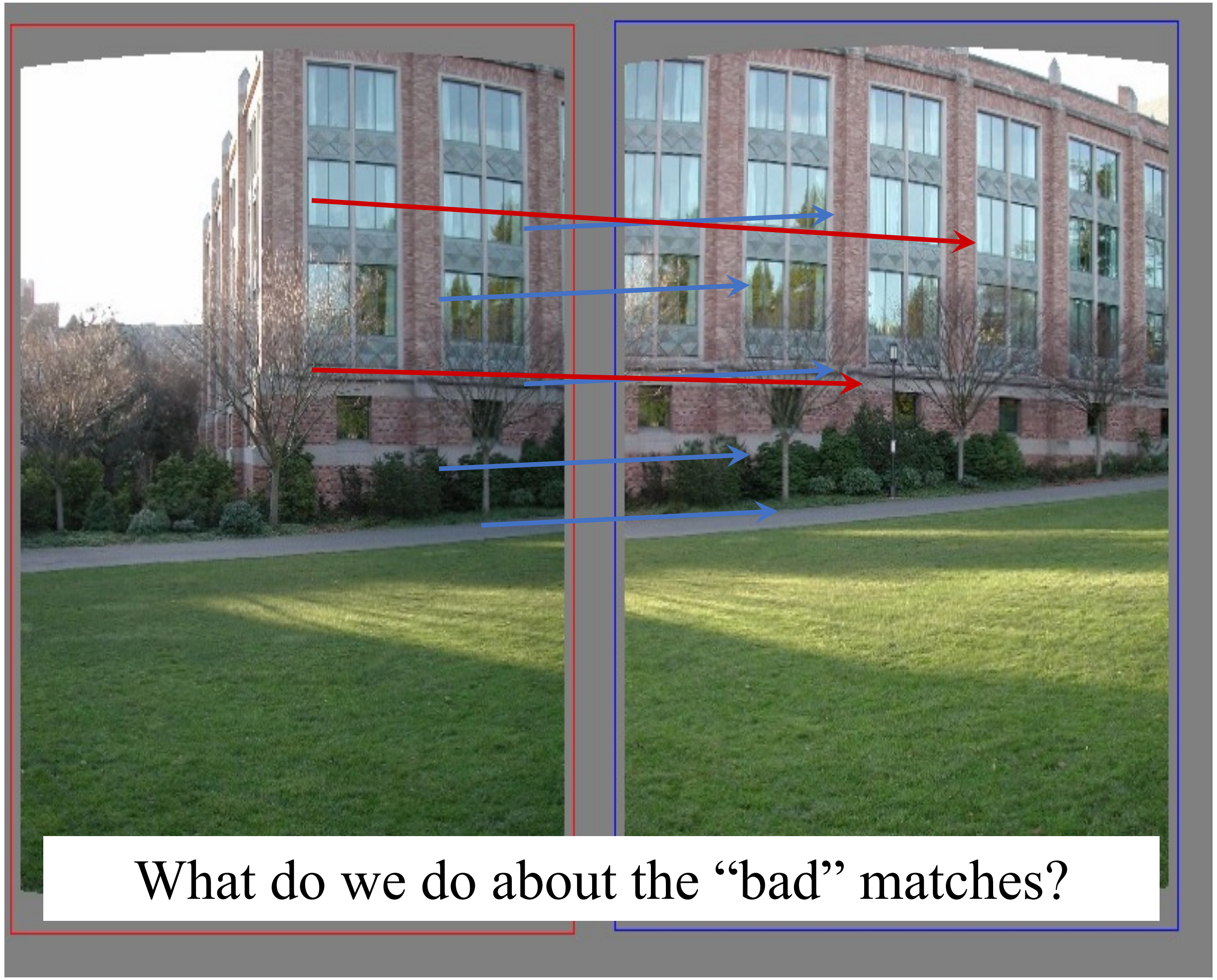
Today’s class

- Fitting with outliers – RANSAC
- Warping
- Blending
- HW3 Motivation

Outliers



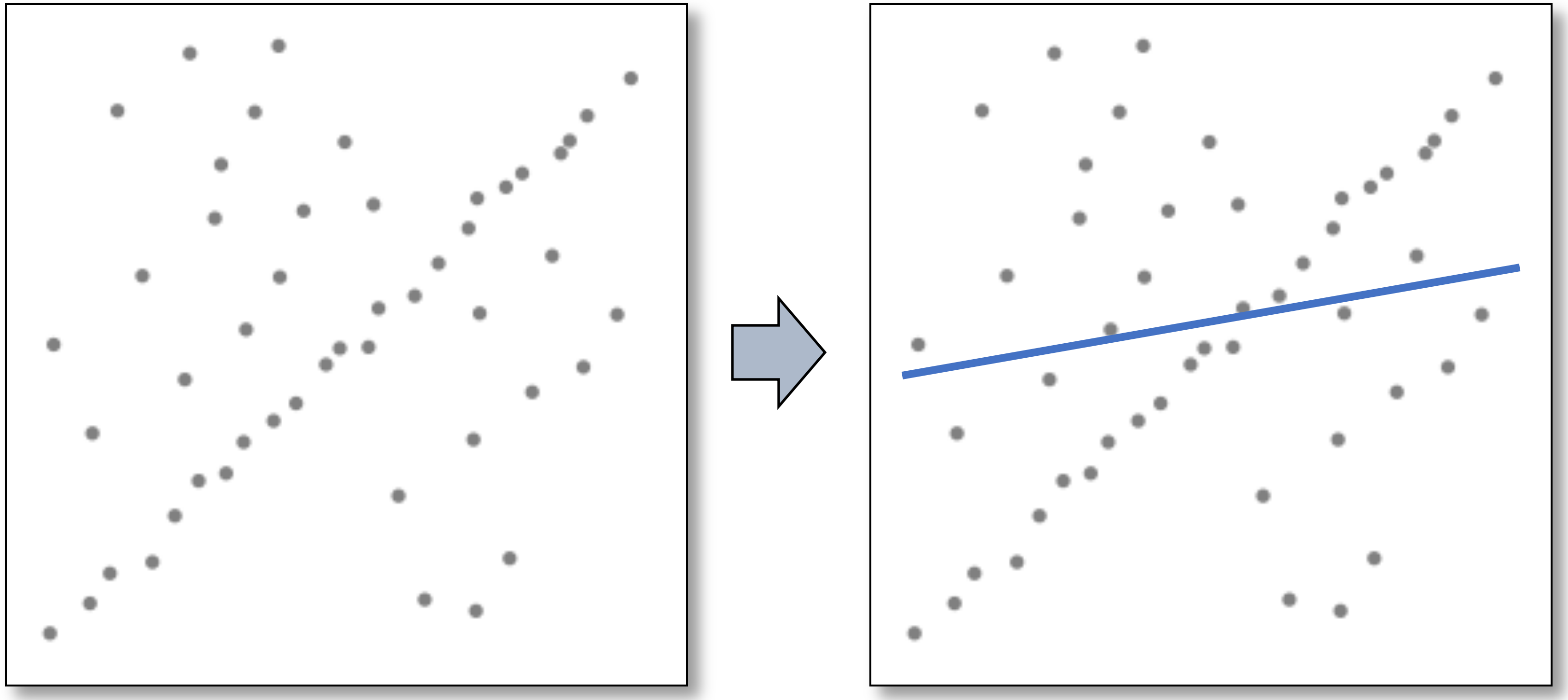
Matching features



What do we do about the “bad” matches?

Robustness

- Let's consider the problem of linear regression



Problem: Fit a line to these datapoints　　　　Least squares fit

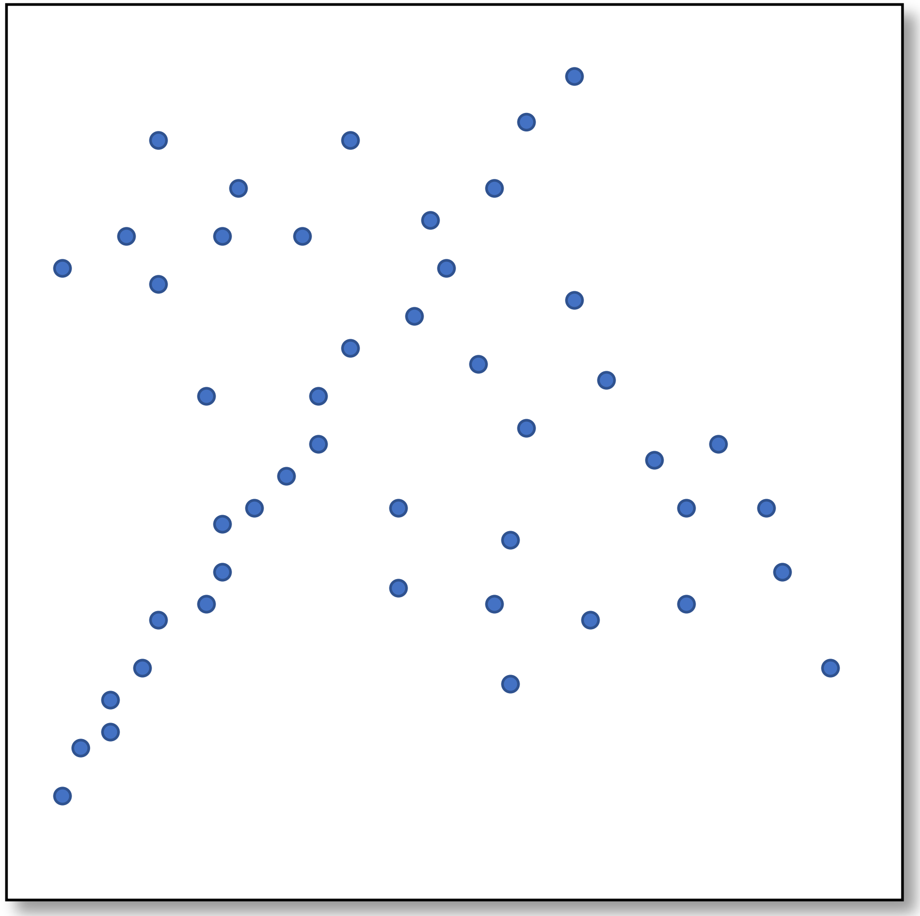
- How can we fix this?

Idea

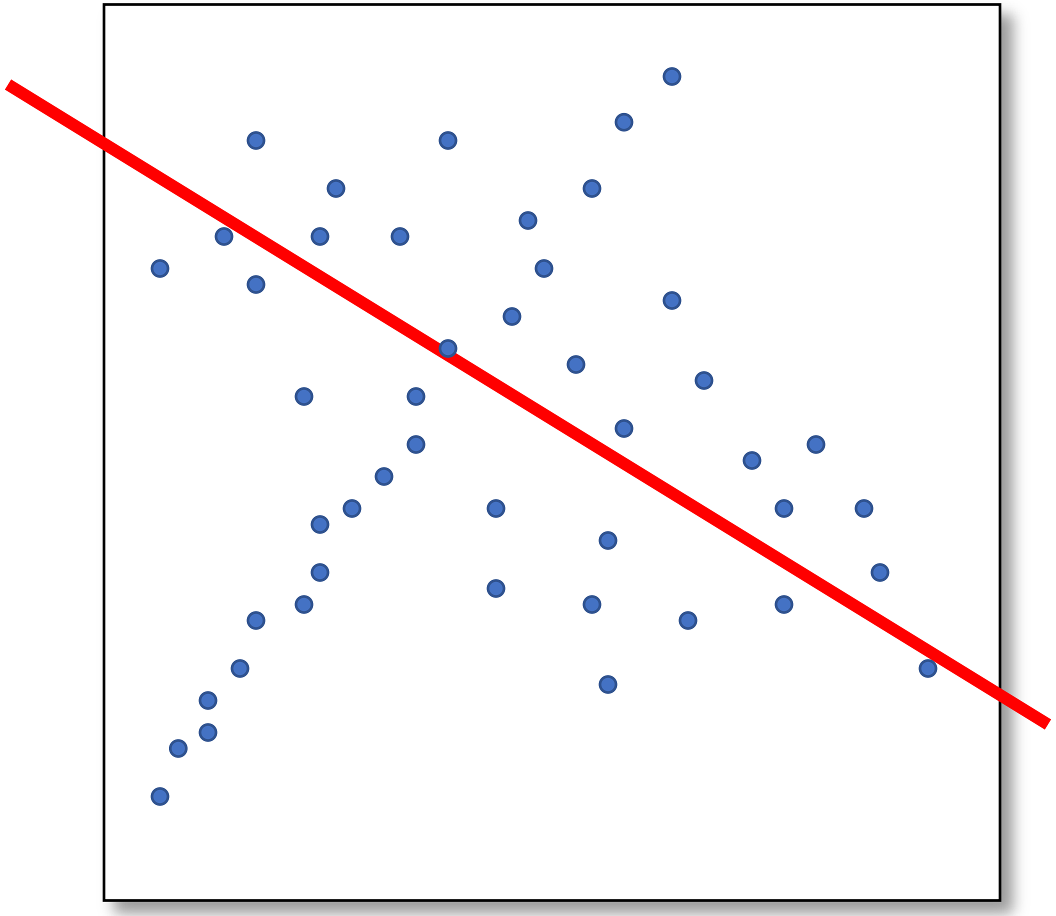
- Given a hypothesized line
- Count the number of points that “agree” with the line
  - “Agree” = within a small distance of the line
  - I.e., the **inliers** to that line

- For all possible lines, select the one with the largest number of inliers

Counting inliers

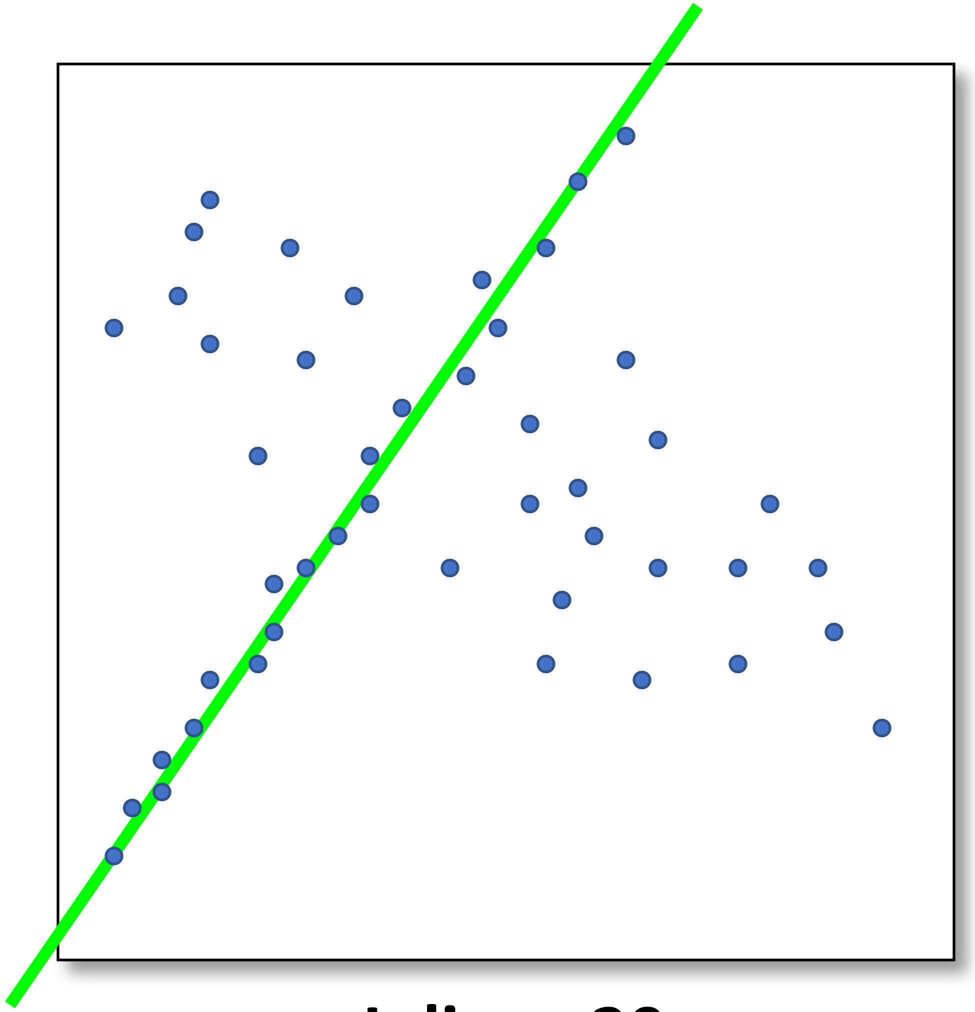


Counting inliers



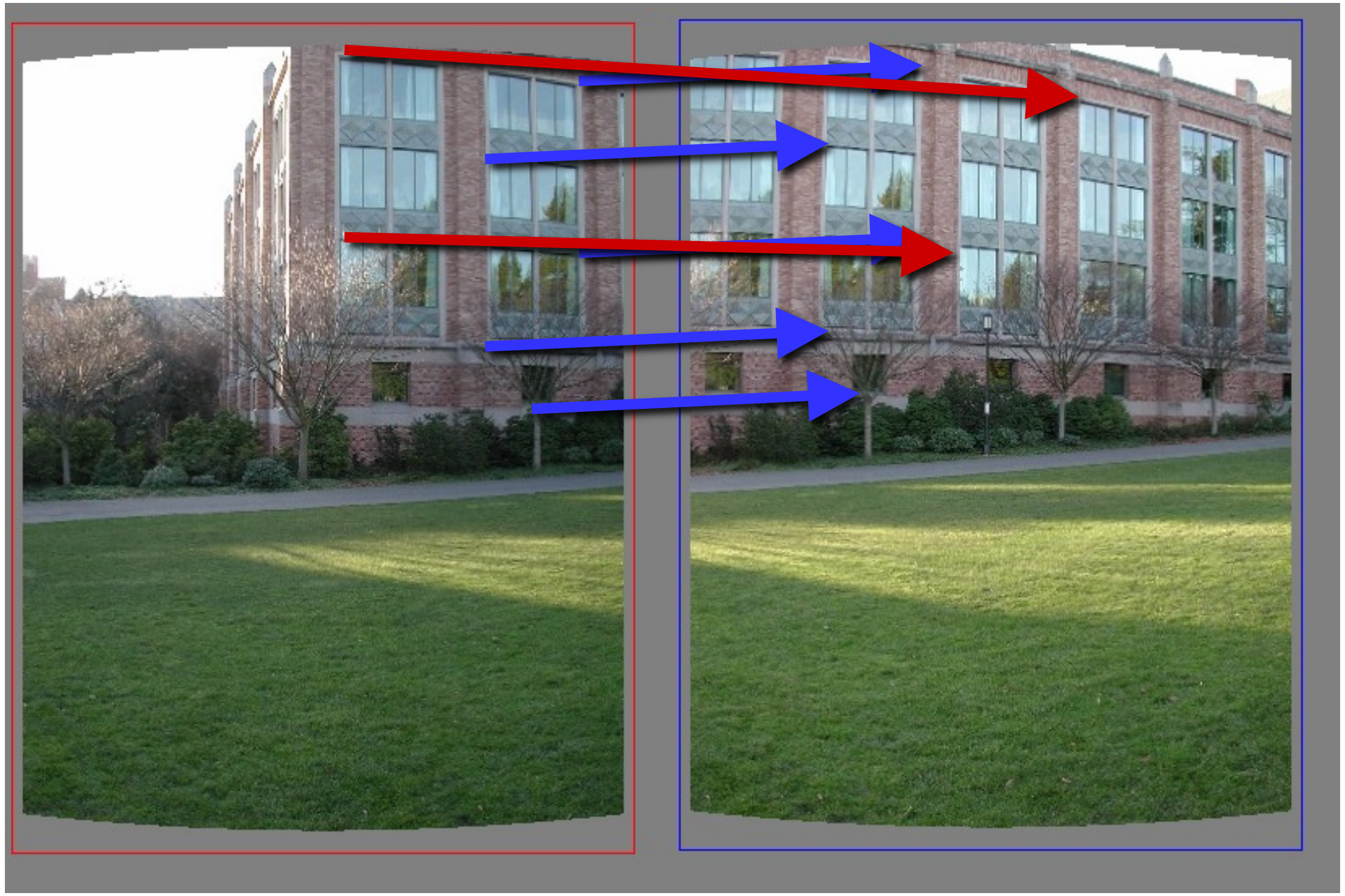
Inliers: 3

Counting inliers

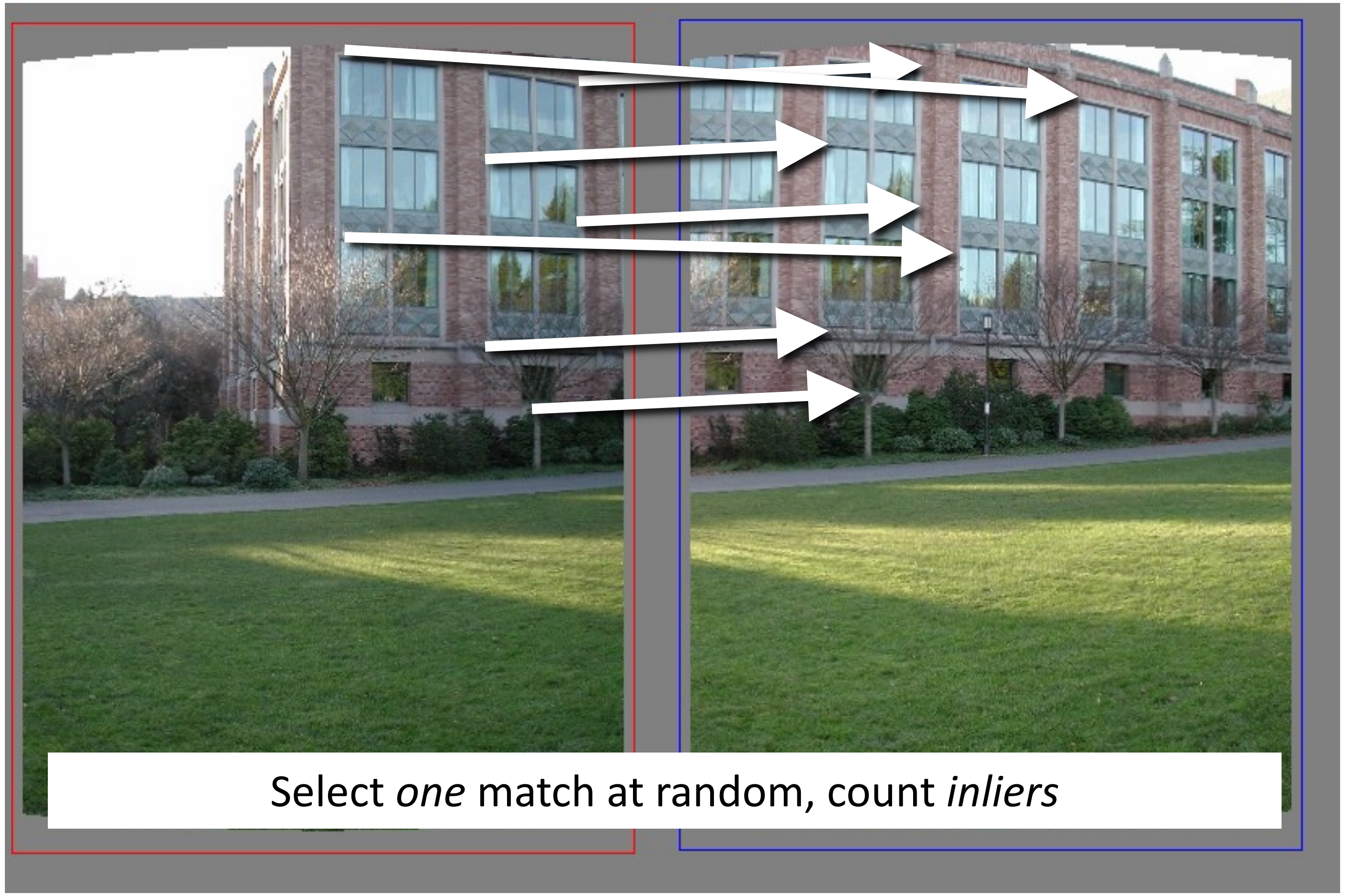


Inliers: 20

Translations

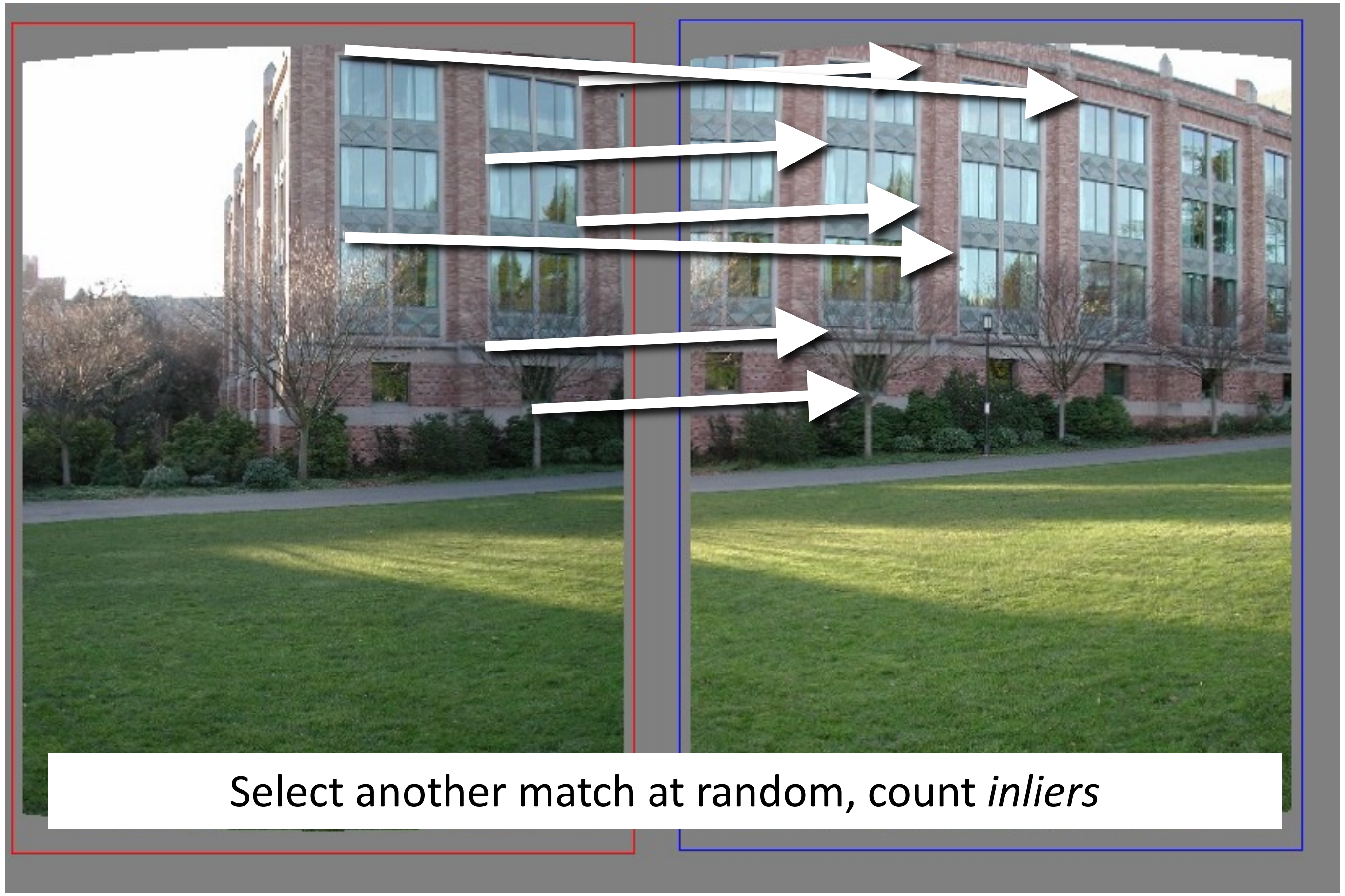


RAndom SAmple Consensus



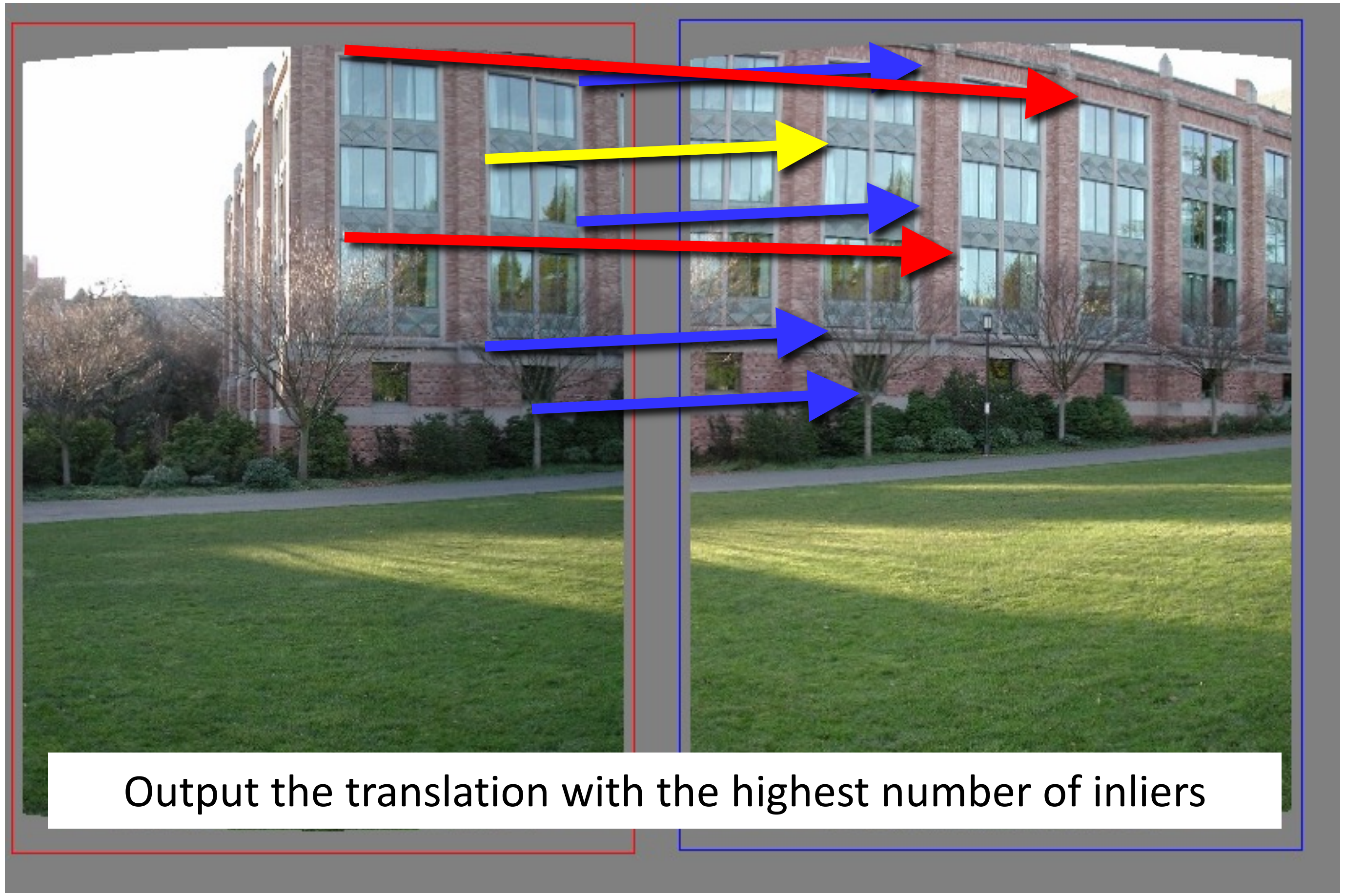
Select *one* match at random, count *inliers*

RAndom SAmple Consensus



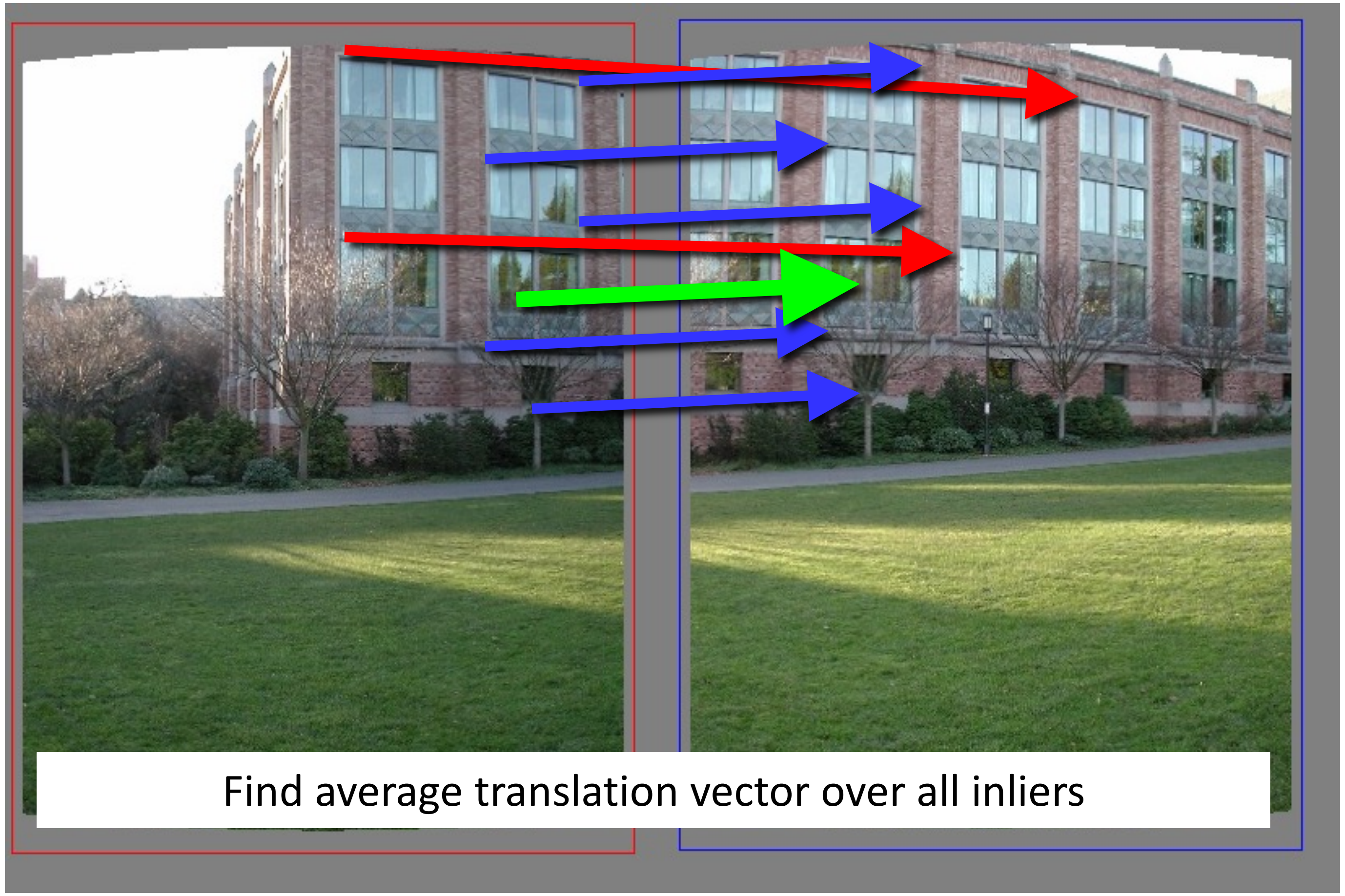
Select another match at random, count *inliers*

RAndom SAmple Consensus



Output the translation with the highest number of inliers

Final step: least squares fit



RANSAC

- Idea:
  - All the inliers will agree with each other on the translation vector; the (hopefully small) number of outliers will (hopefully) disagree with each other
    - RANSAC only has guarantees if there are < 50% outliers
  - "All good matches are alike; every bad match is bad in its own way."

    – Tolstoy via Alyosha Efros

RANSAC

- General version:
  1. Randomly choose *s* samples
     - Typically *s* = minimum sample size that lets you fit a model
  2. Fit a model (e.g., line) to those samples
  3. Count the number of inliers that approximately fit the model
  4. Repeat *N* times
  5. Choose the model that has the largest set of inliers

RANSAC for estimating homography

- RANSAC loop:

1. Select four feature pairs (at random)
2. Compute homography H (exact)
3. Compute *inliers* where $dist(p_i', \mathbf{H}\, p_i) < \varepsilon$
4. Keep largest set of inliers
5. Re-compute least-squares H estimate on all of the inliers

How many rounds?

- If we have to choose *s* samples each time
  - with an outlier ratio *e*
  - and we want the right answer with probability *p*

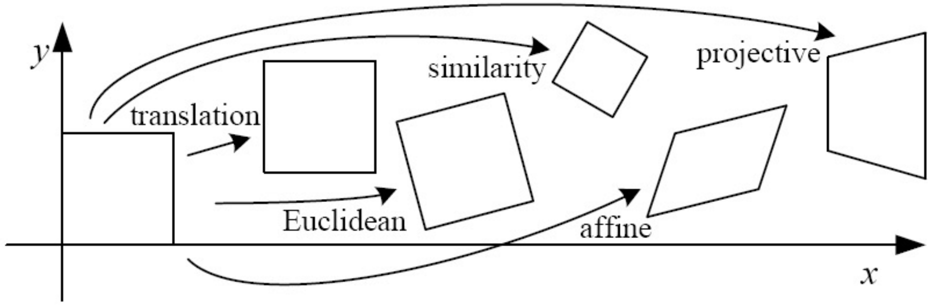
$$N \geq \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

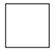
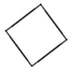
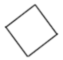
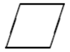
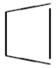
	proportion of outliers *e*						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

$p = 0.99$

How big is *s*?

- For alignment, depends on the motion model
  - Here, each sample is a correspondence (pair of matching points)



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

RANSAC pros and cons

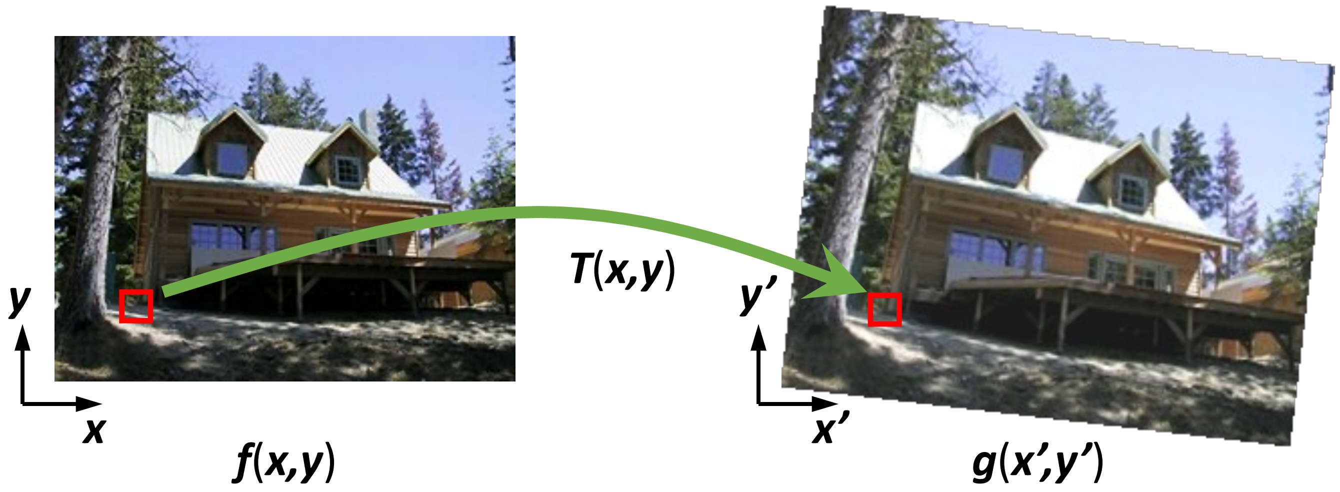
- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Parameters to tune
  - Sometimes too many iterations are required
  - Can fail for extremely low inlier ratios
  - We can often do better than brute-force sampling

Today's class

- Fitting with outliers – RANSAC
- Warping
- Blending
- HW3 Motivation

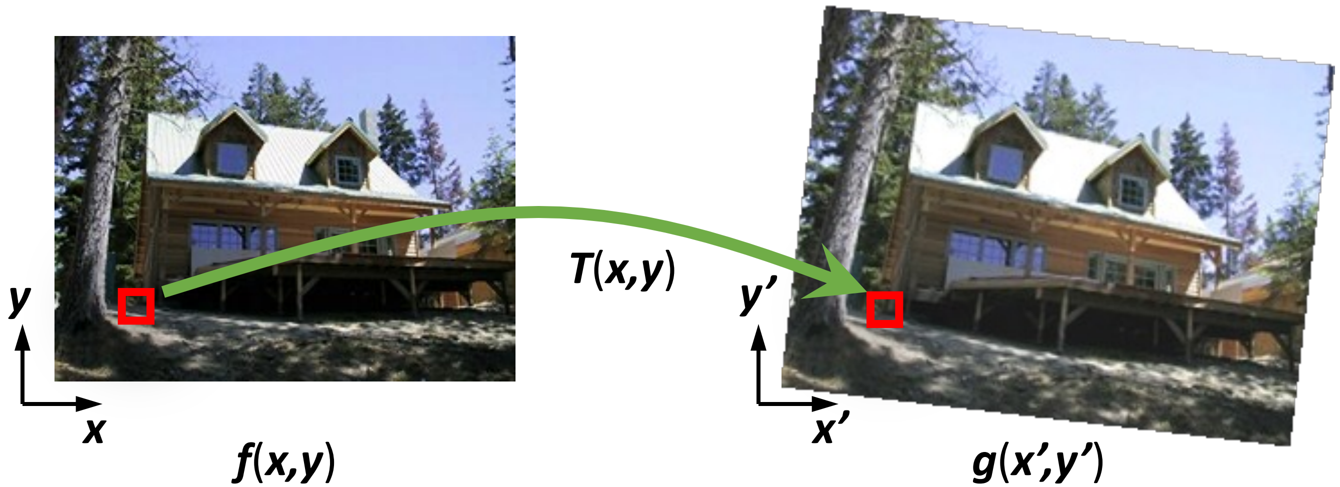
Implementing image warping

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



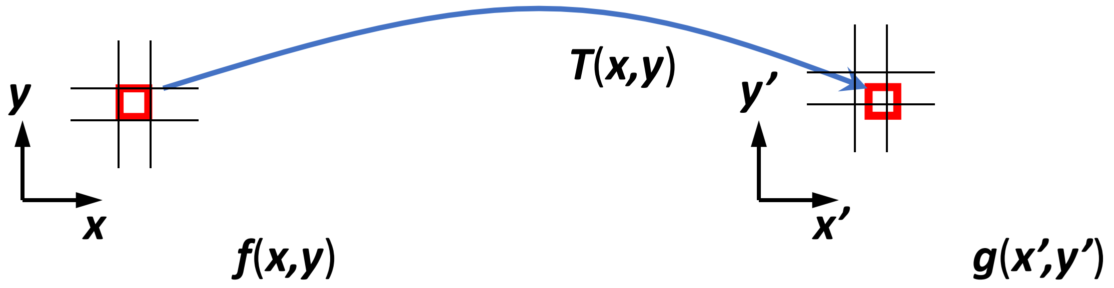
Forward Warping

- Send each pixel (***x,y***) to its corresponding location (***x',y'***) = ***T***(***x,y***) in ***g***(***x',y'***)
    - What if pixel lands "between" two pixels?



Forward Warping

- Send each pixel (***x,y***) to its corresponding location (***x',y'***) = ***T***(***x,y***) in ***g***(***x',y'***)
  - What if pixel lands "between" two pixels?
  - Answer: add "contribution" to several pixels, normalize later (*splatting*)
  - Can still result in holes



Today’s class

- Fitting with outliers – RANSAC
- Warping
- Blending
- HW3 Motivation

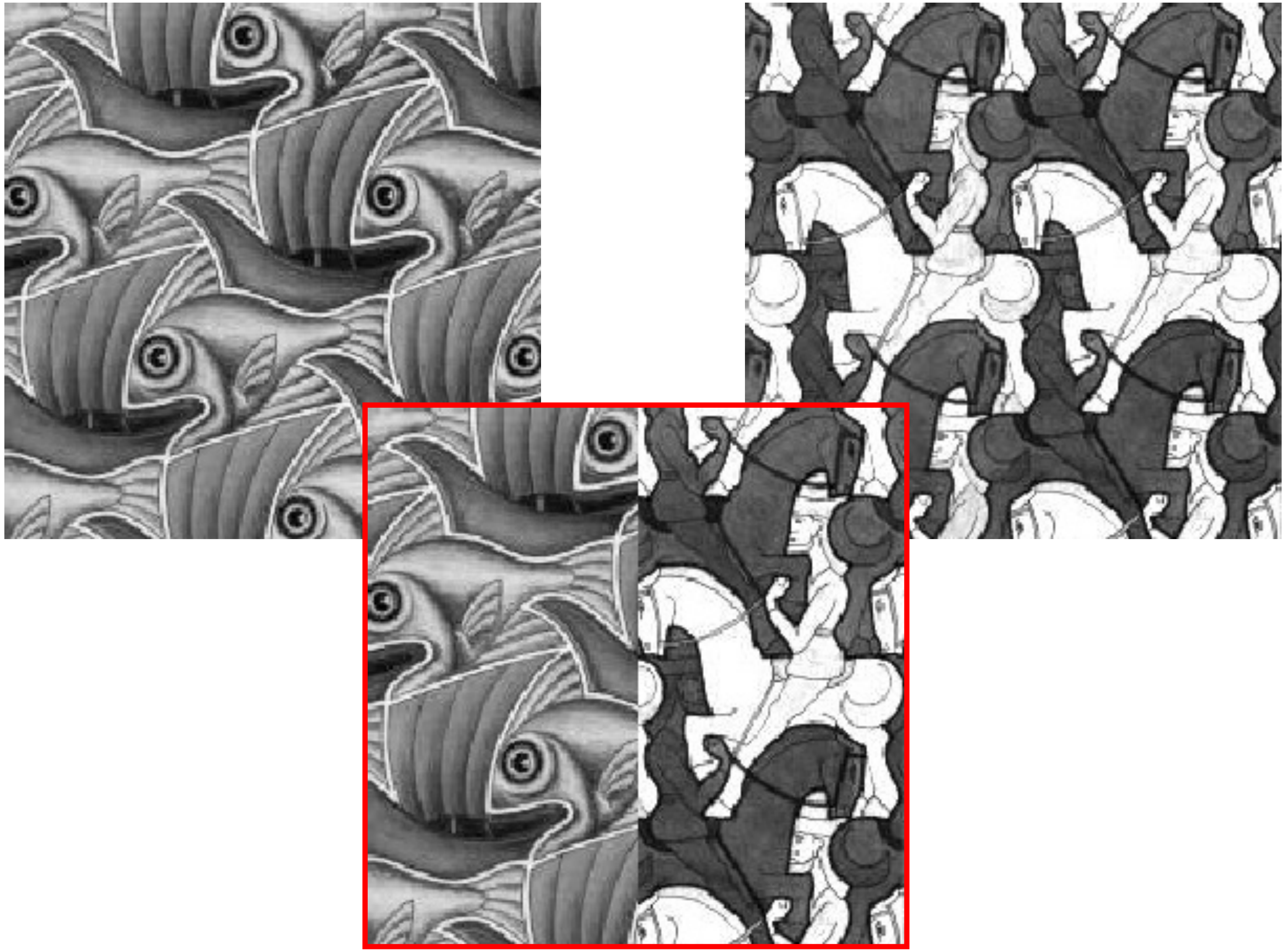
Blending

- We've aligned the images – now what?

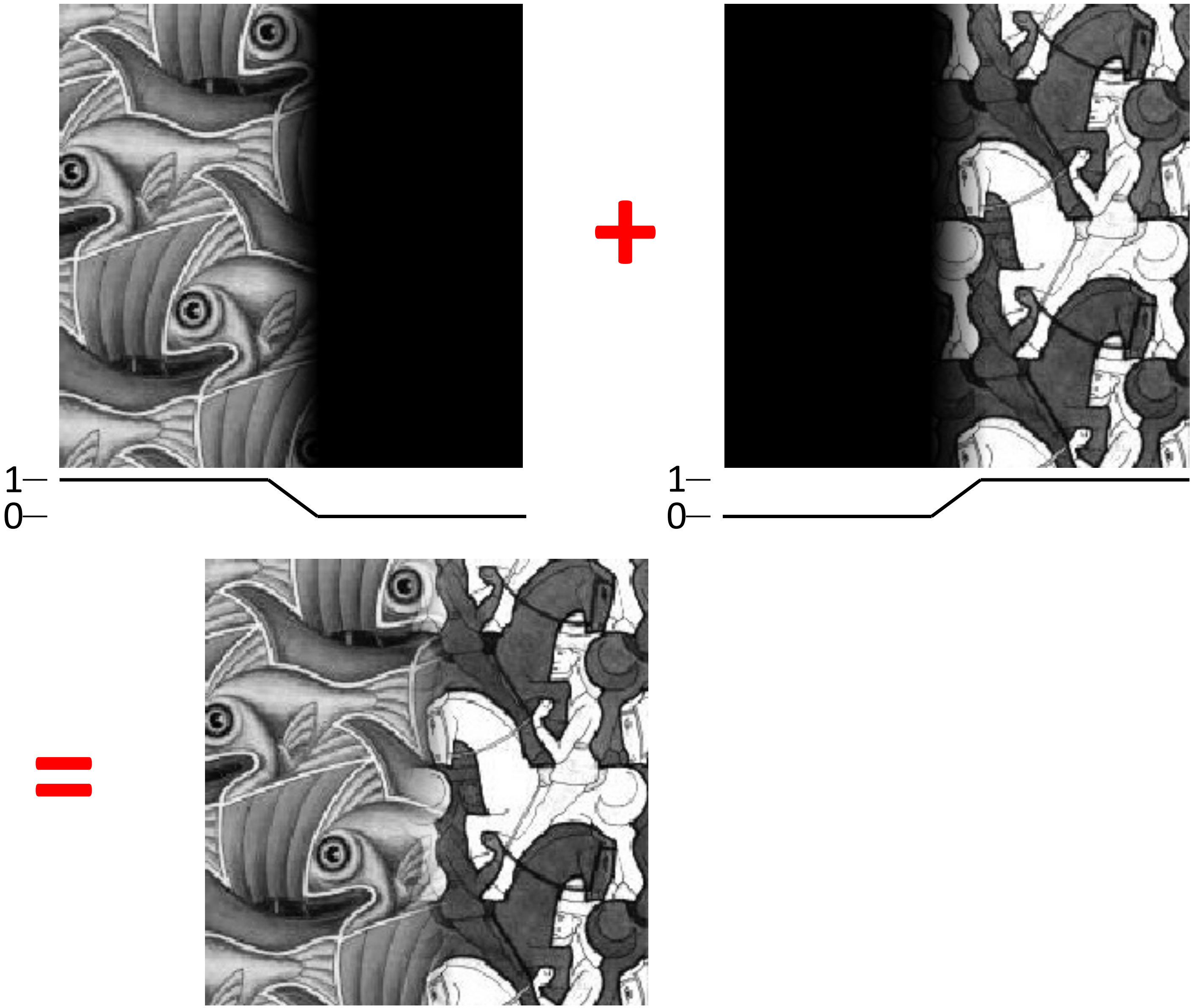
Blending

- Want to seamlessly blend them together

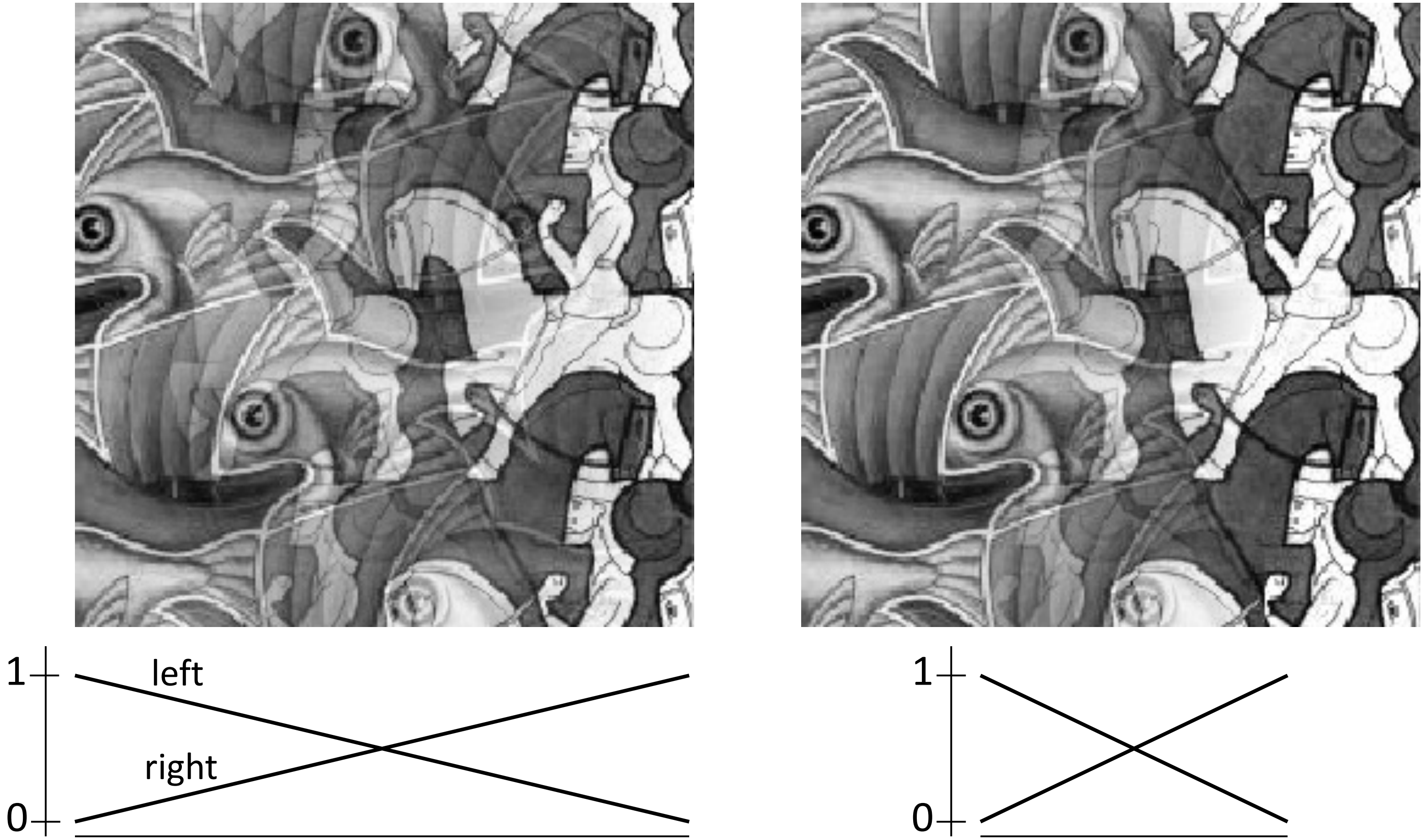
Image Blending



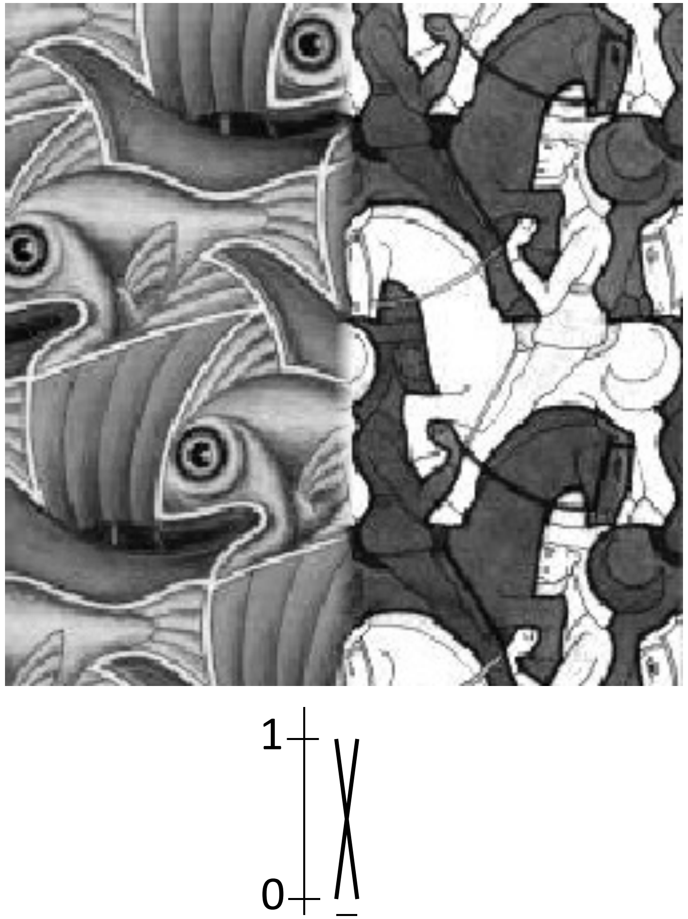
Feathering

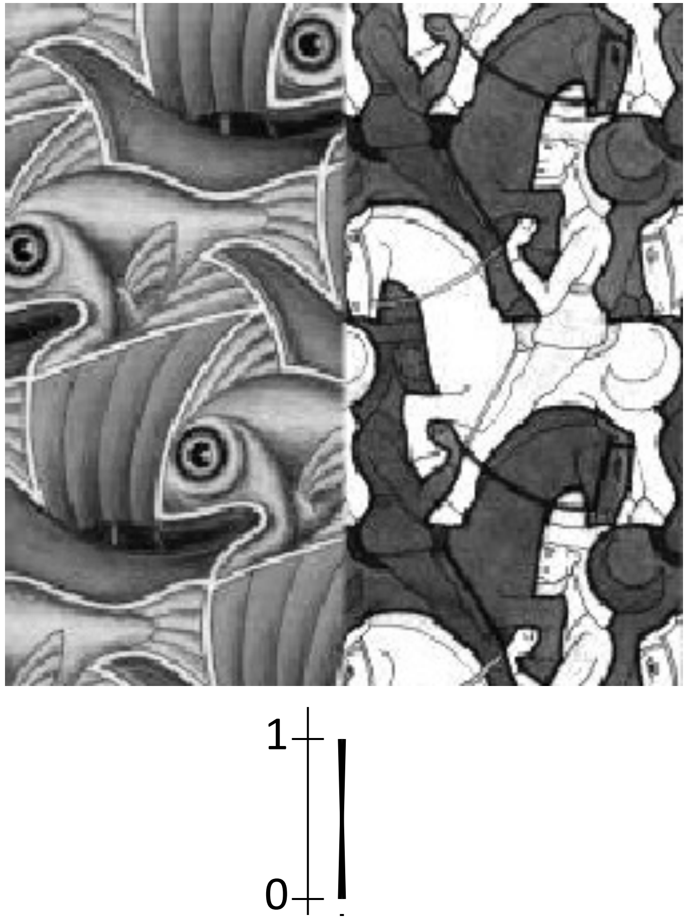


Effect of window size

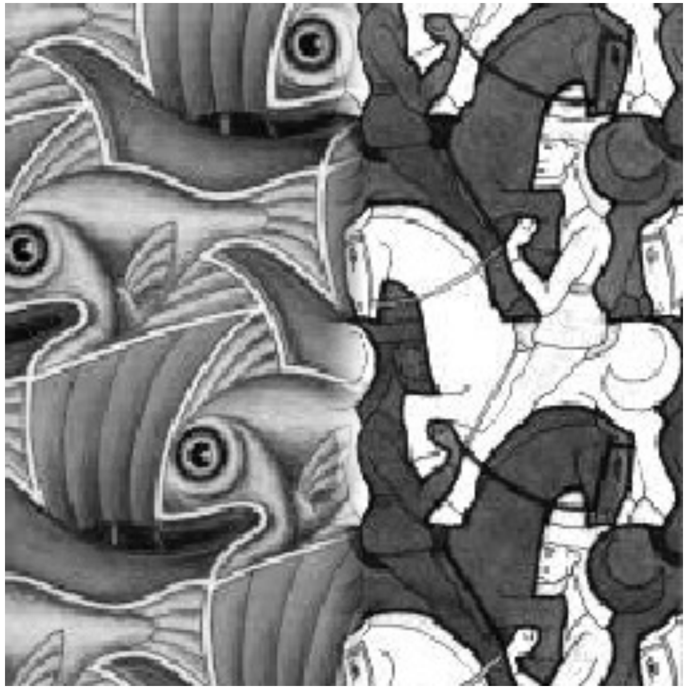


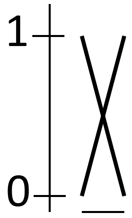
Effect of window size





Good window size

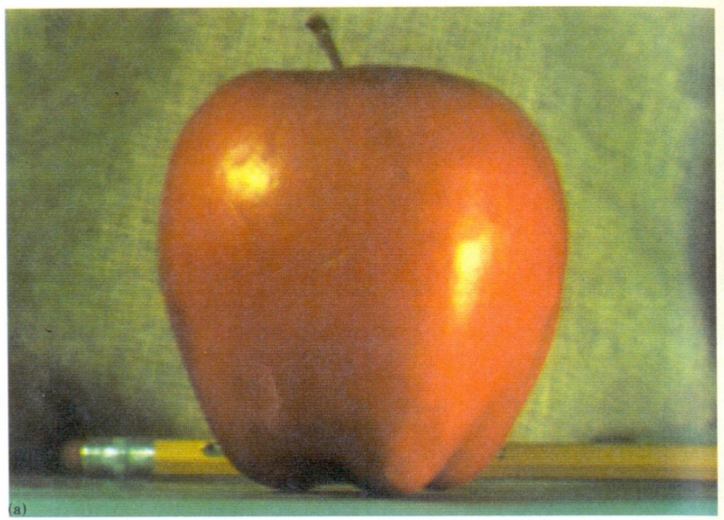


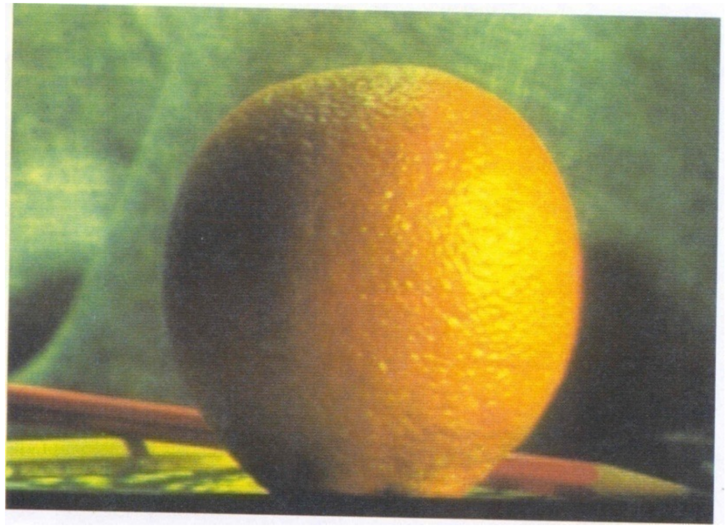


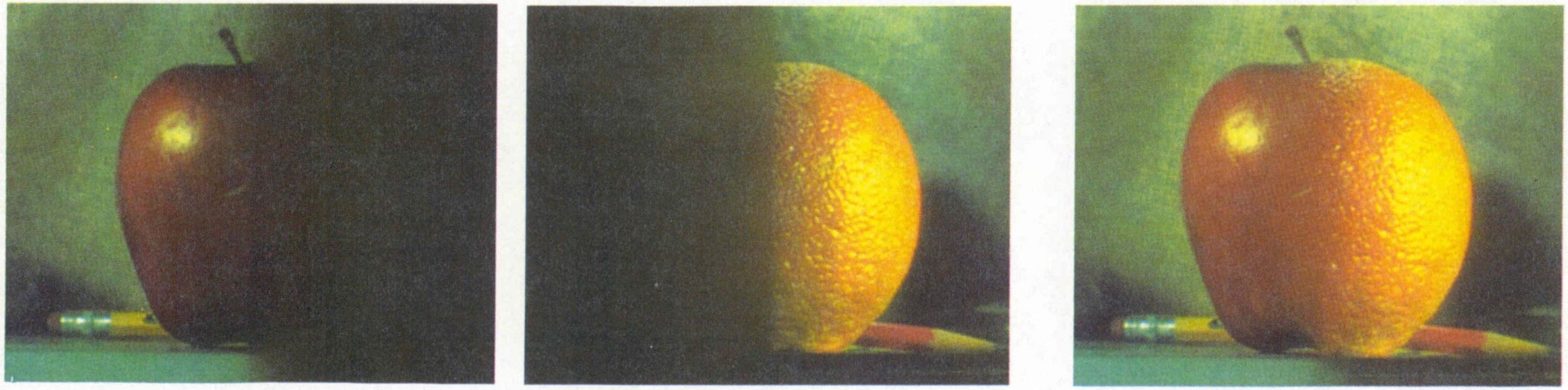
"Optimal" window: smooth but not ghosted

- Doesn't always work...

Pyramid blending





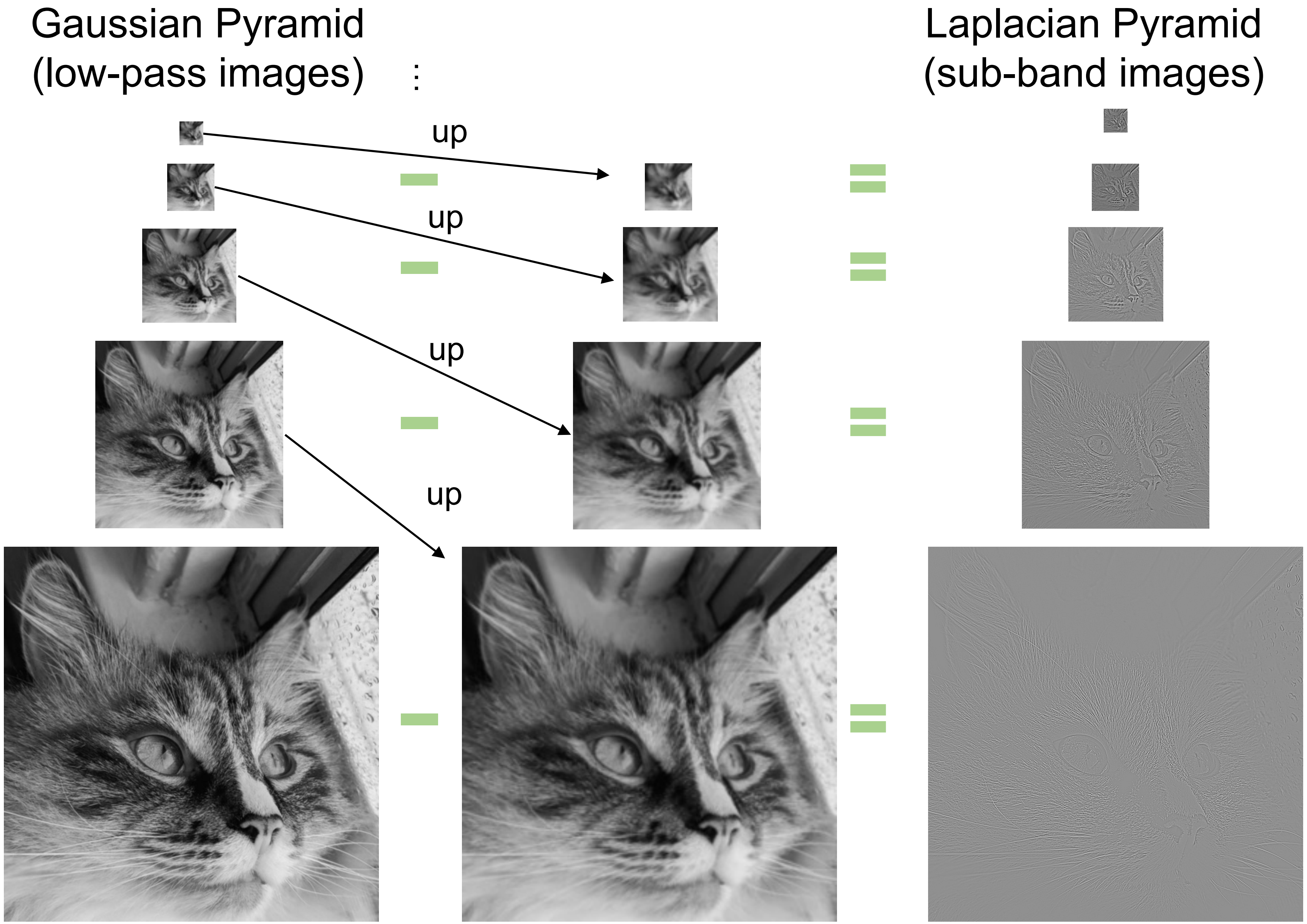


(d) (h) (l)

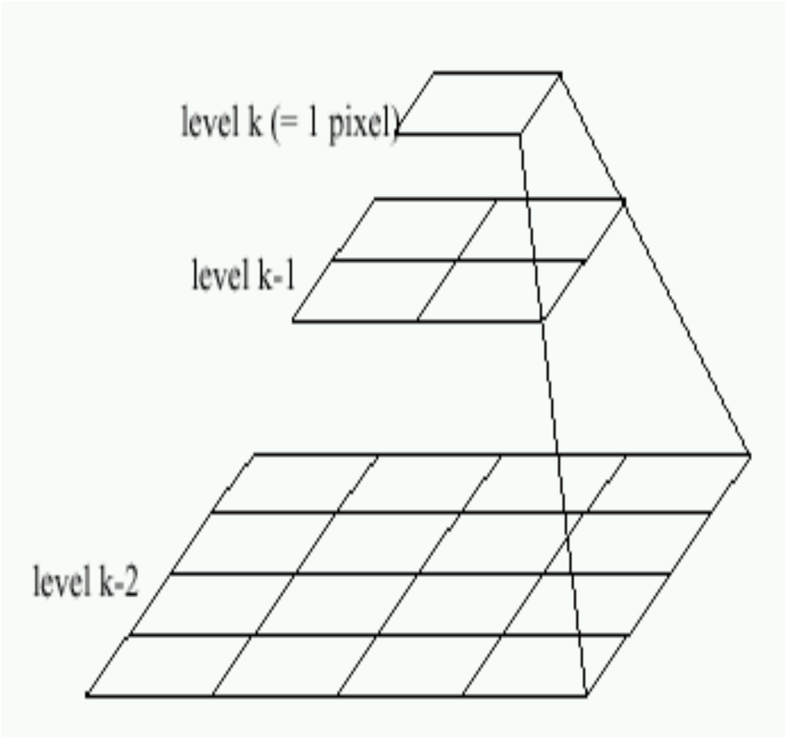
Create a Laplacian pyramid, blend each level

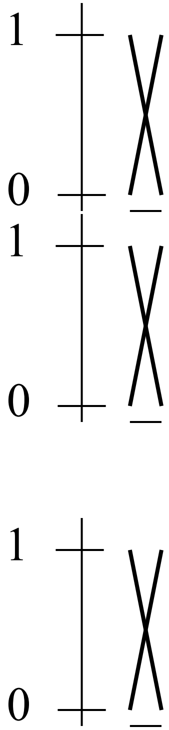
- Burt, P. J. and Adelson, E. H., A multiresolution spline with applications to image mosaics, ACM Transactions on Graphics, 42(4), October 1983, 217-236.

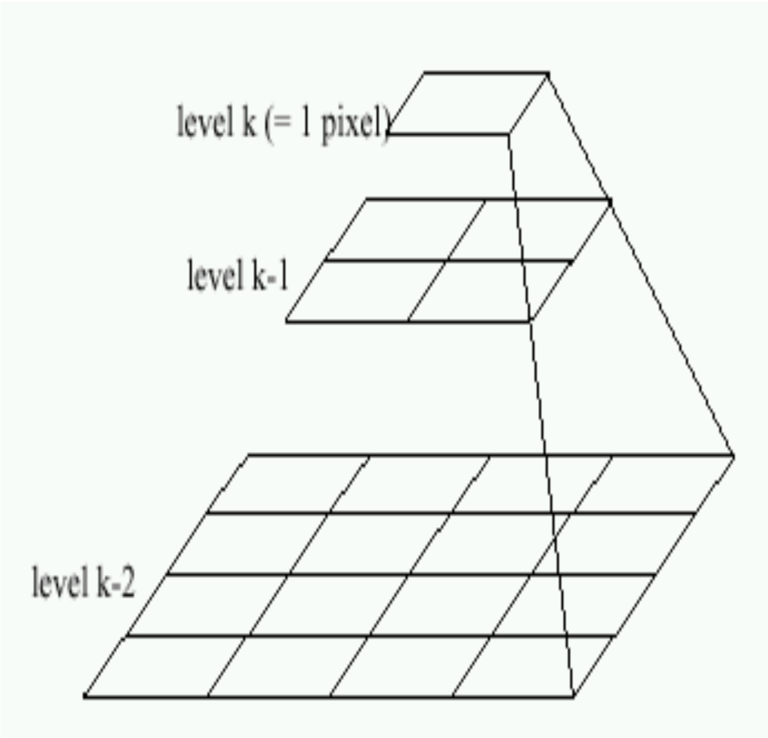
Band-pass filtering in spatial domain



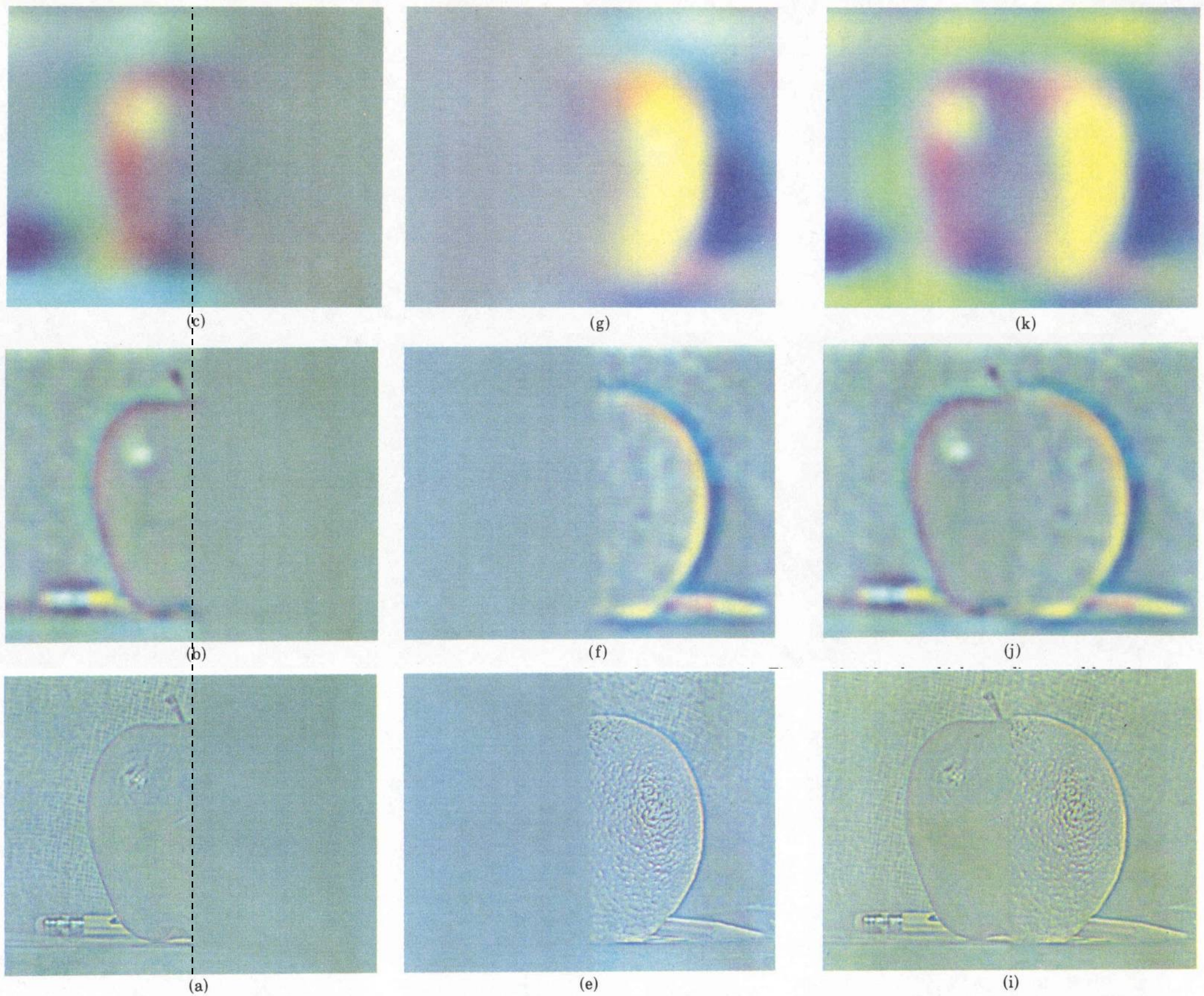
Pyramid Blending







Left pyramid blend Right pyramid



laplacian level 4

laplacian level 2

laplacian level 0

left pyramid

right pyramid

blended pyramid

Laplacian Pyramid: Blending

- General Approach:
  1. Build Laplacian pyramids *LA* and *LB* from images *A* and *B*
  2. Build a Gaussian pyramid *GR* from selected region *R*
  3. Form a combined pyramid *LS* from *LA* and *LB* using nodes of *GR* as weights:
     - *LS(i,j) = GR(I,j,)*LA(I,j) + (1-GR(I,j))*LB(I,j)*
  4. Collapse the *LS* pyramid to get the final blended image

Poisson Image Editing

For more info: [Perez et al, SIGGRAPH 2003]

Today’s class

- Fitting with outliers – RANSAC
- Warping
- Blending
- HW3 Motivation

Fun with homographies

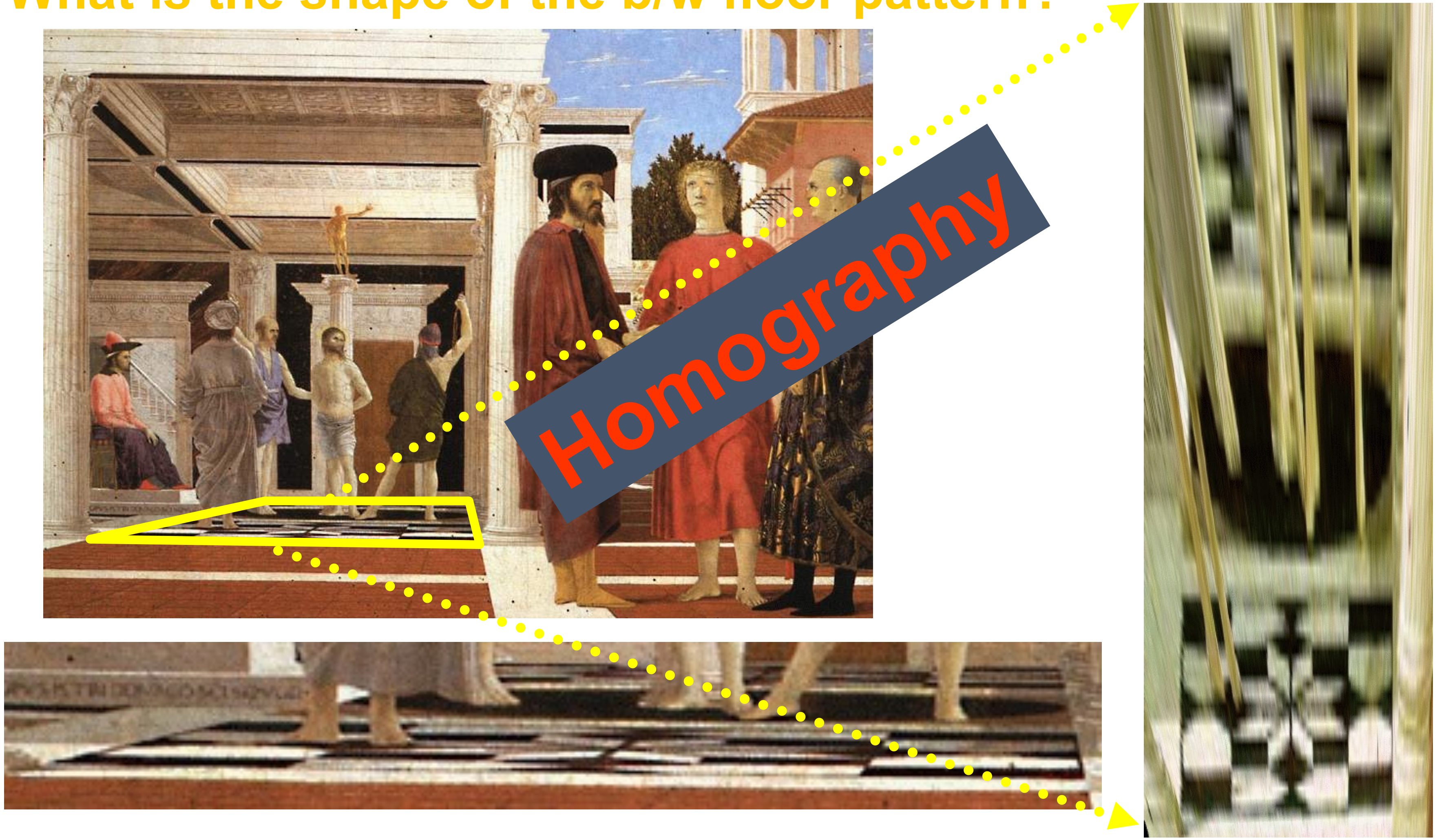
Original image

St.Petersburg
photo by A. Tikhonov

Virtual camera rotations

Analysing patterns and shapes

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



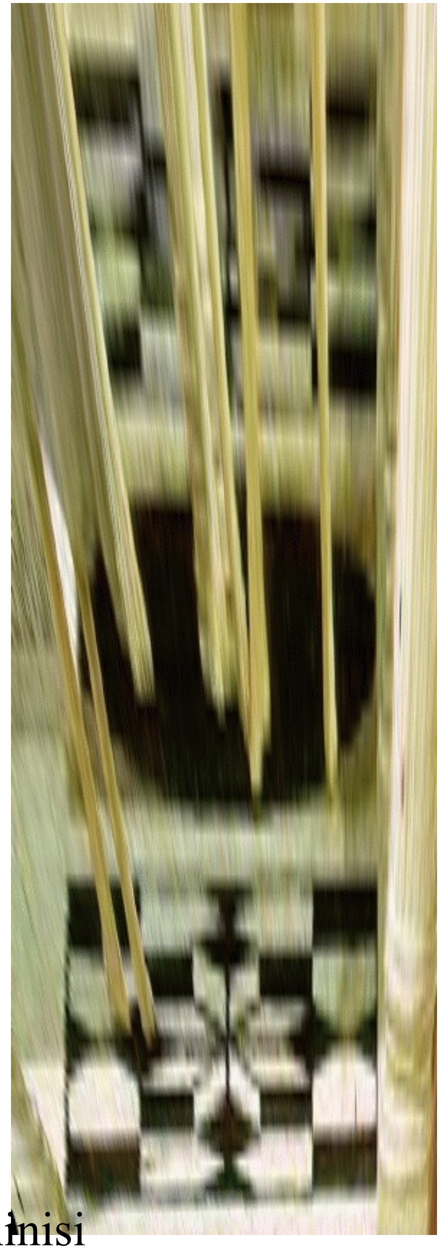
The floor (enlarged)

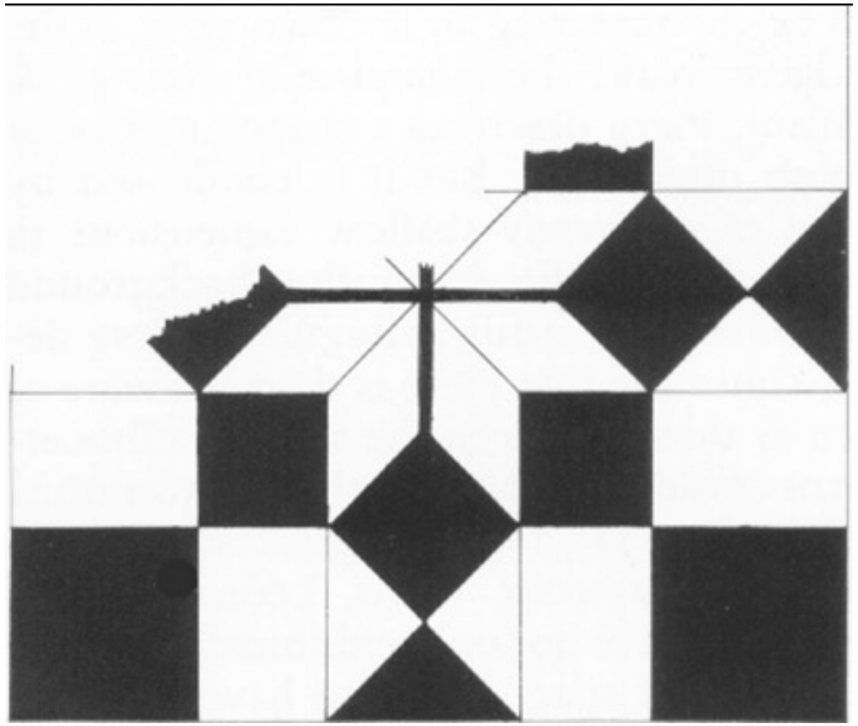
Automatically rectified floor

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**

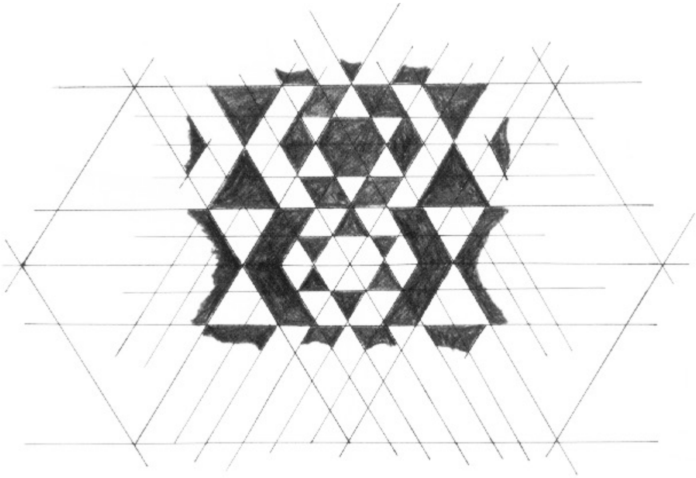
Slide from Criminisi

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

Automatically rectified floor

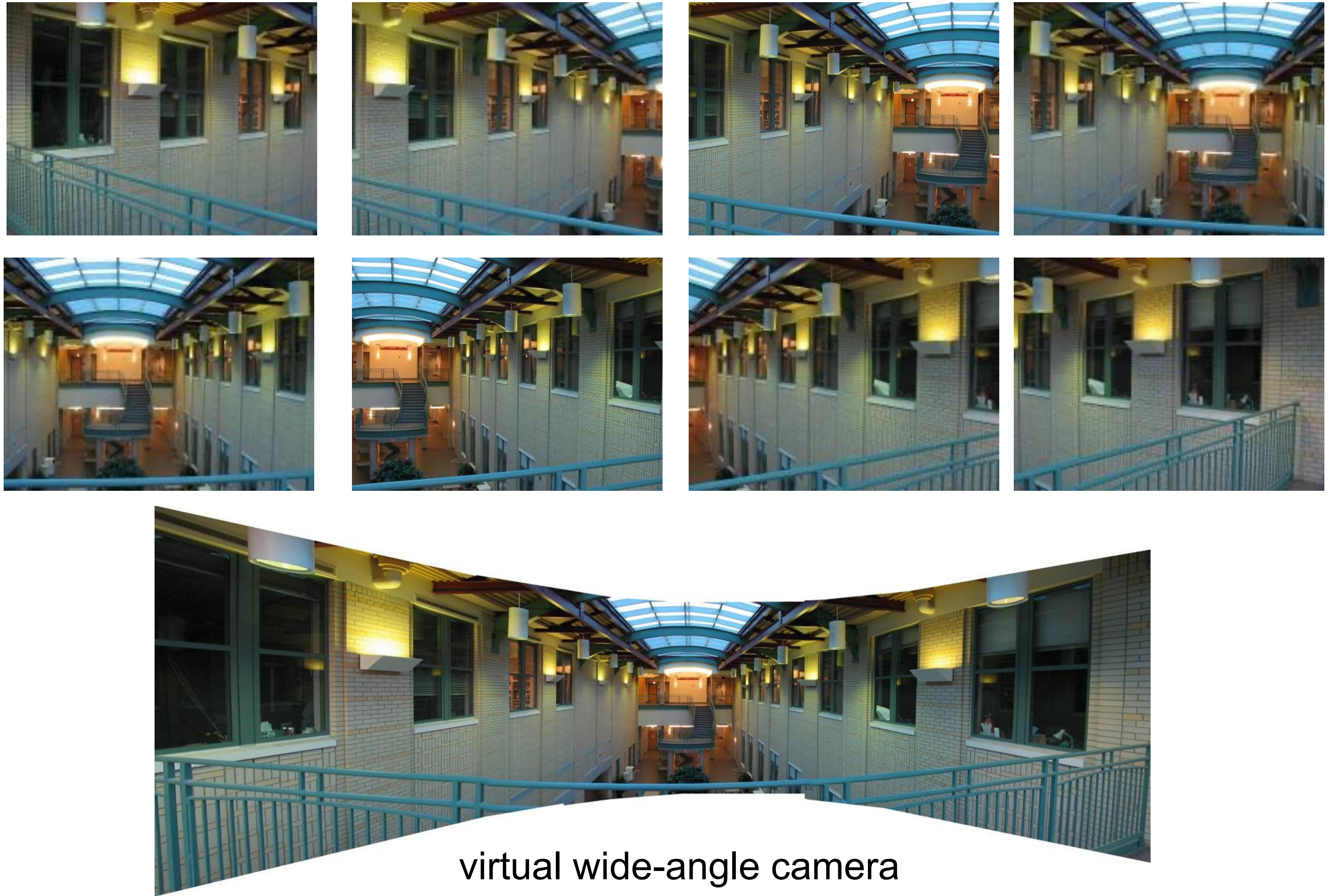
Analysing patterns and shapes

Automatic rectification



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

Slide from Criminisi



virtual wide-angle camera

Some panorama examples

“Before SIGGRAPH Deadline” Photo credit: Doug Zongker

Some panorama examples

- Every image on Google Streetview

Slide Credits

- CS5670, Introduction to Computer Vision, **Cornell Tech, by Noah Snavely.**
- CS 194-26/294-26: Intro to Computer Vision and Computational Photography, **UC Berkeley, by Alyosha Efros.**