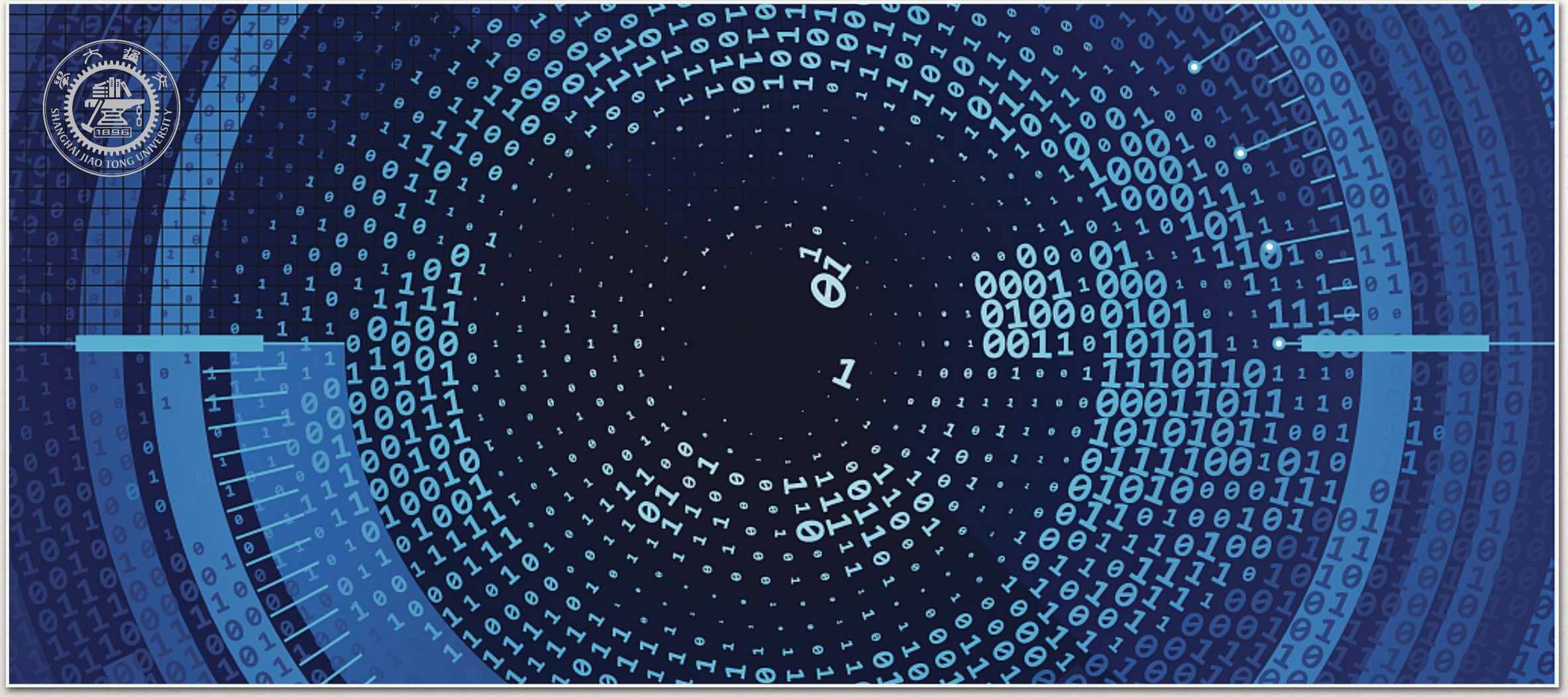


Shanghai Jiao Tong University

Computer Vision

Instructor: Xu Zhao

Class No.: C032703 F032528

Spring 2020

Xu Zhao @ Shanghai Jiao Tong university

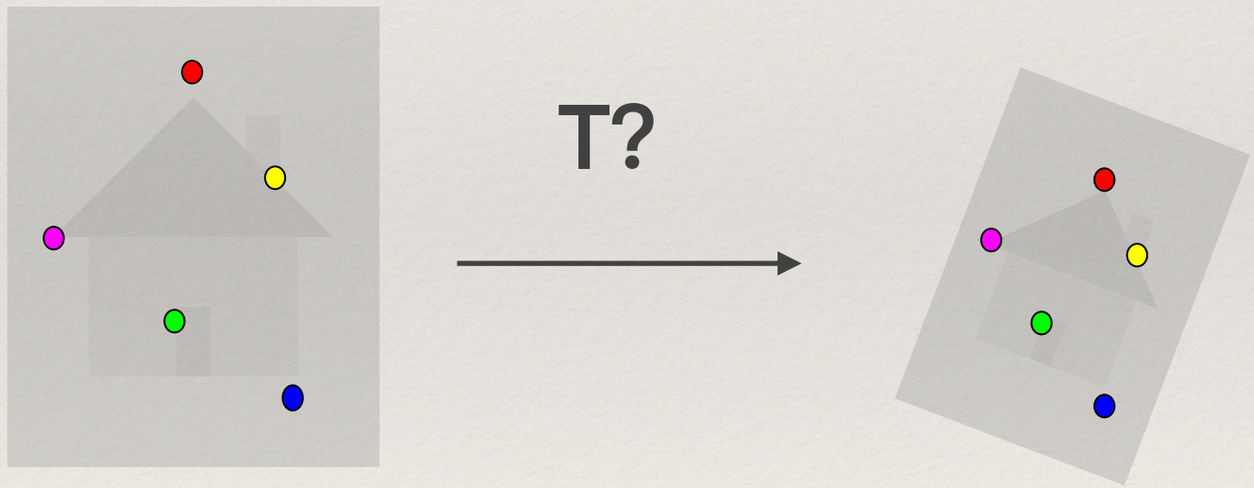
Lecture 7: Registration

Contents

- RANSAC for registration
- Registration with ICP
- Image mosaics
- Deformable object registration

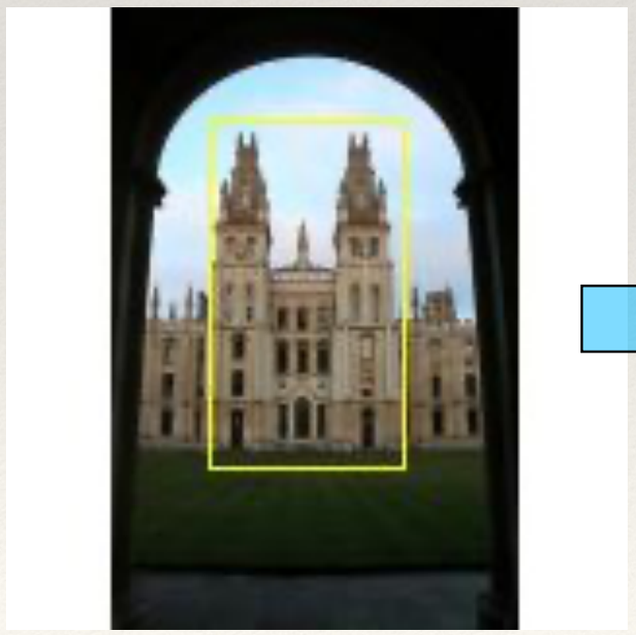
The problem of registration (alignment)

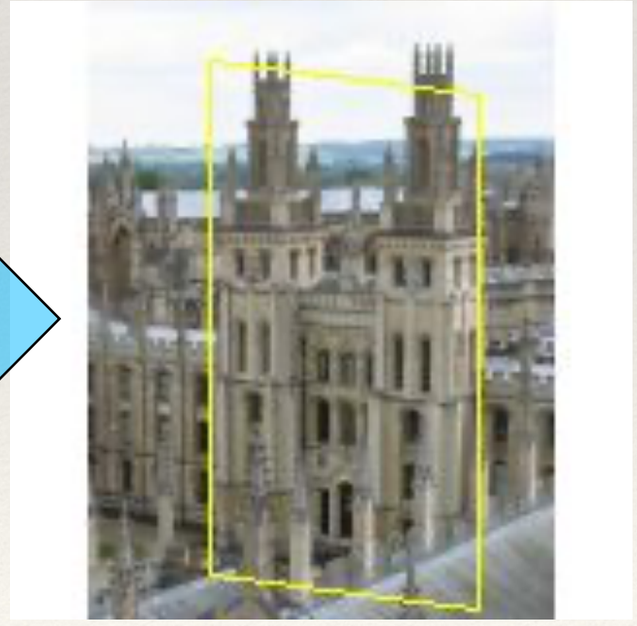
- Registration: finding a **transformation** that takes one dataset to another
- Data：3D ↔ 3D, 2D ↔ 2D， register 3D data to 3D data or 2D data to 2D data
- Transformation
  - Similarity: scale, rotation, translation
  - Affine
  - Homograph
- Object
  - Rigid object
  - Deformable object
- Algorithms
  - Least squares
  - RANSAC
  - ICP…



Applications: Recognition

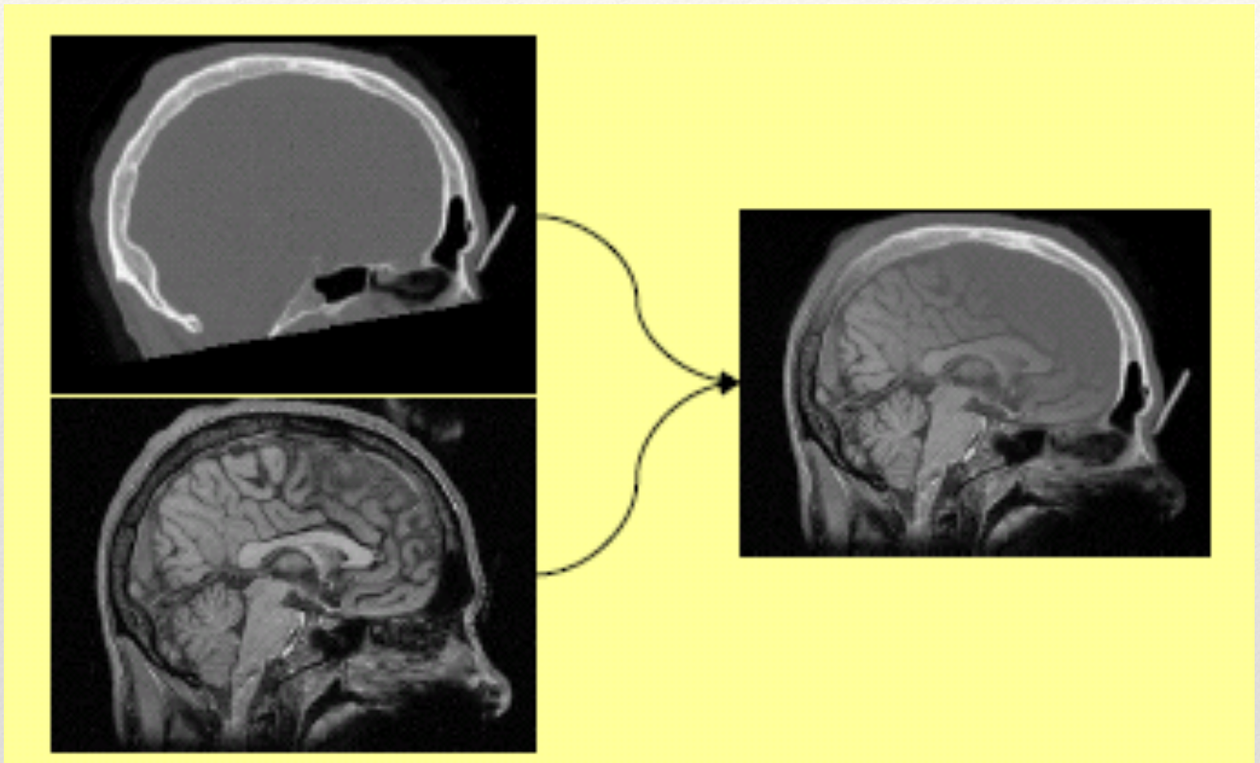
Figures from David Lowe





Figures from Kristen Grauman

Applications: medical image registration



Figures from Kristen Grauman

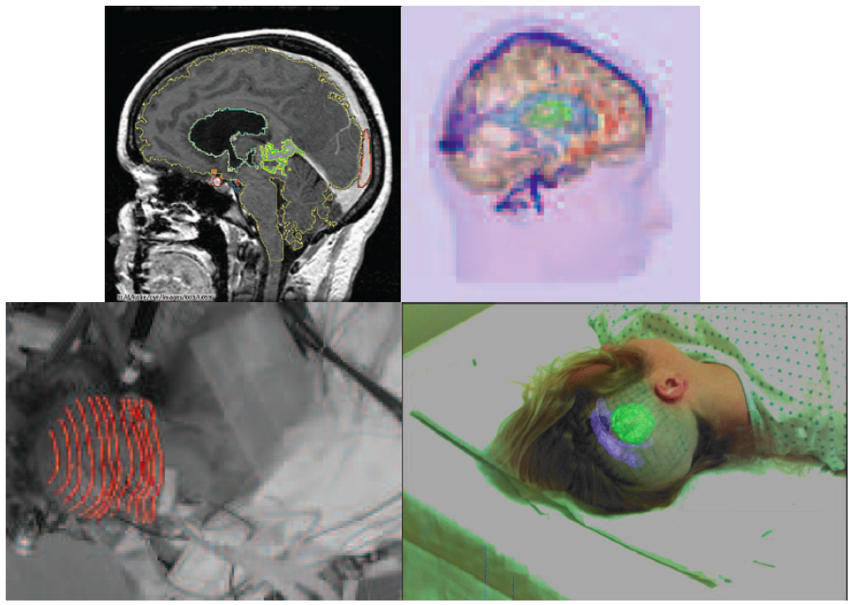


FIGURE 12.11: On the **top left**, a single slice of MRI data with an automatically acquired segmentation overlaid. The segmentation outlines the brain, vacuoles within the brain, and the tumor. MRI produces a sequence of slices, which yield a volume model; a view of a segmented volume model, with different colors showing different regions, is shown at the **top right**. Once this data is obtained, it is registered to a patient lying on a table. Registration is obtained using depth data measured by a laser ranger; the **bottom-left** figure shows a camera view of a patient with laser ranger data overlaid. By registering the segmented data to the patient on the operating table using this laser ranger data and the surface of the MRI data, we can display a processed version of the MRI imagery overlaid on the patient for the surgeon's information (**bottom right**). *Figures by kind permission of Eric Grimson; further information can be obtained from his web site,* `http://www.ai.mit.edu/people/welg/welg.html`.

Figures from D. Forsyth & J. Ponce

Applications: mosaics

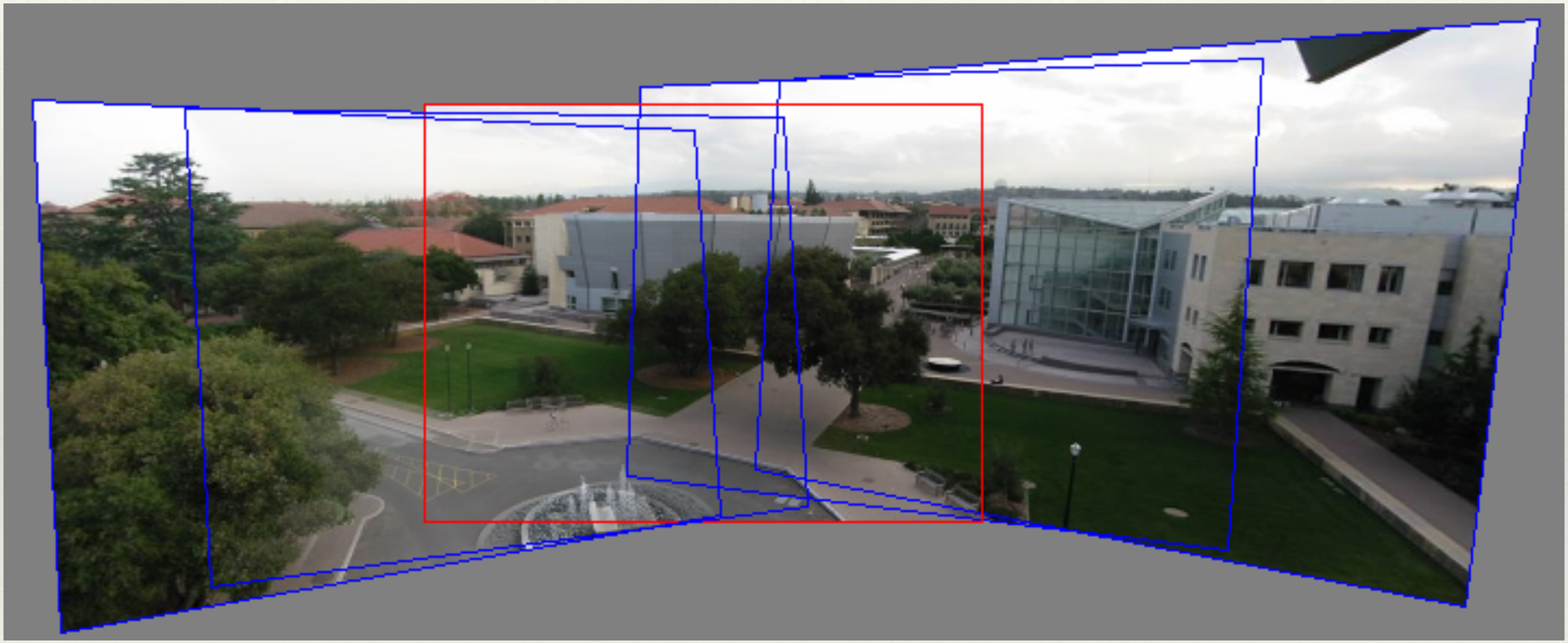


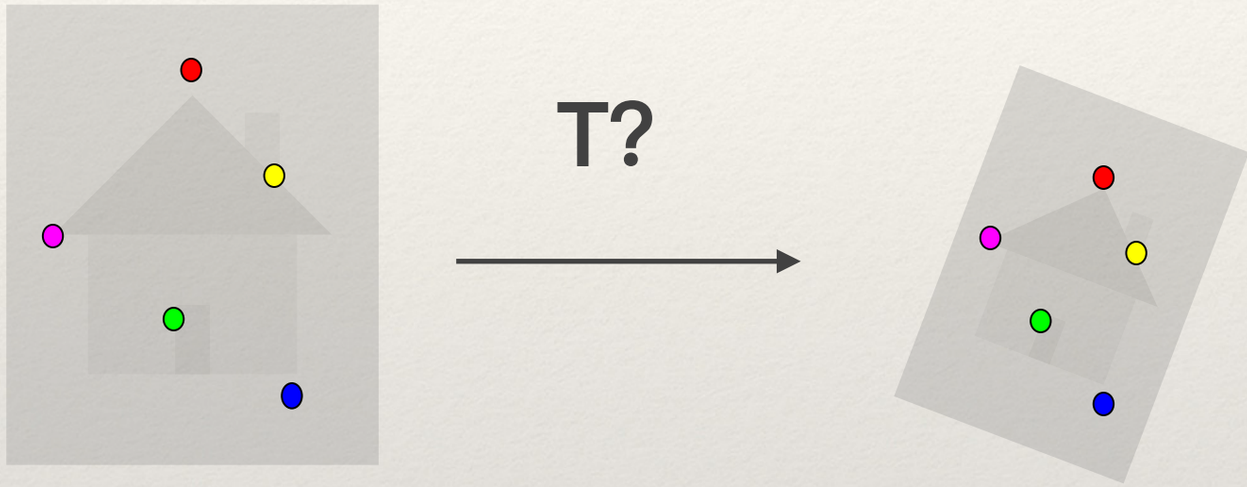
Image from http://graphics.cs.cmu.edu/courses/15-463/2010_fall/

Feature based registration

- General procedures:
  - **Feature detection**: salient and distinctive objects (closed-boundary regions, edges, contours, line intersections, corners, etc.)
  - **Feature matching**: the correspondence between the features detected in the sensed image and those detected in the reference image is established
  - **Transform model estimation:** the type and parameters of the so-called mapping functions, aligning the sensed image with the reference image, are estimated.
  - **Image resampling and transformation (warping)**: the sensed image is transformed by means of the mapping functions.

Transform model estimation

- Example: estimating an affine transformation



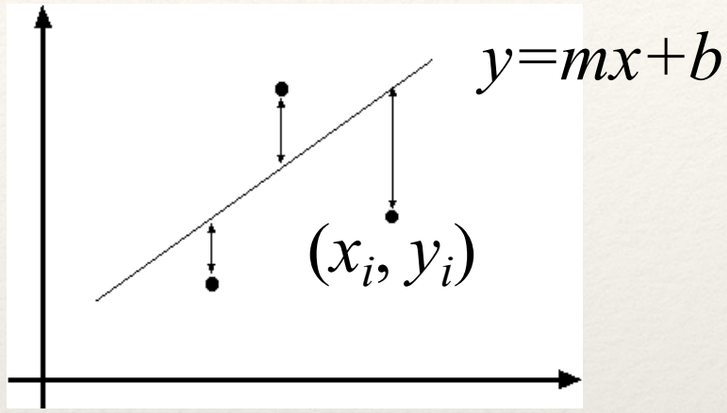
$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

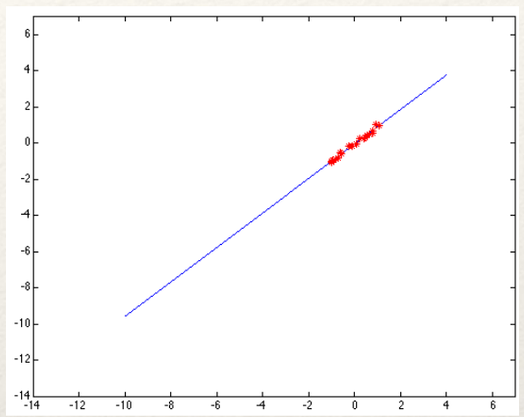
$$A\mathbf{p} = \mathbf{y}$$

Recap: Least squares line fitting

- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m x_i + b$
- Find (m, b) to minimize

$$E = \sum_{i=1}^{n} (y_i - mx_i - b)^2$$





$$E = \sum_{i=1}^{n} \left(\begin{bmatrix} x_i & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - y_i \right)^2 = \left\| \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} m \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \right\|^2 = \|\mathbf{Ap} - \mathbf{y}\|^2$$

$$= \mathbf{y}^T\mathbf{y} - 2(\mathbf{Ap})^T\mathbf{y} + (\mathbf{Ap})^T(\mathbf{Ap})$$

$$\frac{dE}{dp} = 2\mathbf{A}^T\mathbf{Ap} - 2\mathbf{A}^T\mathbf{y} = 0$$

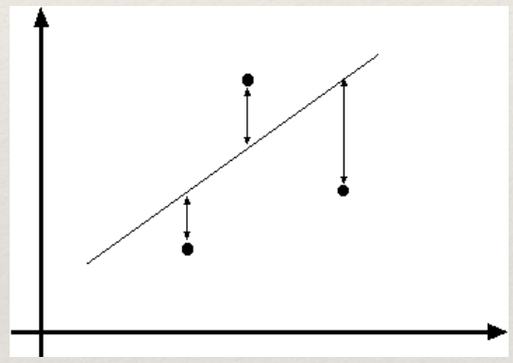
```
Matlab: p = A \ y;
Python:
p = np.linalg.lstsq(A,y)[0]
```

$$\mathbf{A}^T\mathbf{Ap} = \mathbf{A}^T\mathbf{y} \Rightarrow \mathbf{p} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{y}$$ (Closed form solution)

Modified from S. Lazebnik

Problem with “vertical” least squares

- Not rotation-invariant
- Fails completely for vertical lines



Slide from S. Lazebnik

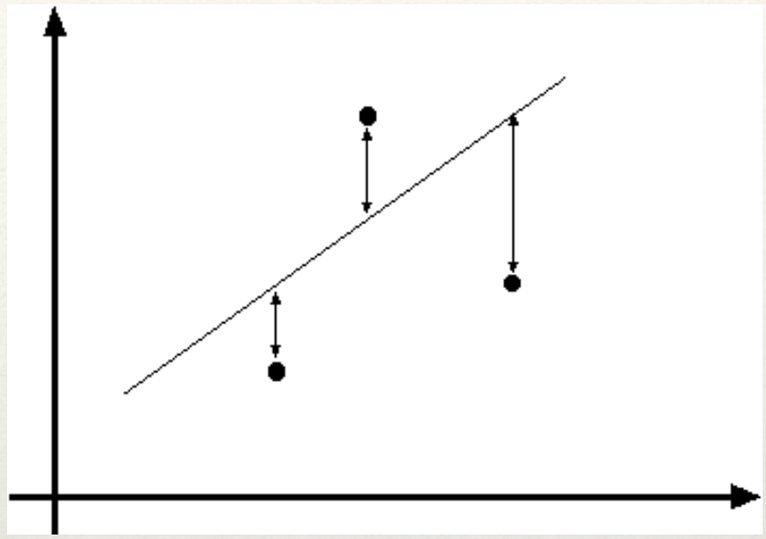
Total least squares

If ($a^2+b^2=1$) then

Distance between point (x_i, y_i) is

$$|ax_i + by_i + c|$$

proof: http://mathworld.wolfram.com/Point-LineDistance2-Dimensional.html

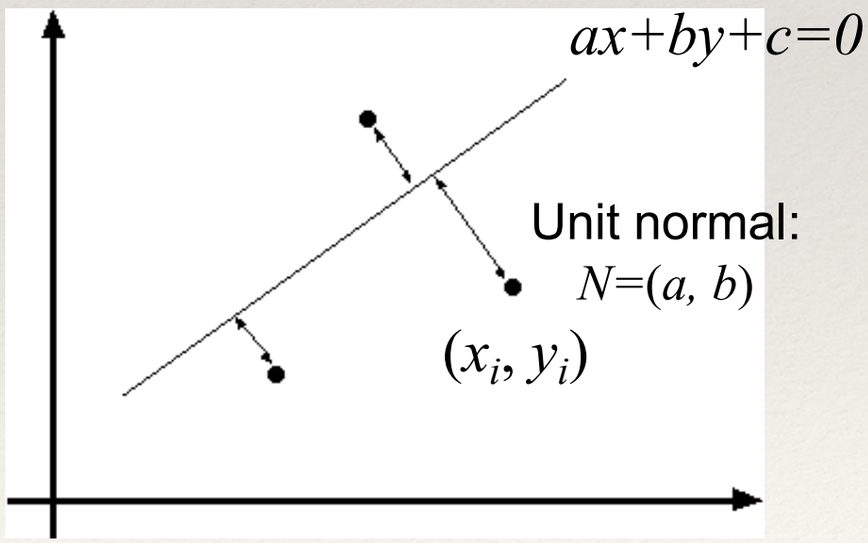


Find (a, b, c) to minimize the sum of squared perpendicular distances

$$E = \sum_{i=1}^{n} (ax_i + by_i + c)^2$$

$$\frac{\partial E}{\partial c} = \sum_{i=1}^{n} 2(ax_i + by_i + c) = 0$$

$$c = -\frac{a}{n}\sum_{i=1}^{n} x_i - \frac{b}{n}\sum_{i=1}^{n} y_i = -a\bar{x} - b\bar{y}$$



Total least squares

Find (a, b, c) to minimize the sum of squared perpendicular distances

$$E = \sum_{i=1}^{n} (a x_i + b y_i + c)^2$$

$$\frac{\partial E}{\partial c} = \sum_{i=1}^{n} 2(a x_i + b y_i + c) = 0$$

$$c = -\frac{a}{n}\sum_{i=1}^{n} x_i - \frac{b}{n}\sum_{i=1}^{n} y_i = -a\bar{x} - b\bar{y}$$

$$E = \sum_{i=1}^{n} (a(x_i - \bar{x}) + b(y_i - \bar{y}))^2 = \left\| \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2 = \mathbf{p}^T \mathbf{A}^T \mathbf{A} \mathbf{p}$$

$$\text{minimize} \mathbf{p}^T \mathbf{A}^T \mathbf{A} \mathbf{p} \quad \text{s.t. } \mathbf{p}^T \mathbf{p} = 1 \quad \Rightarrow \quad \text{minimize} \frac{\mathbf{p}^T \mathbf{A}^T \mathbf{A} \mathbf{p}}{\mathbf{p}^T \mathbf{p}}$$

Solution is eigenvector corresponding to smallest eigenvalue of A^TA

See "Multiple view geometry in computer vision"

- Sec 4.1 for details (DLT algorithm)
- Sec 4.1.2 (or APPENDIX)

Recap: Two common optimization problems

Problem statement

$$\text{minimize } \|\mathbf{Ax}-\mathbf{b}\|^2$$

$$\text{least squares solution to } \mathbf{Ax}=\mathbf{b}$$

Solution

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

$$\mathbf{x}=\mathbf{A}\setminus\mathbf{b} \quad \text{(matlab)}$$

Problem statement

$$\text{minimize } \mathbf{x}^T\mathbf{A}^T\mathbf{Ax} \quad \text{s.t. } \mathbf{x}^T\mathbf{x}=1$$

$$\text{minimize}\frac{\mathbf{x}^T\mathbf{A}^T\mathbf{Ax}}{\mathbf{x}^T\mathbf{x}}$$

$$\text{non - trivial lsq solution to } \mathbf{Ax}=0$$

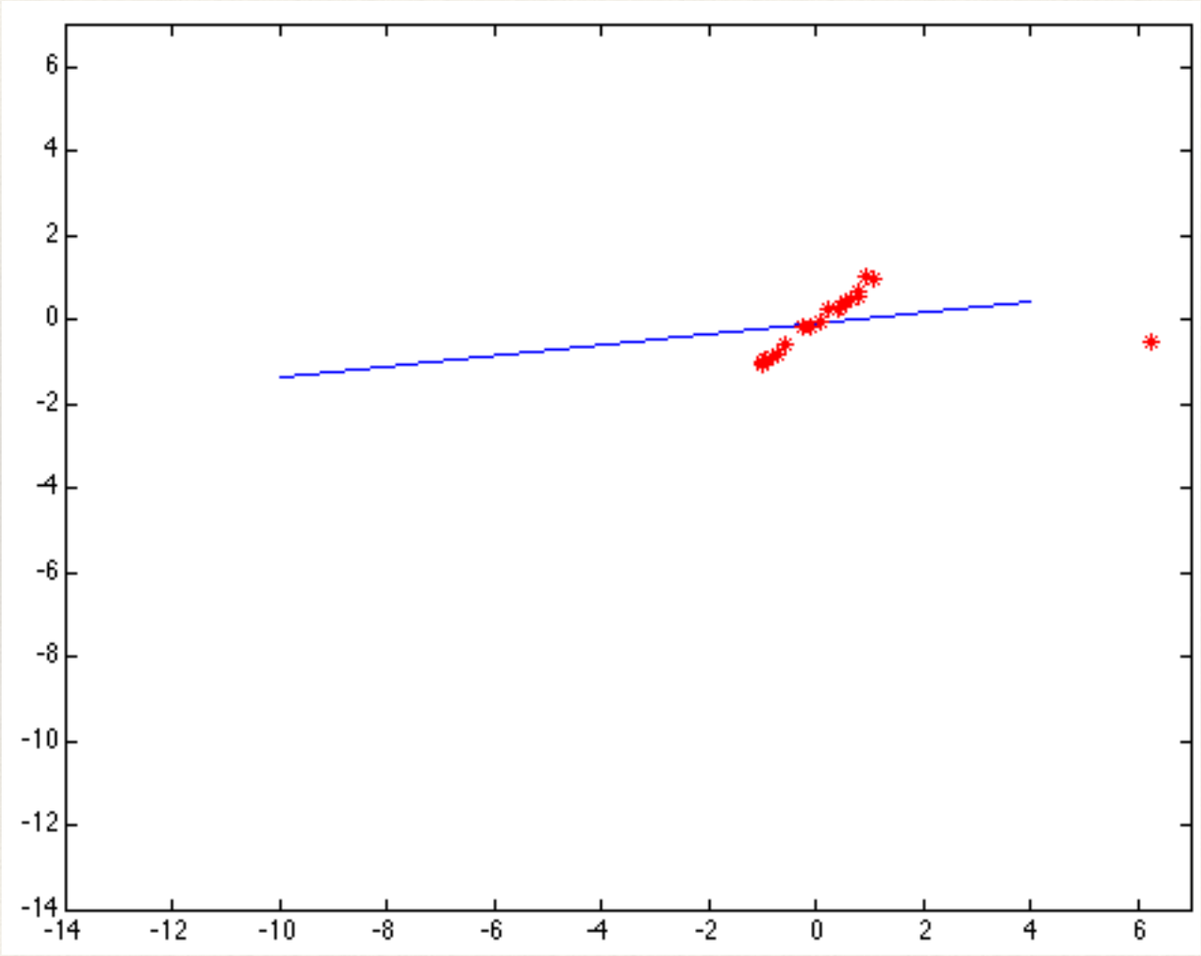
Solution

$$[\mathbf{v},\lambda]=\text{eig}(\mathbf{A}^T\mathbf{A})$$

$$\lambda_1<\lambda_{2..n}:\mathbf{x}=\mathbf{v}_1$$

Least squares: Robustness to noise

- Least squares fit with an outlier:



Robust least squares (to deal with outliers)

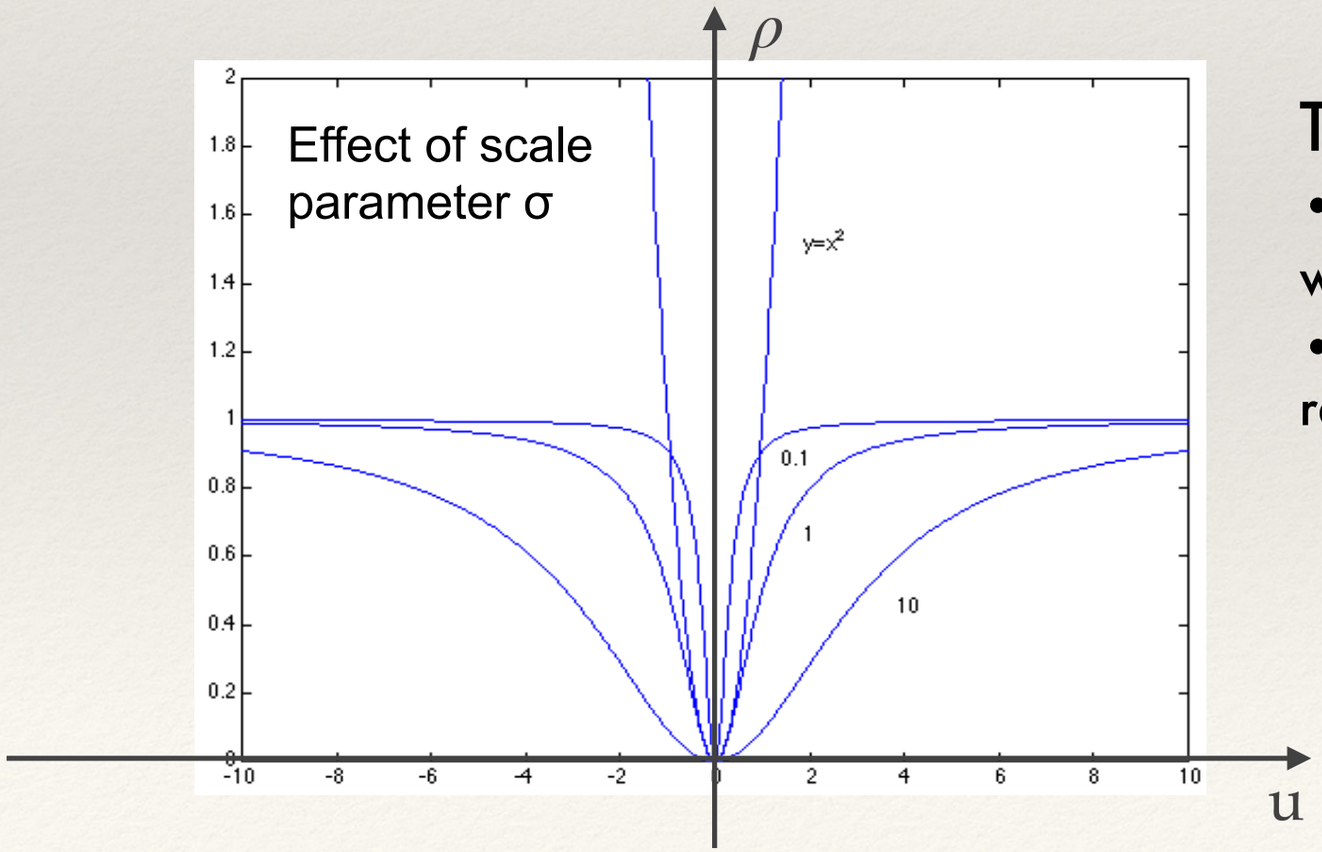
- Least squares fit with an outlier:

General approach:
minimize

$$\sum_i \rho\left(u_i\left(x_i, \theta\right); \sigma\right) \qquad u^2 = \sum_{i=1}^{n} (y_i - mx_i - b)^2$$

$u_i(x_i, \theta)$ – residual of ith point w.r.t. model parameters Θ

ρ – robust function with scale parameter σ



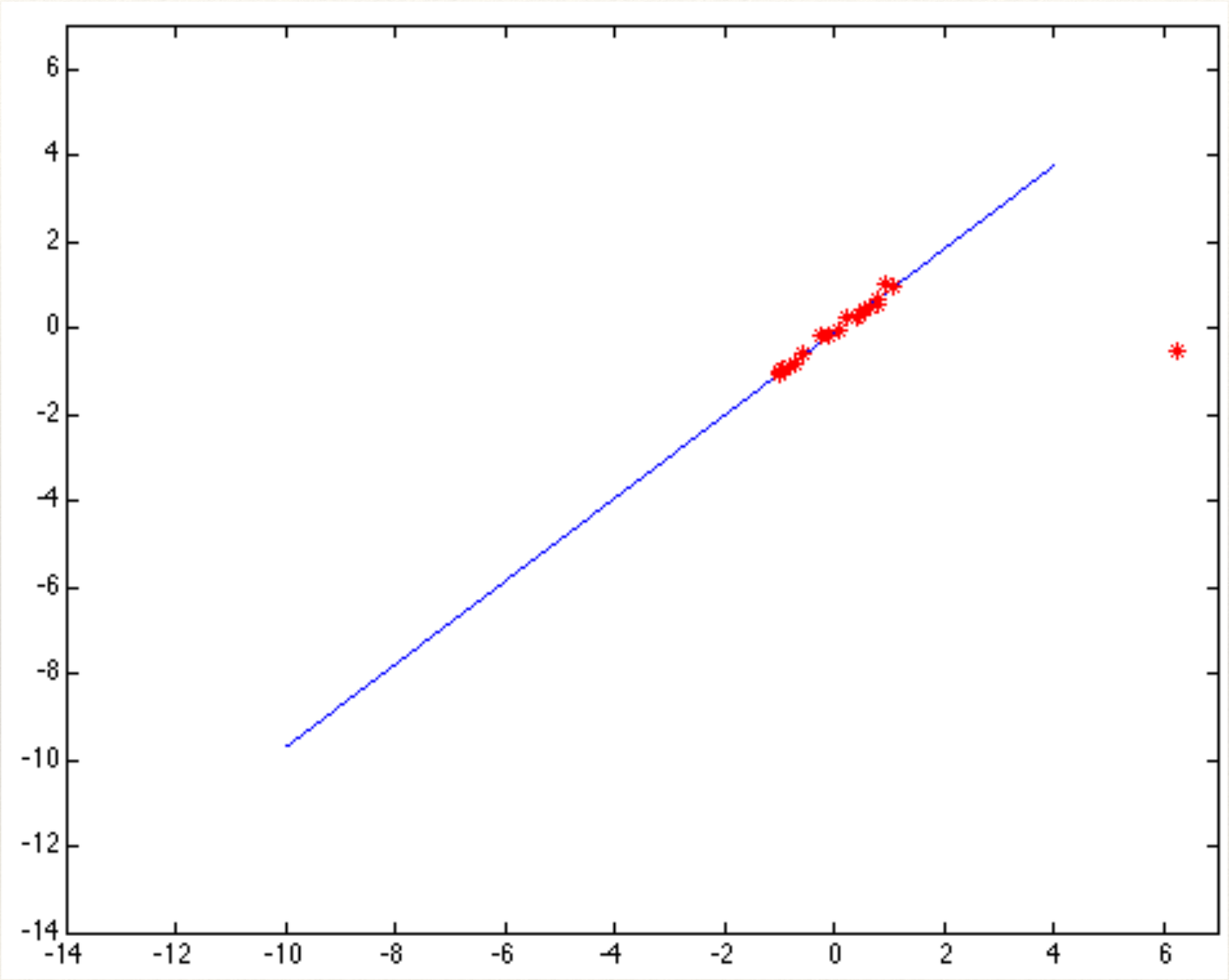
The robust function ρ

- Favors a configuration with small residuals
- Constant penalty for large residuals

$$\rho(u; \sigma) = \frac{u^2}{\sigma^2 + u^2}$$

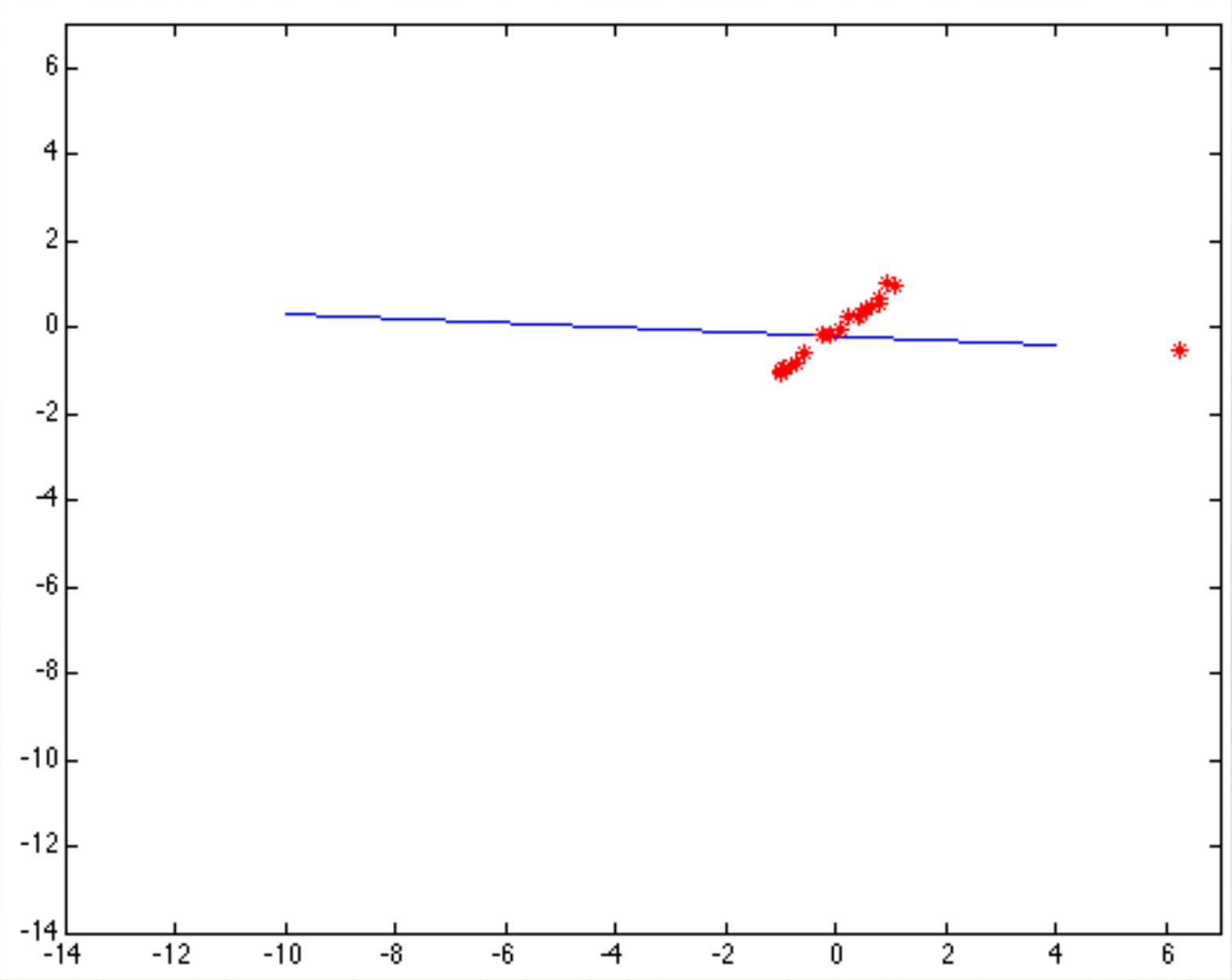
Slide from S. Savarese

Choosing the scale: Just right



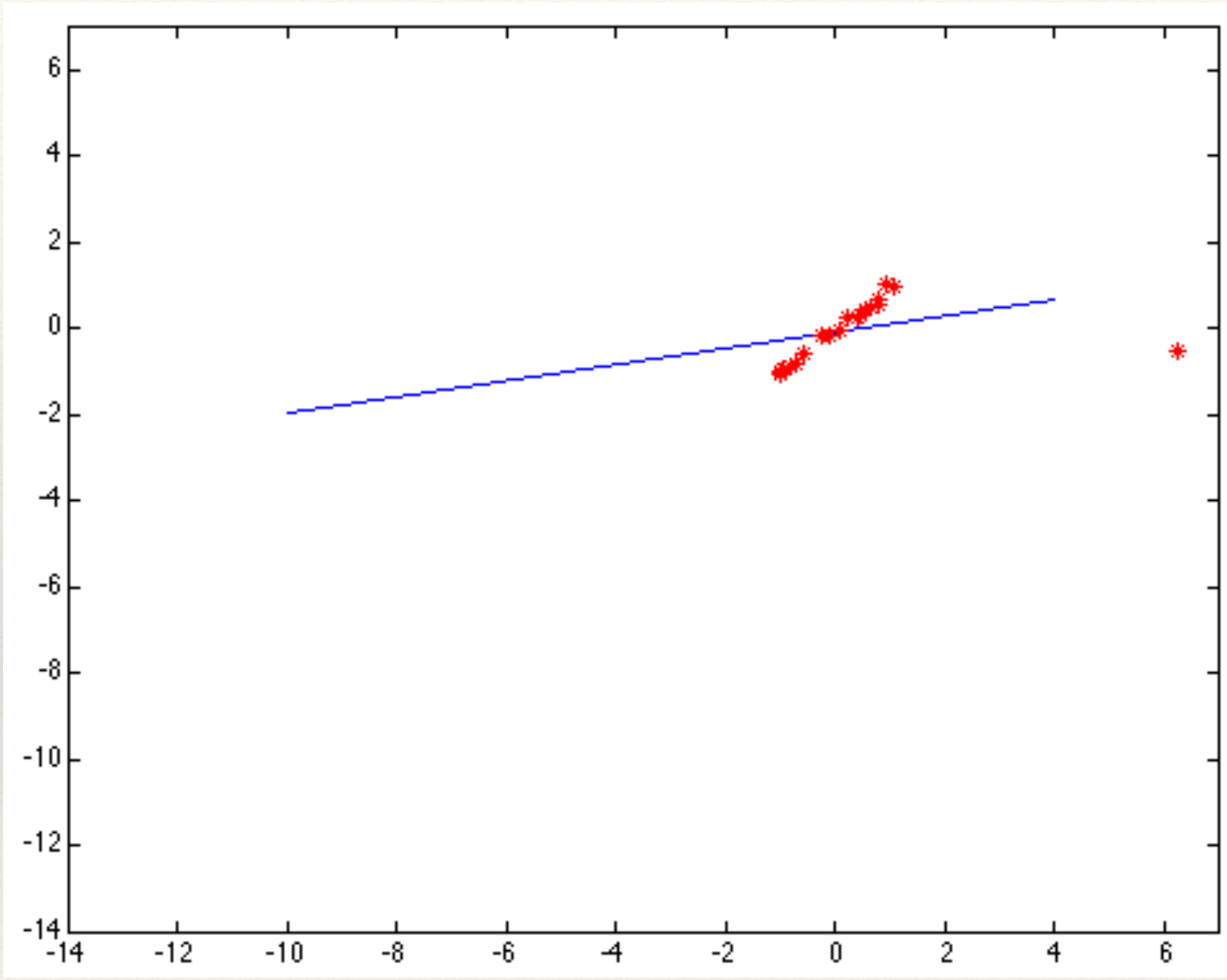
The effect of the outlier is minimized

Choosing the scale: Too small



The error value is almost the same for every point and the fit is very poor

Choosing the scale: Too large



Behaves much the same as least squares

Robust estimation

- Robust fitting is a nonlinear optimization problem that must be solved iteratively
- Scale of robust function should be chosen adaptively based on median residual
- Least squares solution can be used for initialization

RANSAC: RANdom SAmple Consensus

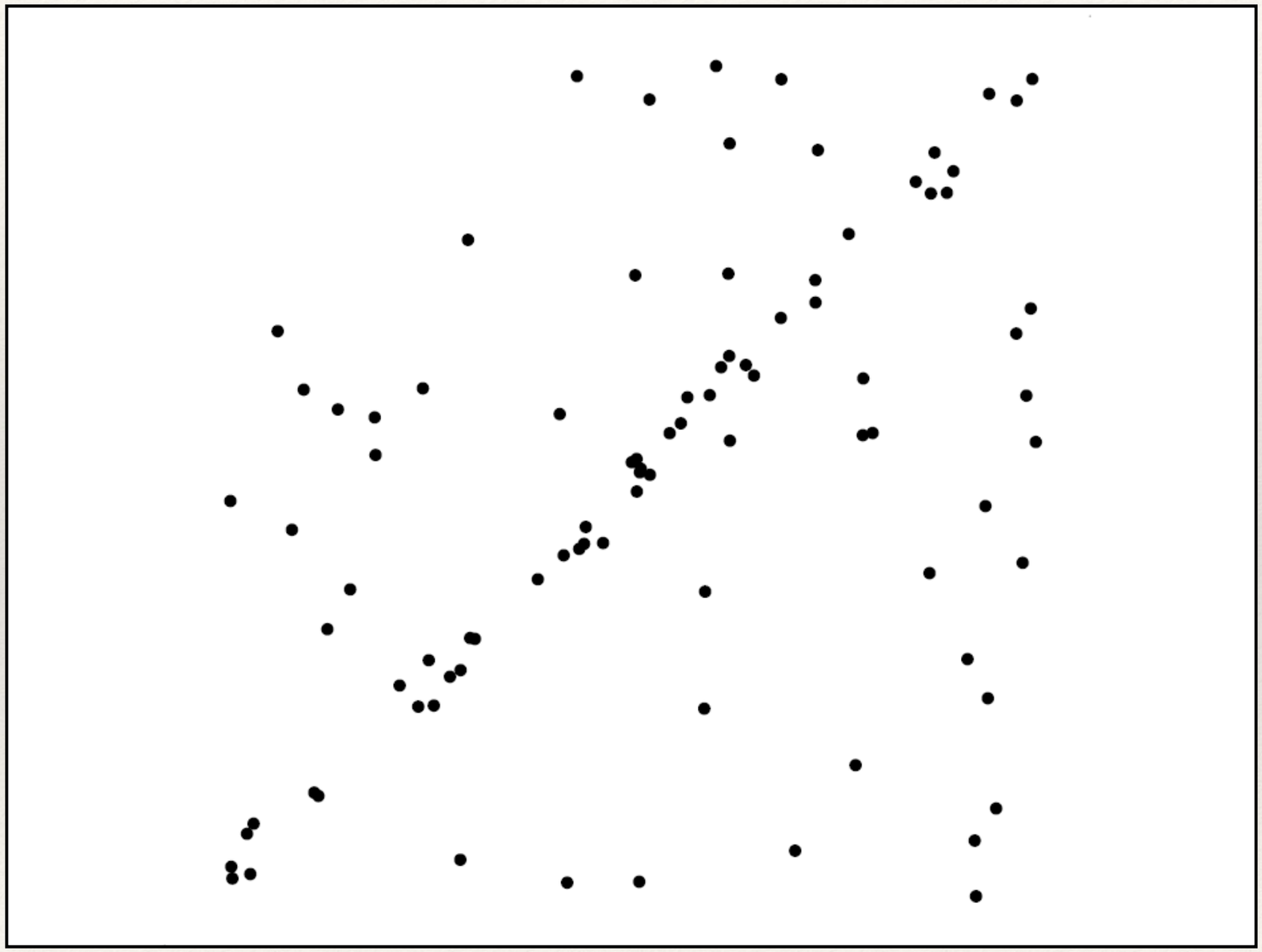
- Fischler & Bolles in ’81
- **Approach**: we want to avoid the impact of outliers, so let’s look for “inliers”, and use those only.
- **Intuition**: if an outlier is chosen to compute the current fit, then the resulting line (transformation) won’t have much support from rest of the points (matches).

Slides from Kristen Grauman

RANSAC for line fitting

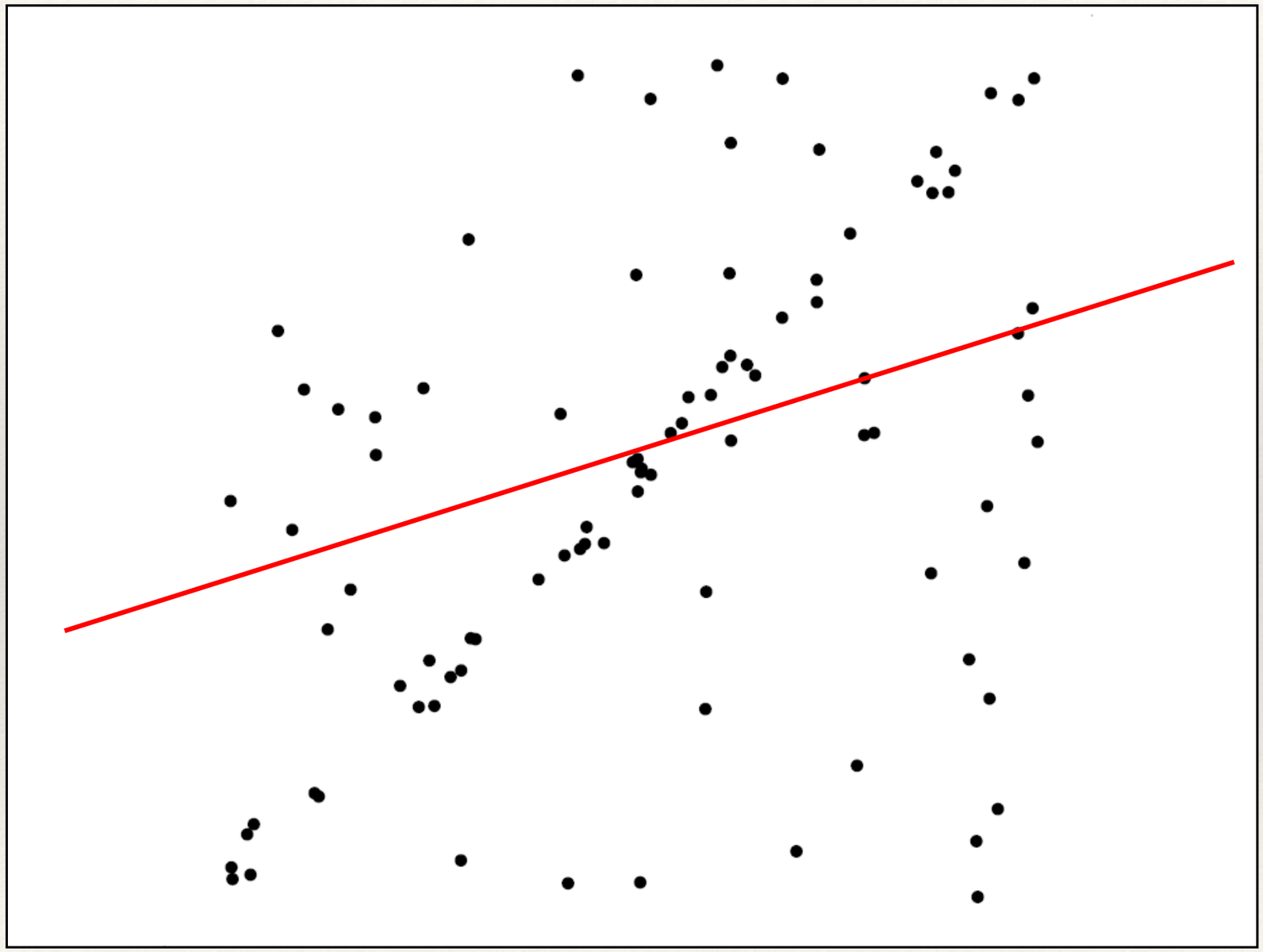
- Repeat ***N*** times:
  - Draw ***s*** points uniformly at random
  - Fit line to these ***s*** points
  - Find inliers to this line among the remaining points (i.e., points whose distance from the line is less than ***t***)
  - If there are ***d*** or more inliers, accept the line and refit using all inliers

RANSAC for line fitting example



Lana Lazebnik

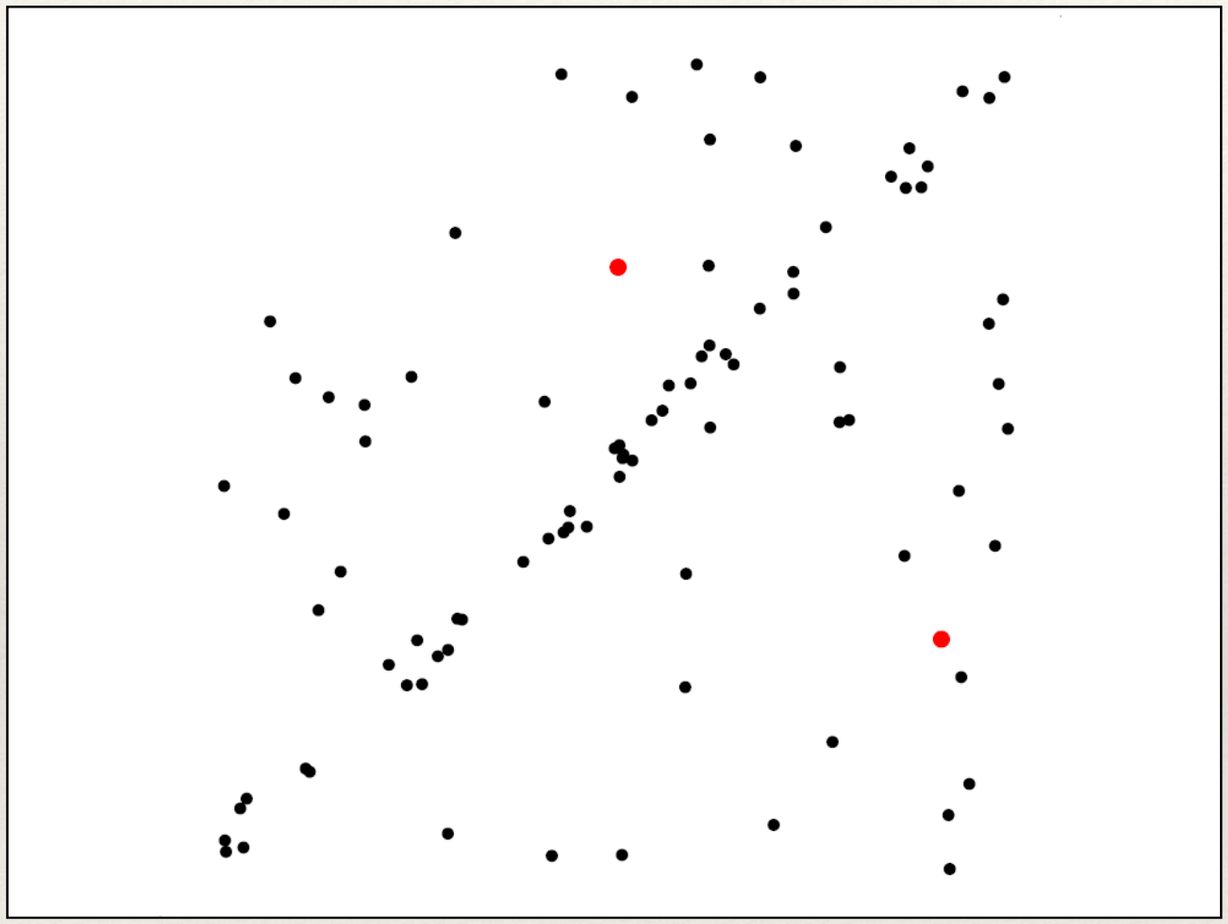
RANSAC for line fitting example



Least-squares fit

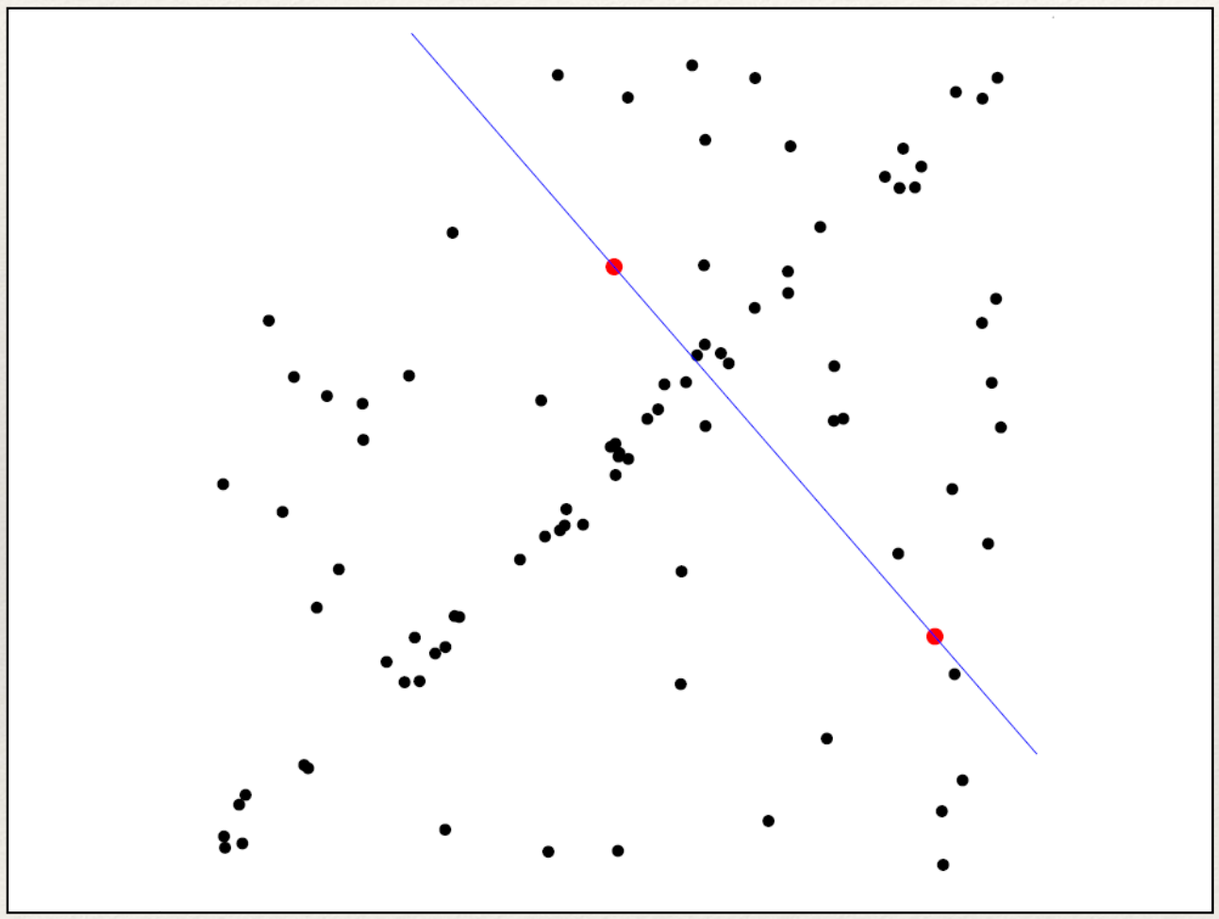
Lana Lazebnik

RANSAC for line fitting example



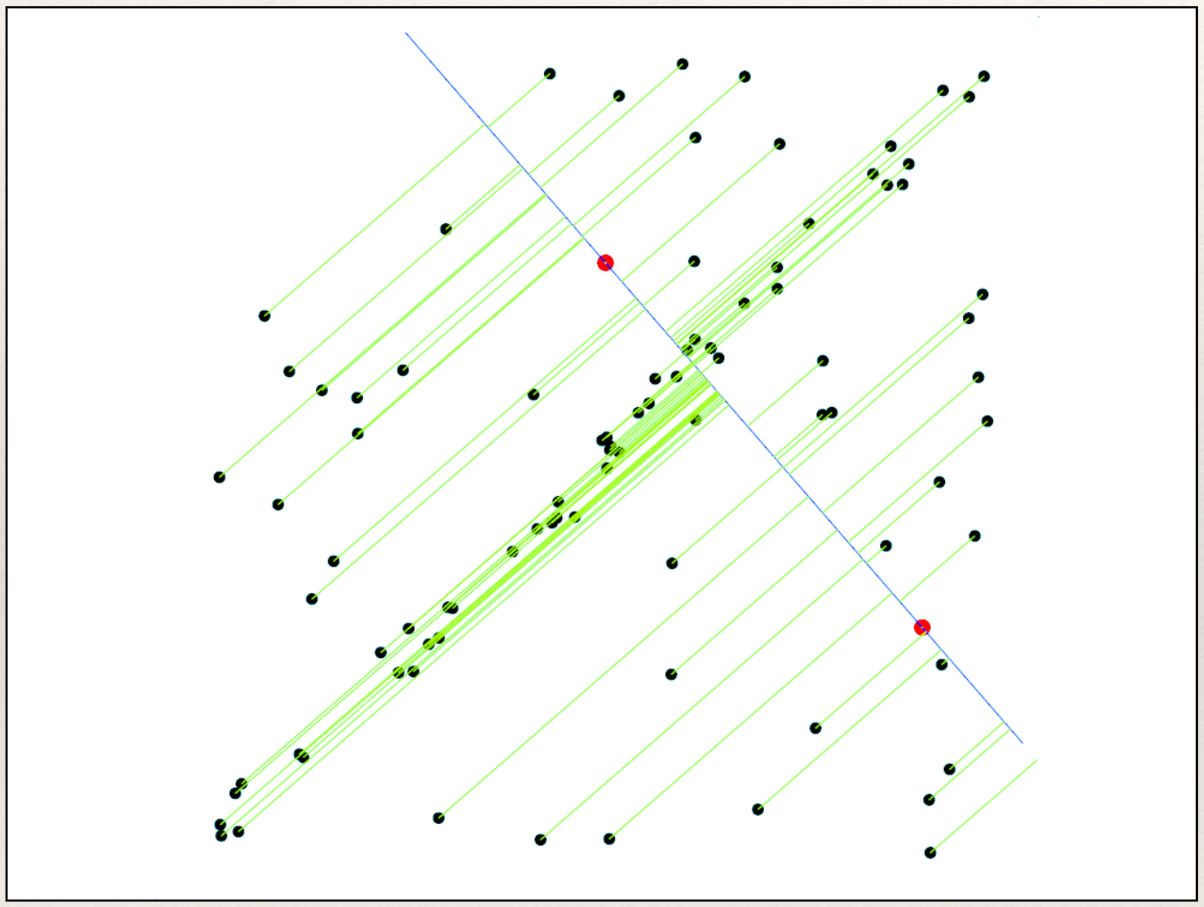
1. Randomly select minimal subset of points

RANSAC for line fitting example



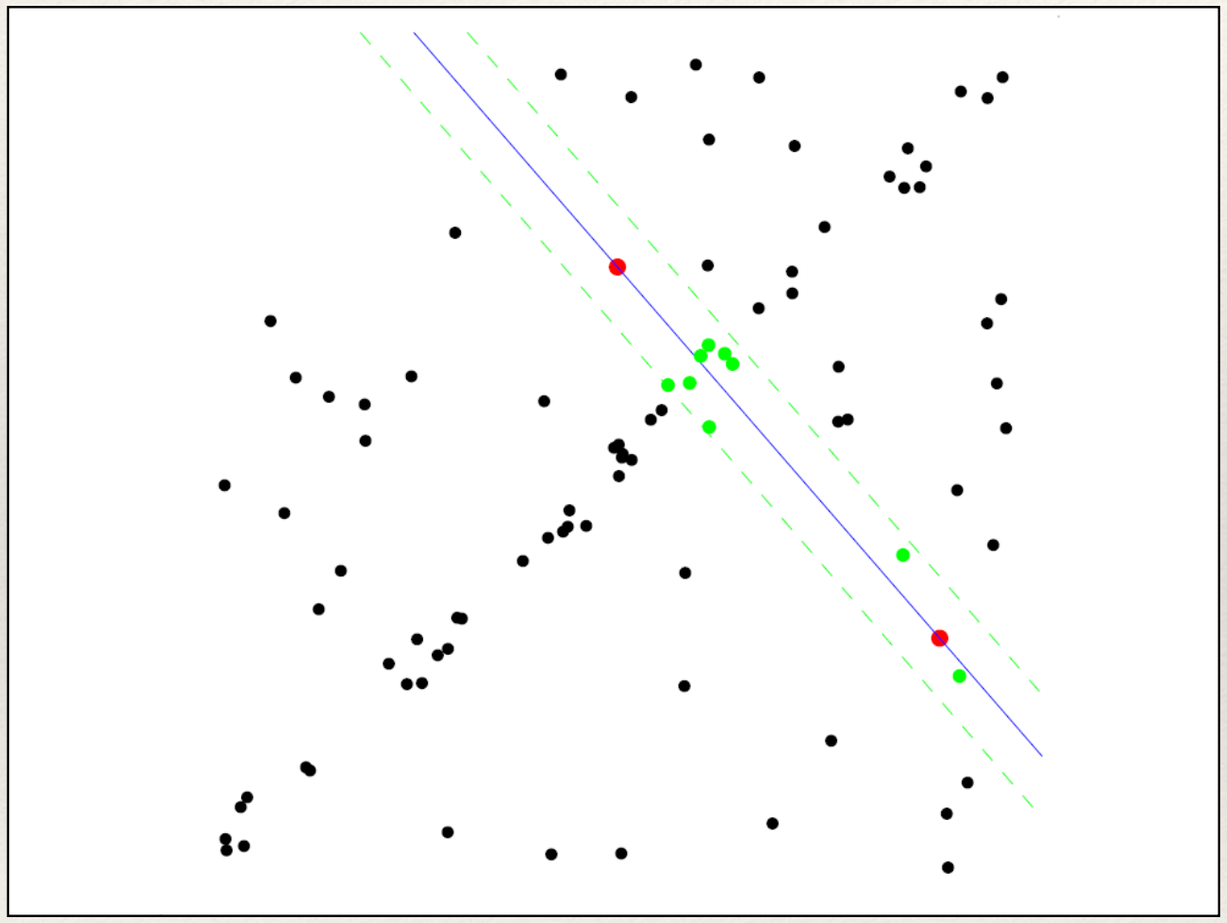
1. Randomly select minimal subset of points
2. Hypothesize a model

RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function

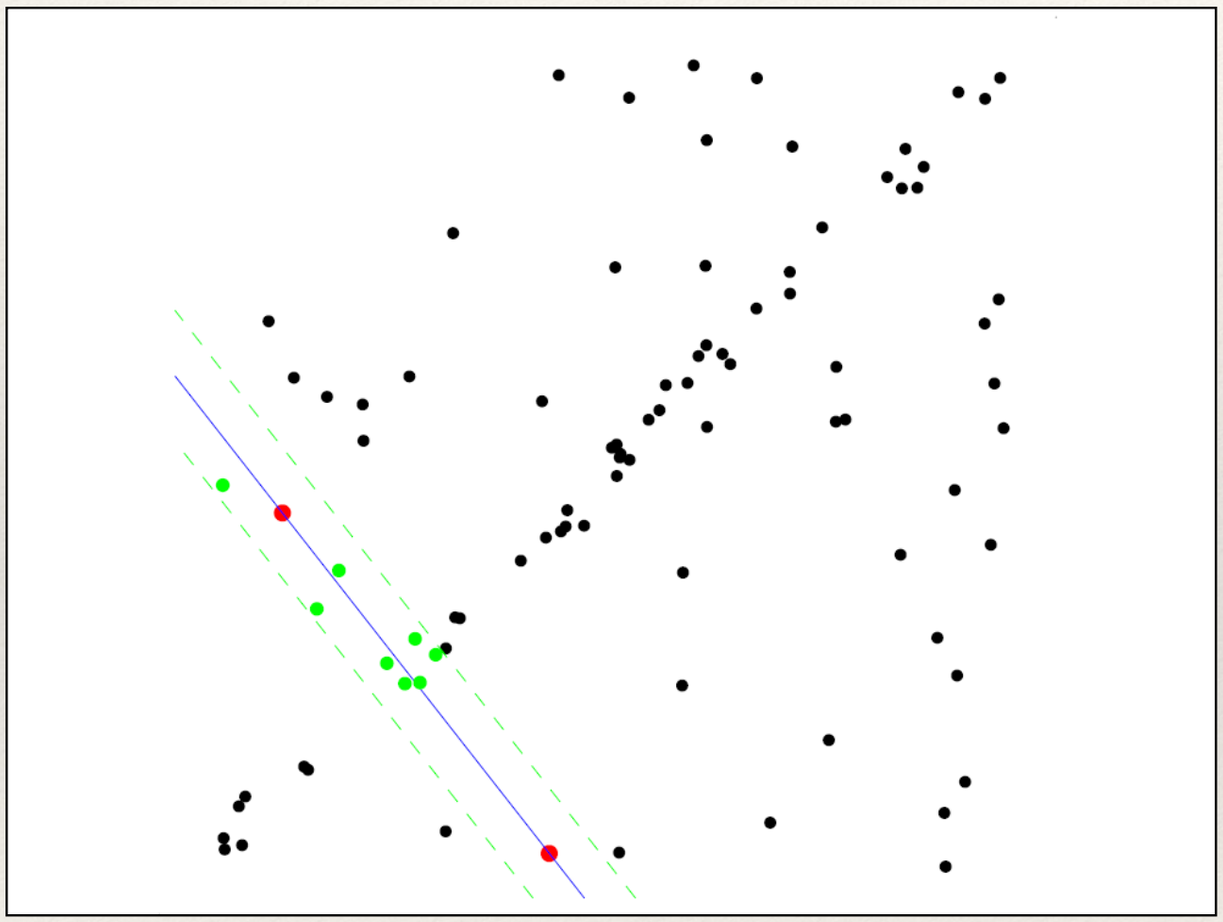
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model

Lana Lazebnik

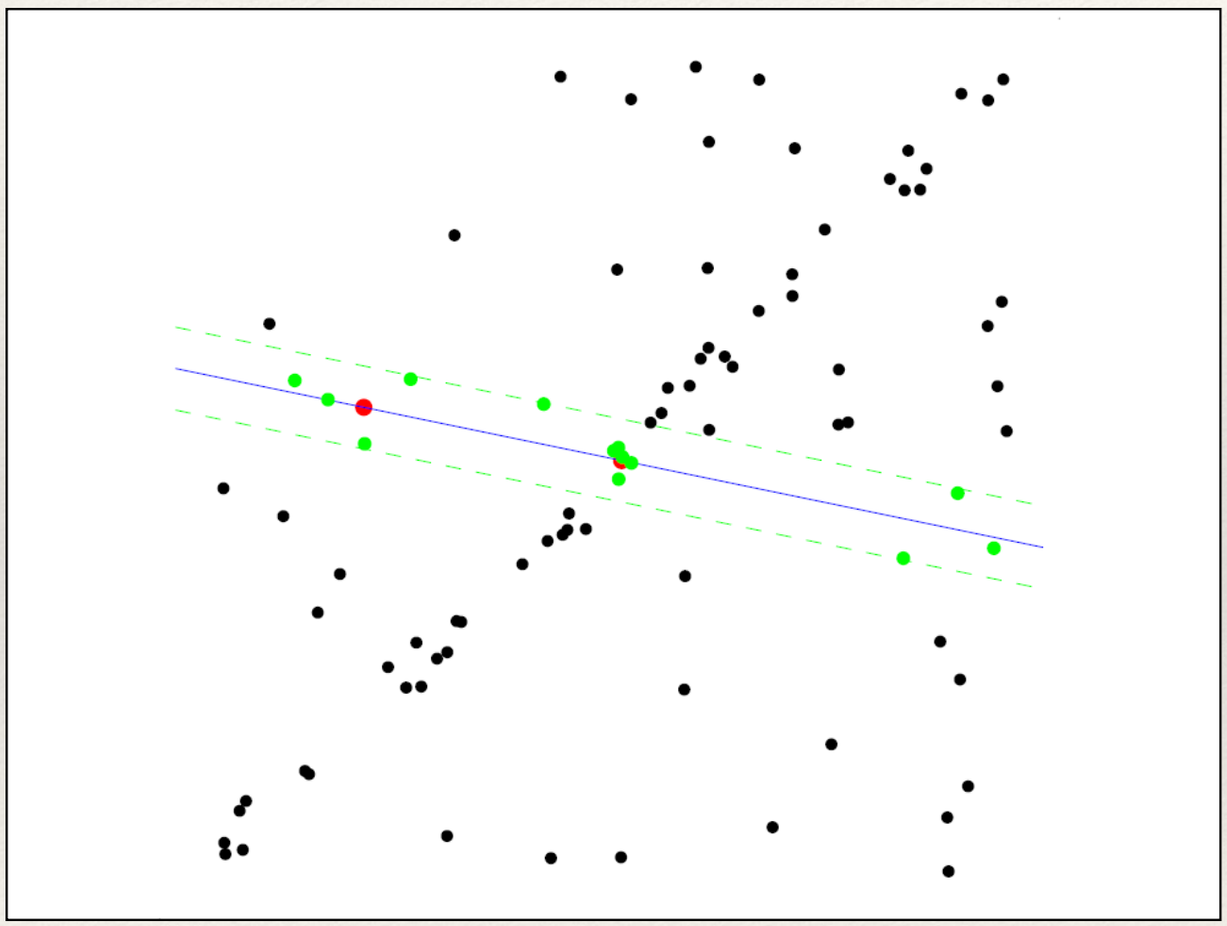
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize-and-verify loop

Lana Lazebnik

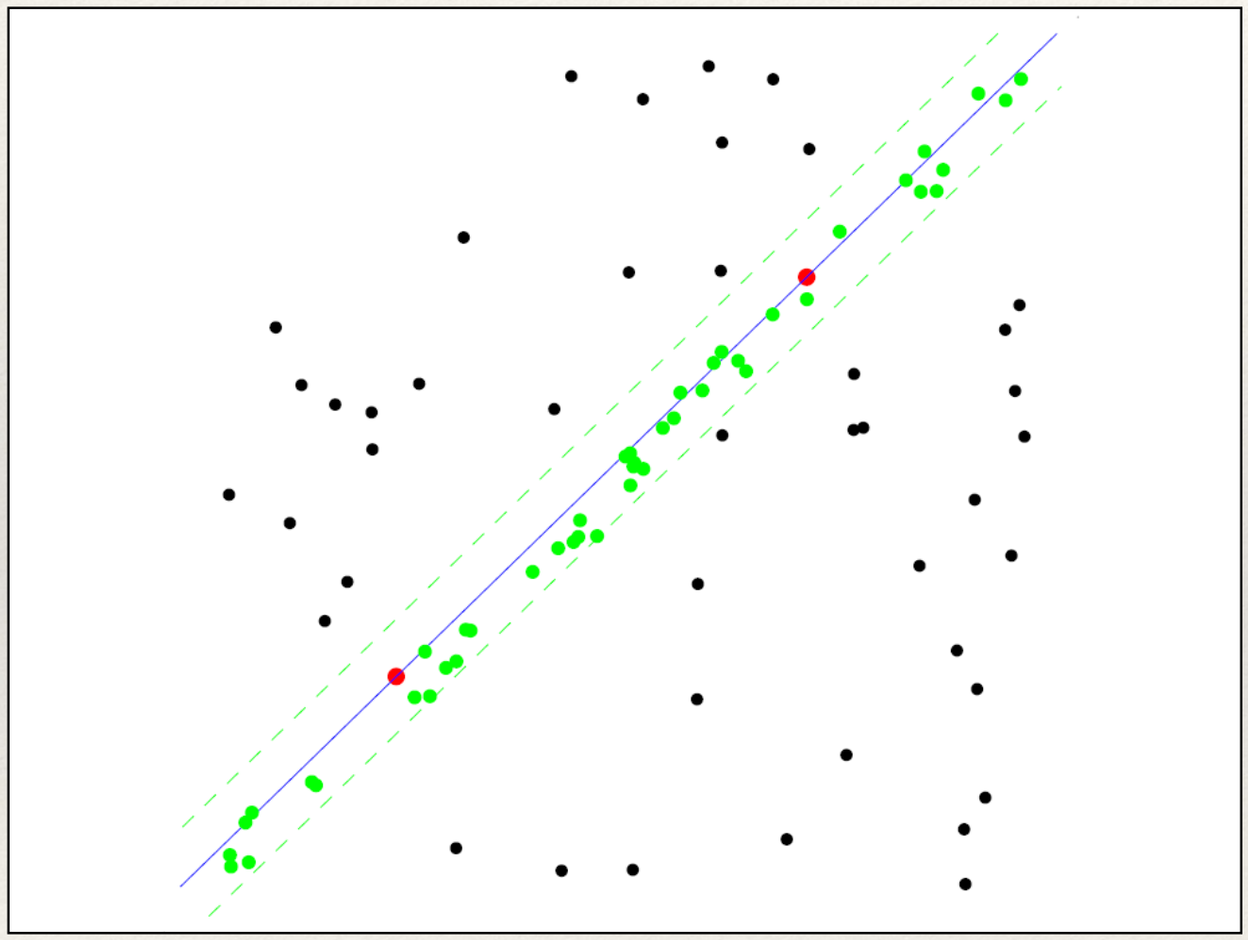
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize-and-verify loop

Lana Lazebnik

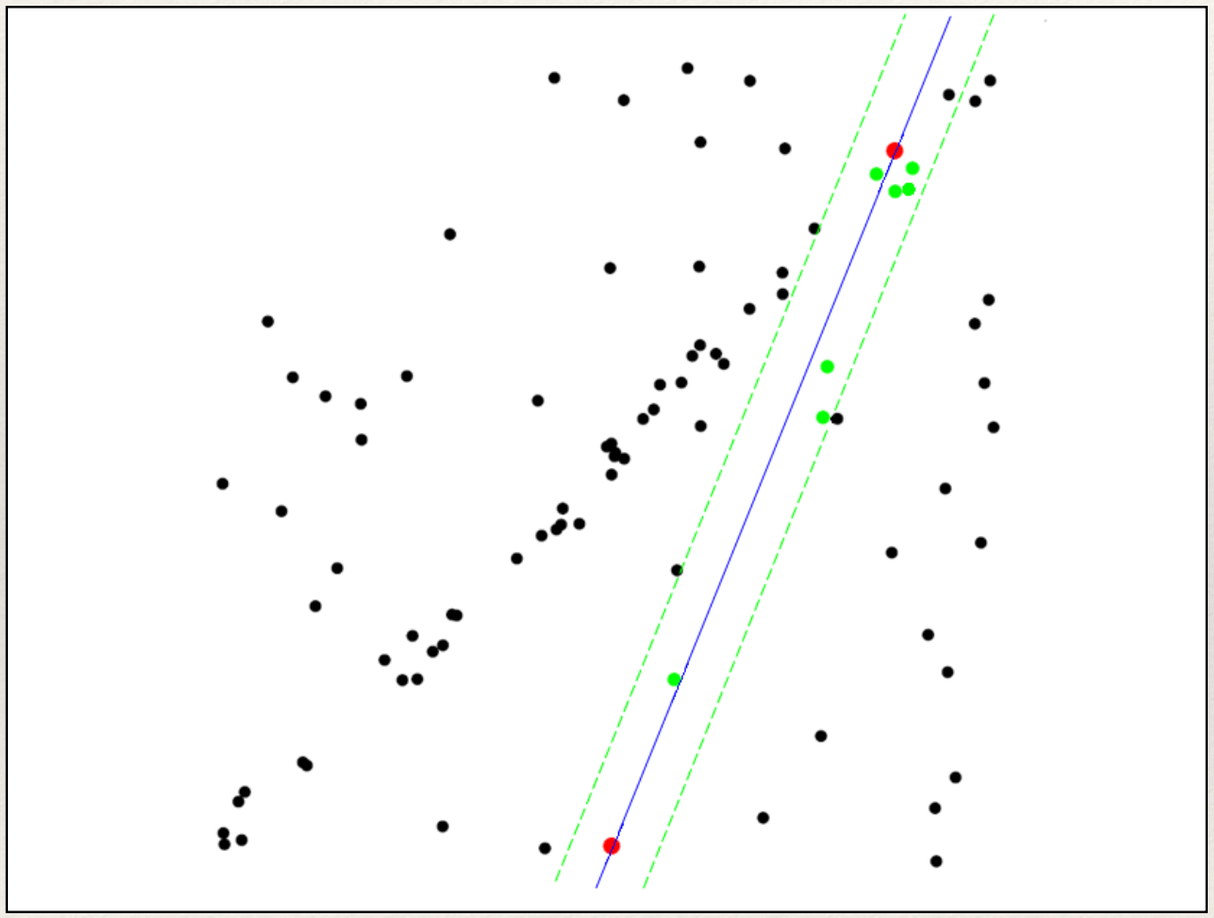
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize-and-verify loop

Lana Lazebnik

RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat hypothesize-and-verify loop

Lana Lazebnik

How many trials for RANSAC?

- To ensure good chance of finding true inliers, need sufficient number of trials, S.
- Let p be probability that any given match is valid
- Let P be to the total prob of success after S trials.
- Likelihood in one trial that all k random samples are inliers is p^k
- Likelihood that all S trials will fail is: $1-P = (1-p^k)^S$
- Required minimum number of trials is

$$S = \log(1-P) \ / \ \log(1-p^k)$$

Slides from Kristen Grauman

RANSAC song

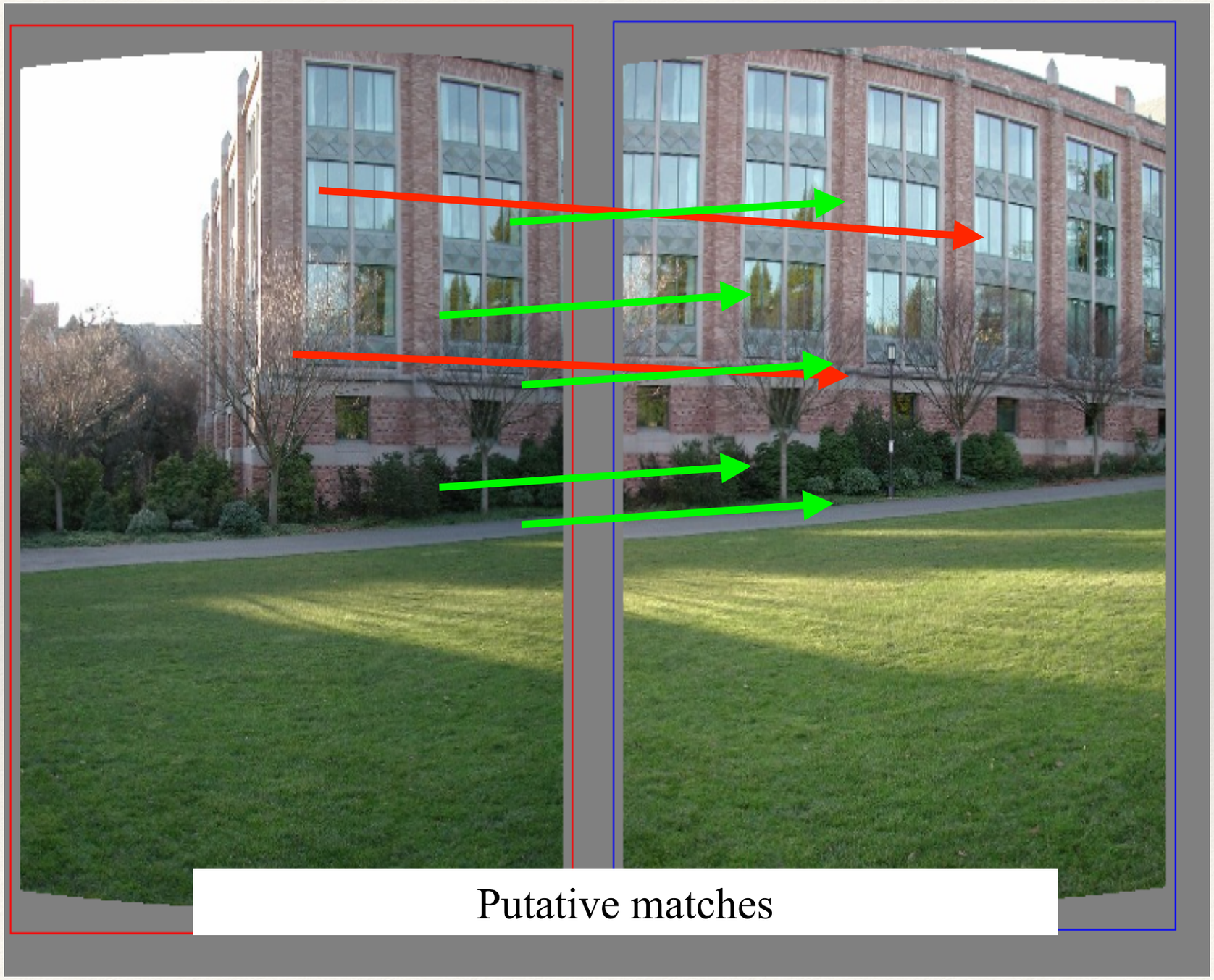
When you have outliers you may face much frustration
if you include them in a model fitting operation.
But if your model's fit to a sample set of minimal size,
the probability of the set being outlier-free will rise.
Brute force tests of all sets will cause computational constipation.
N random samples
will provide an example
of a fitted model uninfluenced by outliers. No need to test all combinations!
Each random trial should have its own unique sample set
and make sure that the sets you choose are not degenerate.
N, the number of sets, to choose is based on the probability
of a point being an outlier, and of finding a set that's outlier free.
Updating *N* as you go will minimise the time spent.
So if you gamble
that *N* samples are ample
to fit a model to your set of points, it's likely that you will win the bet.
Select the set that boasts
that its number of inliers is the most (you're almost there).
Fit a new model just to those inliers and discard the rest,
an estimated model for your data is now possessed!
This marks the end point of your model fitting quest

RANSAC: General form

- RANSAC loop:
  1. Randomly select a *seed group* on which to base transformation estimate (e.g., a group of matches)
  2. Compute transformation from seed group
  3. Find *inliers* to this transformation
  4. If the number of inliers is sufficiently large, re-compute estimate of transformation on all of the inliers
- Keep the transformation with the largest number of inliers

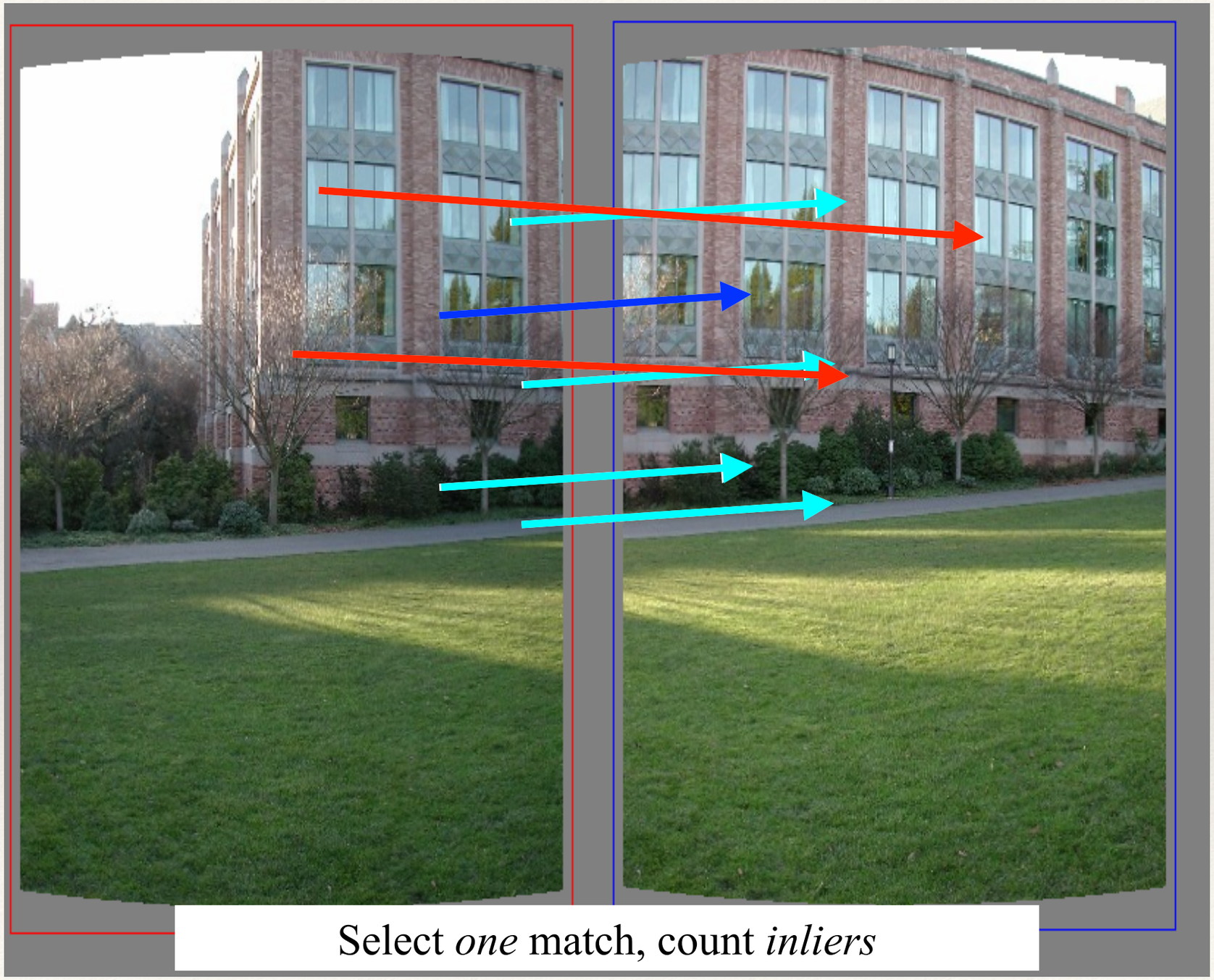
Slides from Kristen Grauman

RANSAC example: Translation



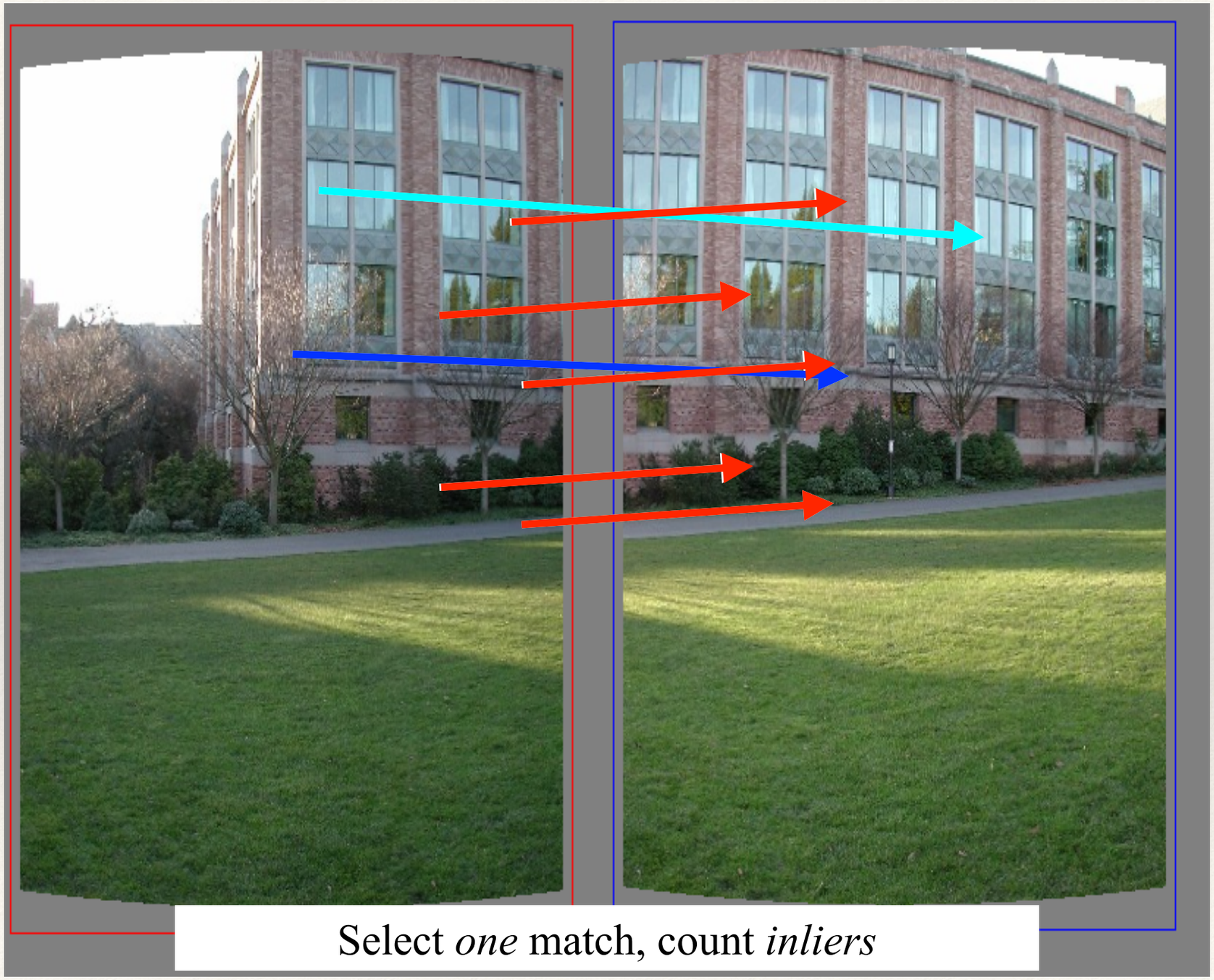
Source: Rick Szeliski

RANSAC example: Translation



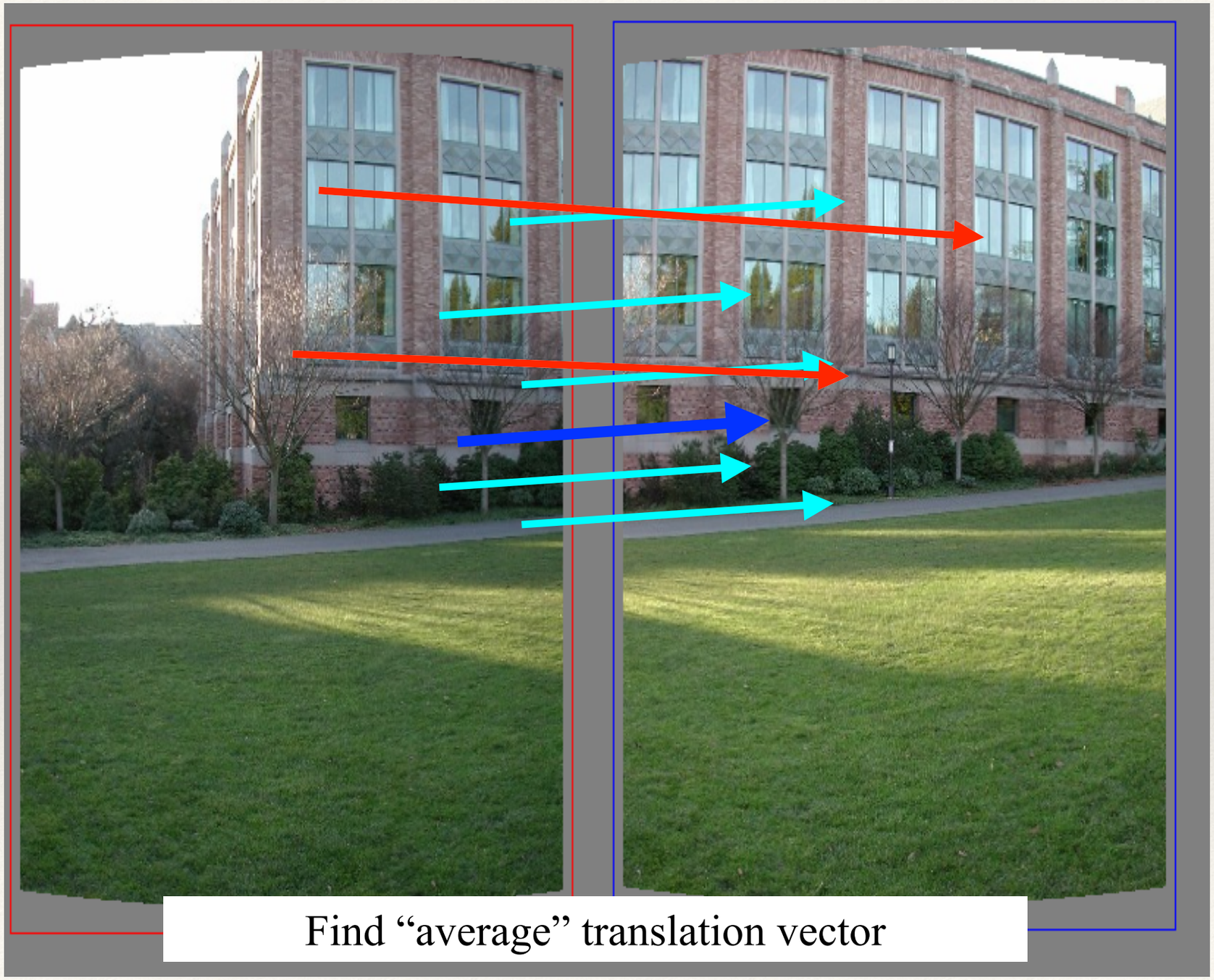
Source: Rick Szeliski

RANSAC example: Translation



Source: Rick Szeliski

RANSAC example: Translation

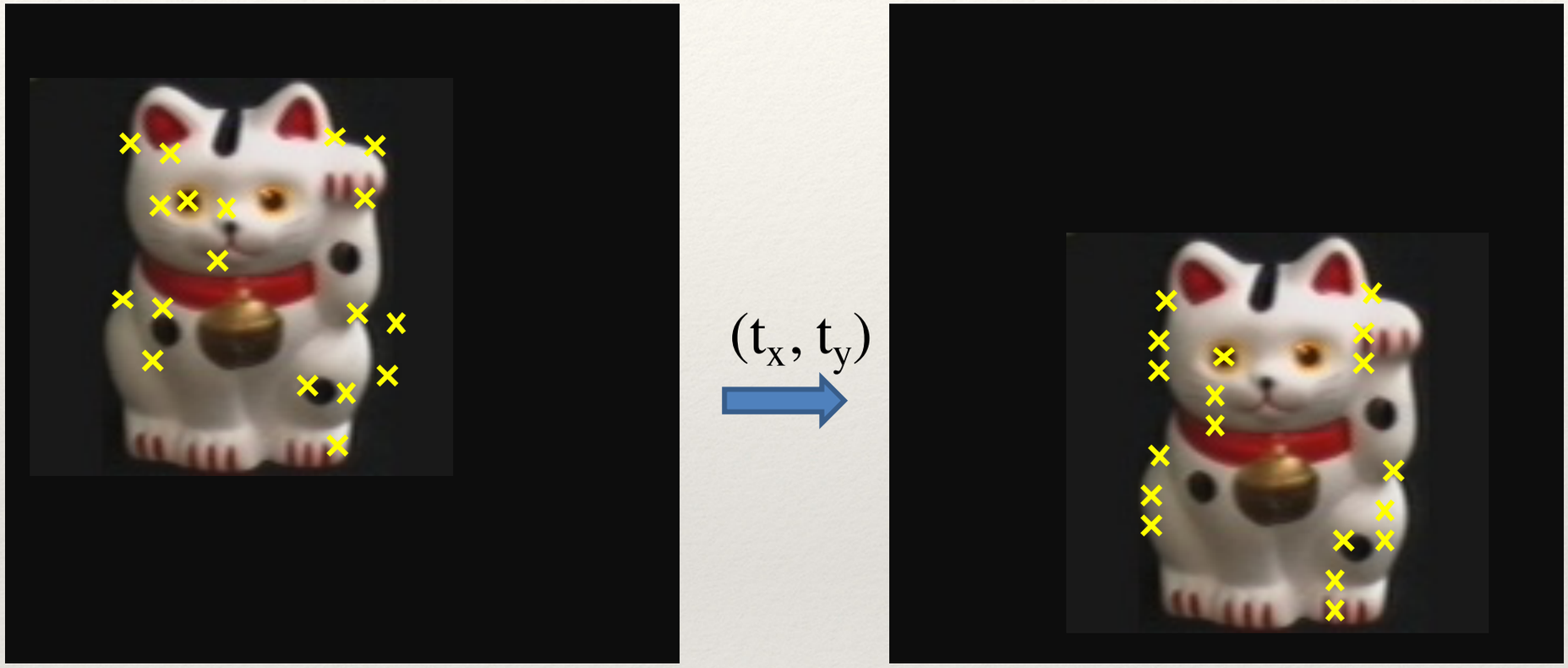


Source: Rick Szeliski

RANSAC pros and cons

- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Parameters to tune
  - Doesn’t work well for low inlier ratios (too many iterations, or can fail completely)
  - Can’t always get a good initialization of the model based on the minimum number of samples

What if we want to align, but we have no matched pairs?



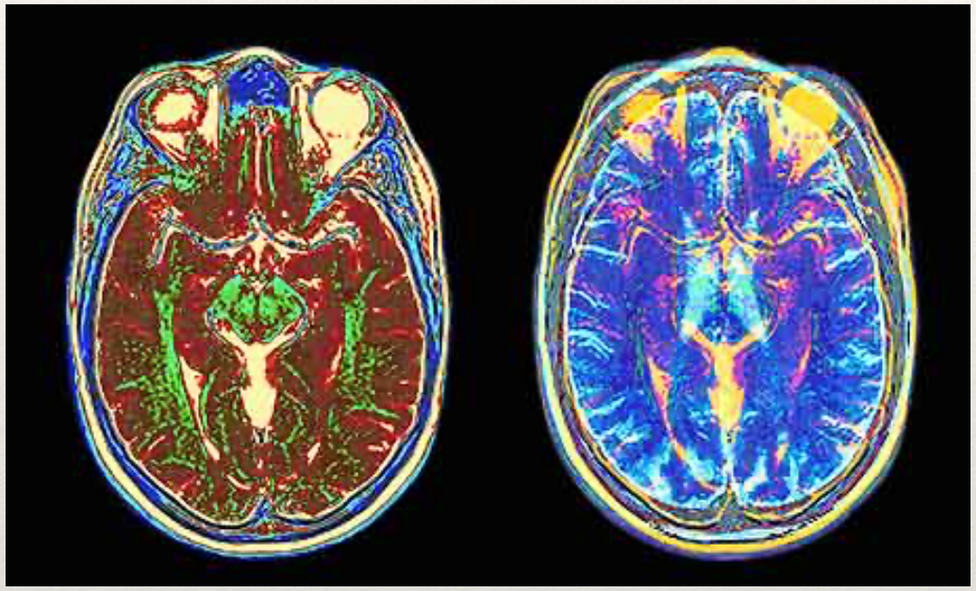
Problem: no initial guesses for correspondence

Iterative Closest Points (ICP) Algorithm

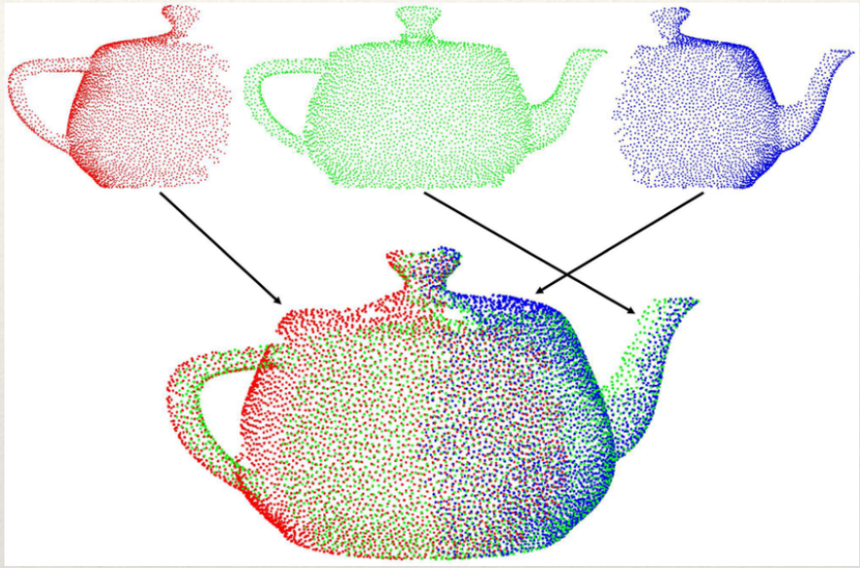
- Goal:
  - Estimate transform between two dense point sets S_1 and S_2

1. **Initialize** transformation
   1. Compute difference in mean positions, subtract
   2. Compute difference in scales, normalize
2. **Assign** each point in S_1 to its nearest neighbor in S_2
3. **Estimate** transformation parameters T
   1. Least squares or robust least squares, e.g., rigid transform
4. **Transform** the points in S_1 using estimated parameters T
5. **Repeat** steps 2-4 until change is very small (convergence)

Applications

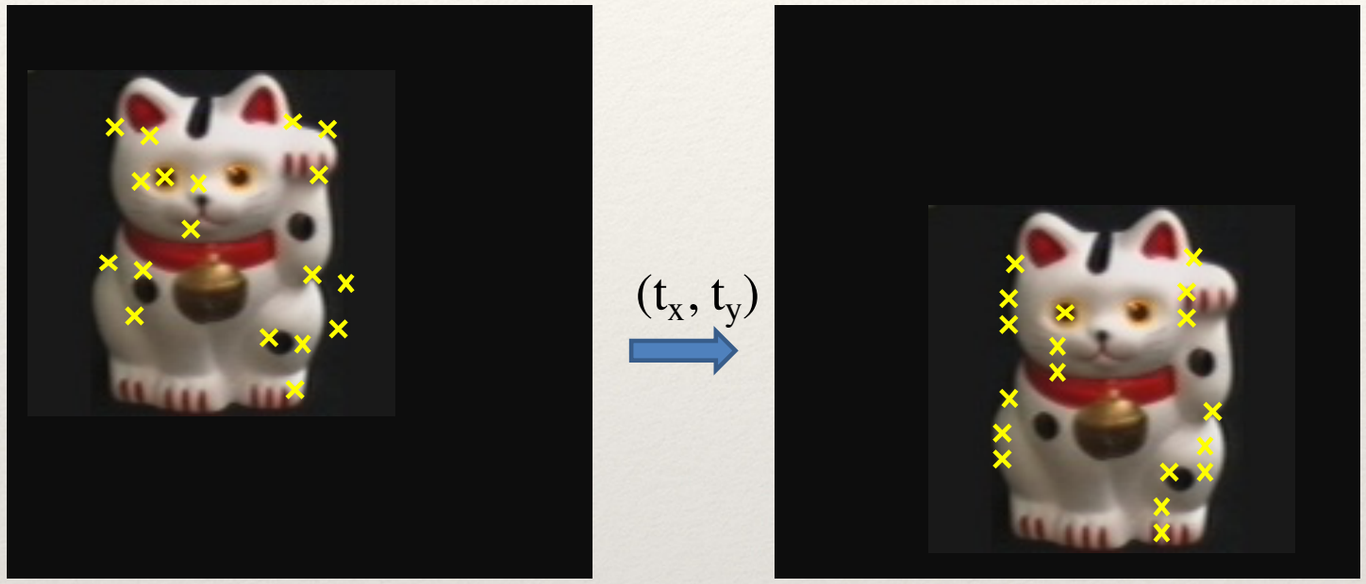


Medical imaging: match brain scans or contours



Vision/Robotics: match point clouds

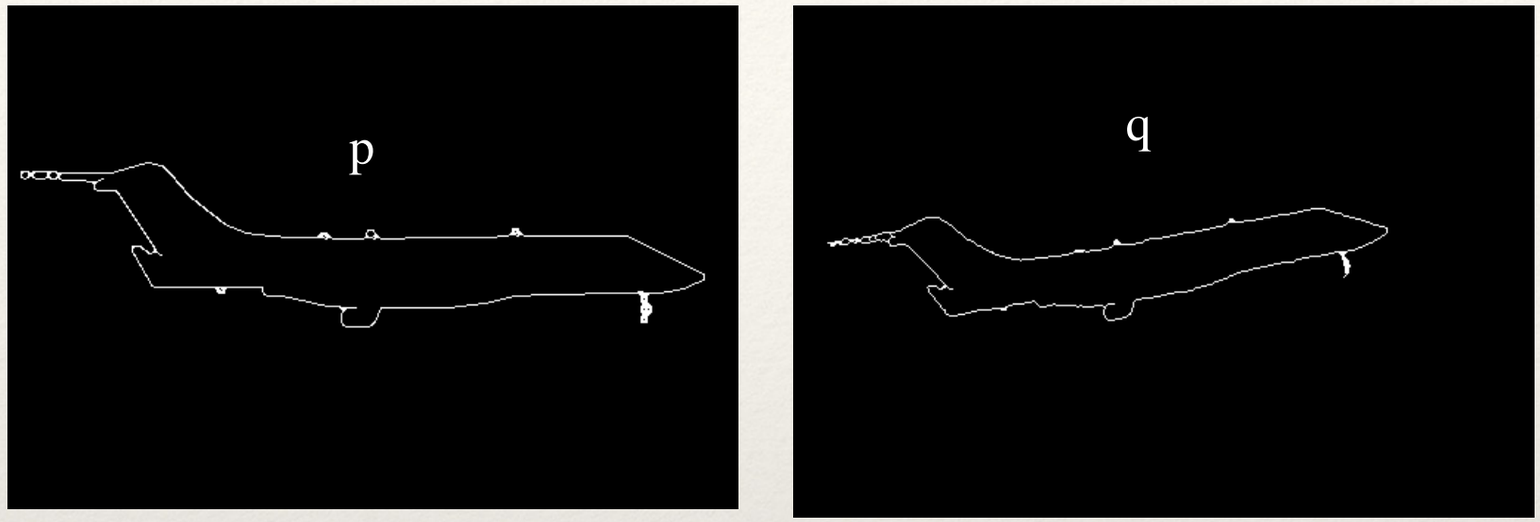
Example: solving for translation



Problem: no initial guesses for correspondence

- ICP solution
  1. Initialize *t* by mean point translation
  2. Find nearest neighbors for each point
  3. Compute transform using matches
  4. Move points using transform
  5. Repeat steps 2-4 until convergence

Example: aligning boundaries



- Extract edge pixels $p_1..p_n$ and $q_1..q_m$
- Compute initial transformation (e.g., compute translation and scaling by center of mass, variance within each image)
- Get nearest neighbors: for each point p_i find corresponding $\text{match}(i) = \underset{j}{\text{argmin}}\ \text{dist}(p_i, q_j)$
- Compute transformation ***T*** based on matches
- Transform points ***p*** according to ***T***
- Repeat 3-5 until convergence

Sparse ICP

Sofien Bouaziz *Andrea Tagliasacchi* *Mark Pauly*

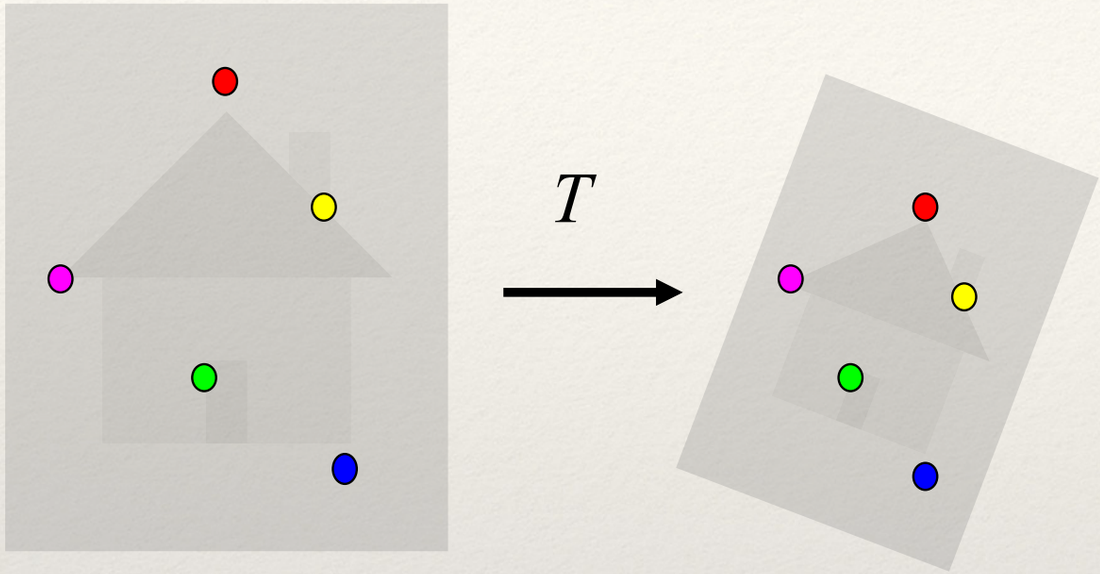
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Image mosaics

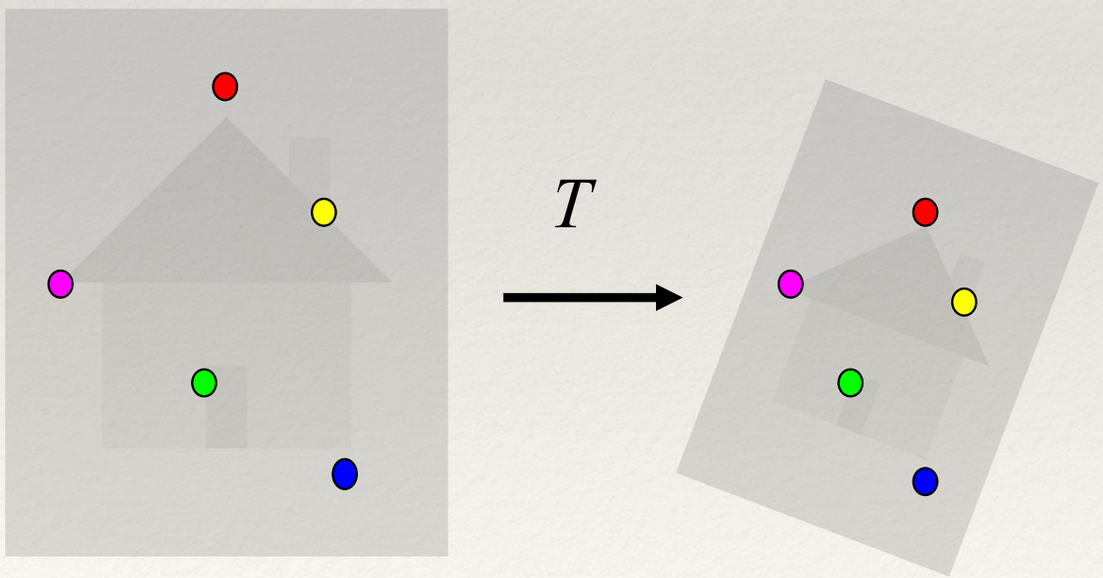
image from S. Seitz

Obtain a wider angle view by combining multiple images.

Main questions



Regisration: Given two images, what is the transformation between them?

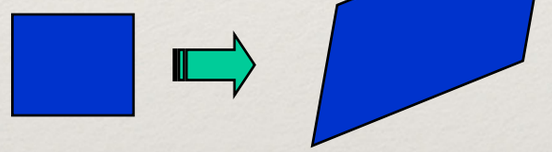


Warping: Given a source image and a transformation, what does the transformed output look like?

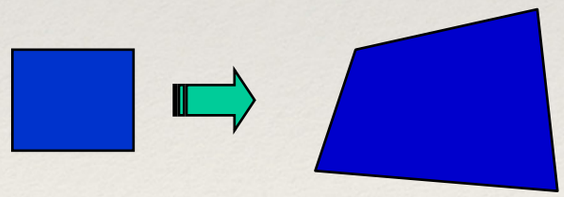
Recap: 2D transformation models

- Similarity (translation, scale, rotation)

- Affine



- Projective (homography)

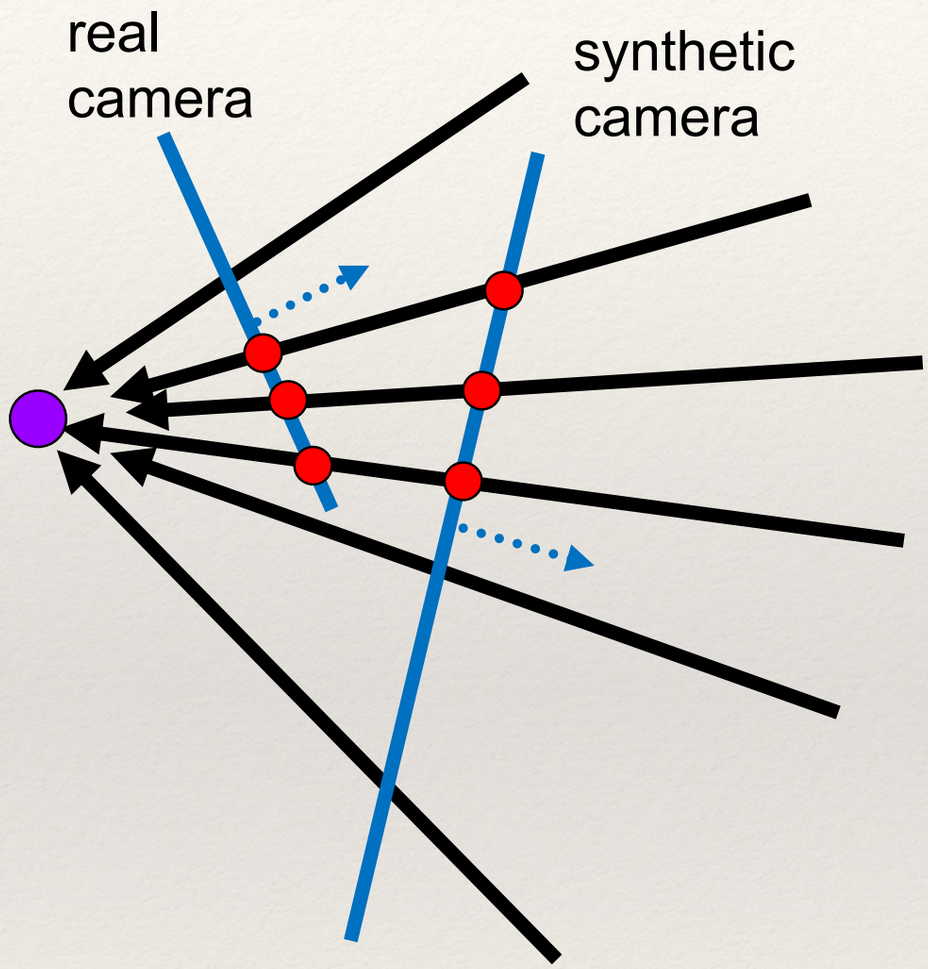


How to stitch together a panorama

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

Source: Steve Seitz

Mosaics: generating synthetic views

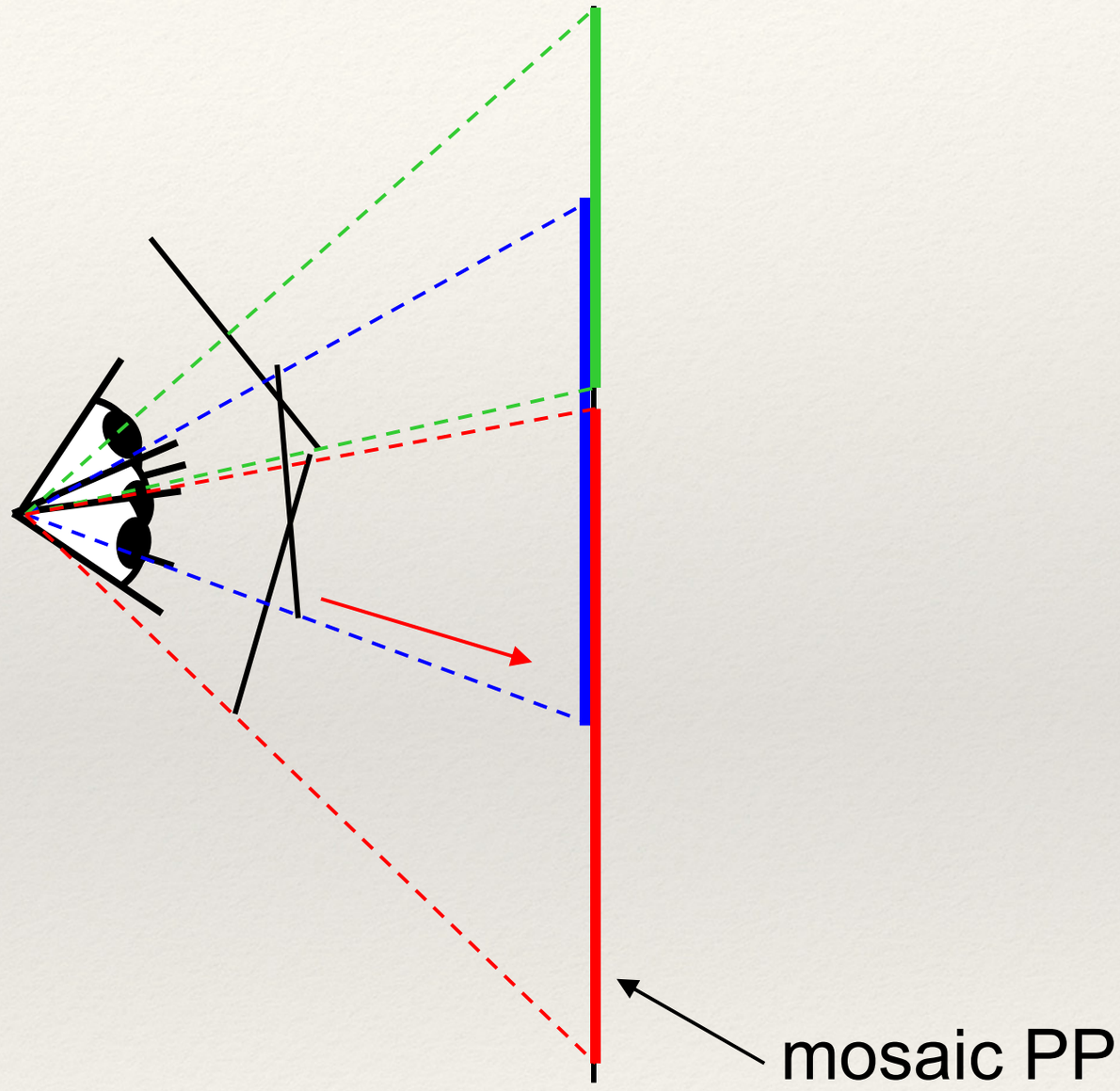


Can generate any synthetic camera view
as long as it has **the same center of projection**!

Source: Alyosha Efros

Image reprojection

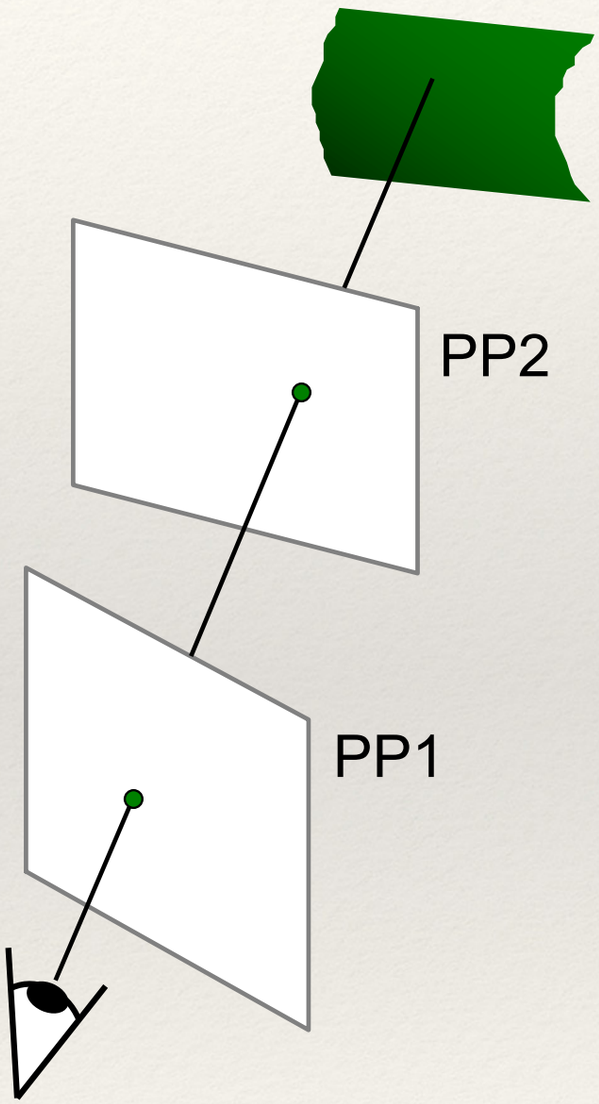
- The mosaic has a natural interpretation in 3D
  - The images are reprojected onto a common plane
  - The mosaic is formed on this plane
  - Mosaic is a synthetic wide-angle camera



Source: Steve Seitz

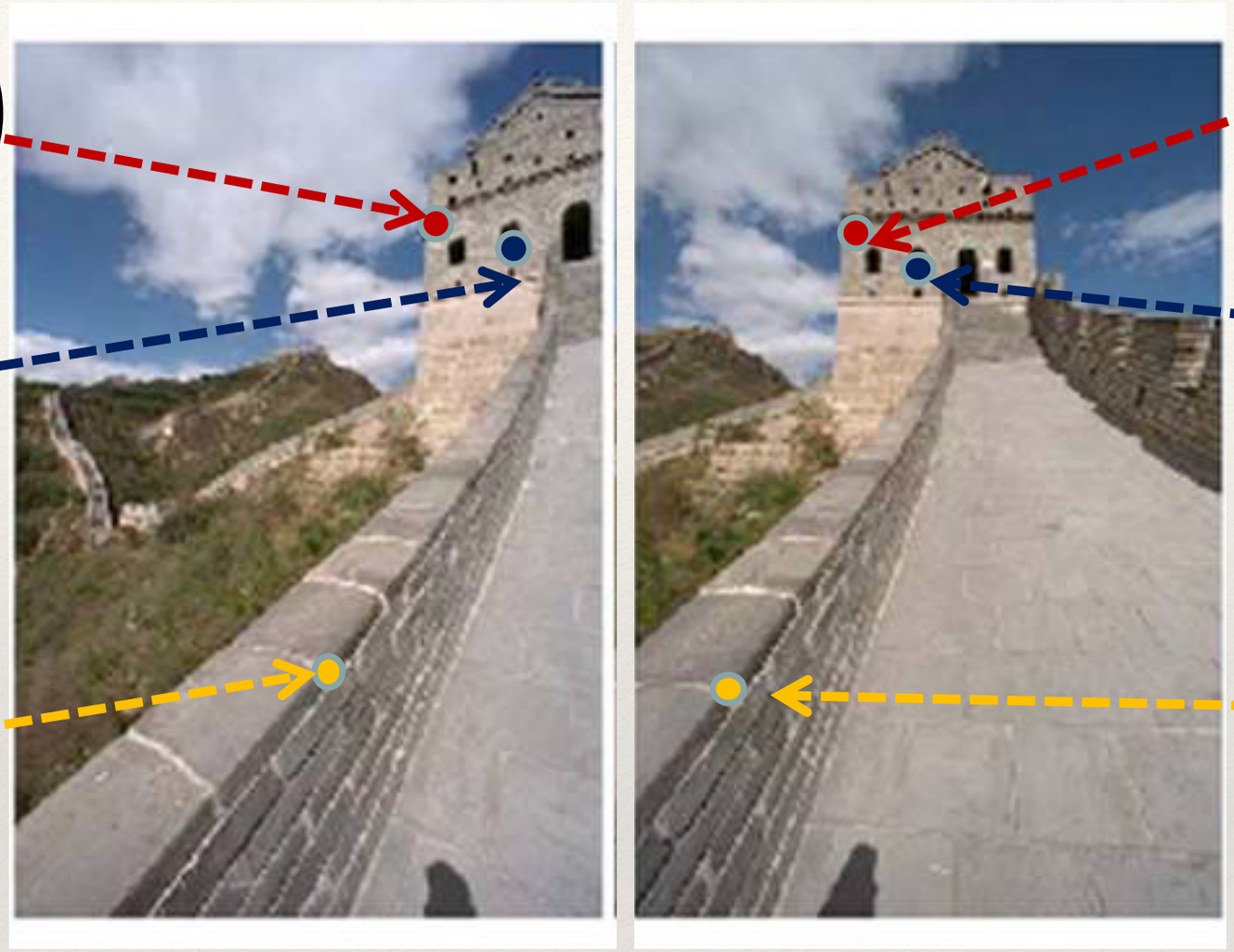
Image reprojection

- Basic question
  - How to relate two images from the same camera center?
    - how to map a pixel from PP1 to PP2
- Answer
  - Cast a ray through each pixel in PP1
  - Draw the pixel where that ray intersects PP2



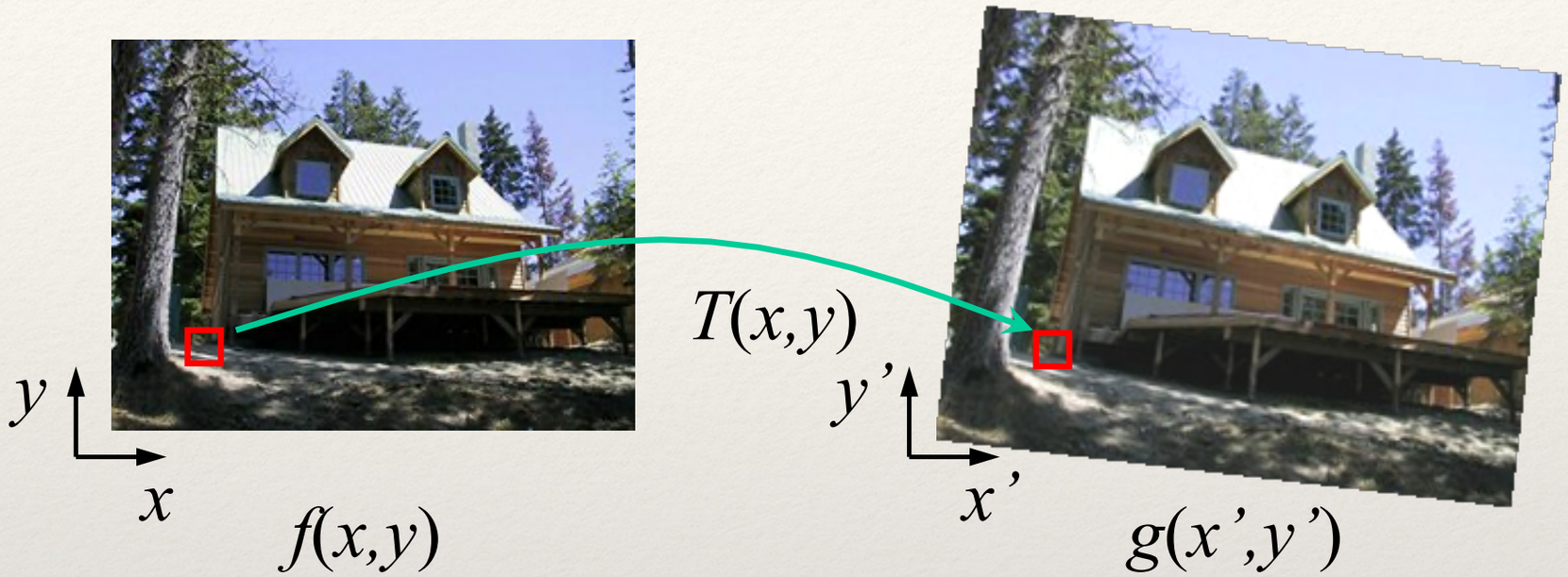
Source: Steve Seitz

Homography



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

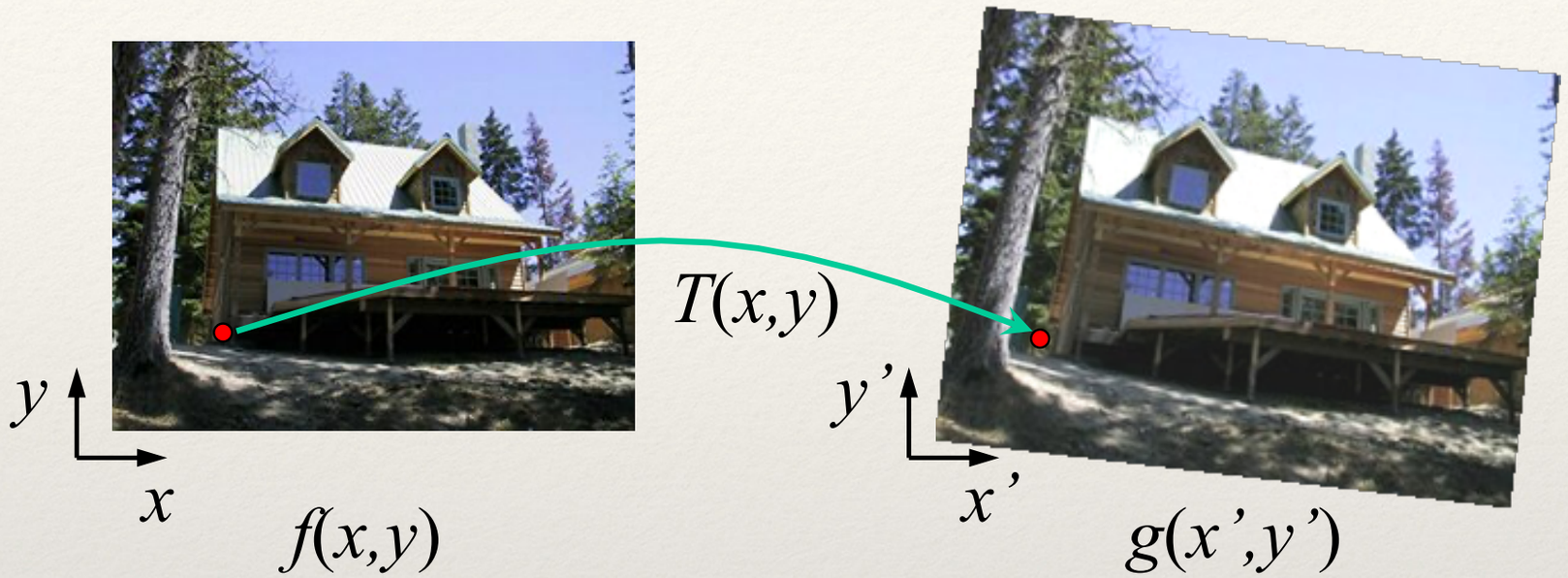
Image warping



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

Slide from Alyosha Efros, CMU

Forward warping



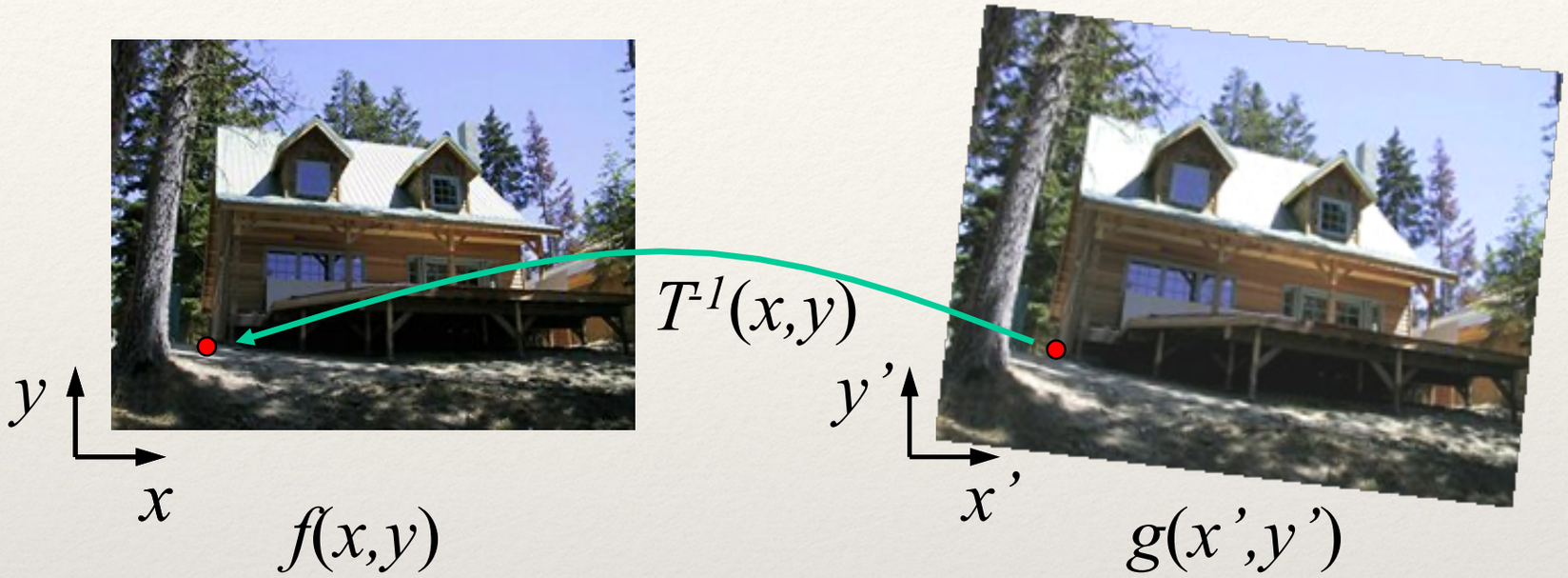
Send each pixel $f(x,y)$ to its corresponding location

$(x',y') = T(x,y)$ in the second image

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

Slide from Alyosha Efros, CMU

Inverse warping



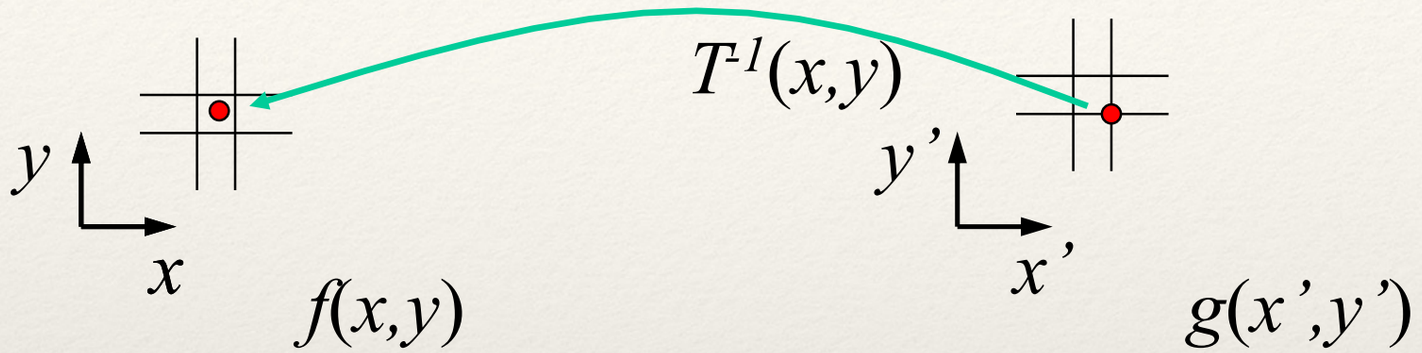
Get each pixel $g(x',y')$ from its corresponding location

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

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

Slide from Alyosha Efros, CMU

Inverse warping



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

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

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

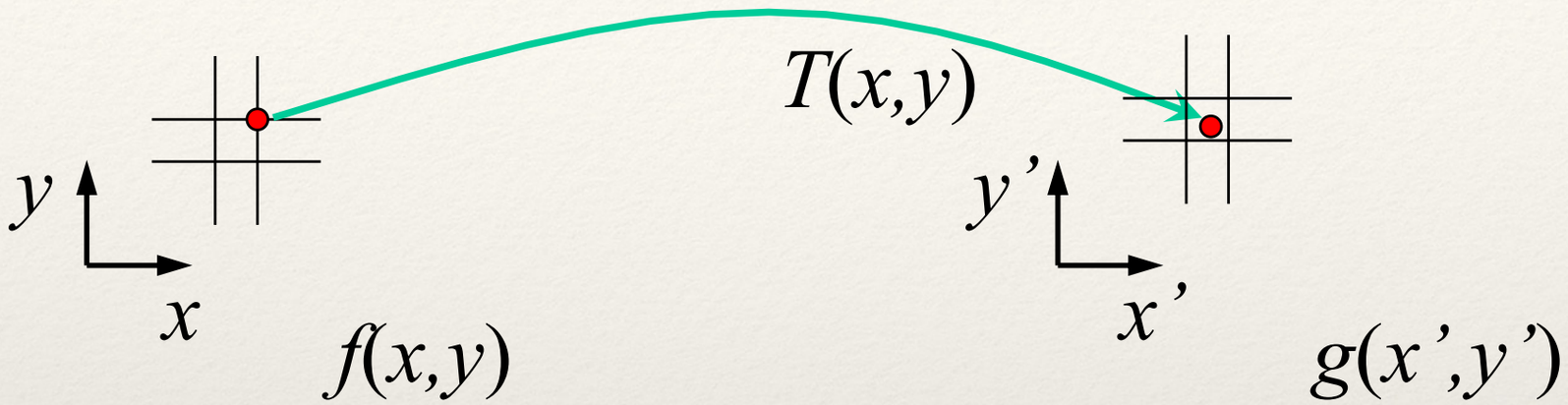
A: *Interpolate* color value from neighbors

– nearest neighbor, bilinear…

`>> help interp2`

Slide from Alyosha Efros, CMU

Forward warping



Send each pixel $f(x,y)$ to its corresponding location

$(x',y') = T(x,y)$ in the second image

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

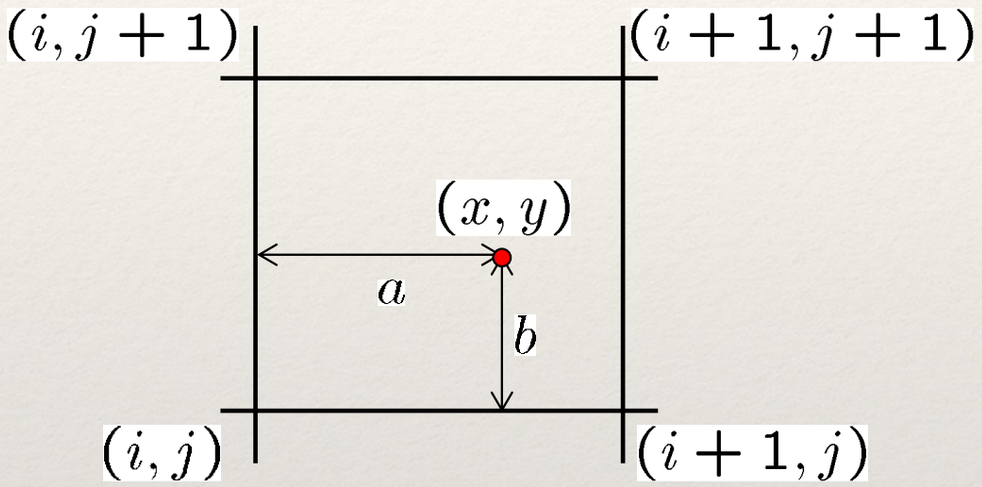
A: distribute color among neighboring pixels (x',y')

– Known as "splatting"

Slide from Alyosha Efros, CMU

Bilinear interpolation

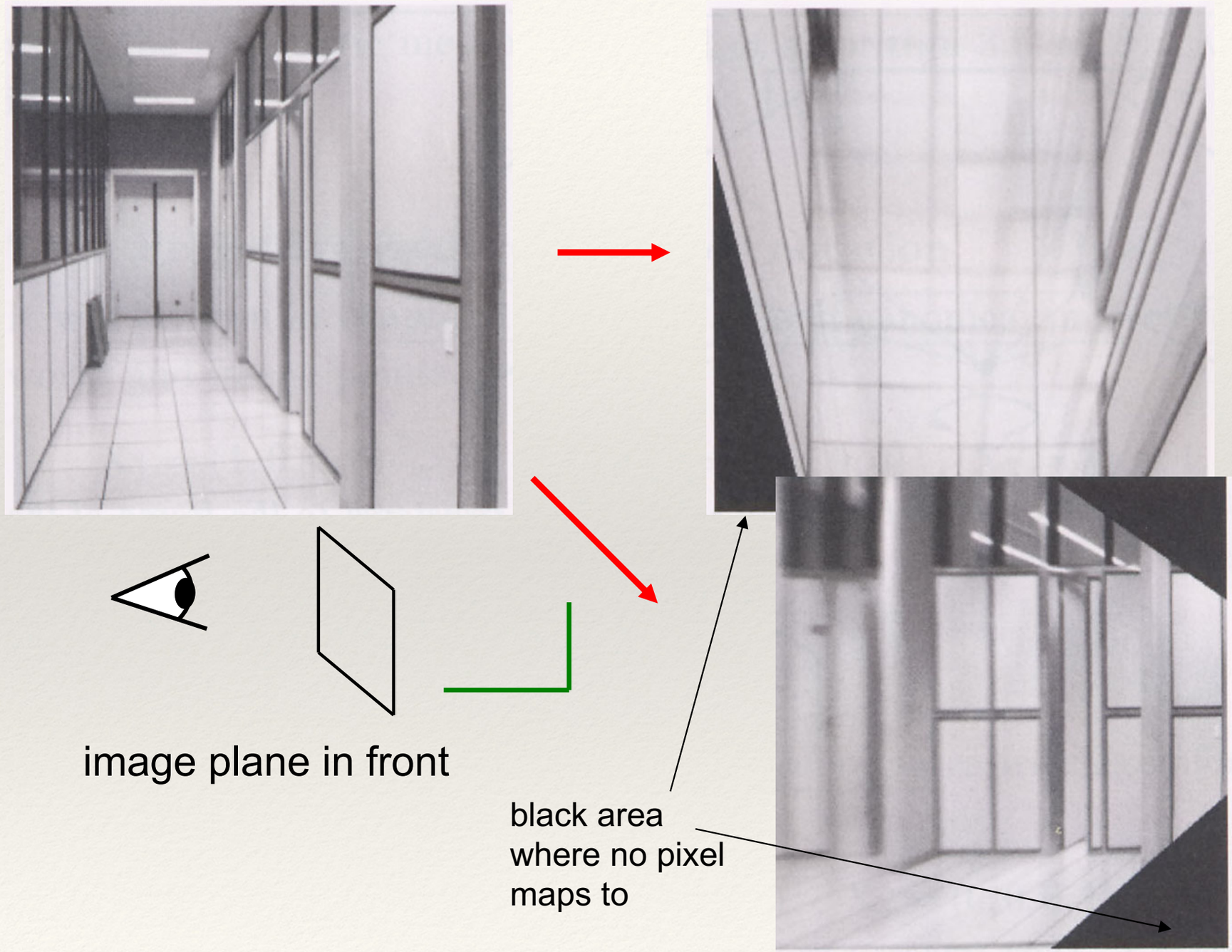
Sampling at *f(x,y):*



$$
\begin{aligned}
f(x,y) = \quad & (1-a)(1-b) \quad f[i,j] \\
& +a(1-b) \quad f[i+1,j] \\
& +ab \quad f[i+1,j+1] \\
& +(1-a)b \quad f[i,j+1]
\end{aligned}
$$

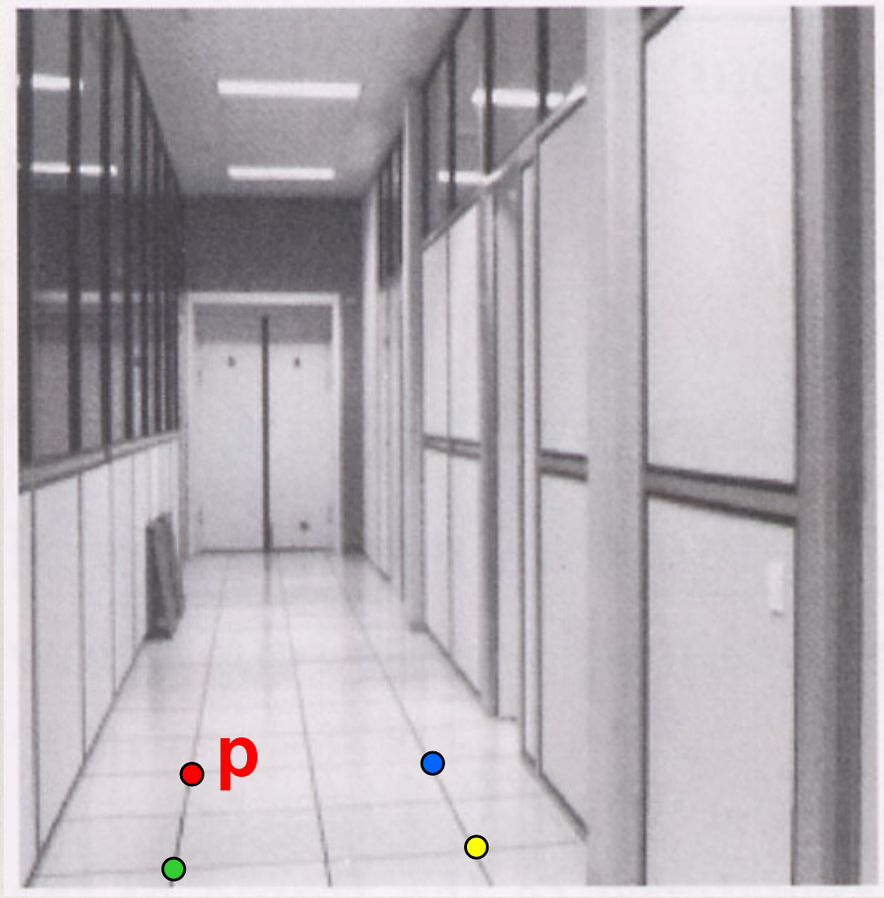
Slide from Alyosha Efros, CMU

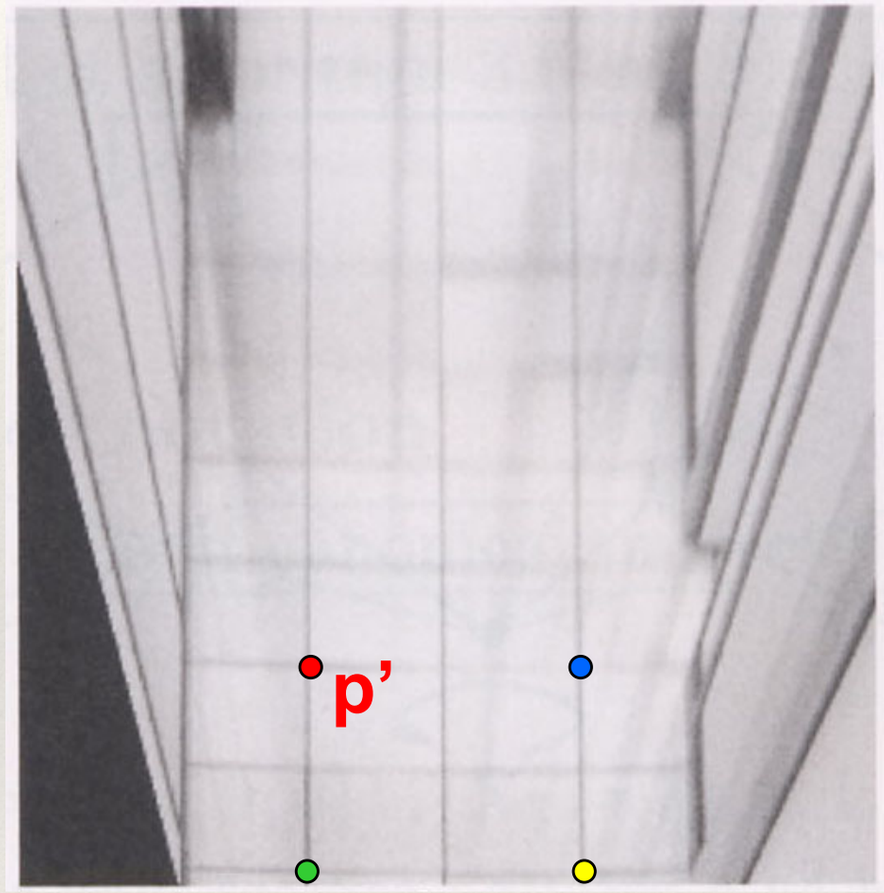
Image warping with homographies



Source: Steve Seitz

Image rectification





Analysing patterns and shapes

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

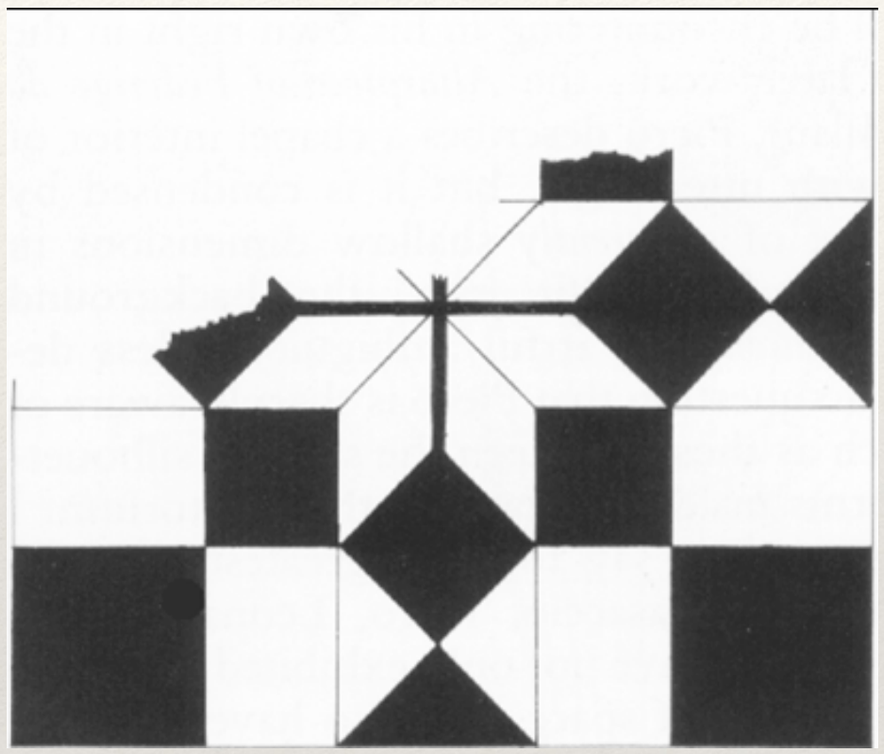
The floor (enlarged)

Automatically rectified floor

Slide from Antonio Criminisi

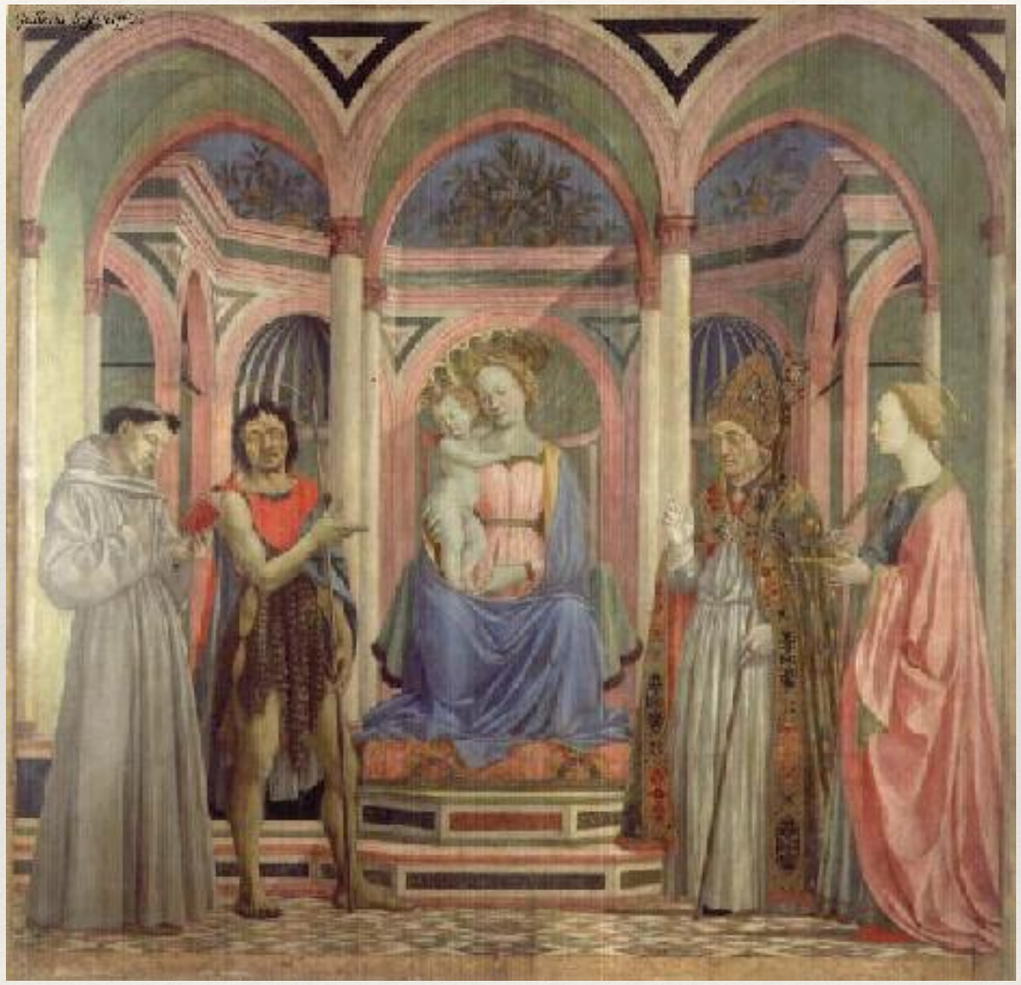
Analysing patterns and shapes

Slide from Antonio Criminisi



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

Analysing patterns and shapes



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

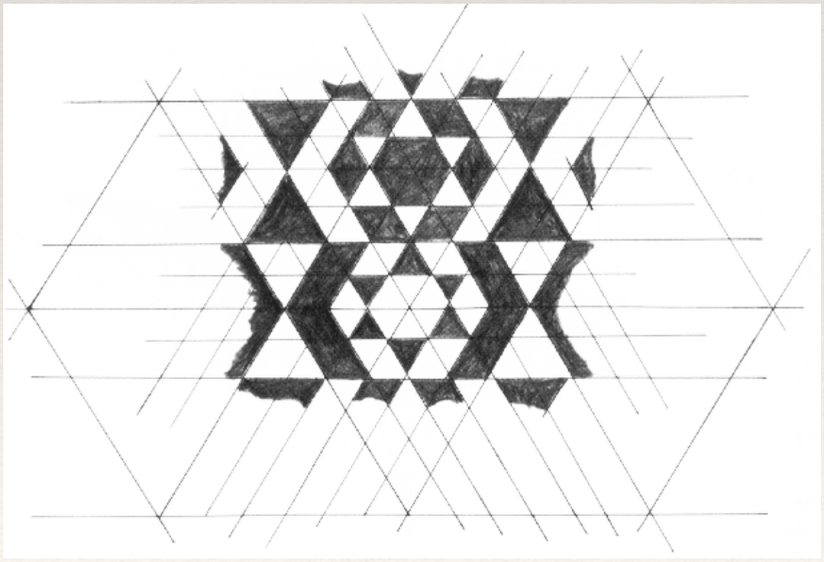
Slide from Criminisi

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

Automatically rectified floor

Analysing patterns and shapes

Automatic rectification

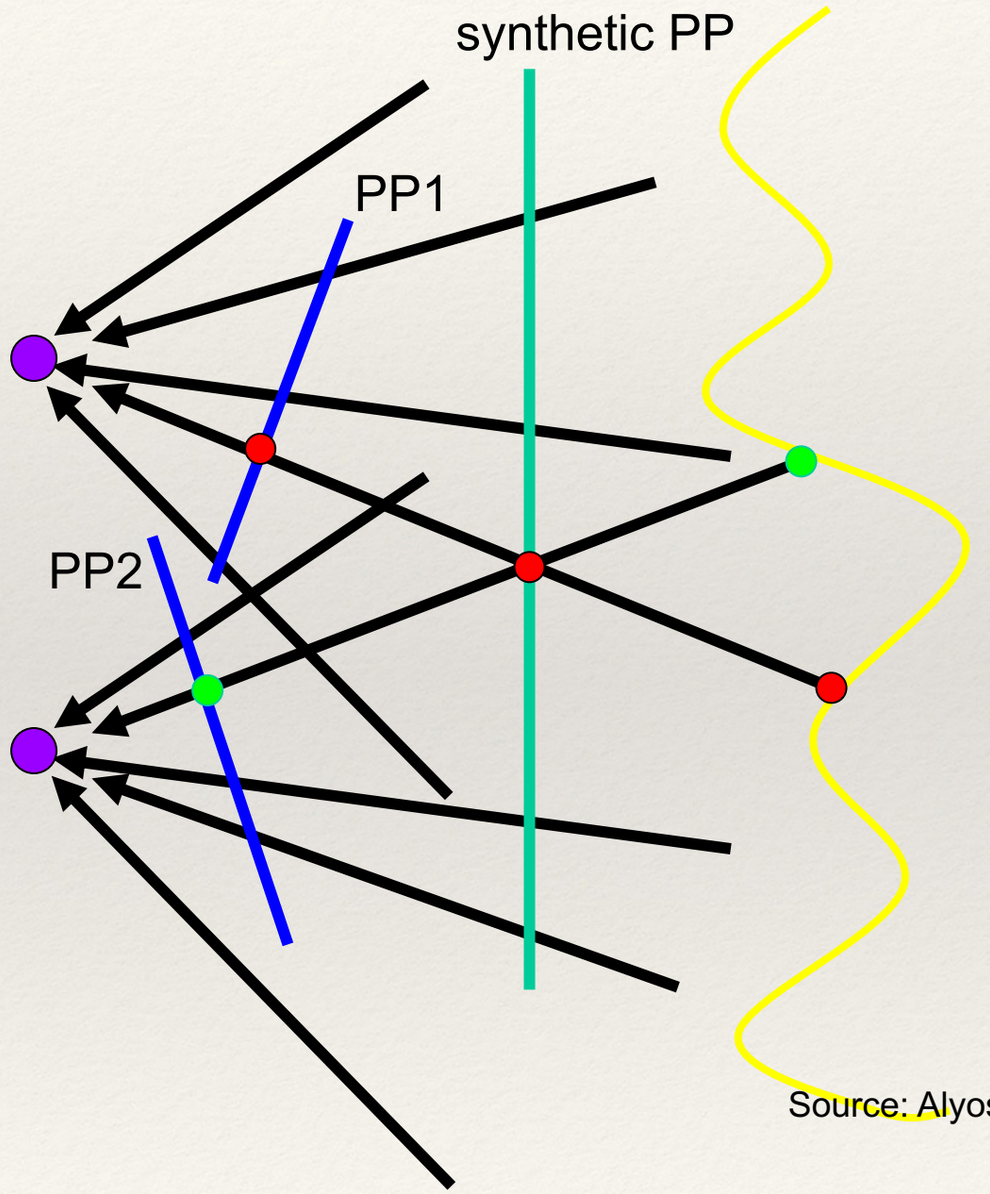


From Martin Kemp, *The Science of Art*
(manual reconstruction)

Slide from Criminisi

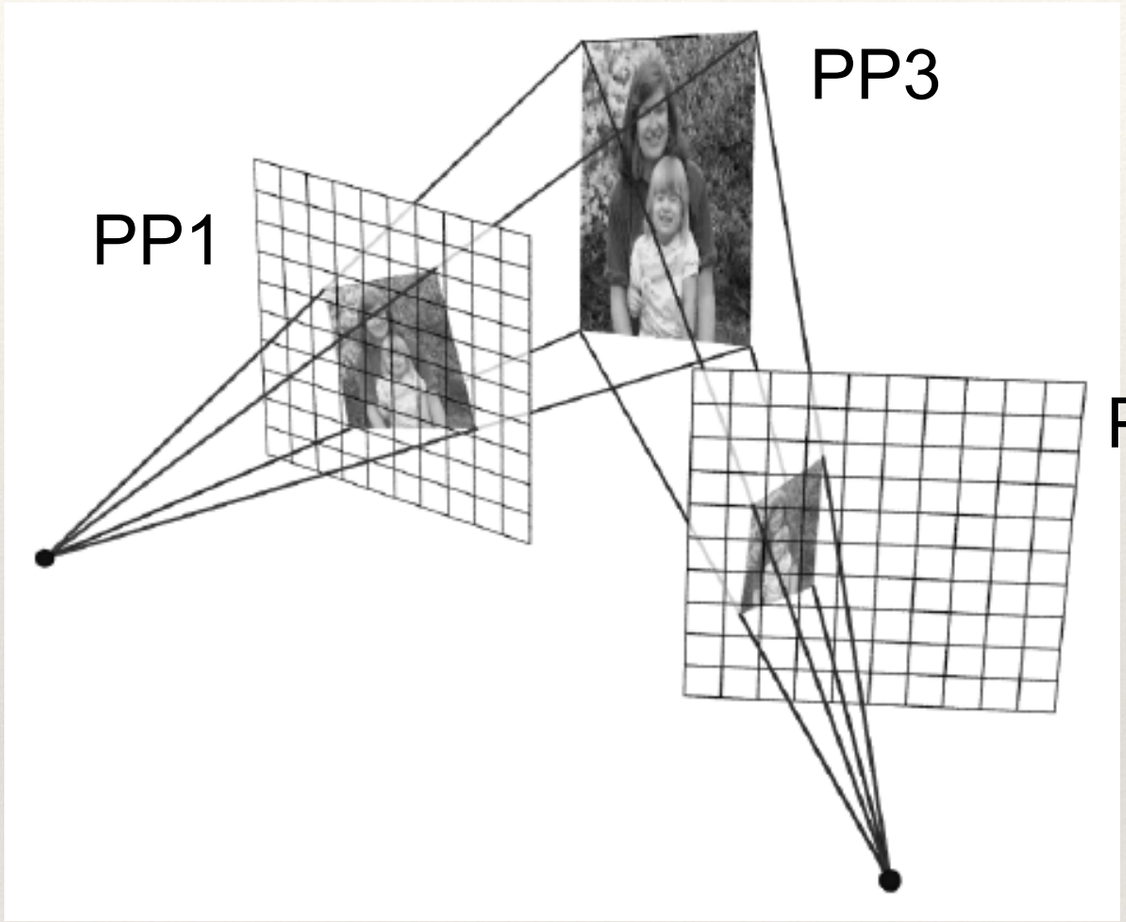
Changing camera center

Does it still work?



Source: Alyosha Efros

Planar scene (or far away)



PP3 is a projection plane of both centers of projection, so we are OK!

This is how big aerial photographs are made

Source: Alyosha Efros

Registering an object model to an image

- Problem
  - Compute rotation and translation of object model in camera coordinates
- Strategy
  - Search for sets of corresponding pairs of model-image features
    - typically, but not always, points
    - size of group depends on camera, model details
      - eg orthographic view of plane object needs three pairs of points
  - Compute rotation, translation from corresponding pairs
  - Verify computed transformation
    - Place model in camera coordinate system using this transformation
    - Render other model features using camera
    - Compare these to image; accept if score is good enough

Slides: D. Forsyth

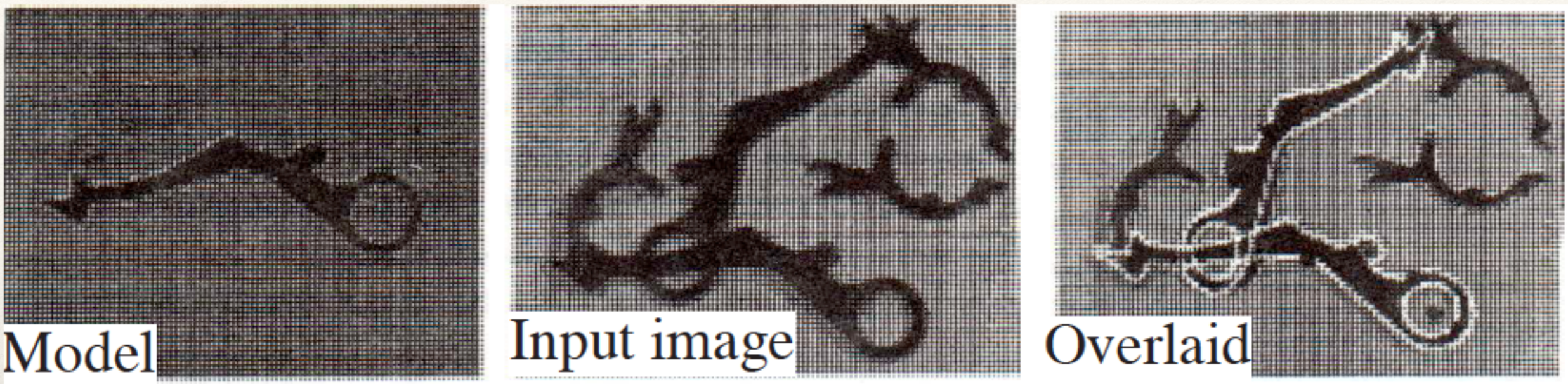


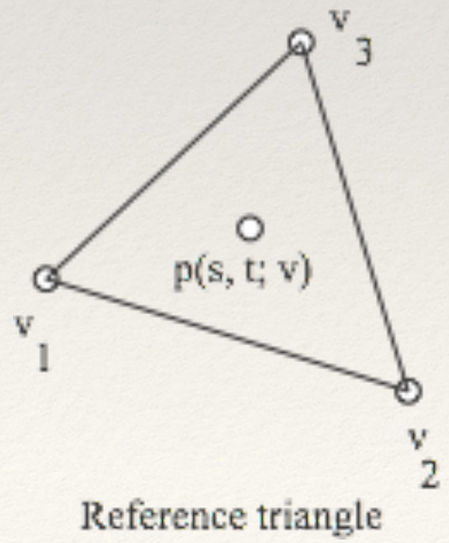
FIGURE 12.6: A plane object registered to an image. On the **left**, an image of an object; in the **center**, an image containing two instances of this object, along with some other stuff (the popular term is *clutter*). Feature points are detected, and then correspondences between groups—in this case, triples of points—are searched; each correspondence gives rise to an affine transformation from the model to the image. Satisfactory correspondences align many model edge points with image edge points, as in the figure on the **left**, which is why the method is sometimes called *alignment*. The images in this figure come from one of the earliest papers on the subject and are affected by the poor reproduction techniques of the time. *This figure was originally published as Figure 7 of "Object recognition using alignment," D.P. Huttenlocher and S. Ullman, Proc. IEEE ICCV, 1986.* © *IEEE, 1986.*

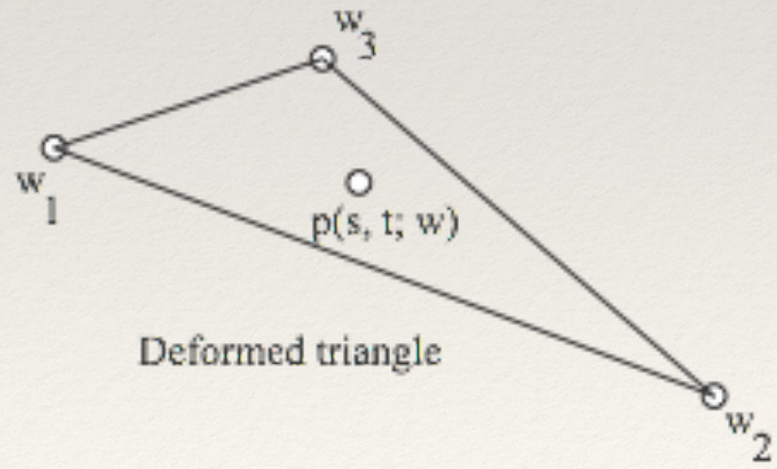
Registering deformable object

- Active appearance models - I
  - Grid of small triangles
    - Vertices are the parameters
  - For a triangle, describe interior point p by barycentric coordinates

$$p(s,t;\mathbf{v}) = s\mathbf{v}_1 + t\mathbf{v}_2 + (1-s-t)\mathbf{v}_3$$

- Crucial property:
  - when points v move to w, then p(s, t; v) goes to p(s, t; w)





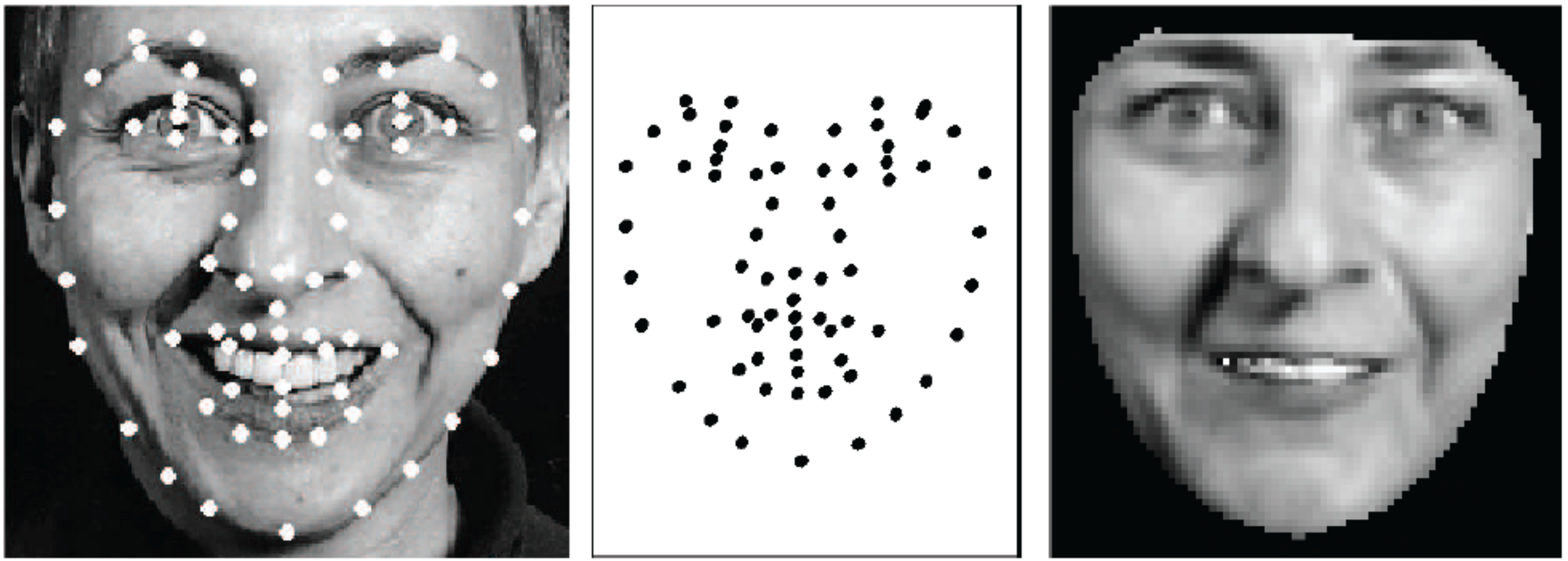
Slides: D. Forsyth

Registering deformable object

- Active appearance models - II
  - Intensities don’t change when mesh deforms
  - So we can find a deformed mesh by comparing intensities at corresponding locations
  - v are verts of reference mesh; In is intensity of reference image; w are verts of deformed mesh; Id is intensity of deformed image; s_j, t_j are samples of s, t.
  - Then we can find w by minimizing the intensity error between corresponding points
  - g could be the identity (not robust) or an M-estimator
  - error takes form

$$\sum_j g(\| \mathcal{I}_d(\boldsymbol{p}(s_j, t_j; \boldsymbol{w})) - \mathcal{I}_n(\boldsymbol{p}(s_j, t_j; \boldsymbol{v})) \|^2)$$

Slides: D. Forsyth



Reference points Relaxed points Relaxed intensity

FIGURE 12.8: A set of reference points placed over a face, on the **left**. At the **center**, these points in a relaxed configuration. Now assume we have a reasonable triangulation of the original set of points. By placing those points in correspondence with the relaxed configuration, we can map the intensities of the reference face to a relaxed configuration (**right**). *This figure was originally published as Figure 1 of "Active Appearance Models," by T. Cootes, G. Edwards, and C. Taylor, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2001, © IEEE, 2001.*

Registering deformable object

- Active appearance models - III
  - Complex triangle grids have too many vertices
    - hard minimization; many local minima; too many deformations
    - write reference vertices in matrix V, deformed vertices in matrix W
    - choose some basis deformations, write in matrices B
- Deformation model is then

$$\mathcal{W} = \mathcal{R}(\mathcal{V} + \sum_l \mathcal{B}_l \theta_l) + \boldsymbol{t}$$

  - where parameters are
    - R - rotation matrix
    - Theta - deformation terms
    - t - translation vector
  - now minimize intensity error as a function of these parameters

Slides: D. Forsyth

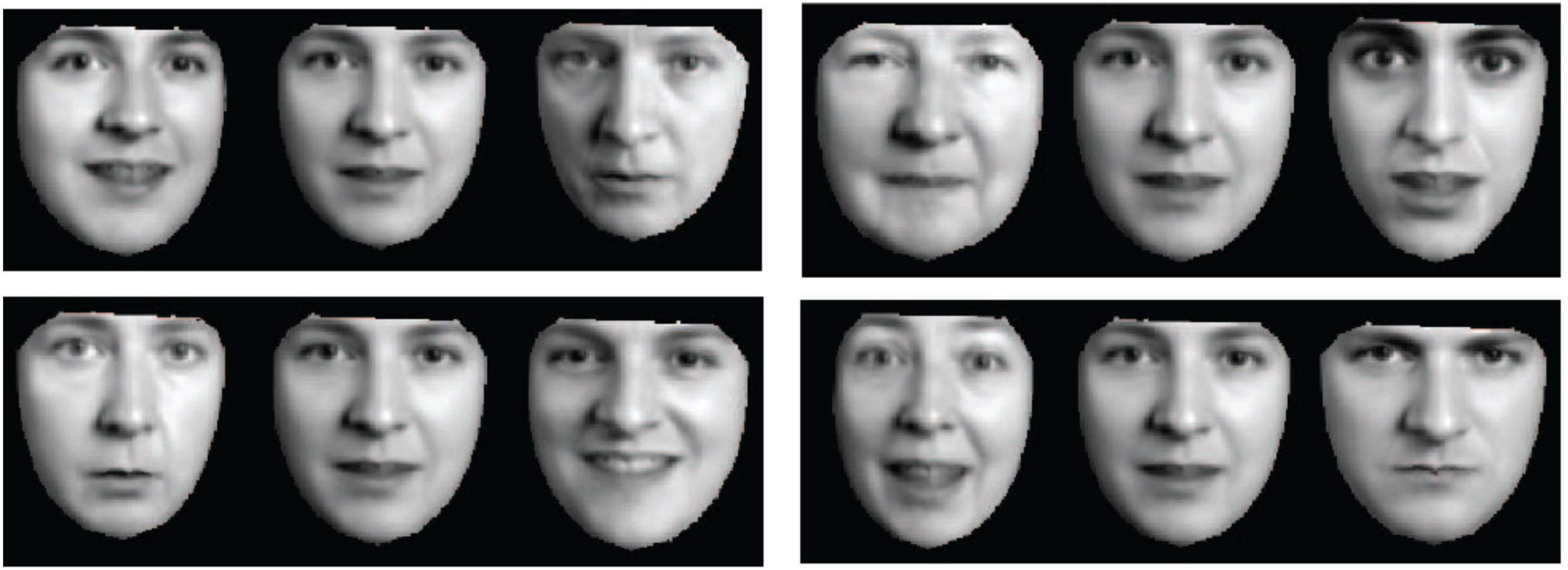


FIGURE 12.9: Different face intensity masks generated by moving deformation parameters to different values. Each block shows the effect of a different parameter; the center of that block shows the parameter at the mean value (where the mean is taken over numerous example faces), and the left (resp. right) of the block shows the parameter at mean plus (resp. minus) three standard deviations. Note how a range of expressions is encoded by these parameter variations. *This figure was originally published as Figure 2 of "Active Appearance Models," by T. Cootes, G. Edwards, and C. Taylor, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2001, © IEEE, 2001.*

FIGURE 12.10: Active appearance models registered to face images. On the **left**, the initial configuration of the model (blurry blob over the face; original face is second from right). As the minimization process proceeds, the search improves the registration to produce, in the final converged state, the registration on the **right**. Once we have this registration, the location of the vertices of the mesh and the deformation parameters encode the shape of the face. *This figure was originally published as Figure 5 of "Active Appearance Models," by T. Cootes, G. Edwards, and C. Taylor, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2001,* © *IEEE, 2001.*

Registration application: Medical Imaging

- Register medical imaging data, virtual surgery data, with real patient
  - to improve surgeon's visualization of what is there
- Register medical imaging data obtained using different types of imaging
  - provides improved resolution, information about tissue
- Register images of the same structure at different times
  - to compare progress of disease; to improve contrast of reagents
- Register to an atlas (standard models of anatomical structures)
  - to provide priors for image segmentation

Slides: D. Forsyth

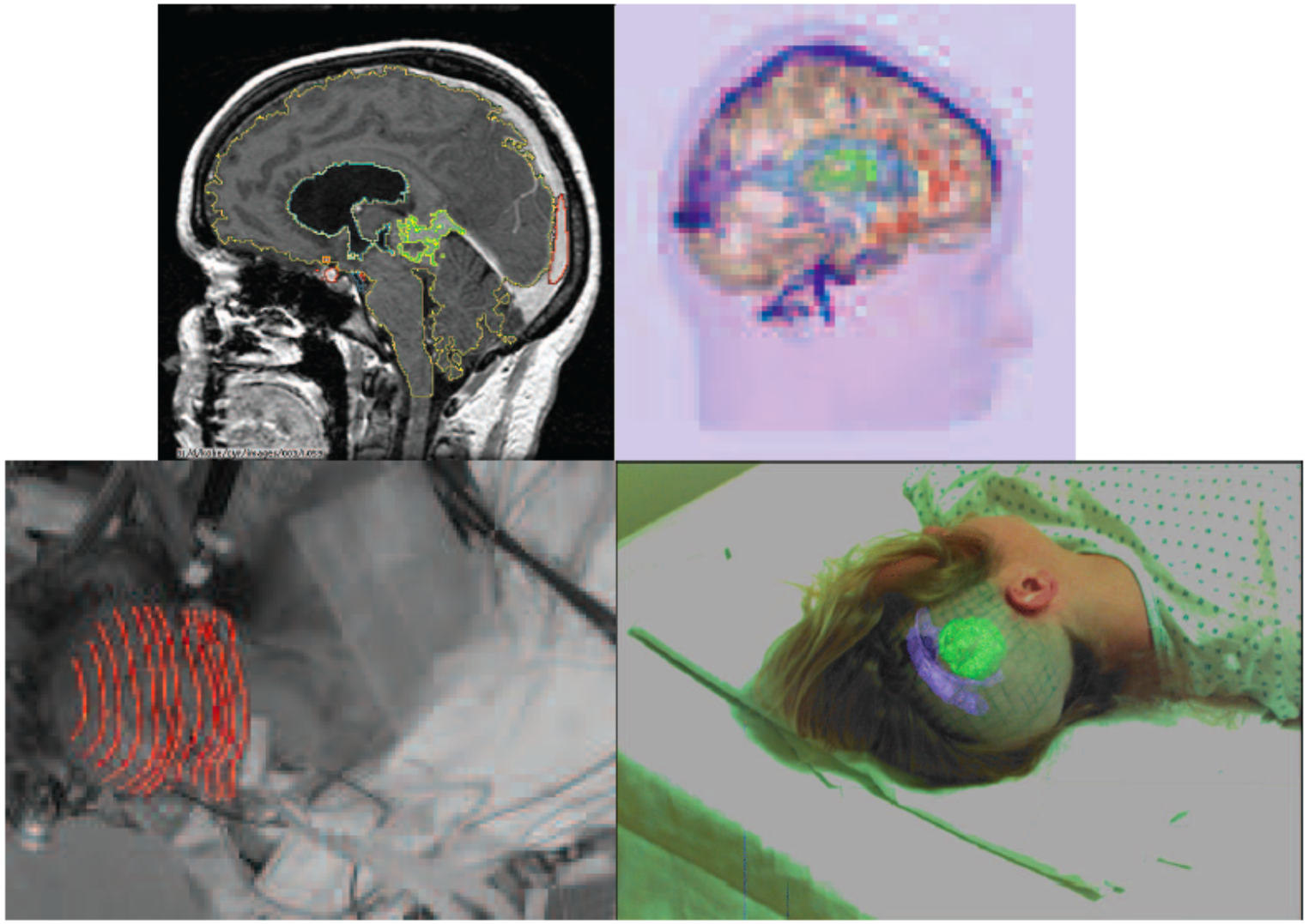


FIGURE 12.11: On the **top left**, a single slice of MRI data with an automatically acquired segmentation overlaid. The segmentation outlines the brain, vacuoles within the brain, and the tumor. MRI produces a sequence of slices, which yield a volume model; a view of a segmented volume model, with different colors showing different regions, is shown at the **top right**. Once this data is obtained, it is registered to a patient lying on a table. Registration is obtained using depth data measured by a laser ranger; the **bottom-left** figure shows a camera view of a patient with laser ranger data overlaid. By registering the segmented data to the patient on the operating table using this laser ranger data and the surface of the MRI data, we can display a processed version of the MRI imagery overlaid on the patient for the surgeon's information (**bottom right**). *Figures by kind permission of Eric Grimson; further information can be obtained from his web site,* `http://www.ai.mit.edu/people/welg/welg.html`.

Registering imaging, model data to real patient

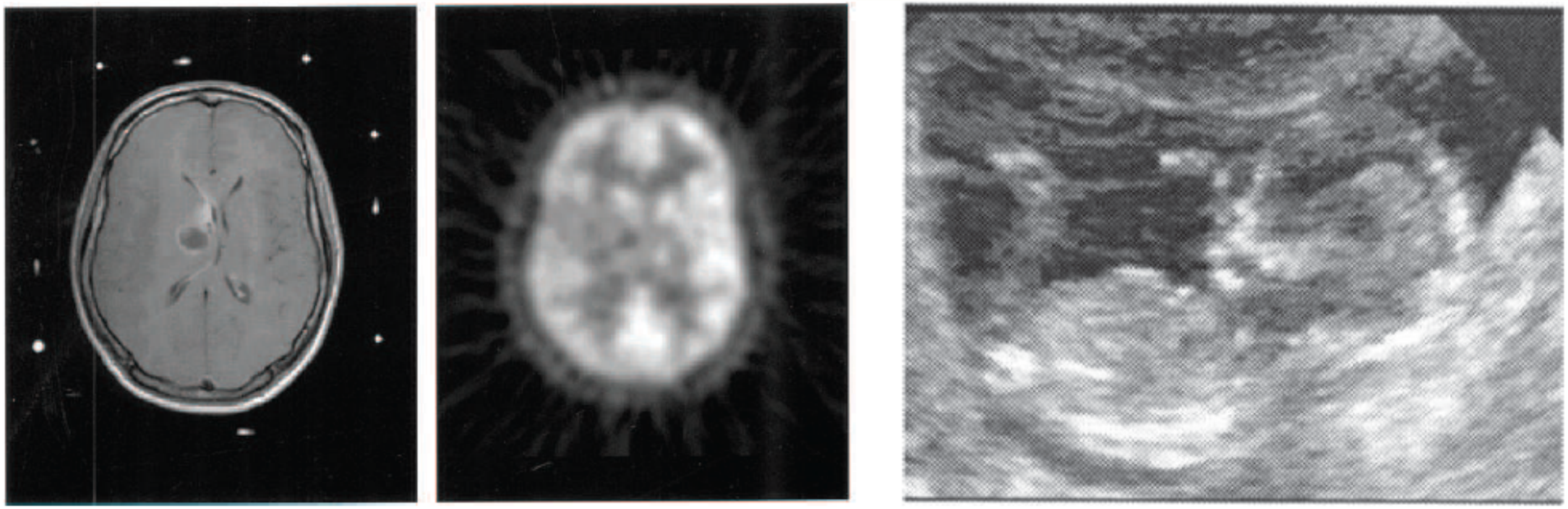
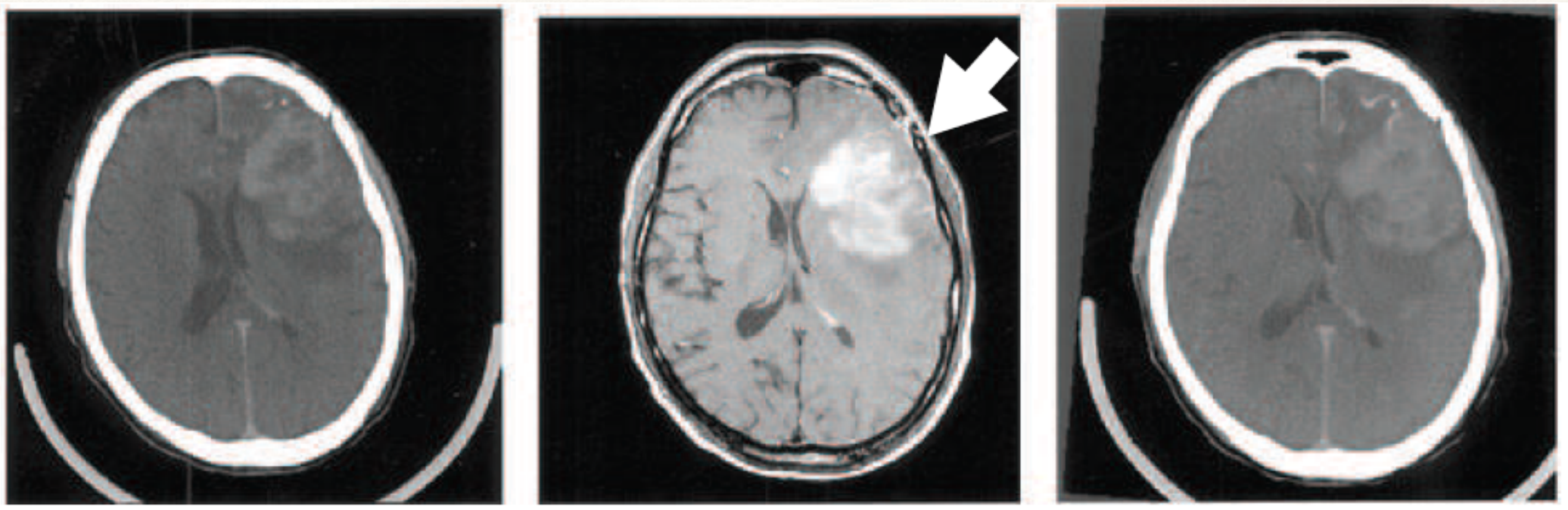


FIGURE 12.12: Images obtained with three different imaging modes. **Left**, an MR image of a brain, obtained with a patient wearing markers (the bright spots outside the skull). **Center**, a positron emission tomography (PET, a kind of NMI) image of the same brain. **Right**, a US image of a fetus in a womb. Notice how each modality shows different detail in different ways; there is high-resolution detail of the brain in the MR image. Compare this with the brain in the CT image of Figure 12.15, where the skull is much more visible. Notice the NM image is at low resolution, but in fact reflects function because regions that respond strongly have taken up some reagent. Finally, the US image has a significant noise component but shows details of soft tissue—you should be able to see a leg, the body, the head, and a hand of the fetus. *Part of this figure was originally published as Figure 10 of "Medical Image Registration using Mutual Information" by F. Maes, D. Vandermeulen and P. Suetens, Proc. IEEE, 2003* © *IEEE, 2003.*

Different imaging modes



CT image slice

MR image slice

Slice of registered volume

FIGURE 12.15: On the **left**, a 2D slice of a 3D CT image of a brain. **Center**, a 2D slice of a 3D MR image of a brain. On the **right**, a slice through the registered volumes. Notice how some rotation was required to register the volumes. The two volume boundaries don't overlap exactly in the right image, and the line separating the hemispheres of the brain in the CT image needs to be rotated a few degrees to overlap the same line in the MR image. Some deformation may have been applied here, too. Notice also that each image emphasizes a different type of structure. In the CT image, the bone is clearly visible, but there isn't much contrast between different soft tissues. In the MR image, soft tissue detail is visible, and a lesion can be seen (arrow). This means that registering by lining up pixel values probably will work poorly, and this registration required the mutual information methods described in the text. By registering the two volumes, we have the most information about each voxel. *This figure was originally published as Figure 1 of "Medical Image Registration using Mutual Information," by F. Maes, D. Vandermeulen, and P. Suetens, Proc. IEEE, 2003* © *IEEE, 2003.*

Registering different imaging modes

Changes in an organ exposed by registration

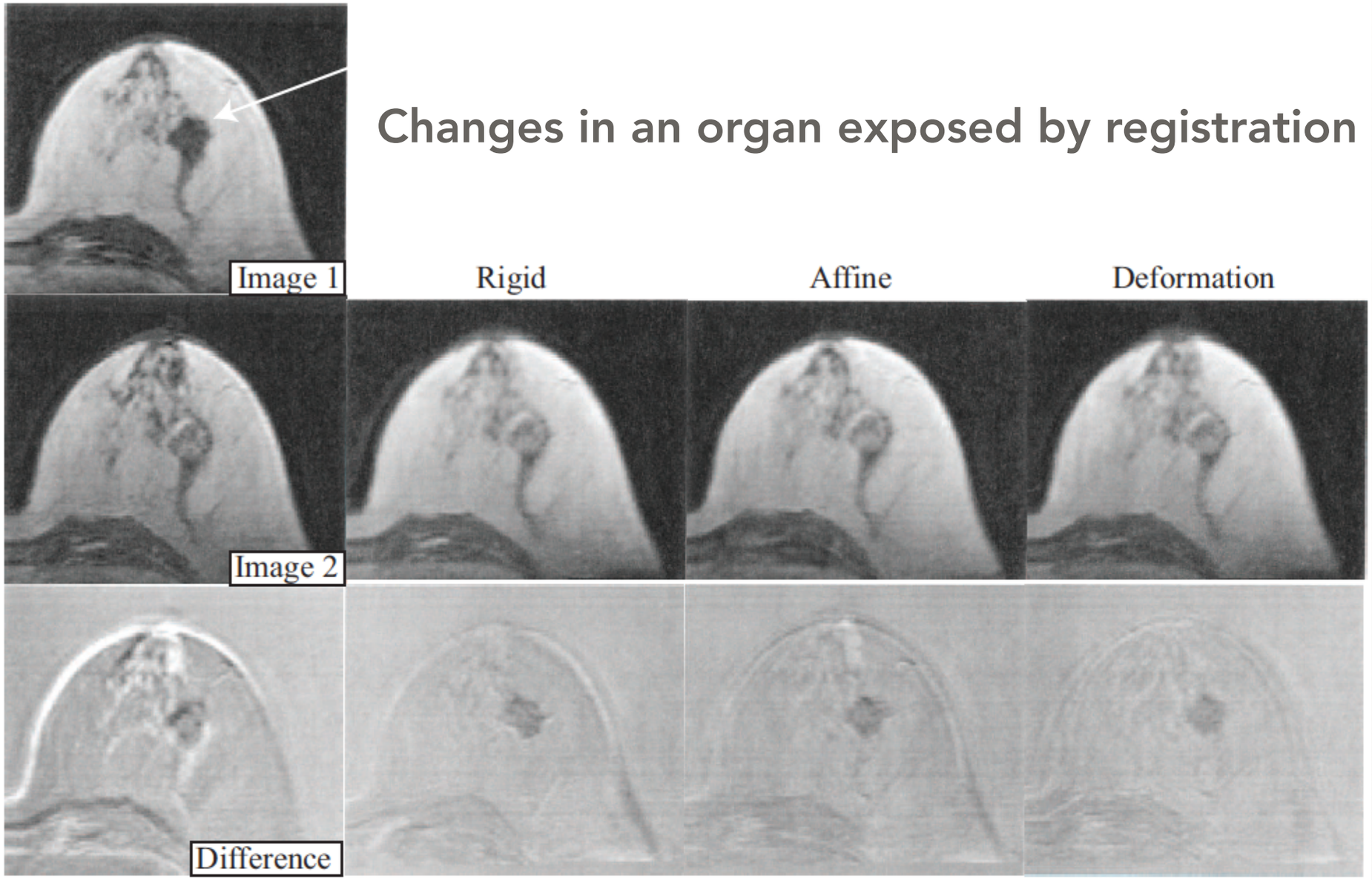


FIGURE 12.13: Registering images can expose changes in an organ. These are images of a breast, where a contrast medium is moving around (the dark material indicated by the arrow). Notice that in image 2, the contrast medium has moved. If these images are just superimposed and subtracted (**right** column), then structures in the difference image show that the breast moved between the images, too (notice the bright section on the edge; this means that the breast has shifted somewhat, which you can confirm by comparing the images). The other columns show registered (resp. difference) images under different models of motion. A rigid motion clearly improves the situation, as does an affine motion, but because breasts are deformable, a deformable registration gets a difference image that more clearly exposes the movement of the contrast medium. *This figure was originally published as Figures 6 and 7 of "Nonrigid registration using free-form deformation," by Ruekert* et al., *IEEE Trans. Medical Imaging, v18, n8, 1999* © *IEEE, 1999.*

Registering to an atlas

FIGURE 12.14: On the **left**, an image of a brain, showing enlarged ventricles (the dark butterfly-shaped blob in the middle). This is a volume of cerebro-spinal fluid, or CSF, inside the brain. It is desirable to segment the CSF, to measure the volume of the ventricles. One way to do this is to register this image to an atlas, a generic image of a brain that will be used to provide priors for the segmentation method. This brain will not be exactly the same in shape as the imaged brain. In the **center**, the CSF segmented by registering an atlas to the image using an affine transform; because the registration aligns the atlas to the brain relatively poorly, the segmentation shows poor detail. On the **right**, the same method applied to an atlas registered with a deformable model; notice the significant improvement in detail. *This figure was originally published as Figure 15 of "Medical Image Registration using Mutual Information," by F. Maes, D. Vandermeulen, and P. Suetens, Proc. IEEE, 2003* © *IEEE, 2003.*