CSCI 4830 / 5722
Computer Vision

University of Colorado **Boulder**

CSCI 4830 / 5722
Computer Vision

Dr. Ioana Fleming
Spring 2019
Lecture 14

University of Colorado **Boulder**

Reminders

Submissions:

- Take a break

Readings:

- Szeliski:
  - chapter 6.1 (Least squares, ITP, RANSAC)
  - chapter 4.1 (Feature detection – Points and patches)
- P&F:
  - chapter 10.2 (Least quares), 10.4 (RANSAC)
  - chapter 5 (Local features – corners, SIFT features)

Today

- Fitting
- Least squares – again
- RANSAC

Fitting and Alignment: Methods

- Global optimization / Search for parameters
  - Least squares fit
  - Robust least squares
  - Iterative closest point (ICP)

- Hypothesize and test
  - Generalized Hough transform
  - RANSAC

Slide from Derek Hoiem

Recap: Two Common Optimization Problems

Problem statement	Solution
minimize $\lVert \mathbf{Ax} - \mathbf{b} \rVert^2$	$\mathbf{x} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}$
least squares solution to $\mathbf{Ax} = \mathbf{b}$	$\mathbf{x} = \mathbf{A} \setminus \mathbf{b}$ (matlab)

Problem statement	Solution
minimize $\mathbf{x}^T\mathbf{A}^T\mathbf{Ax}$ s.t. $\mathbf{x}^T\mathbf{x} = 1$	$[\mathbf{v}, \lambda] = \mathrm{eig}(\mathbf{A}^T\mathbf{A})$
minimize $\dfrac{\mathbf{x}^T\mathbf{A}^T\mathbf{Ax}}{\mathbf{x}^T\mathbf{x}}$	$\lambda_1 < \lambda_{2..n} : \mathbf{x} = \mathbf{v}_1$
non - trivial lsq solution to $\mathbf{Mx} = 0$	

Singular value decomposition (SVD)

- SVD decomposes any mxn matrix **A** as

$$\underset{m\times n}{\mathbf{A}} = \underset{m\times m}{\mathbf{U}} \ \underset{m\times n}{\Sigma} \ \underset{n\times n}{\mathbf{V}^T}$$

- Properties
  - Σ is a diagonal matrix containing singular values of A
    - ***diagonal entries sorted from largest to smallest***
    - ***The non-zero singular values of A (found on the diagonal entries of Σ) are the square roots of the non-zero eigenvalues of both A^TA and AA^T***
  - columns of U are eigenvectors of AA^T
  - columns of V are eigenvectors of A^TA

Eigenvalues & Eigenvectors

- **Eigenvectors** (for a square $m \times m$ matrix $\mathbf{S}$)

$$\boxed{\mathbf{S}\mathbf{v} = \lambda \mathbf{v}}$$

(right) eigenvector

$\mathbf{v} \in \mathbb{R}^m \neq \mathbf{0}$

eigenvalue

$\lambda \in \mathbb{R}$

Example

$$\begin{pmatrix} 6 & -2 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

- **How many eigenvalues** are there at most?

Example

- Let $S = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$ ← Real, symmetric.

- Then $S - \lambda I = \begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix} \Rightarrow (2-\lambda)^2 - 1 = 0.$

- The eigenvalues are 1 and 3 (nonnegative, real).
- The eigenvectors are orthogonal (and real):

$$\begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Plug in these values and solve for eigenvectors.

Singular Value Decomposition

For an $m \times n$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

$$A = U\Sigma V^T$$

$m \times m$ | $m \times n$ | V is $n \times n$

The columns of ***U*** are orthogonal eigenvectors of $\boldsymbol{AA^T}$.

The columns of ***V*** are orthogonal eigenvectors of $\boldsymbol{A^TA}$.

Eigenvalues $\lambda_1 \ldots \lambda_r$ of $\boldsymbol{AA^T}$ are the eigenvalues of $\boldsymbol{A^TA}$.

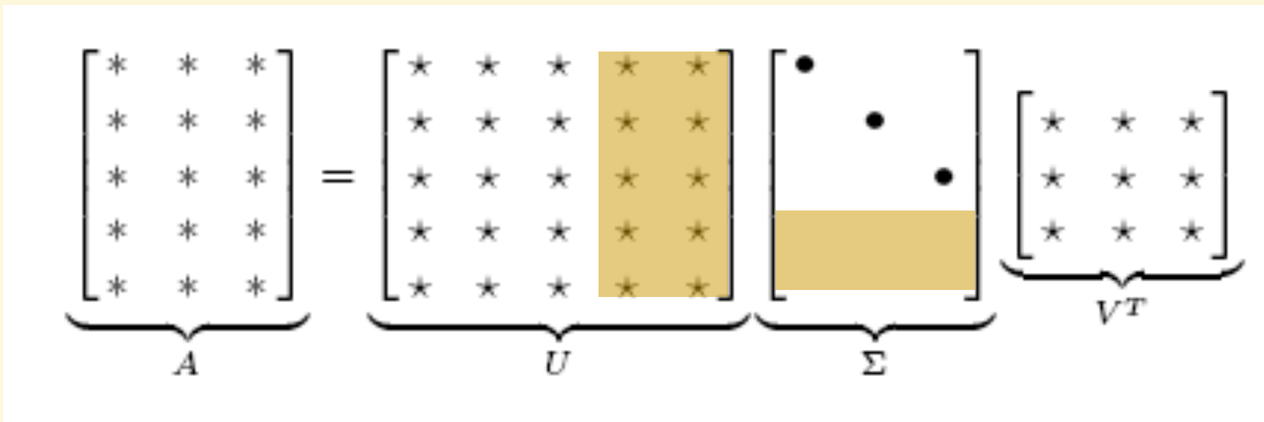
$$\sigma_i = \sqrt{\lambda_i}$$

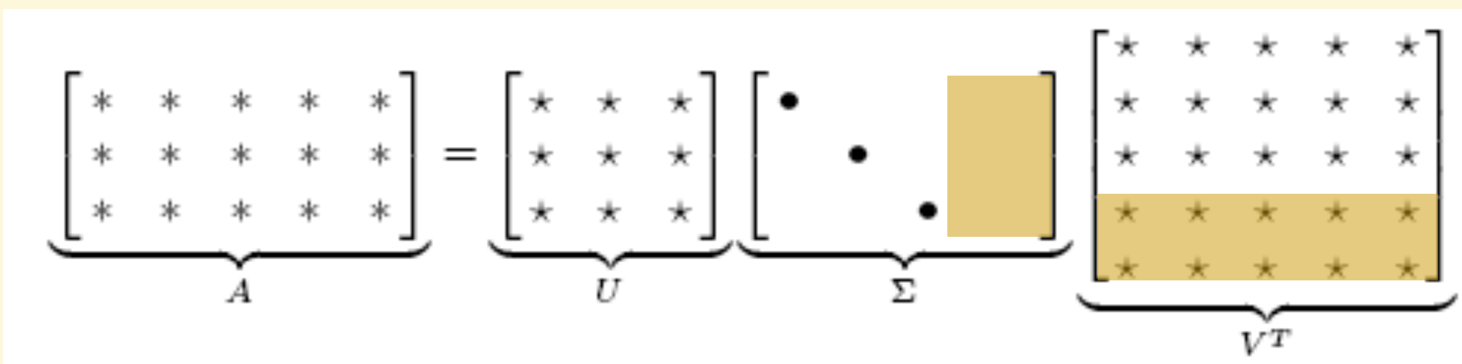
$$\Sigma = diag(\sigma_1 \ldots \sigma_r)$$

Singular values.

Singular Value Decomposition

- Illustration of SVD dimensions and sparseness





SVD example

Let $$A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Thus m=3, n=2. Its SVD is

$$\begin{bmatrix} 0 & 2/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

Typically, the singular values arranged in decreasing order.

Singular value decomposition (SVD)

$$\underset{m\times n}{\mathbf{A}} = \underset{m\times m}{\mathbf{U}} \; \underset{m\times n}{\Sigma} \; \underset{n\times n}{\mathbf{V}^T}$$

- If A is singular (e.g., has rank 3)
  - only first 3 singular values are nonzero
  - we can throw away all but first 3 columns of U and V

  - Choose M' = U', S' = $\Sigma' V'^T$

$$\underset{m\times n}{\mathbf{A}} = \underset{3\times m}{\mathbf{U}'} \; \underset{3\times 3}{\Sigma'} \; \underset{3\times n}{\mathbf{V}'^T}$$

Dimensionality Reduction

Singular value decomposition (SVD)

Dimensionality reduction is a process for reducing the number of features to be used in an analysis or modeling exercise.

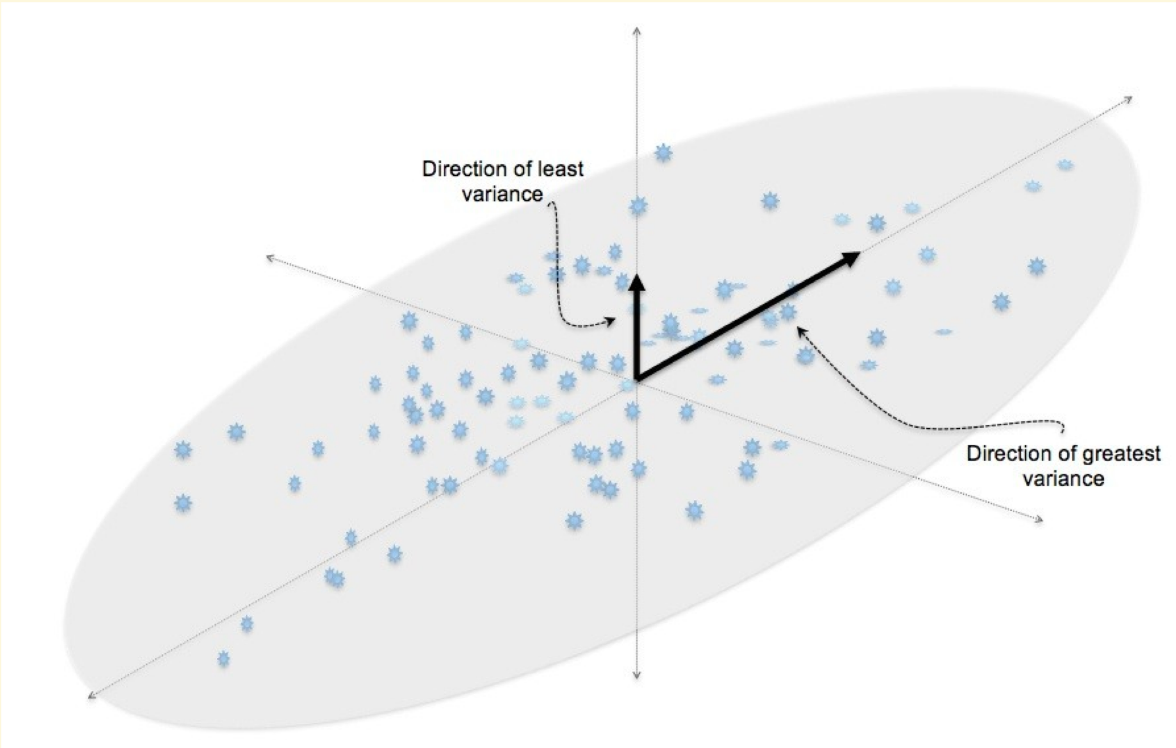
When is it useful?

- **PCA (**Principal Component Analysis)
- **Image Compression**
- **Least Squares**

Singular value decomposition (SVD)

Dimensionality Reduction: when is it useful?

- **PCA (**Principal Component Analysis)



Singular value decomposition (SVD)

Dimensionality Reduction: when is it useful?

- **Image Compression**

… singular values contain most of the information about the image.

Keep largest singular values, nullify the others!

Singular value decomposition (SVD)

Dimensionality Reduction: when is it useful?

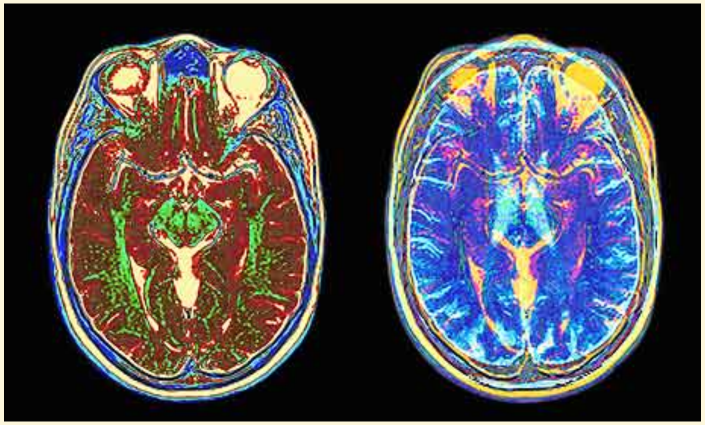
- **Least Squares**

Just like PCA: The directions along which there is greatest variance are referred to as the "principal components" (of variation in the data)

What if you want to align but have no prior matched pairs?

- Hough transform and RANSAC not applicable

- Important applications



Medical imaging: match brain scans or contours

Robotics: match point clouds

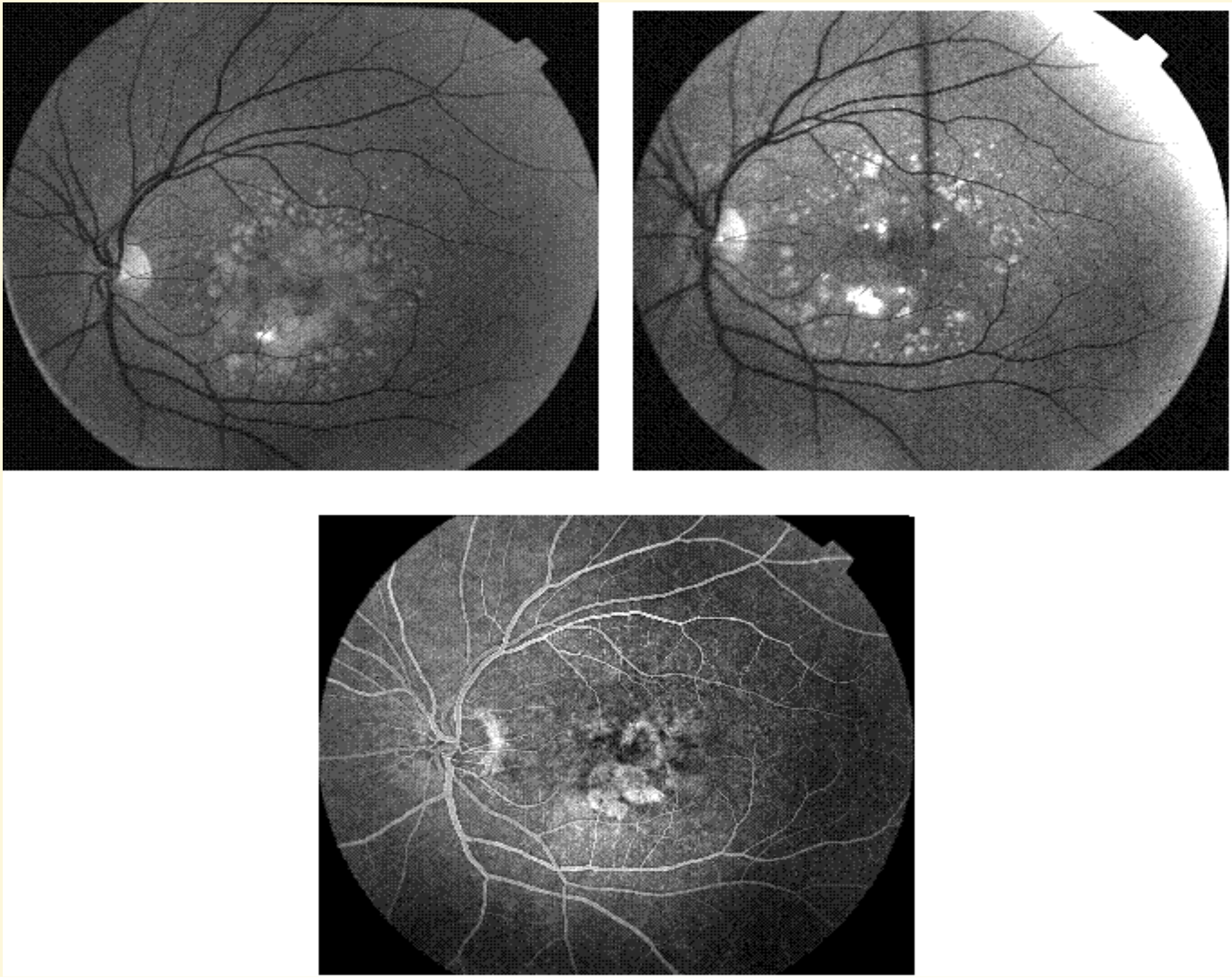
Iterative Closest Points (ICP) Algorithm

Goal: estimate transform between two dense sets of points

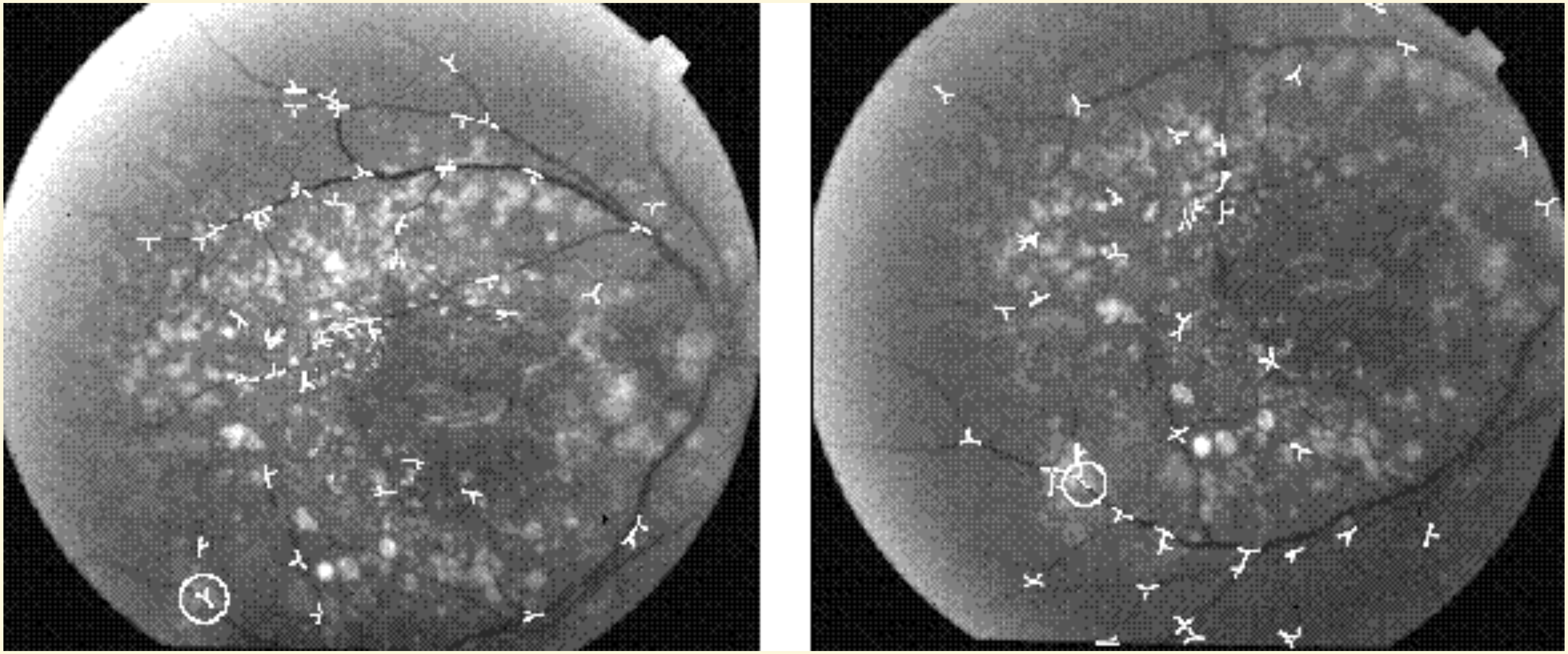
1. **Assign** each point in {Set 1} to its nearest neighbor in {Set 2}
2. **Estimate** transformation parameters
   – e.g., least squares or robust least squares
3. **Transform** the points in {Set 1} using estimated parameters
4. **Repeat** steps 1-3 until change is very small

Project Goal

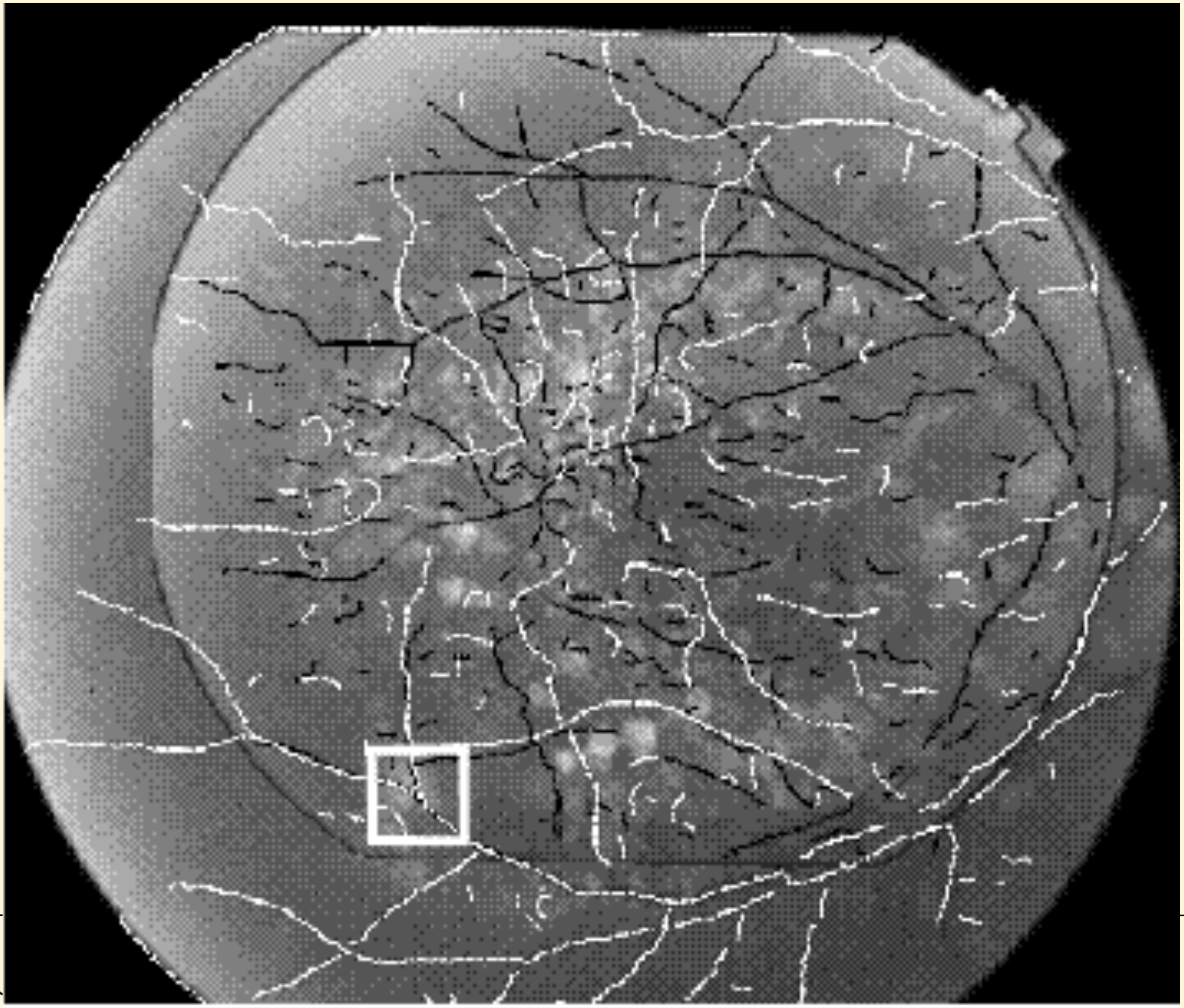
- Registration and tracking for retinal image targets
- C. V. Stewart, C.-L. Tsai, and B. Roysam, "The Dual Bootstrap Iterative Closest Point Algorithm with Application to Retinal Image Registration," *IEEE Trans on Medical Imaging*,22(11), pp. 1379-1394, 2003.



Registration – Retinal Images



Local vs. Global

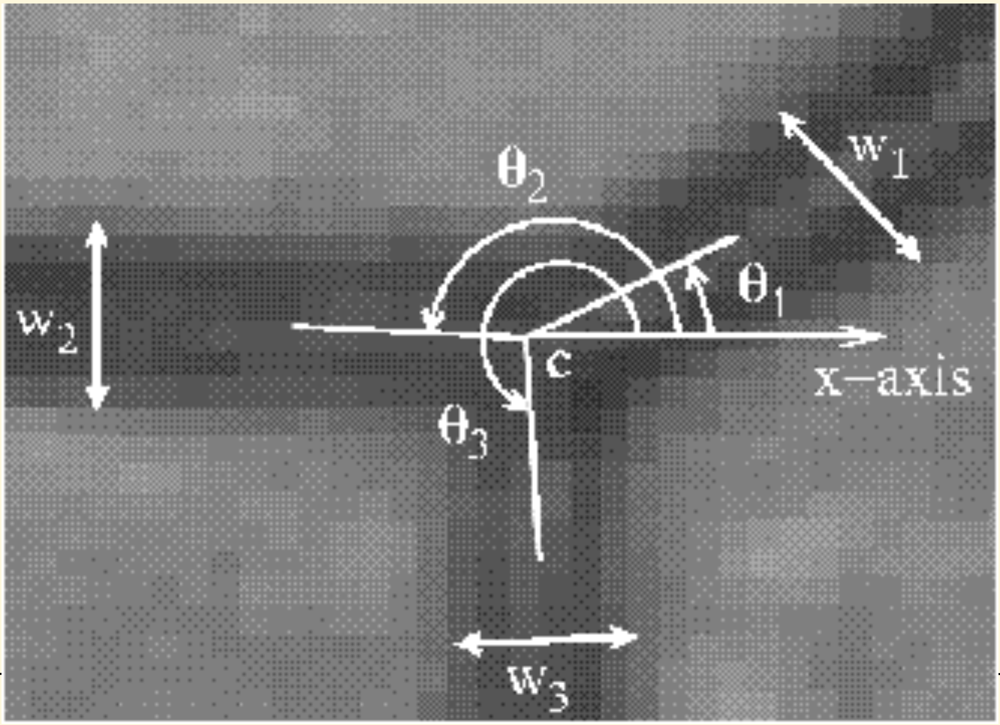


The algorithm

1. Pre-computation
2. Estimating the transformation
3. Region Bootstrapping
4. Model Bootstrapping

Pre-computation

- Feature extraction (tracing) → produce sets;
- Match landmarks → initial correspondences
  - comparing invariants



Transformation Estimate

For each initial correspondence:

- Initialize lowest order model;
- Compute initial transformation estimate;
- Establish initial bootstrap region – $\alpha = 10$;

While not converged:

- Estimate transformation parameters: ICP, covariance

Bootstrapping the Model

Transformation	Equation	DoF	Accuracy
Similarity	$\mathbf{p}' = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} & 0 & 0 & 0 \\ \theta_{21} & -\theta_{13} & \theta_{12} & 0 & 0 & 0 \end{pmatrix} \mathbf{X}(\mathbf{p} - \mathbf{p}_0)$	4	5.05 pixels
Affine	$\mathbf{p}' = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} & 0 & 0 & 0 \\ \theta_{21} & \theta_{22} & \theta_{23} & 0 & 0 & 0 \end{pmatrix} \mathbf{X}(\mathbf{p} - \mathbf{p}_0)$	6	4.58 pixels
Reduced quadratic	$\mathbf{p}' = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} & \theta_{14} & 0 & \theta_{14} \\ \theta_{21} & -\theta_{13} & \theta_{12} & \theta_{24} & 0 & \theta_{24} \end{pmatrix} \mathbf{X}(\mathbf{p} - \mathbf{p}_0)$	6	2.41 pixels
Quadratic	$\mathbf{p}' = \begin{pmatrix} \theta_{11} & \theta_{12} & \theta_{13} & \theta_{14} & \theta_{15} & \theta_{16} \\ \theta_{21} & \theta_{22} & \theta_{23} & \theta_{24} & \theta_{25} & \theta_{26} \end{pmatrix} \mathbf{X}(\mathbf{p} - \mathbf{p}_0)$	12	0.64 pixels

Bootstrapping the Model

- Apply model selection technique;
- If different model:
  - Recompute estimate and covariance matrix

DUAL BOOTSTRAP = simultaneous growth in the bootstrap region and the transformation model order

Bootstrapping the Region

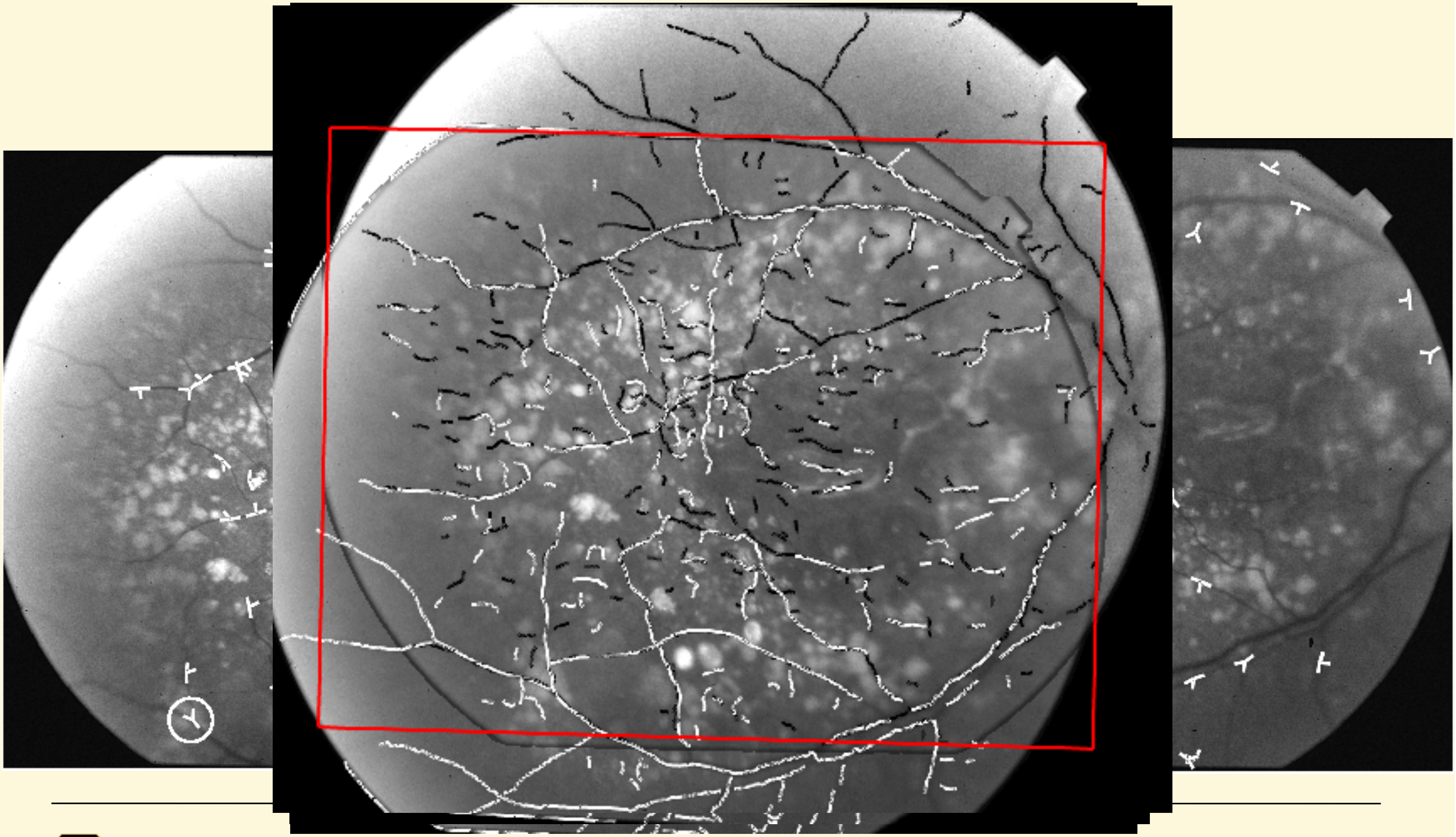
- Uncertainty in the transformation estimate:

  - deriving the uncertainty from the covariance of the transformation parameters;

  - developing region expansion equations using the transfer error;

  - growing each side independently

Retina Example

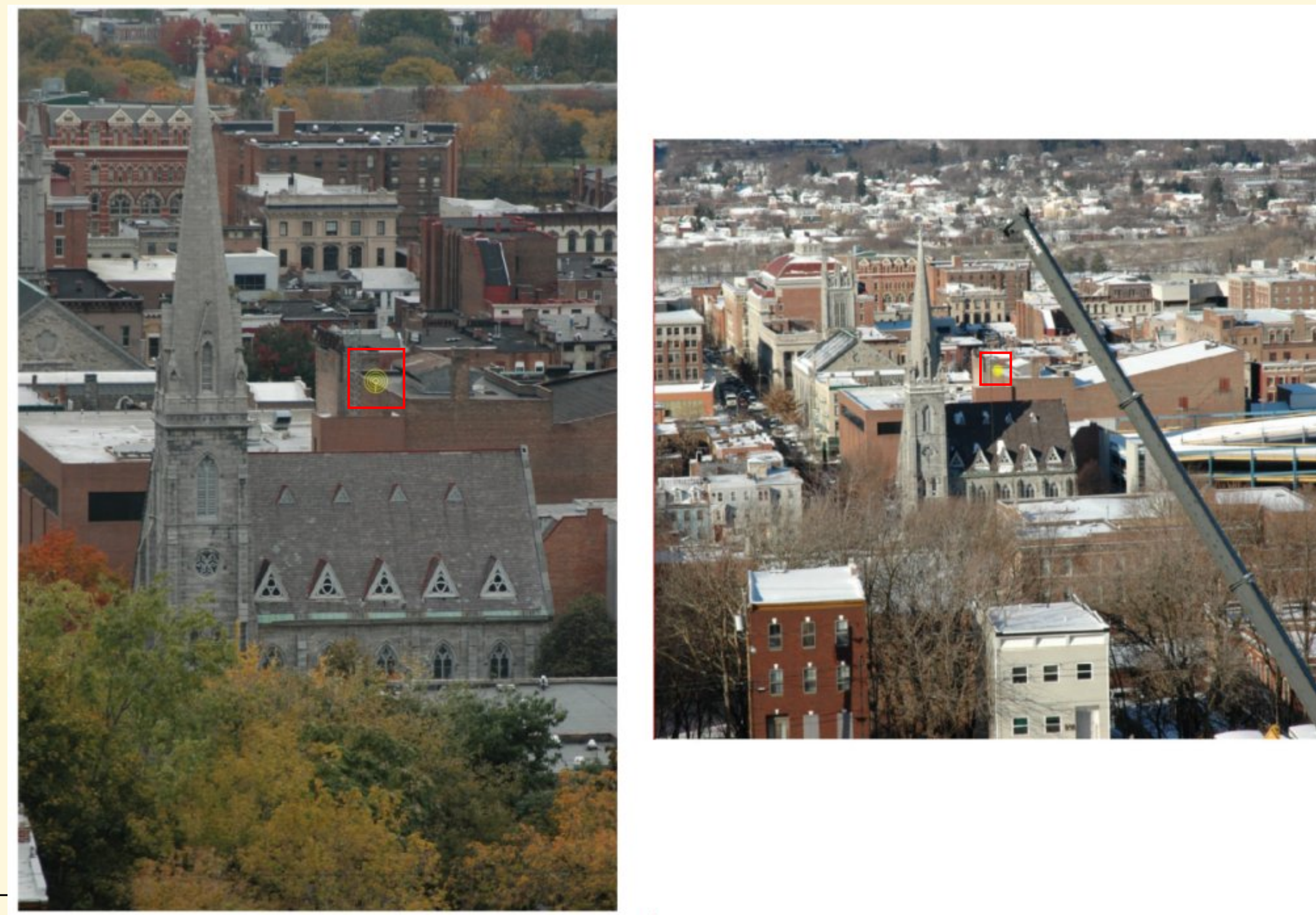


Results

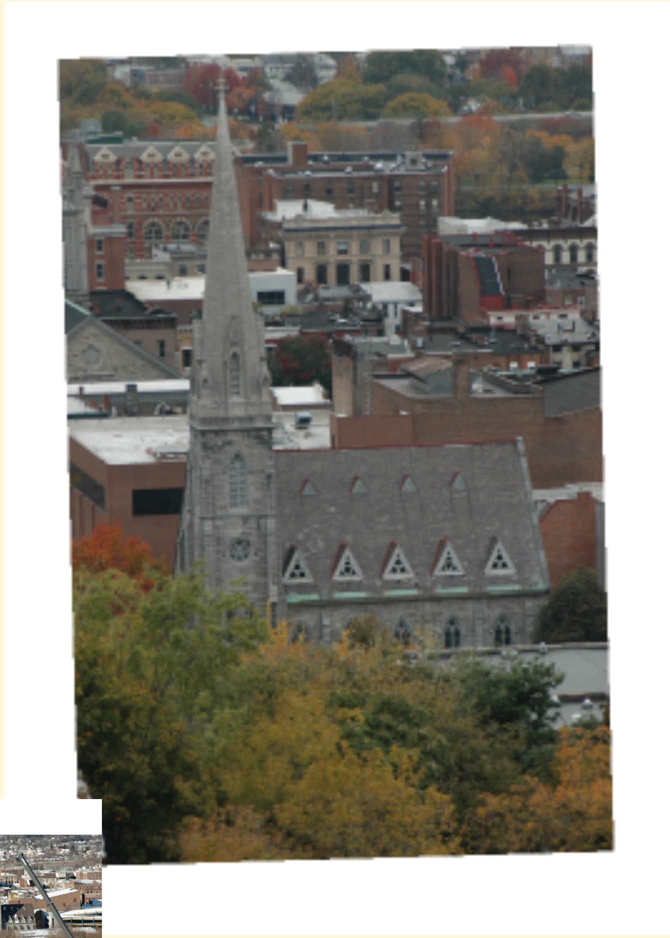
- **Over 4000 image pairs;**
- **For > 35% overlap and at least 1 correspondence there were no failures;**
- **Accuracy: .64 pixels**

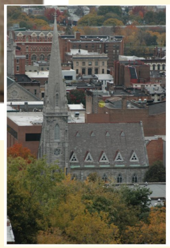
	All pairs	One landmark pair
Healthy(%)	97.0	99.5
Pathology(%)	97.8	100

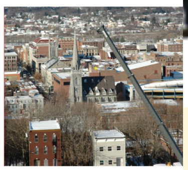
Winter-Summer Example



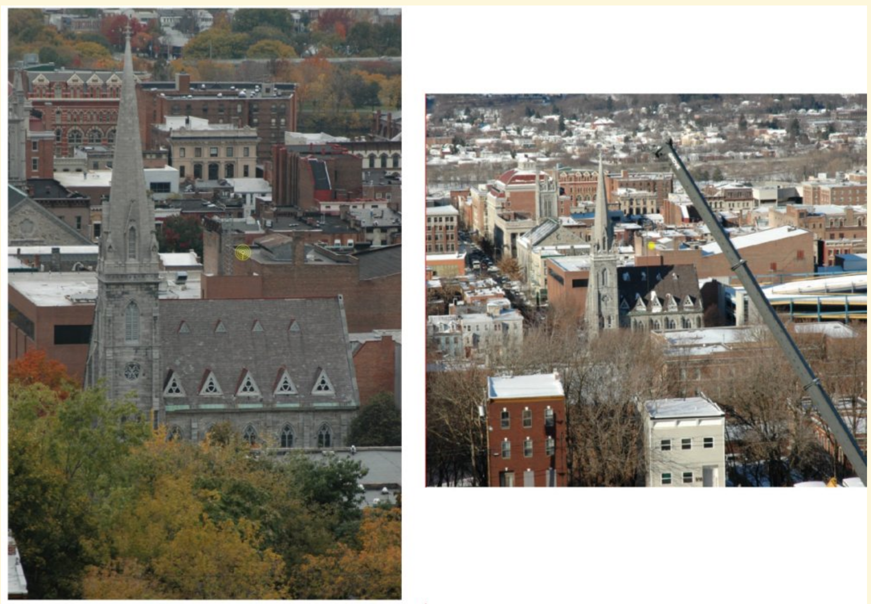
Winter-Summer Example

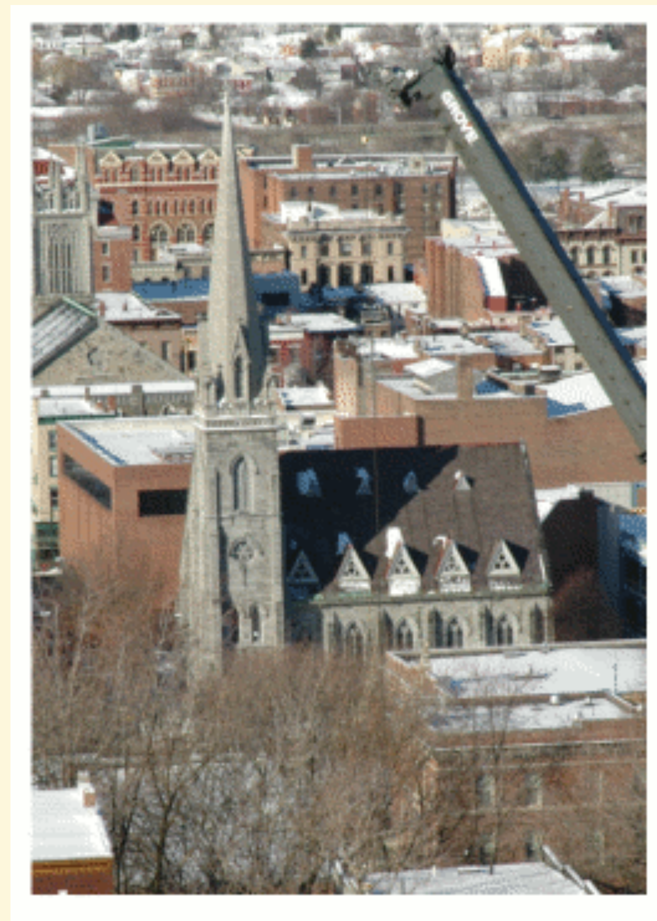




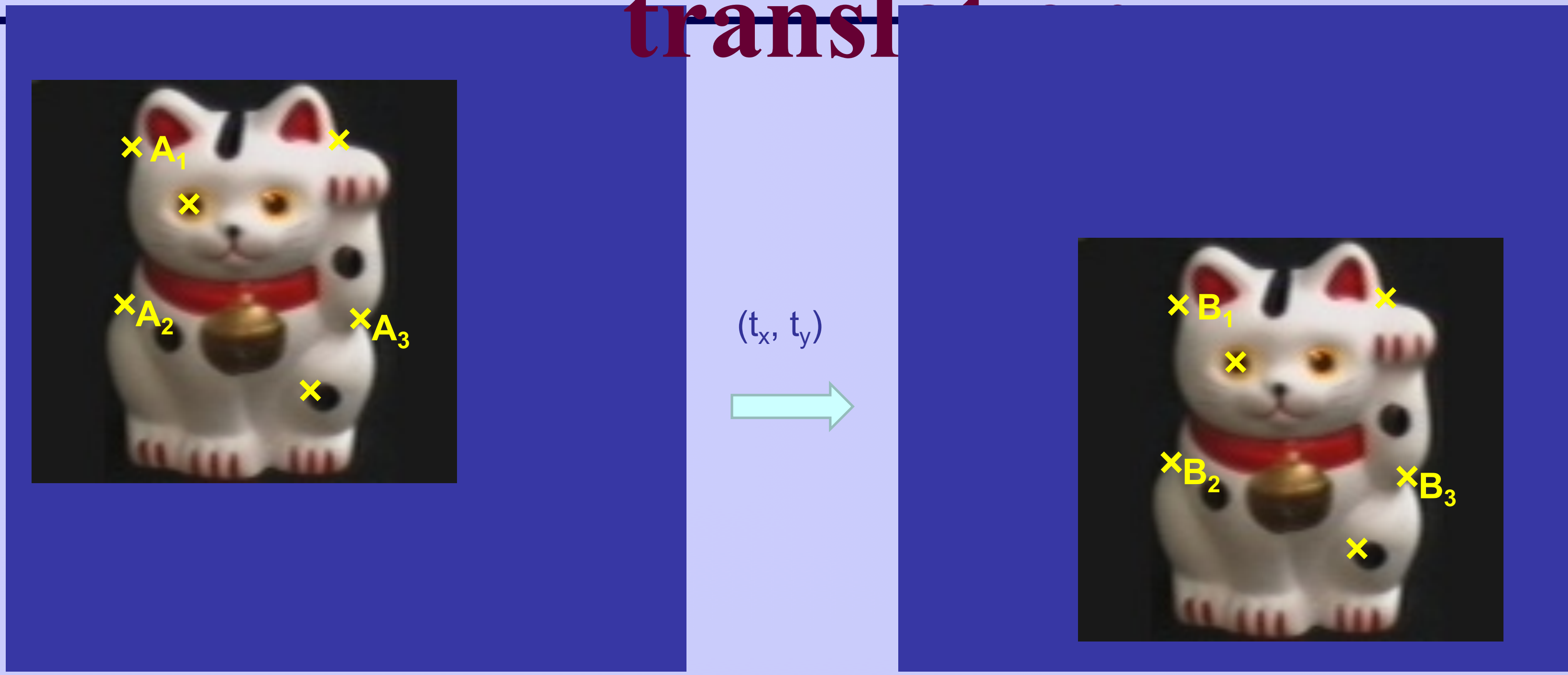


Winter-Summer Example





Example: solving for translation



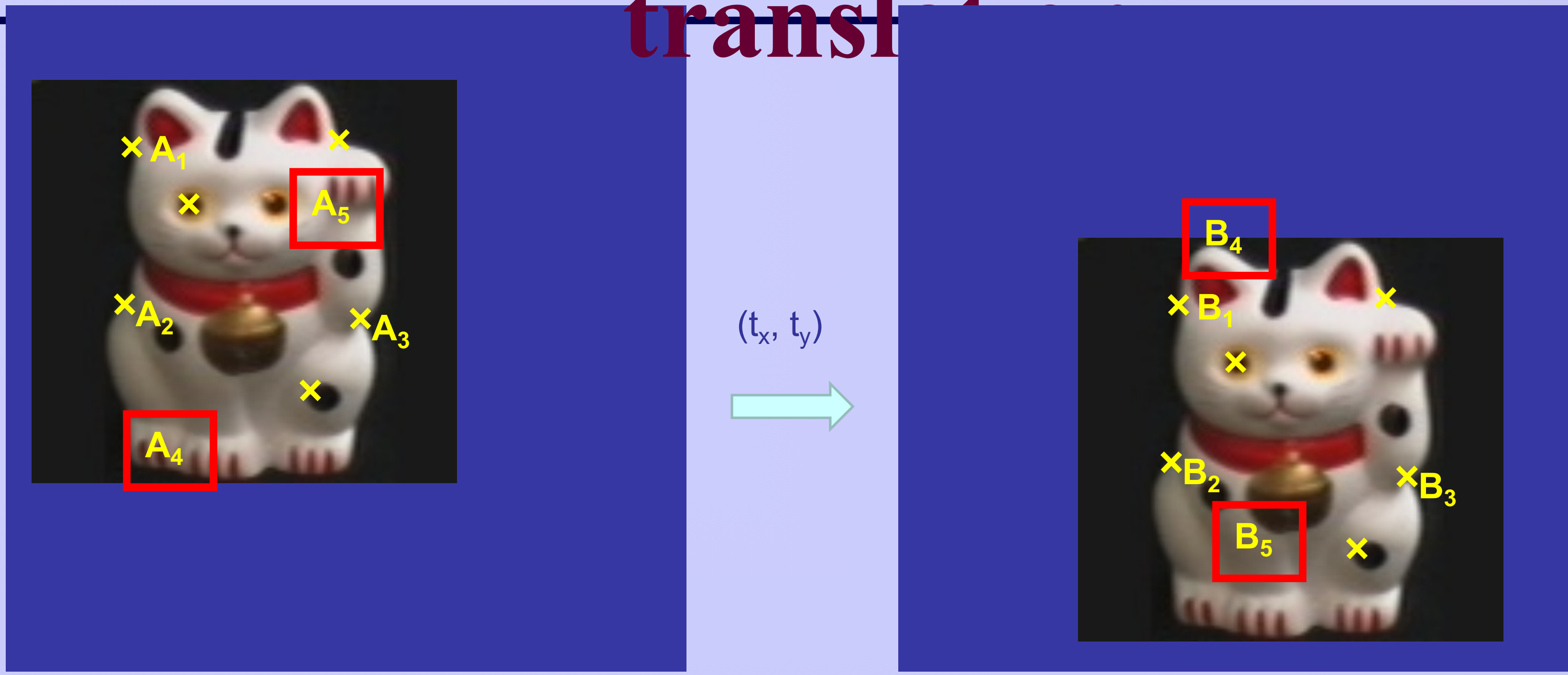
Least squares solution

1. Write down objective function
2. Derived solution
   a) Compute derivative
   b) Compute solution
3. Computational solution
   a) Write in form Ax=b
   b) Solve using pseudo-inverse or eigenvalue decomposition

$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} x_1^B - x_1^A \\ y_1^B - y_1^A \\ \vdots \\ x_n^B - x_n^A \\ y_n^B - y_n^A \end{bmatrix}$$

Example: solving for translation



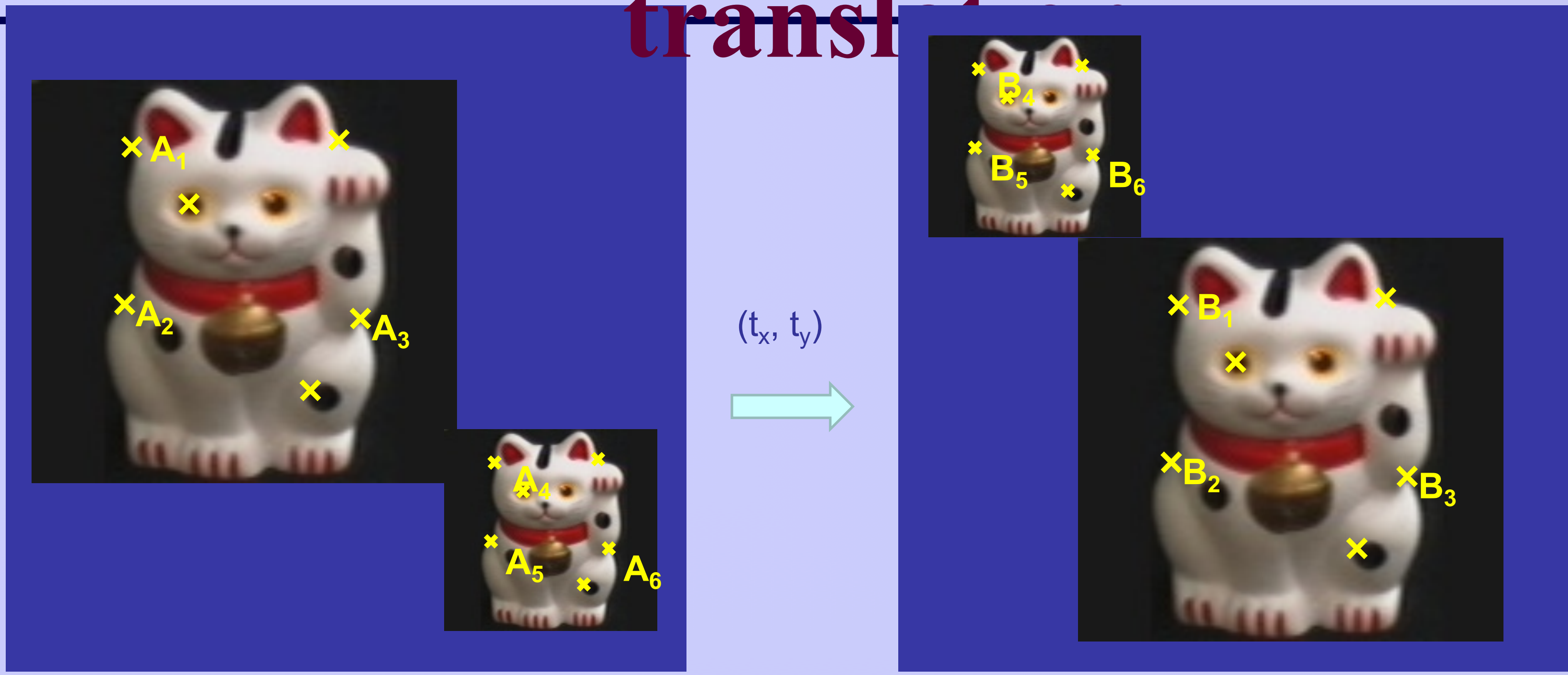
Problem: outliers

RANSAC solution

1. Sample a set of matching points (1 pair)
2. Solve for transformation parameters
3. Score parameters with number of inliers
4. Repeat steps 1-3 N times

$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Example: solving for translation



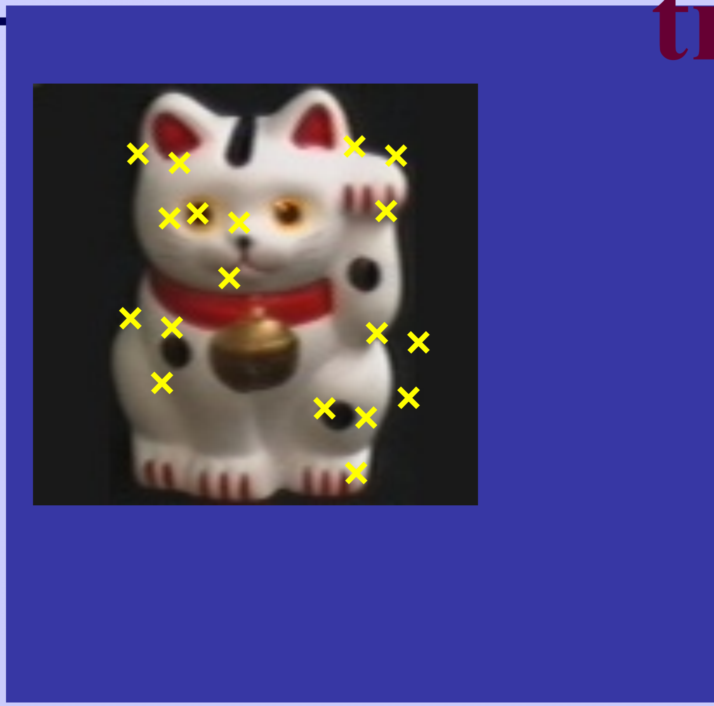
Problem: outliers, multiple objects, and/or many-to-one matches

Hough transform solution

1. Initialize a grid of parameter values
2. Each matched pair casts a vote for consistent values
3. Find the parameters with the most votes
4. Solve using least squares with inliers

$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Example: solving for translation



(t_x, t_y)

Problem: no initial guesses for correspondence

ICP solution

1. Find nearest neighbors for each point
2. Compute transform using matches
3. Move points using transform
4. Repeat steps 1-3 until convergence

$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$