CMPSCI 670: Computer Vision

Alignment

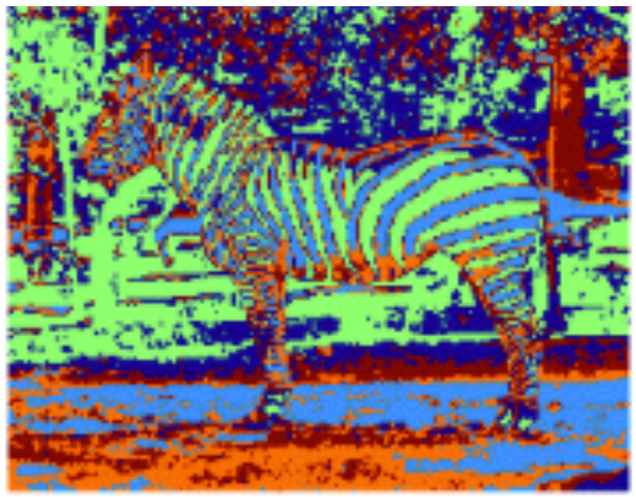
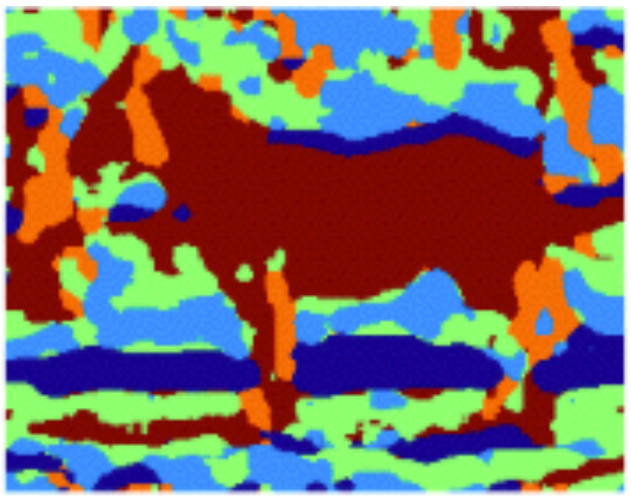
University of Massachusetts, Amherst

October 20, 2014

Instructor: Subhransu Maji

Administrivia

- Homework 2 grades posted via. email
- Homework 3 deadline is now Wed., Oct. 22
- Homework 4 will be posted on Wed.
  - Texture and grouping

- Project abstract due: October 27 (next week)
  - Won't be graded, but it is required
    - I'll provide feedback based on this

Evaluating segmentations

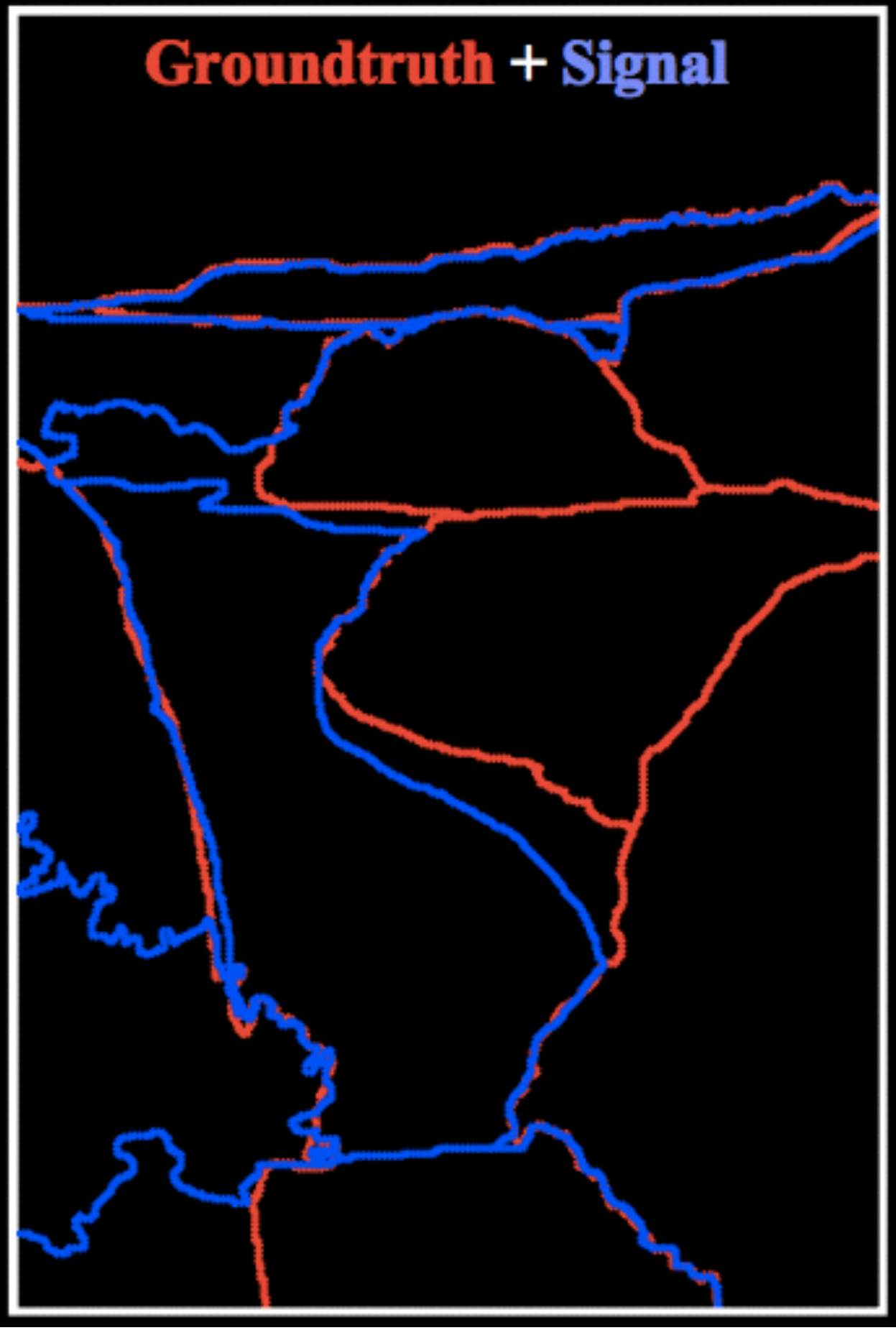
- **How do you know when a segmentation is good?**
- ~~The result should look good on these two images~~
- ~~Higher performance on the final goal we are interested in~~
  - grades, happiness, survival, …
- It did well on a standard segmentation benchmark

http://www.cis.upenn.edu/~jshi/GraphTutorial/Tutorial-ImageSegmentationGraph-cut2-Martin.pdf

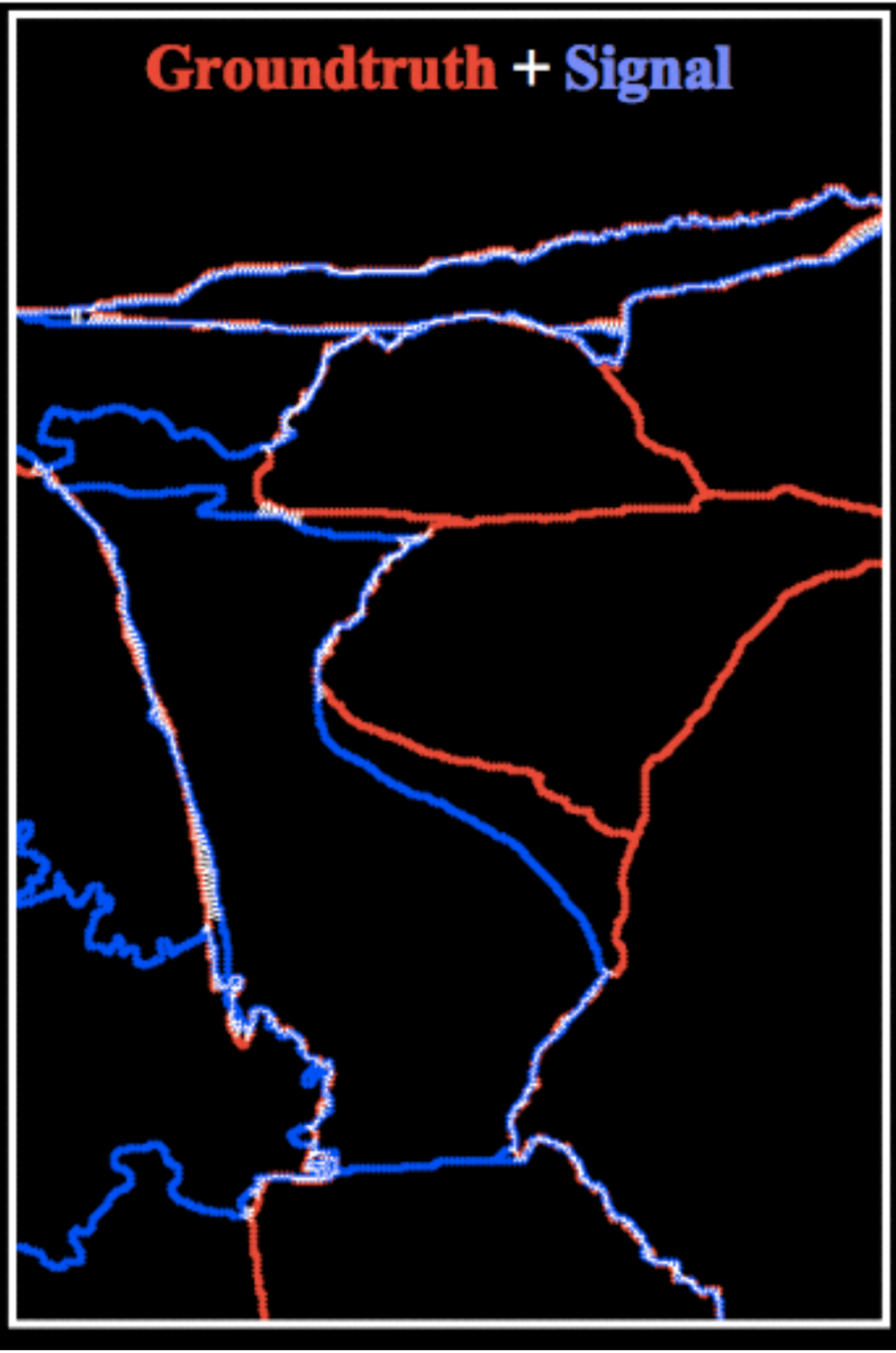
Berkeley segmentation database

Measuring accuracy

how do we correspond the signal with groundtruth?

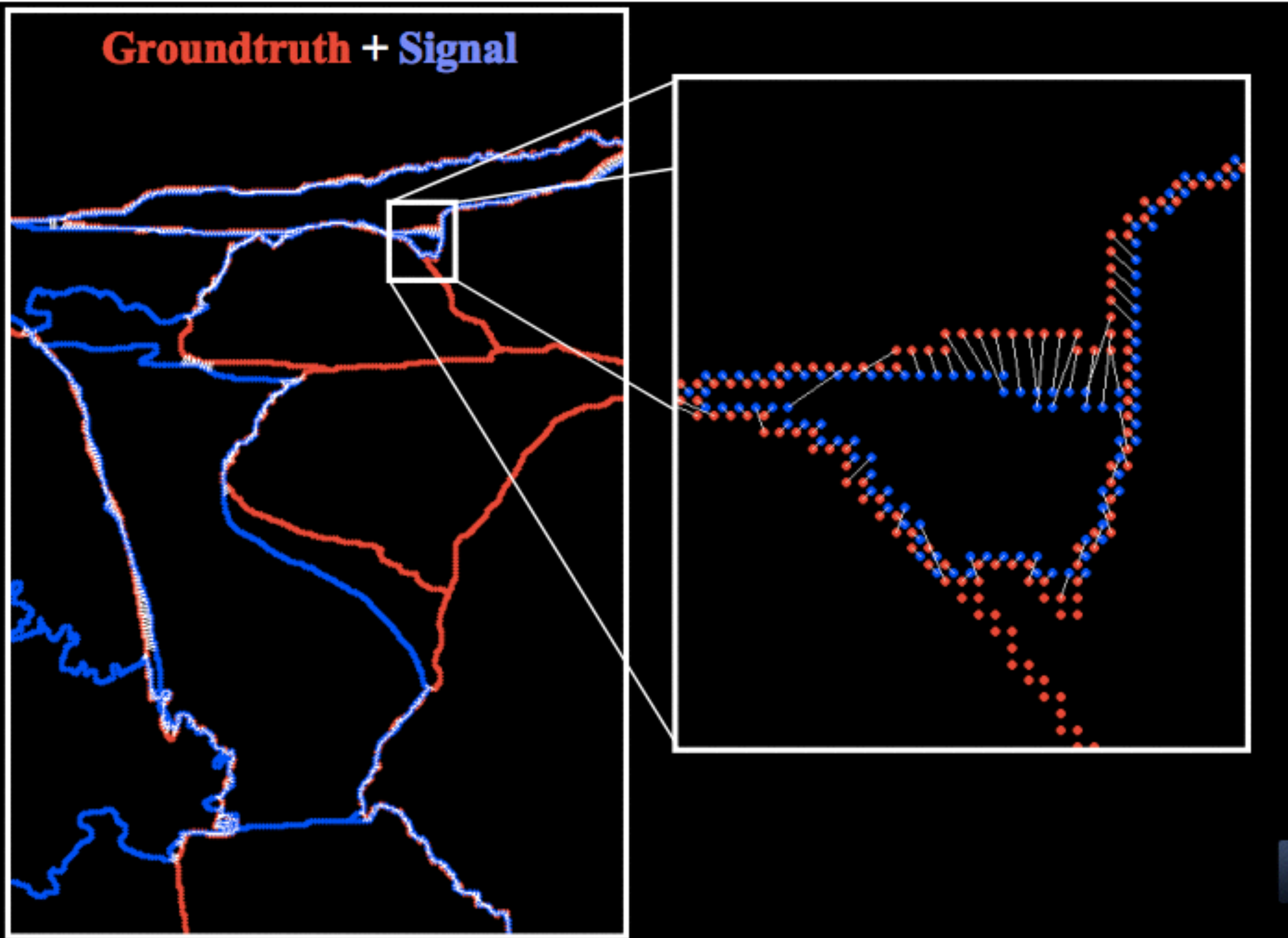


Simply overlaying does not work…

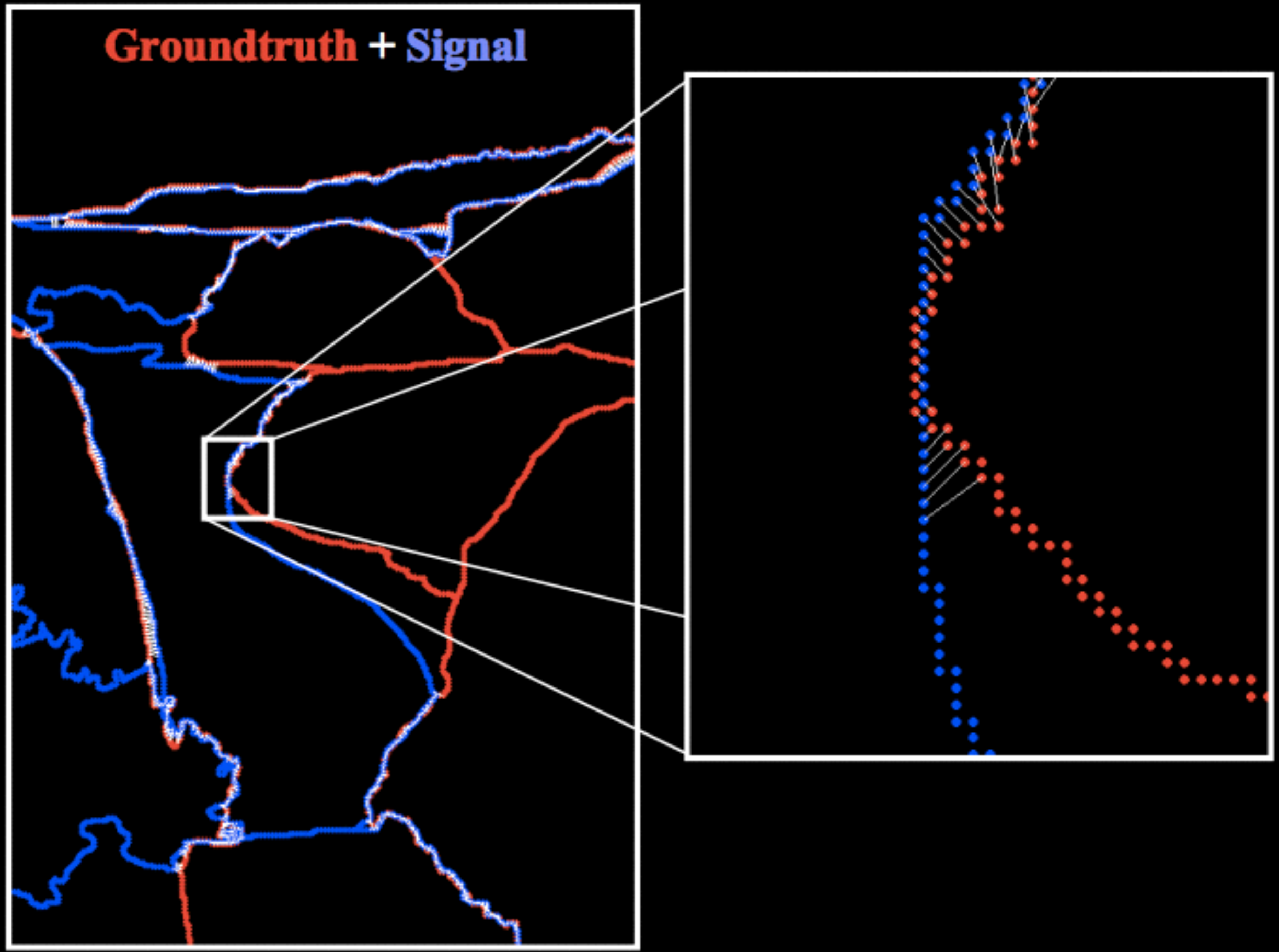


Compute correspondence via min-cost assignment on bipartite graph.

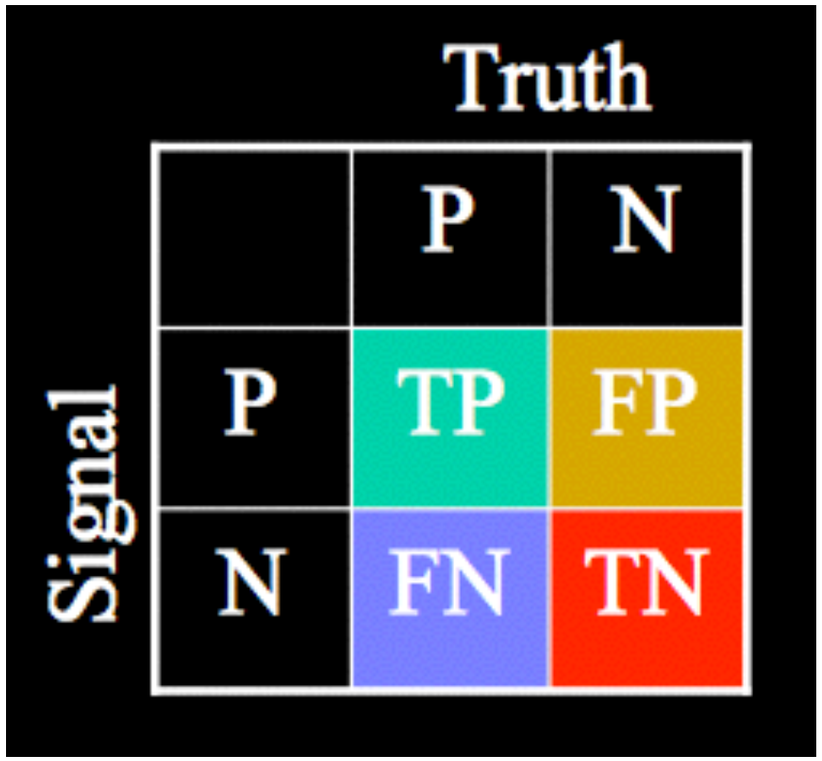
http://www.cis.upenn.edu/~jshi/GraphTutorial/Tutorial-ImageSegmentationGraph-cut2-Martin.pdf



http://www.cis.upenn.edu/~jshi/GraphTutorial/Tutorial-ImageSegmentationGraph-cut2-Martin.pdf

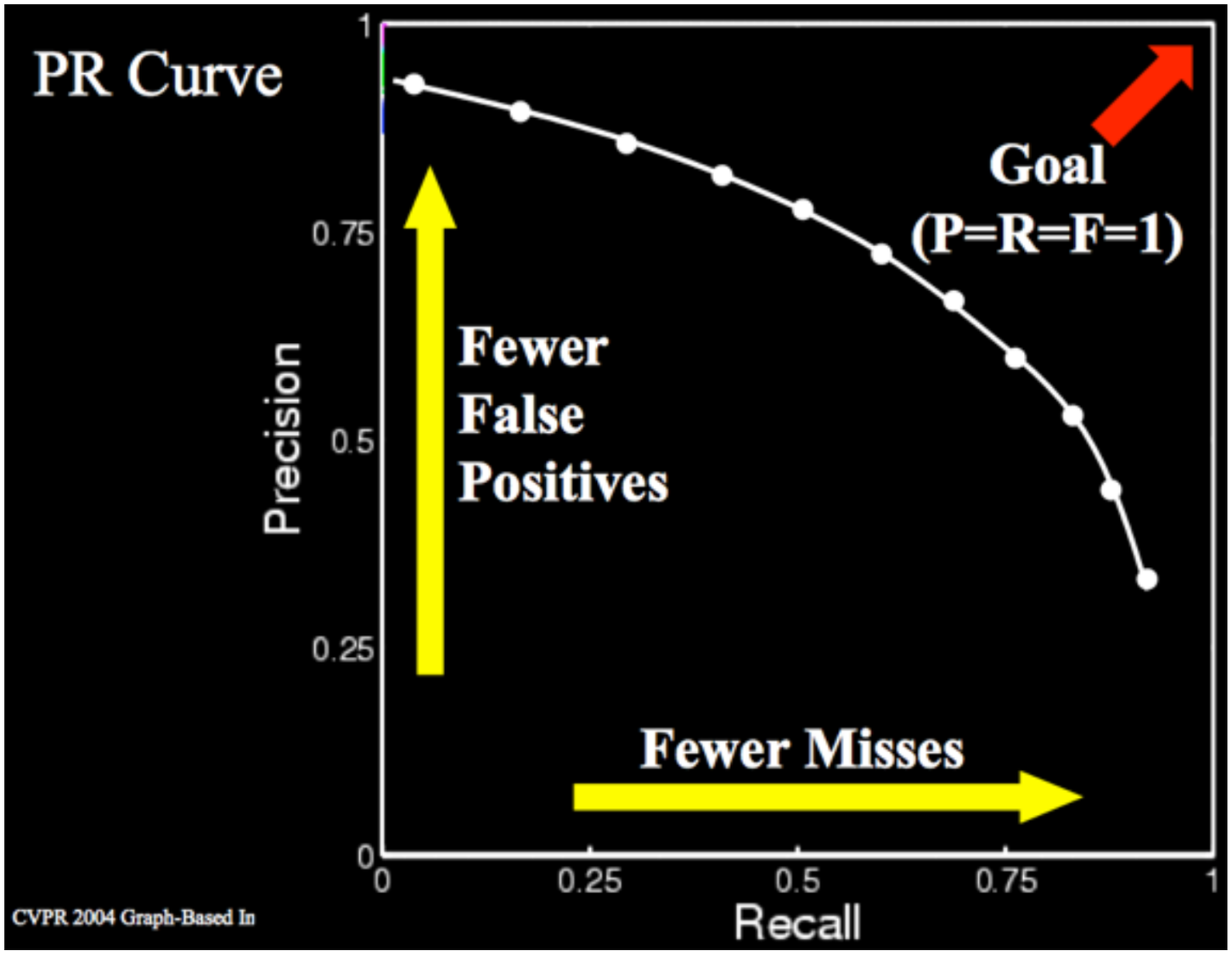

Groundtruth + Signal

Precision and Recall



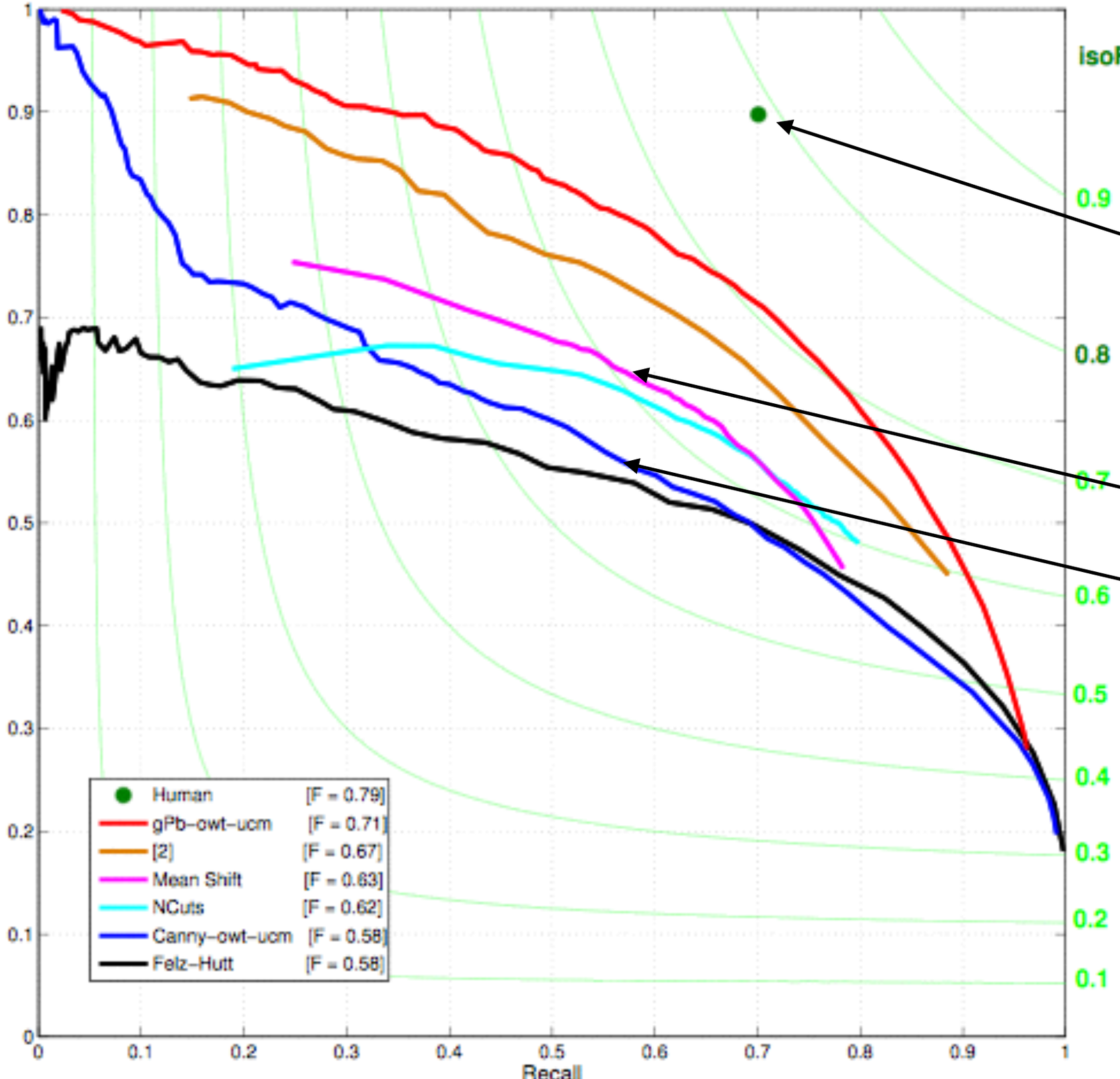
Precision = TP / (TP+FP)

Recall = TP / (TP+FN)



- Vary the threshold and plot precision vs. recall curve

Current methods on BSDS



Humans (not perfect)

Meanshift

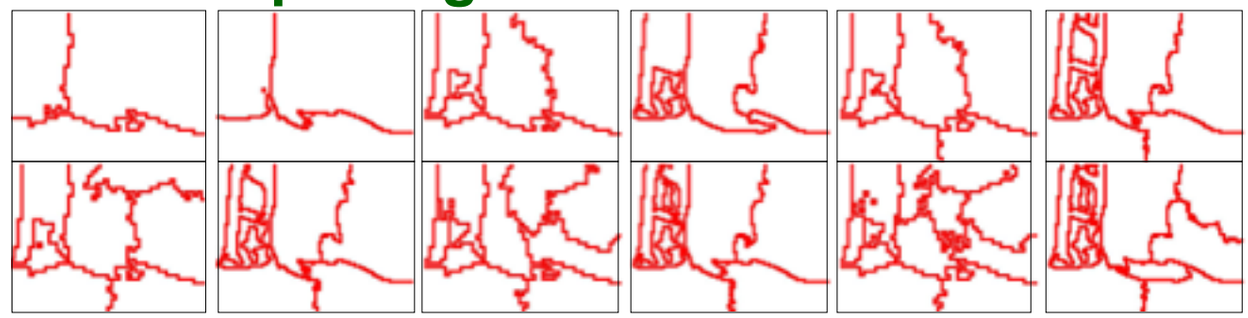
Canny

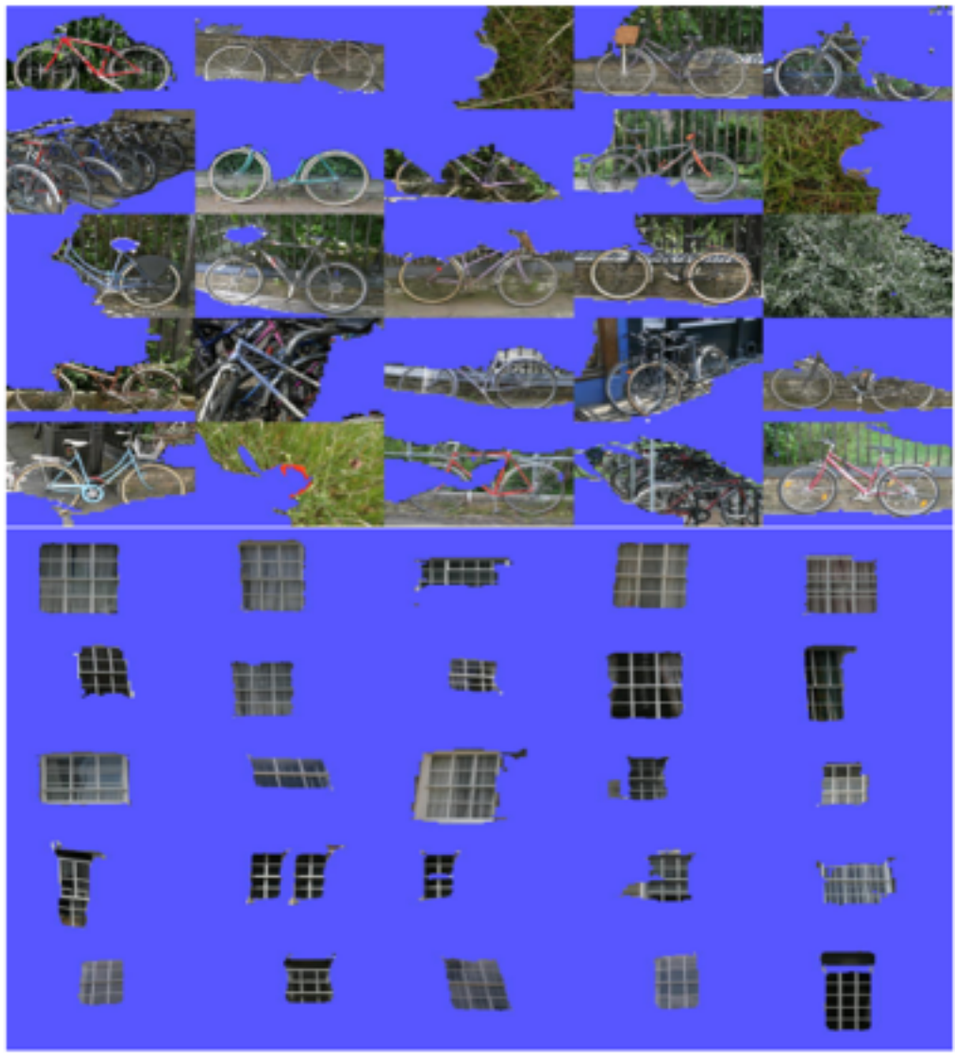
Arbelaez et al., CVPR 2009

Some applications of segmentation

Segments as primitives for recognition

Multiple segmentations



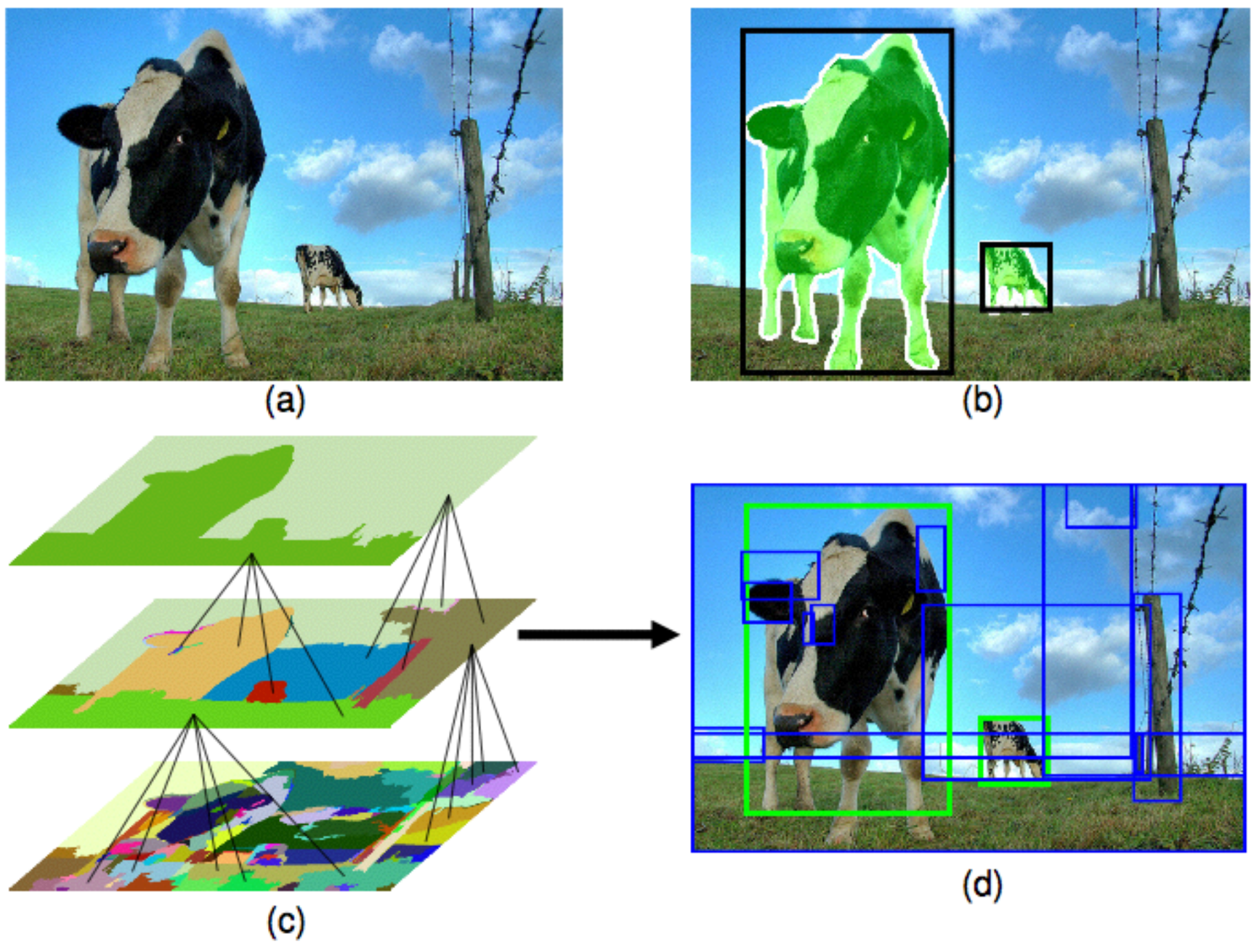


B. Russell et al., **"Using Multiple Segmentations to Discover Objects and their Extent in Image Collections,"** CVPR 2006

Slide credit: Lana Lazebnik

Regions proposals for detection

- Generate a set of regions for further classification



Van de Sande et al., ICCV 2011

Motion segmentation

grouping pixels over space and time

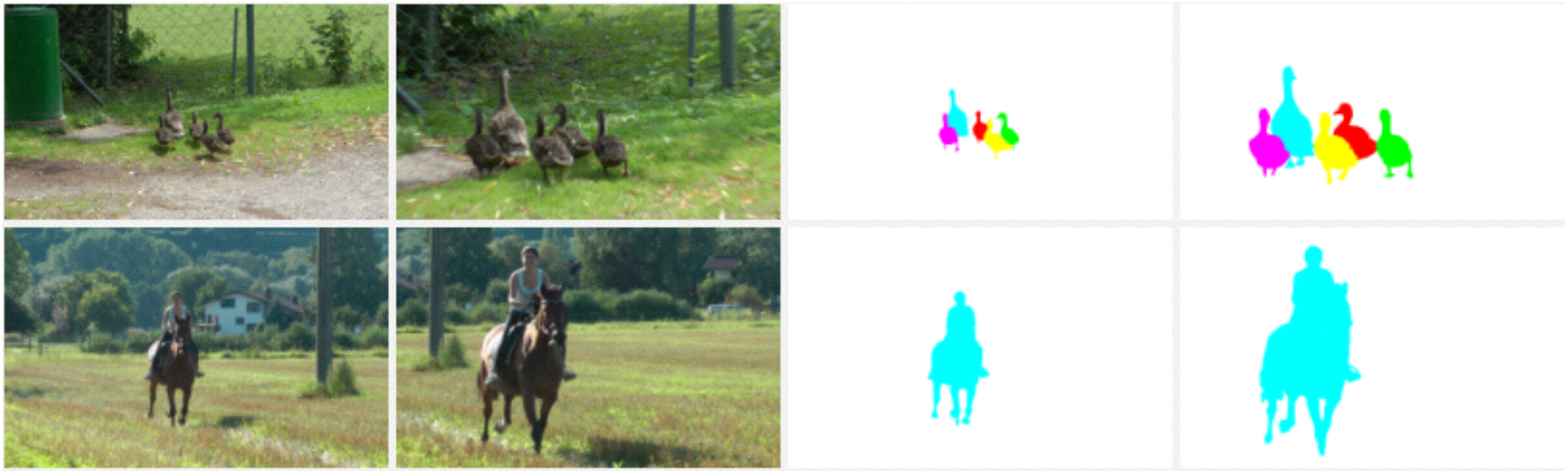


image sequence segmentation

http://lmb.informatik.uni-freiburg.de/resources/datasets/moseg.en.html

P. Ochs, J. Malik, T. Brox
Segmentation of moving objects by long term video analysis,
IEEE Transactions on Pattern Analysis and Machine Intelligence, 2014.

Summary of grouping

- Segmentation to find object boundaries or mid-level regions, tokens.
- Bottom-up segmentation via clustering
  - Choices — features, affinity functions, clustering algorithms
- Grouping methods also useful for quantization, can create new feature summaries
  - Texton histograms for texture within local region
- Example clustering algorithms
  - k-means
  - mean shift
  - min cut, normalized cut
- Segmentation quality **can** be measured (BSDS)

Further thoughts and readings ..

- Gestalt psychology **http://en.wikipedia.org/wiki/Gestalt_psychology**
- Chapter 5, Richard Szeliski's book
- **Berkeley segmentation database and benchmark**
  - Also read about the Berkeley boundary detector
- **http://www.cis.upenn.edu/~jshi/GraphTutorial/**
- Image segmentation via. graph cuts
  - Boykov and Jolly, **Interactive graph cuts for optimal boundary & region segmentation of objects in ND images**, ICCV 2001
- Normalized cuts for image segmentation (Shi and Malik)
  - **http://www.cs.berkeley.edu/~malik/papers/SM-ncut.pdf**
- Biased normalized cuts
  - **http://people.cs.umass.edu/~smaji/projects/biasedNcuts/index.html**

Today: Alignment

- What are the alignment problems in computer vision?
  - rigid vs. deformable (non-rigid)

- Good features to match
  - Invariance properties
  - Local features: the SIFT descriptor

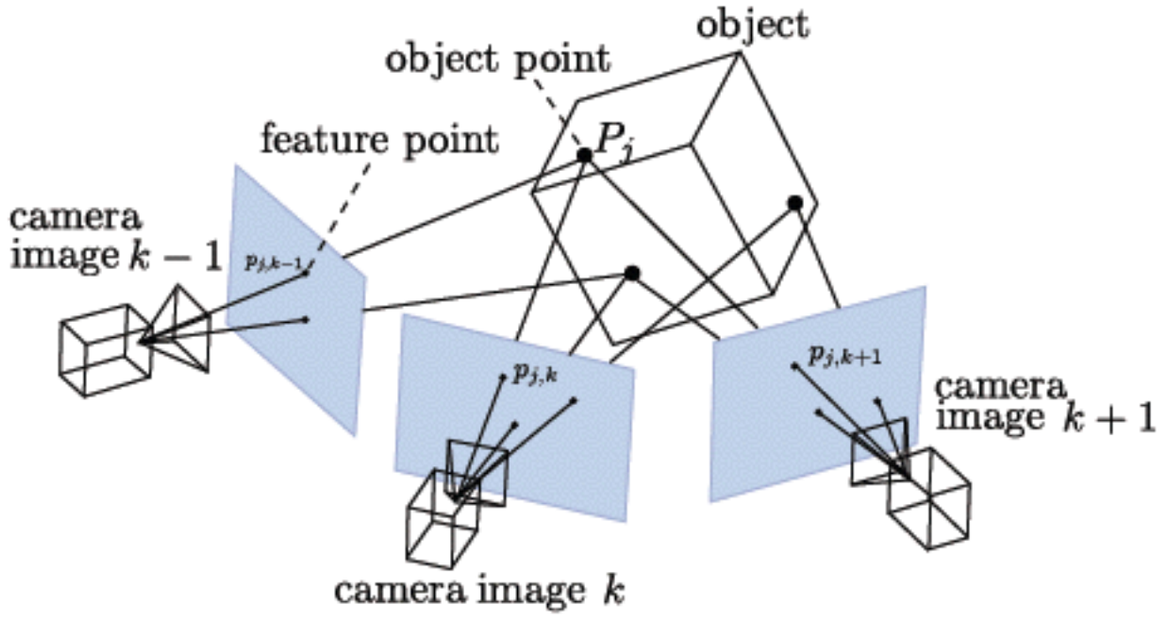
- Alignment algorithms
  - Rigid alignment: RANSAC
    - Application: panoramic photo stitching
  - Non-rigid alignment
    - Application: shape matching

Panoramic photo stitching

http://stayhealthy-stayhappy.com/2013/03/13/get-outside-but-remember/

- Align several photographs to form a large one

Estimating 3D structure



http://openmvg.readthedocs.org/en/latest/software/SfM/SfM/

http://www.cs.cornell.edu/~snavely/bundler/

Large-scale instance retrieval

- Matching under changes in viewpoint, illumination, translation and scale

http://www.robots.ox.ac.uk/~vgg/research/vgoogle/index.html

Oxford building search

http://www.robots.ox.ac.uk/~vgg/research/oxbuildings/index.html

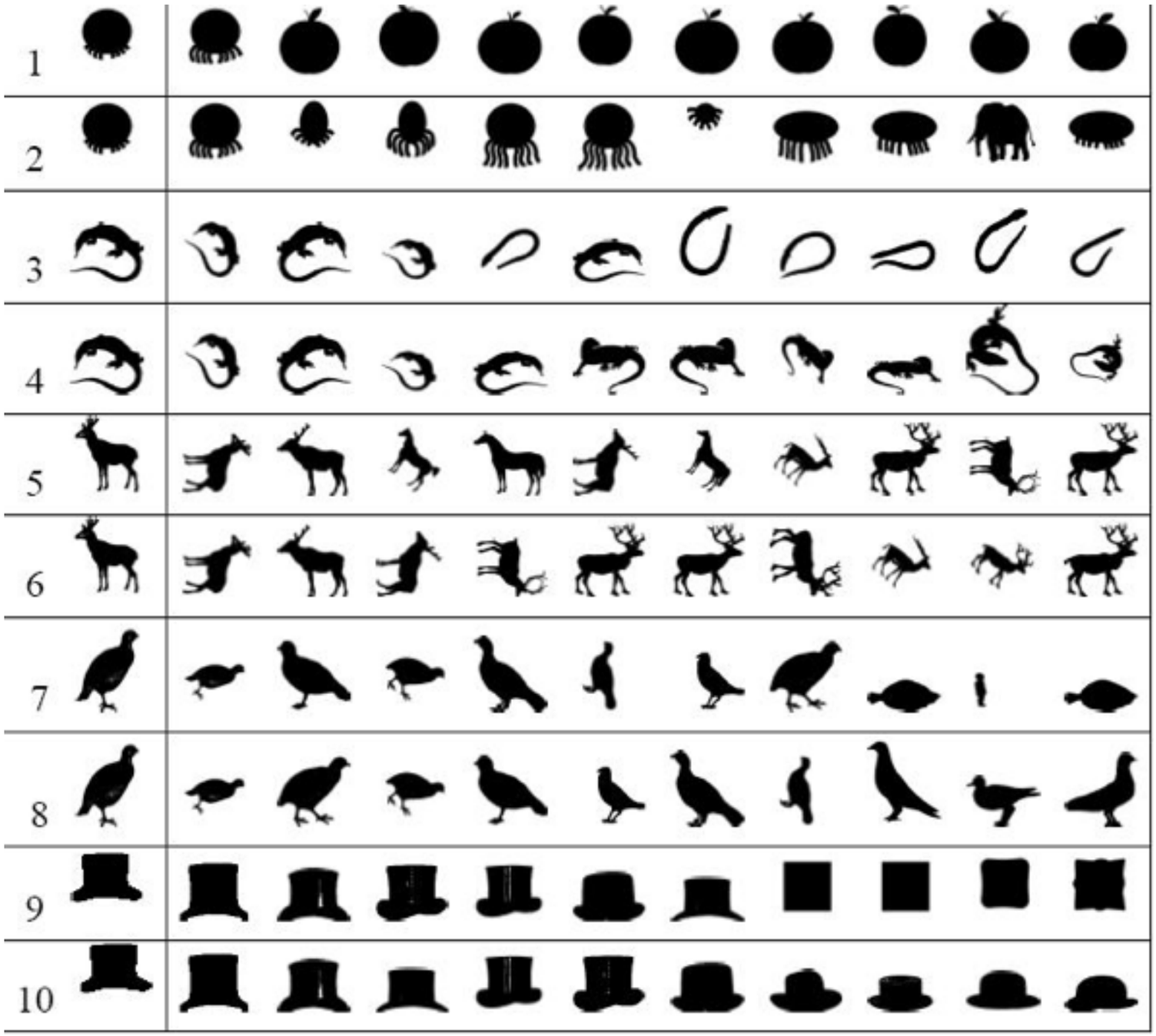
Face alignment

http://vis-www.cs.umass.edu/faceAlignment/index.html

Y. Zhou, W. Zhang, X. Tang, and H. Shum. A Bayesian mixture model for multi-view face alignment. CVPR, 2005

- Complicated deformation

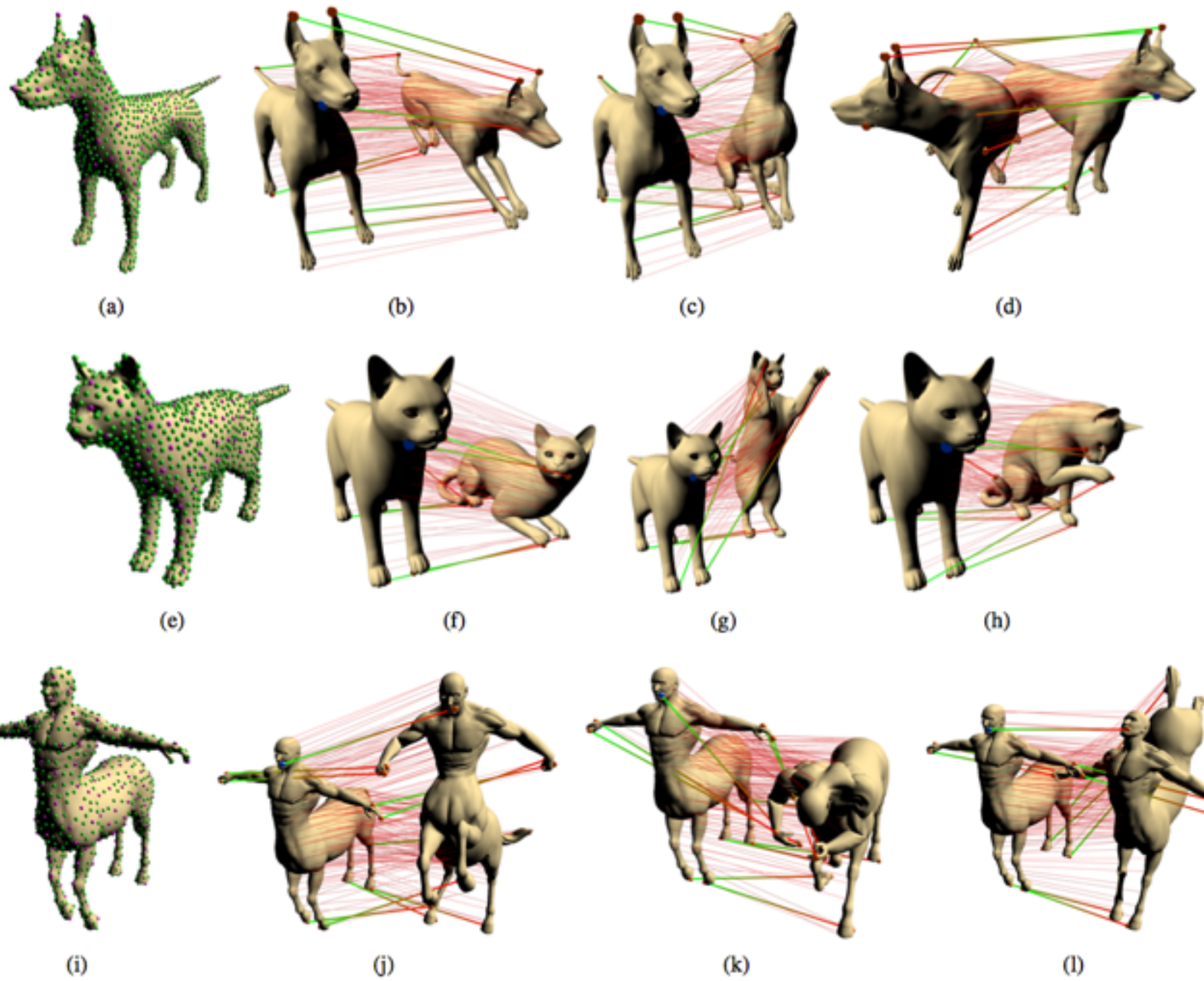
Shape matching



http://www.dabi.temple.edu/~shape/MPEG7/

- Retrieve objects within the same class

3D shape matching



http://www.tevs.eu/project_eg11.html

- Partial matching of object instances in 3D

Alignment problems

- **Rigid alignment**
  - matching objects under viewpoint change
  - deformations: scale, translation, rotation, affine.
- Examples:
  - Panoramic photo stitching
  - 3D shape reconstruction
  - Retrieval of buildings, objects, etc., from large datasets

- **Deformable alignment**
  - matching objects under more general changes
  - class of deformations potentially very large
- Examples:
  - aligning deformable objects such as faces, handwritten digits, articulated humans
  - 2d shape matching
  - 3d shape matching

Alignment

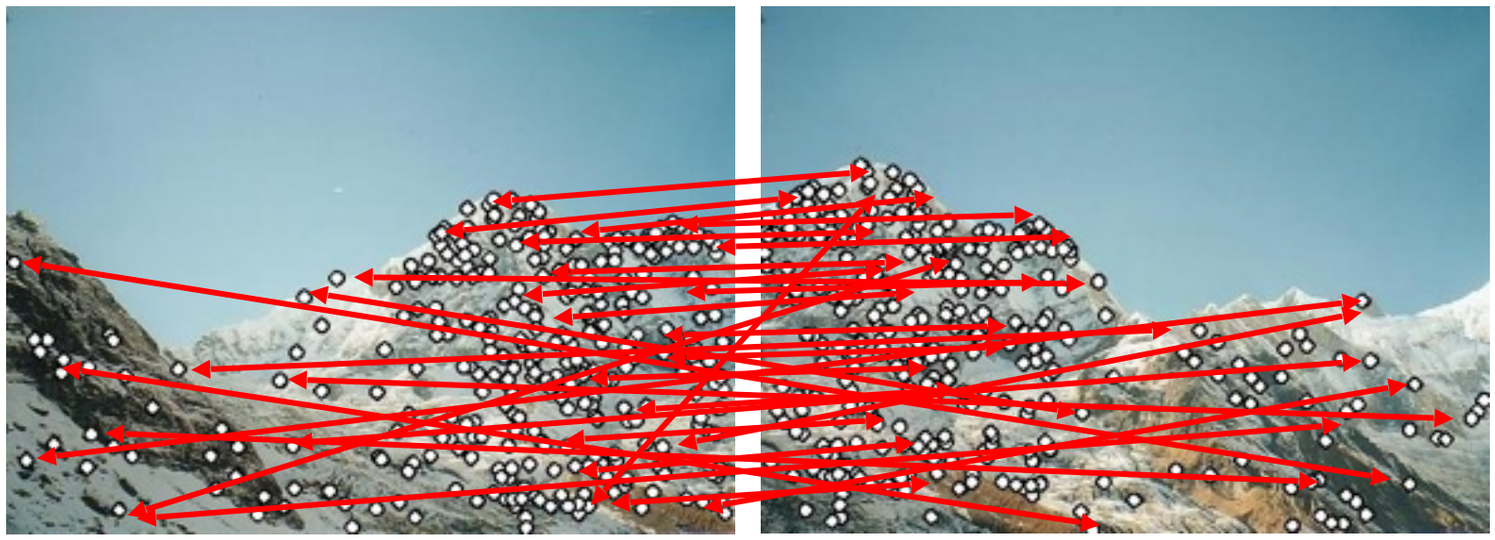
- What are the alignment problems in computer vision?
  - rigid vs. deformable (non-rigid)

- Good features to match
  - Invariance properties
  - Local features: the SIFT descriptor

- Alignment algorithms
  - Rigid alignment: RANSAC
    - Application: panoramic photo stitching
  - Non-rigid alignment
    - Application: shape matching

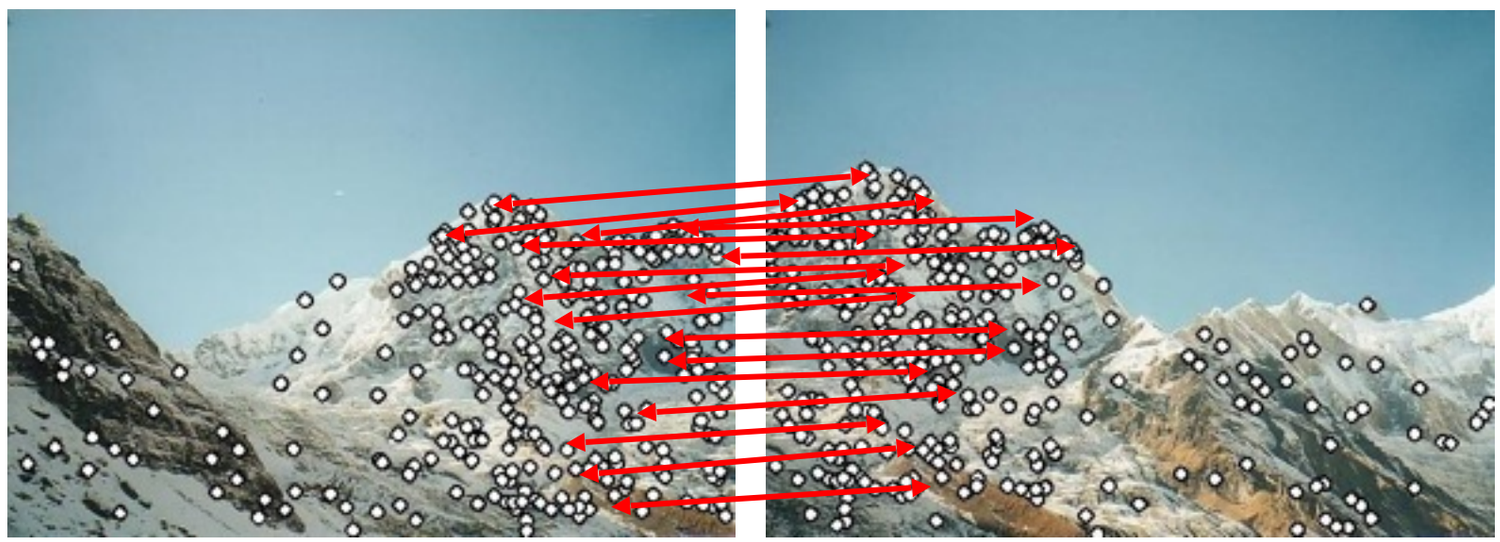
A framework for alignment

- Matching local features
  - Local information used, can contain outliers
  - But hopefully enough of these matches are good

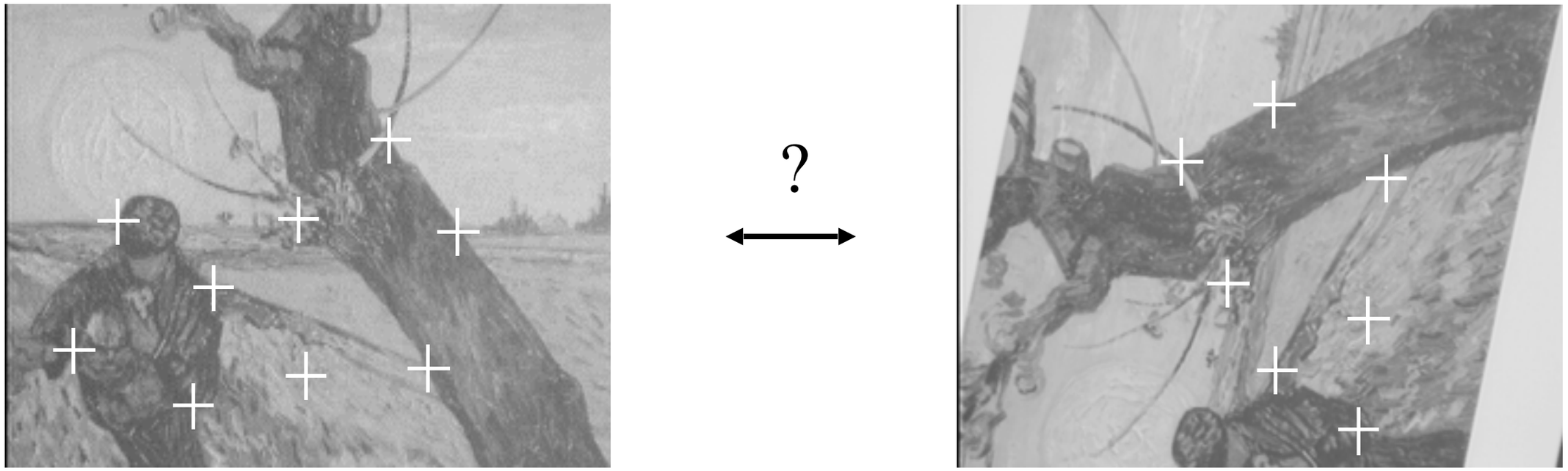


A framework for alignment

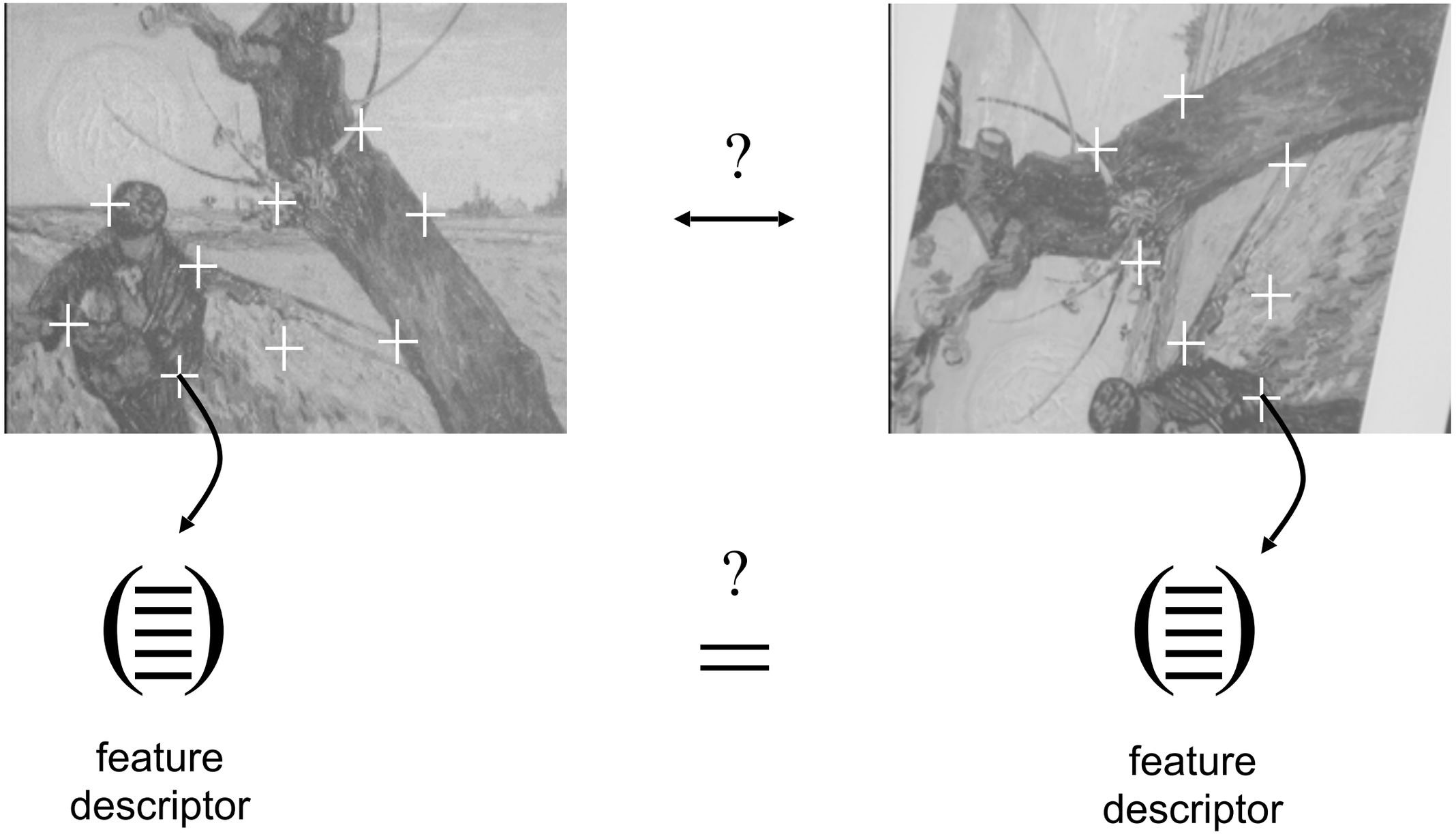
- Matching local features
  - Local information used, can contain outliers
  - But hopefully enough of these matches are good
- Consensus building
  - Aggregate the good matches and find a transformation that explains these matches



Generating putative correspondences



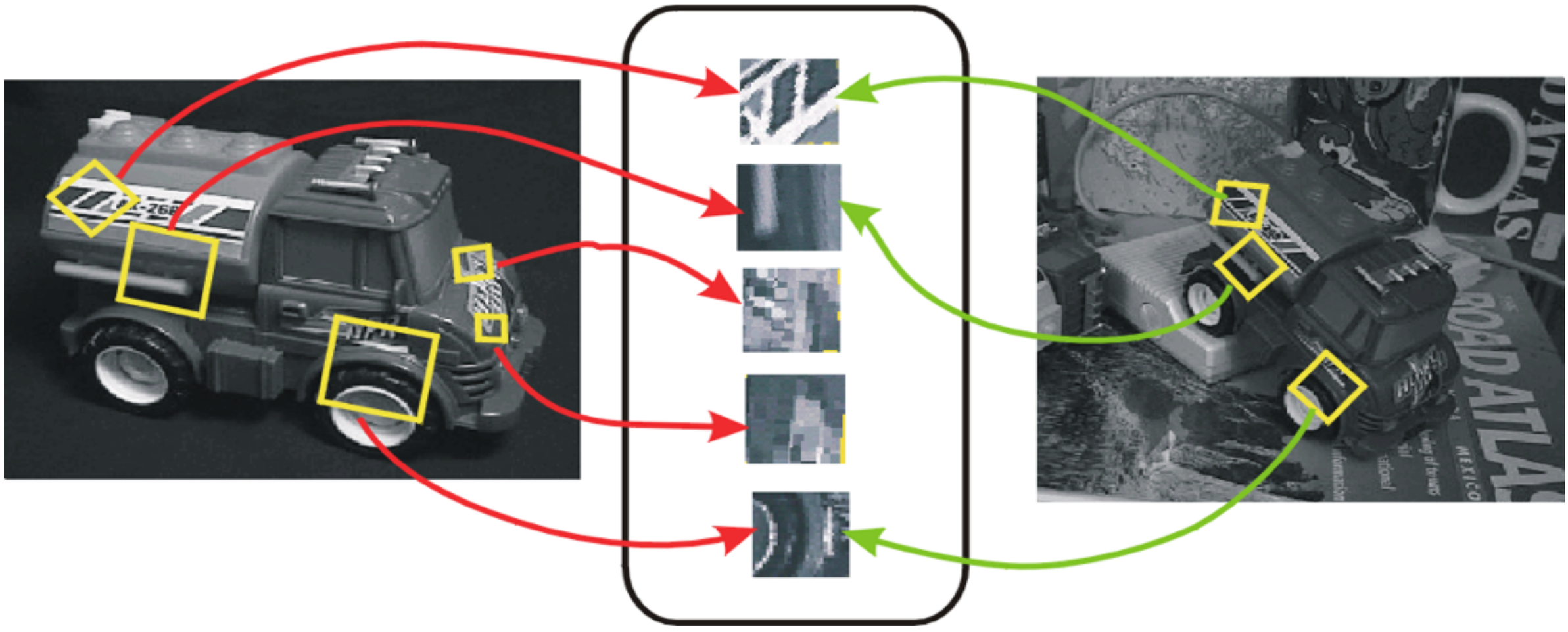
Generating putative correspondences



- Need to compare *feature descriptors* of local patches surrounding interest points

Feature descriptors

- **Recall**: feature detection and description



Feature descriptors

- Simplest descriptor: vector of raw intensity values
- How to compare two such vectors?
  - Sum of squared differences (SSD)

$$\mathrm{SSD}(\mathbf{u},\mathbf{v}) = \sum_i (u_i - v_i)^2$$

    - Not invariant to intensity change

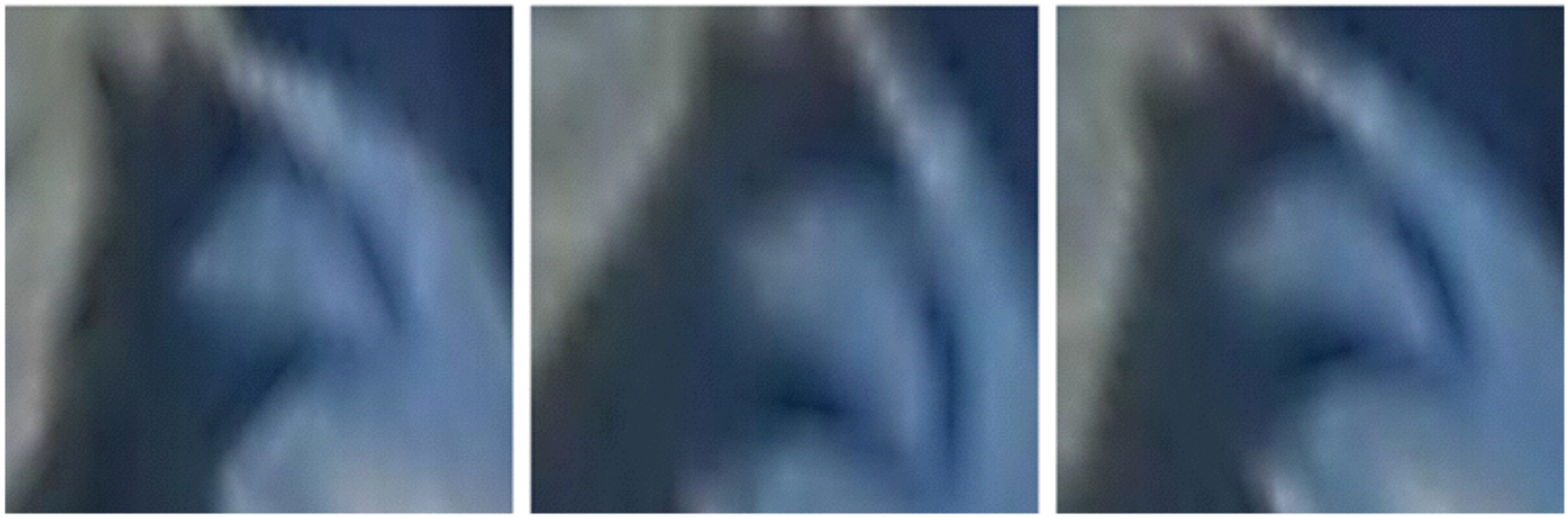
  - Normalized correlation

$$\rho(\mathbf{u},\mathbf{v}) = \frac{(\mathbf{u}-\bar{\mathbf{u}})}{\|\mathbf{u}-\bar{\mathbf{u}}\|} \cdot \frac{(\mathbf{v}-\bar{\mathbf{v}})}{\|\mathbf{v}-\bar{\mathbf{v}}\|} = \frac{\sum_i (u_i - \bar{\mathbf{u}})(v_i - \bar{\mathbf{v}})}{\sqrt{\left(\sum_j (u_j - \bar{\mathbf{u}})^2\right)\left(\sum_j (v_j - \bar{\mathbf{v}})^2\right)}}$$

    - Invariant to affine intensity change

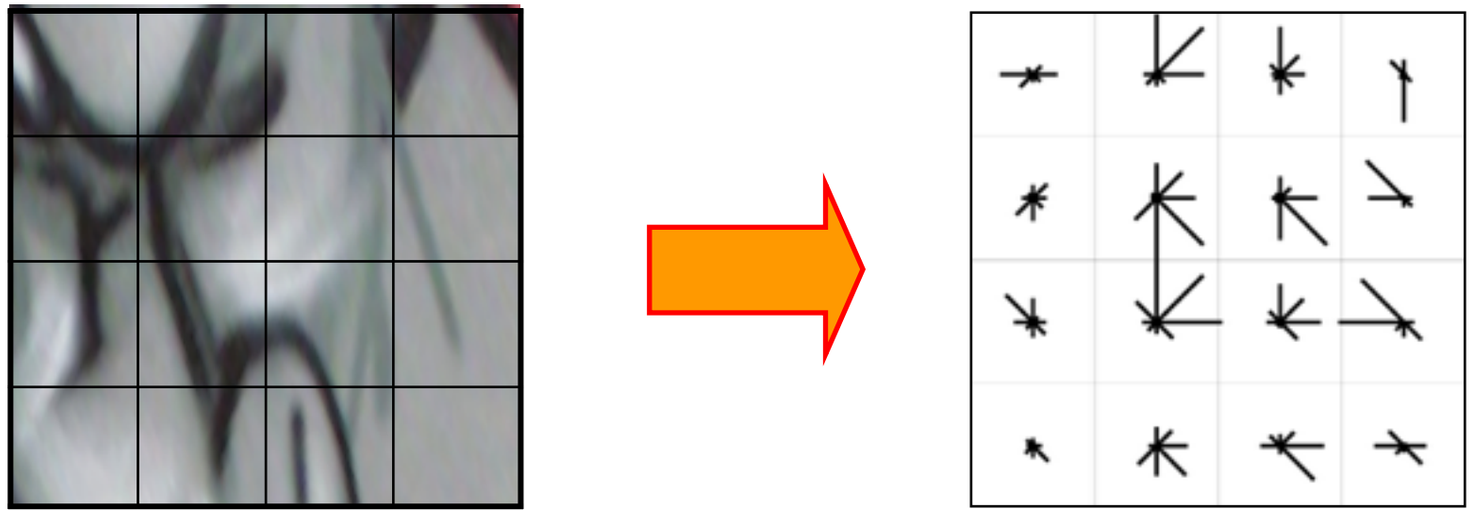
Disadvantage of intensity vectors as descriptors

- Small deformations can affect the matching score a lot



Feature descriptors: SIFT

- Descriptor computation:
  - Divide patch into 4x4 sub-patches
  - Compute histogram of gradient orientations (8 reference angles) inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions



David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

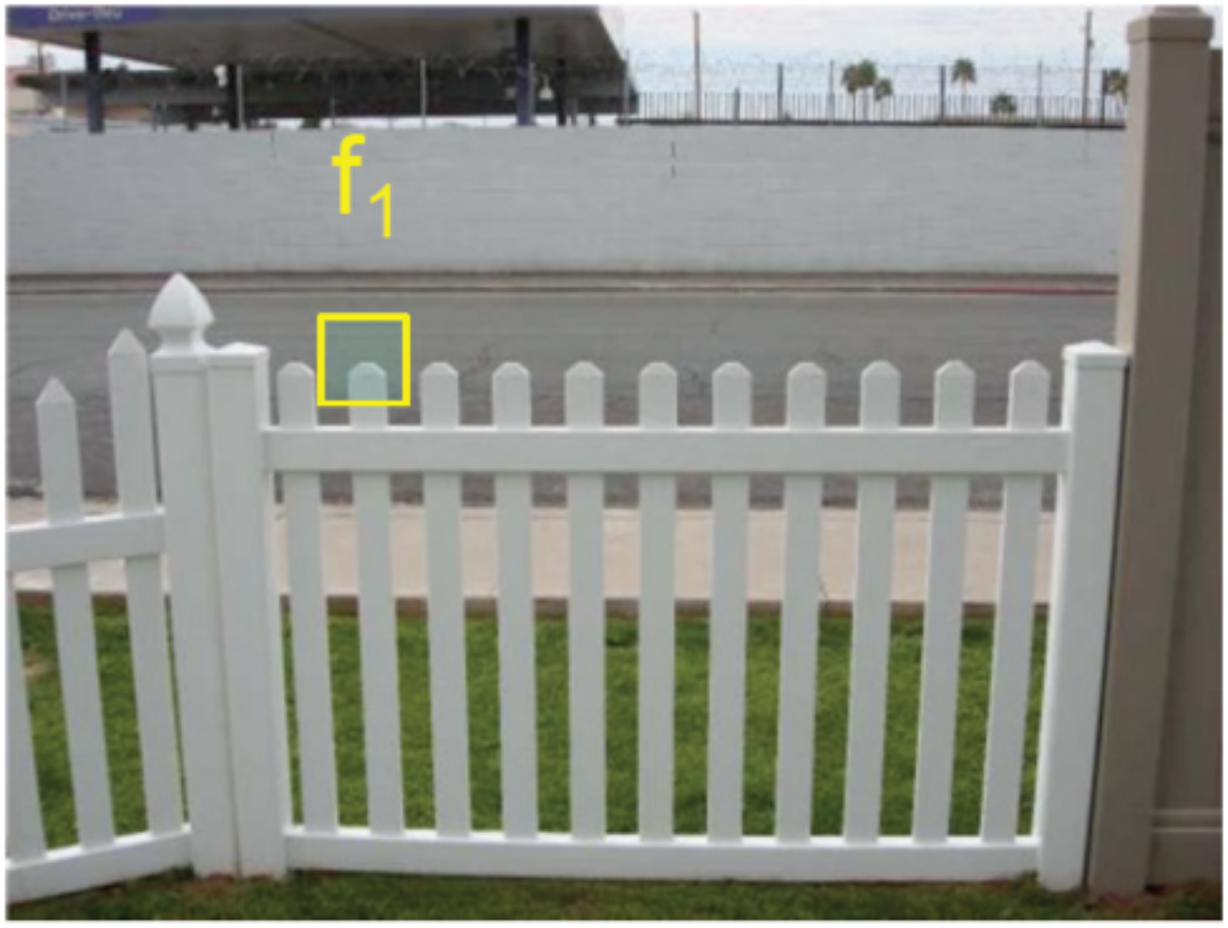
Feature descriptors: SIFT

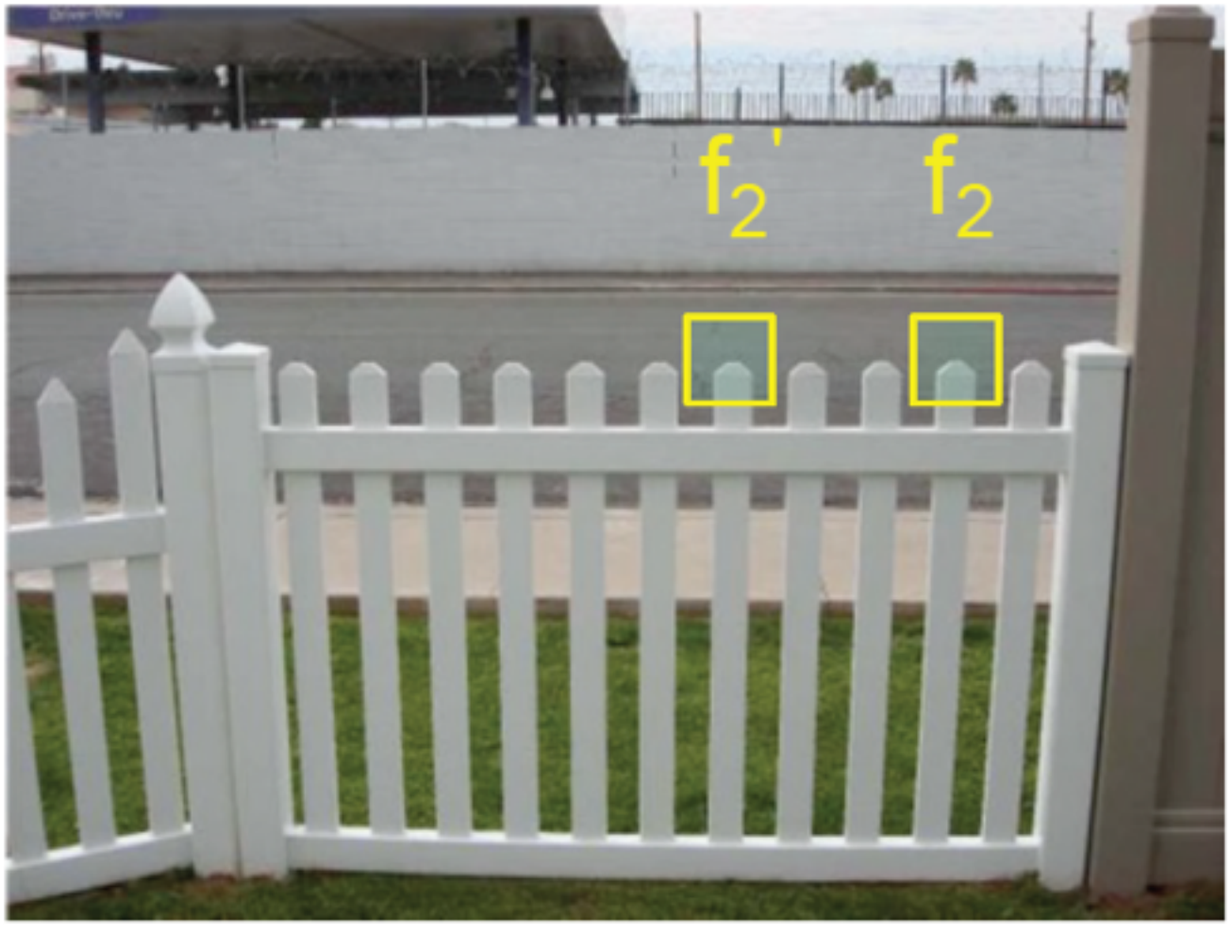
- Descriptor computation:
  - Divide patch into 4x4 sub-patches
  - Compute histogram of gradient orientations (8 reference angles) inside each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions

- Advantage over raw vectors of pixel values
  - Gradients less sensitive to illumination change
  - Pooling of gradients over the sub-patches achieves robustness to small shifts, but still preserves some spatial information

David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Problem: Ambiguous putative matches

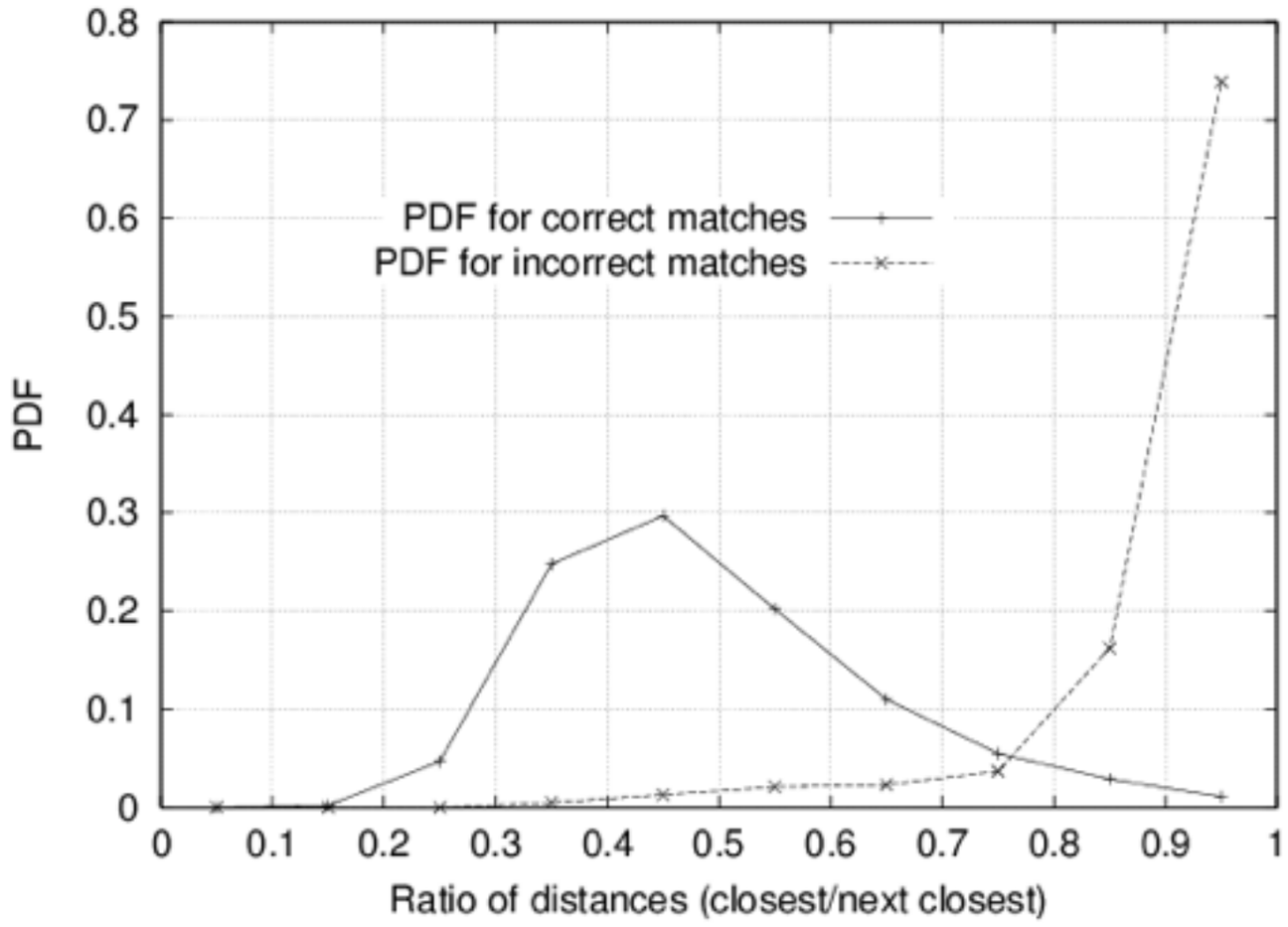




Source: Y. Furukawa

Rejection of unreliable matches

- How can we tell which putative matches are more reliable?
- Heuristic: compare distance of **nearest** neighbor to that of **second** nearest neighbor
  - Ratio of closest distance to second-closest distance will be *high* for features that are *not* distinctive



Threshold of 0.8 provides good separation

David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Alignment

- What are the alignment problems in computer vision?
  - rigid vs. deformable (non-rigid)

- Good features to match
  - Invariance properties
  - Local features: the SIFT descriptor

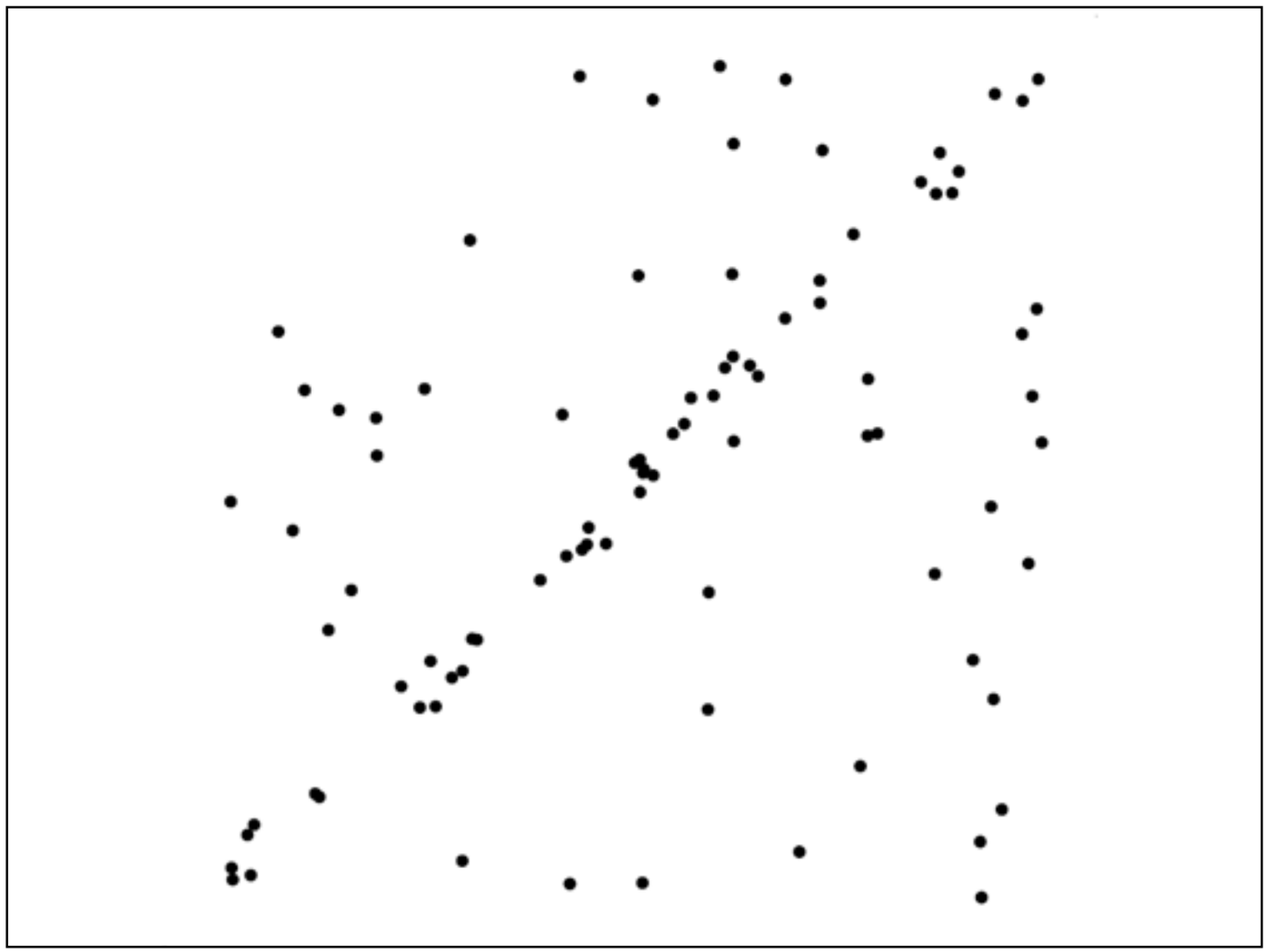
- Alignment algorithms
  - Rigid alignment: RANSAC
    - Application: panoramic photo stitching
  - Non-rigid alignment
    - Application: shape matching

RANSAC

- **Ran**dom **Sa**mple **C**onsensus
  - Choose a small subset of points uniformly at random
  - Fit a model to that subset
  - Find all remaining points that are “close” to the model and reject the rest as outliers
  - Do this many times and choose the best model
- For rigid transformation we can estimate the parameters of the **transformation**, e.g., rotation angle, scaling, translation, etc, from **putative** correspondence matches
- Lets see how RANSAC works for a simple example.

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

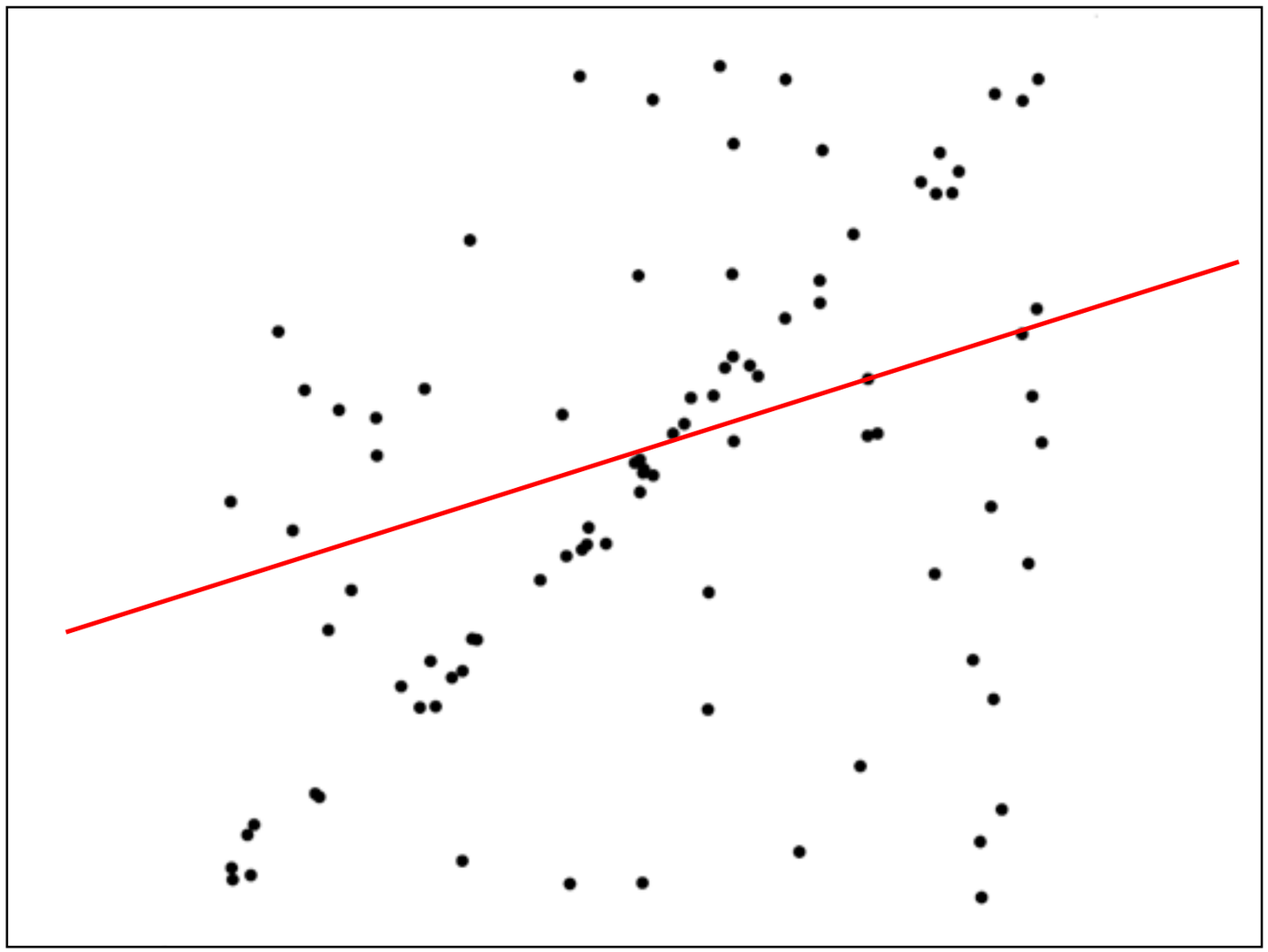
RANSAC for line fitting example



Source: R. Raguram

RANSAC for line fitting example

$$\min_{a,b} \sum_i (ax_i + b - y_i)^2$$



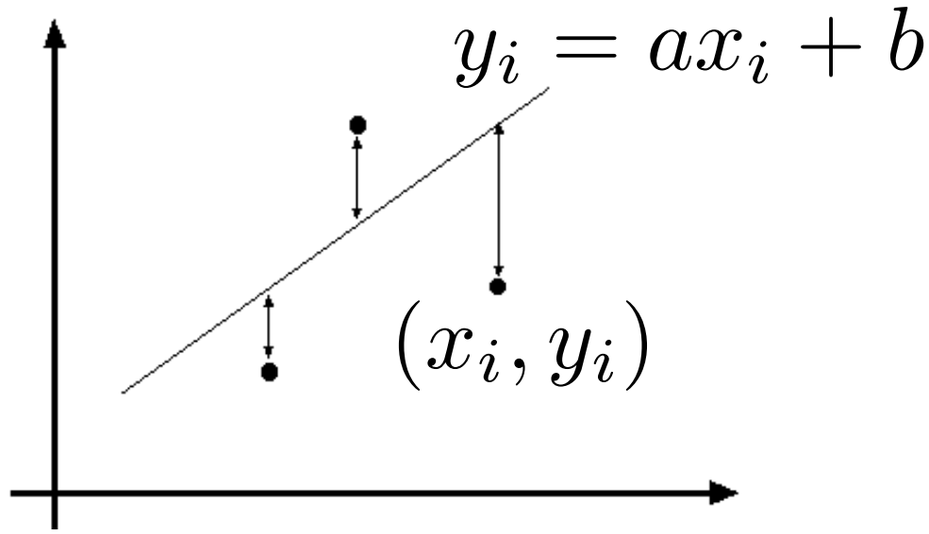
Least-squares fit

Source: R. Raguram

Least-squares line fitting

- Data: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Line equation: $y_i = ax_i + b$

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

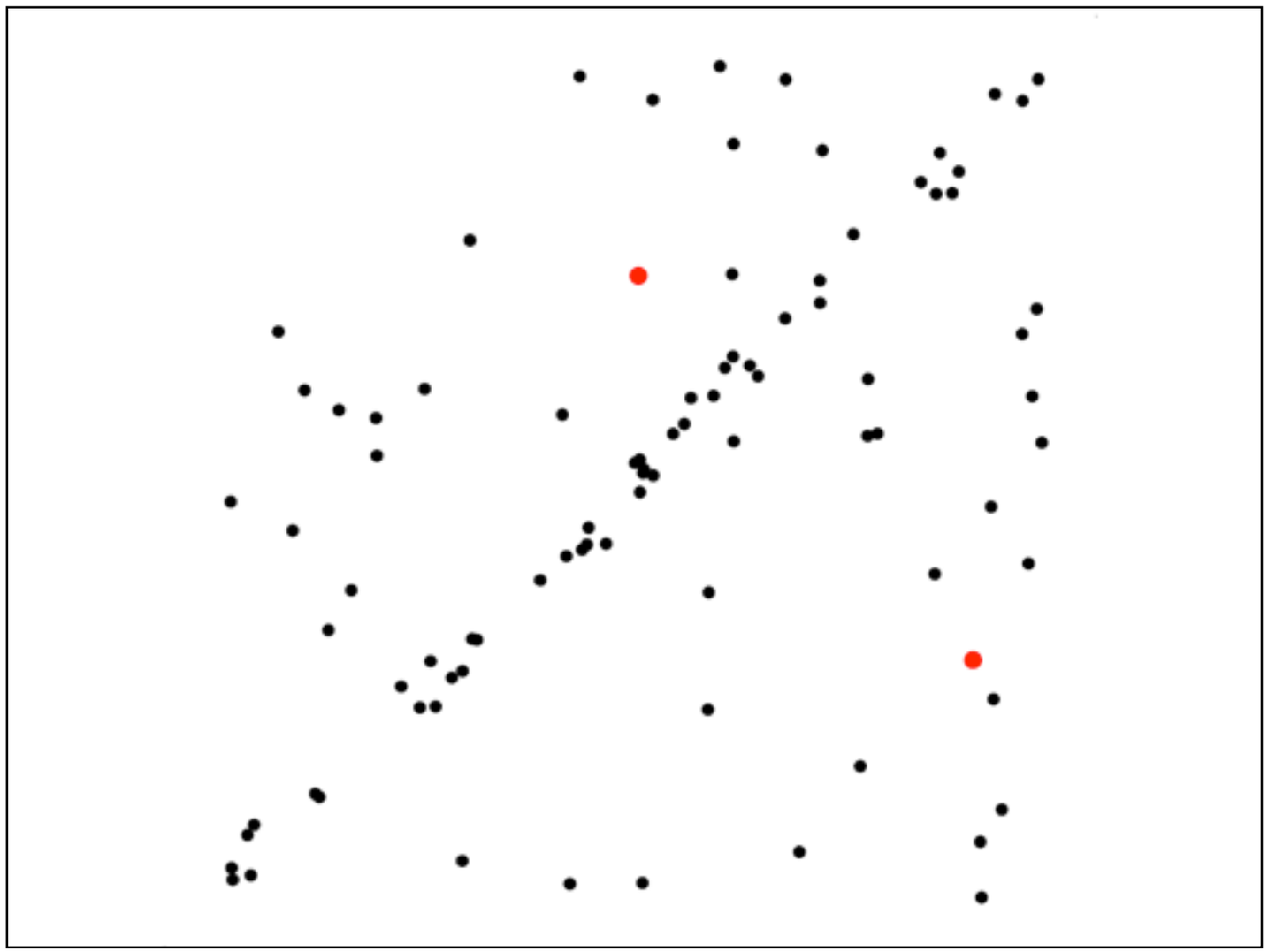


$$\frac{dE}{da} = \sum_{i=1}^{n} 2\,(ax_i + b - y_i)\,x_i = 0 \qquad \frac{dE}{db} = \sum_{i=1}^{n} 2\,(ax_i + b - y_i) = 0$$

$$a = \frac{n\sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

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

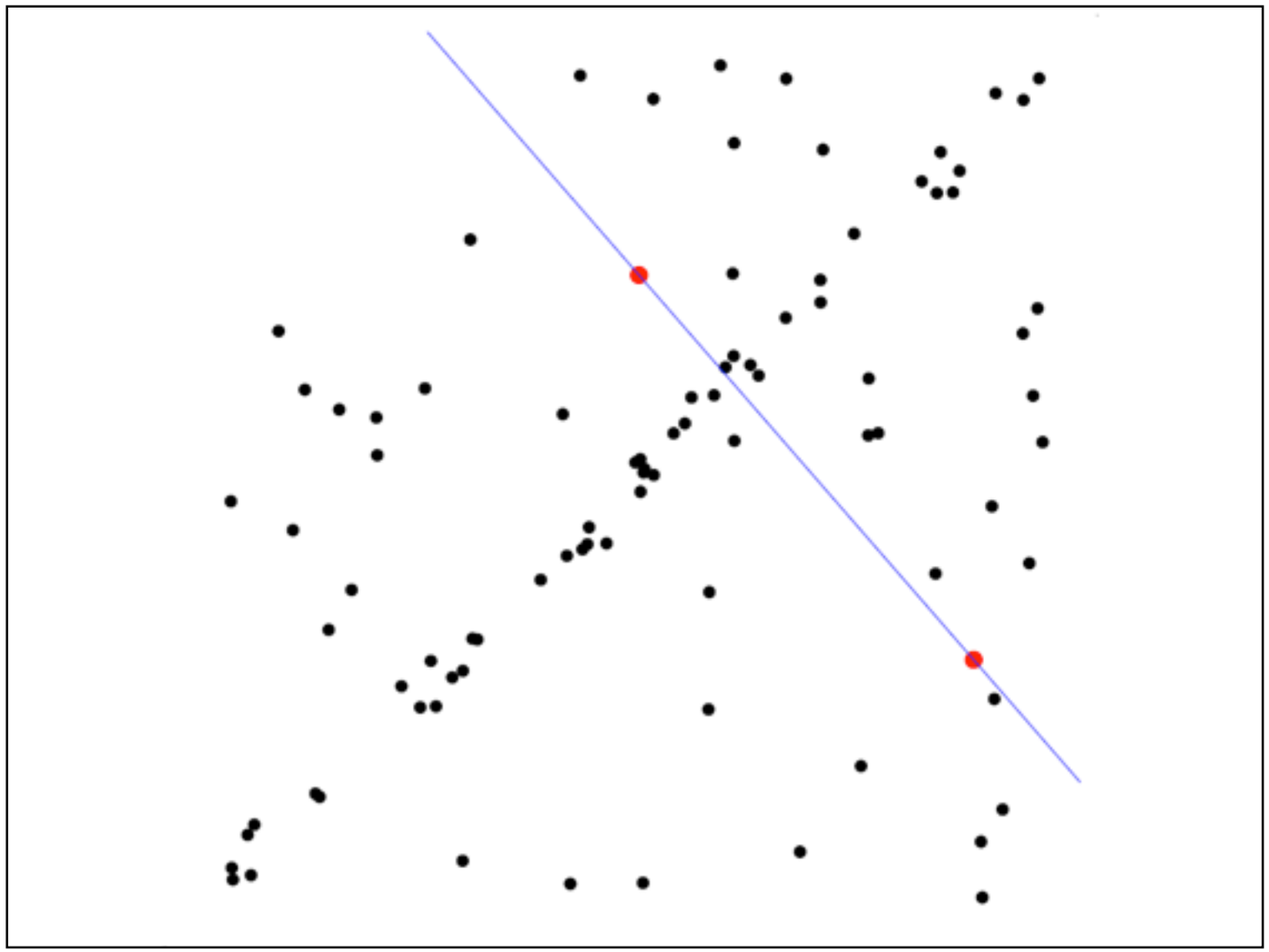
RANSAC for line fitting example



1. Randomly select minimal subset of points

Source: R. Raguram

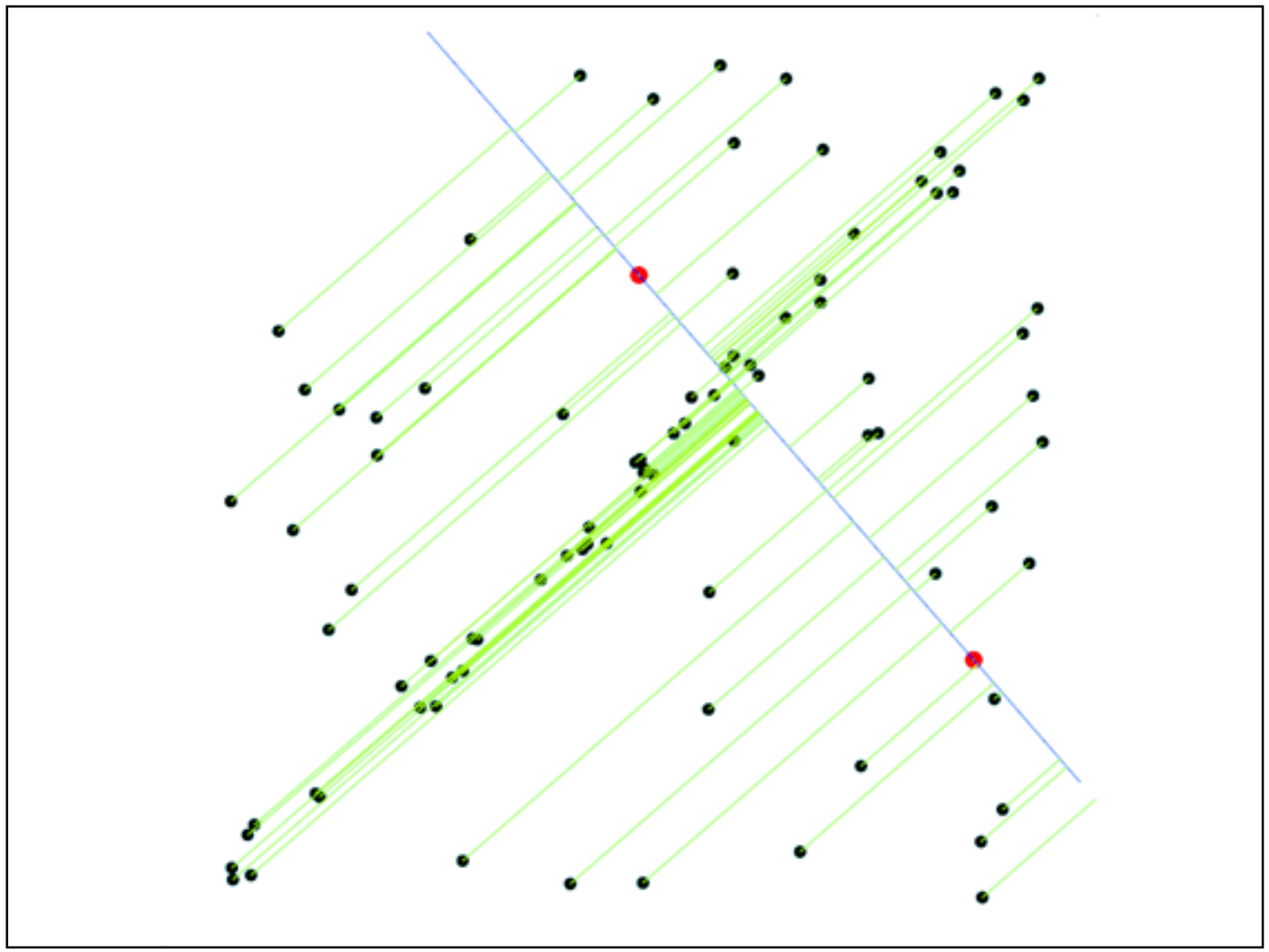
RANSAC for line fitting example



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

Source: R. Raguram

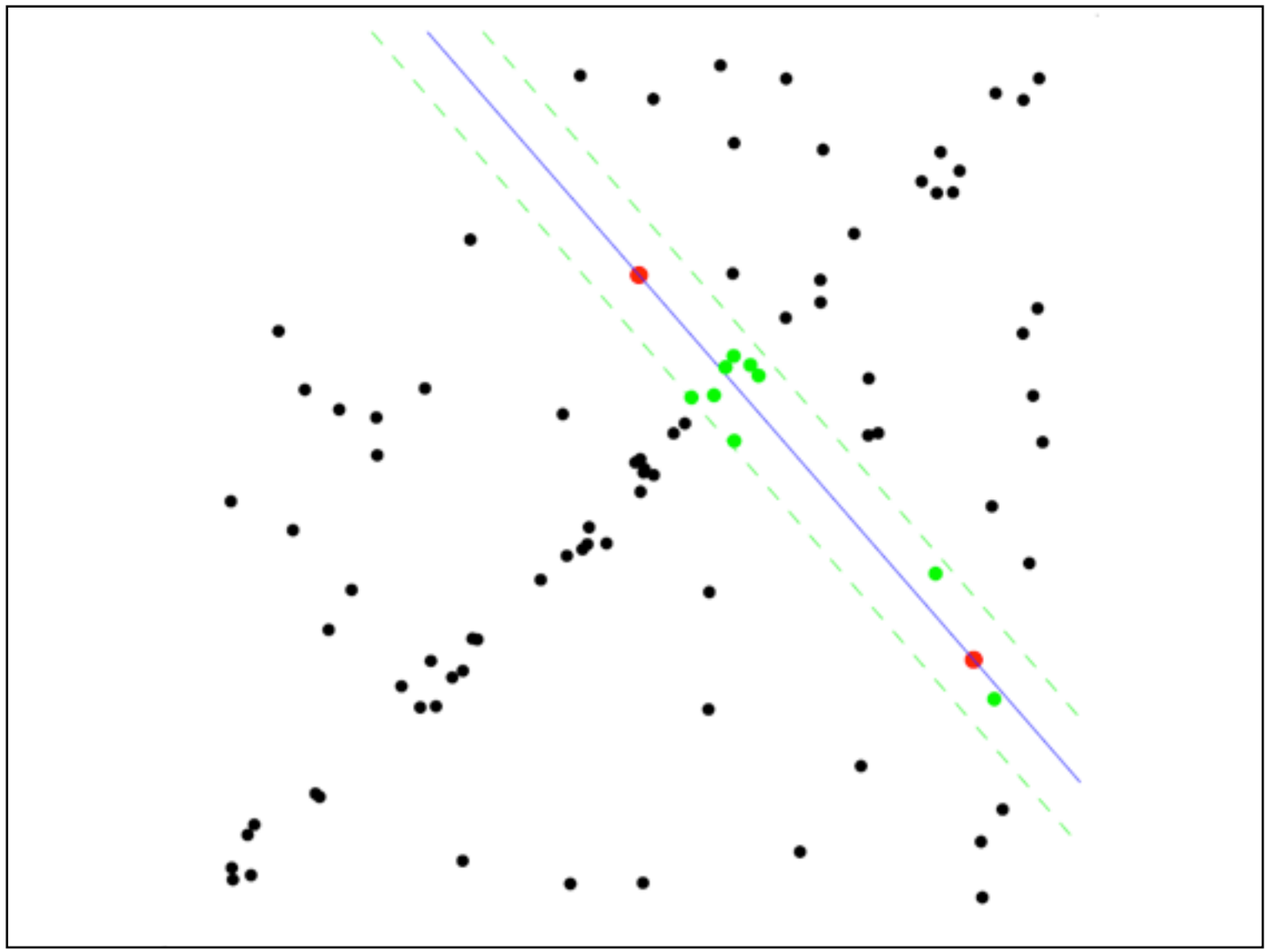
RANSAC for line fitting example



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

Source: R. Raguram

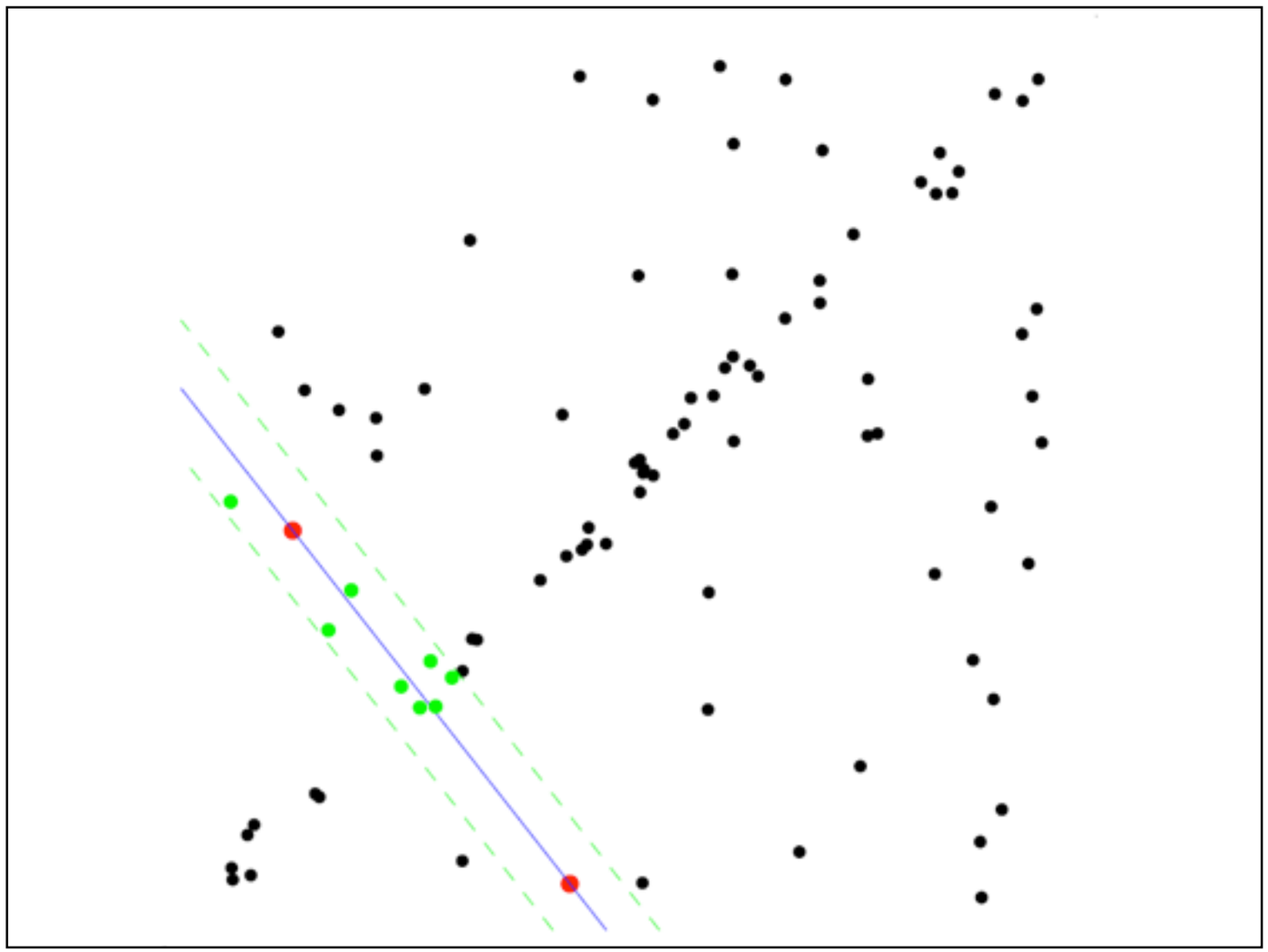
RANSAC for line fitting example



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

Source: R. Raguram

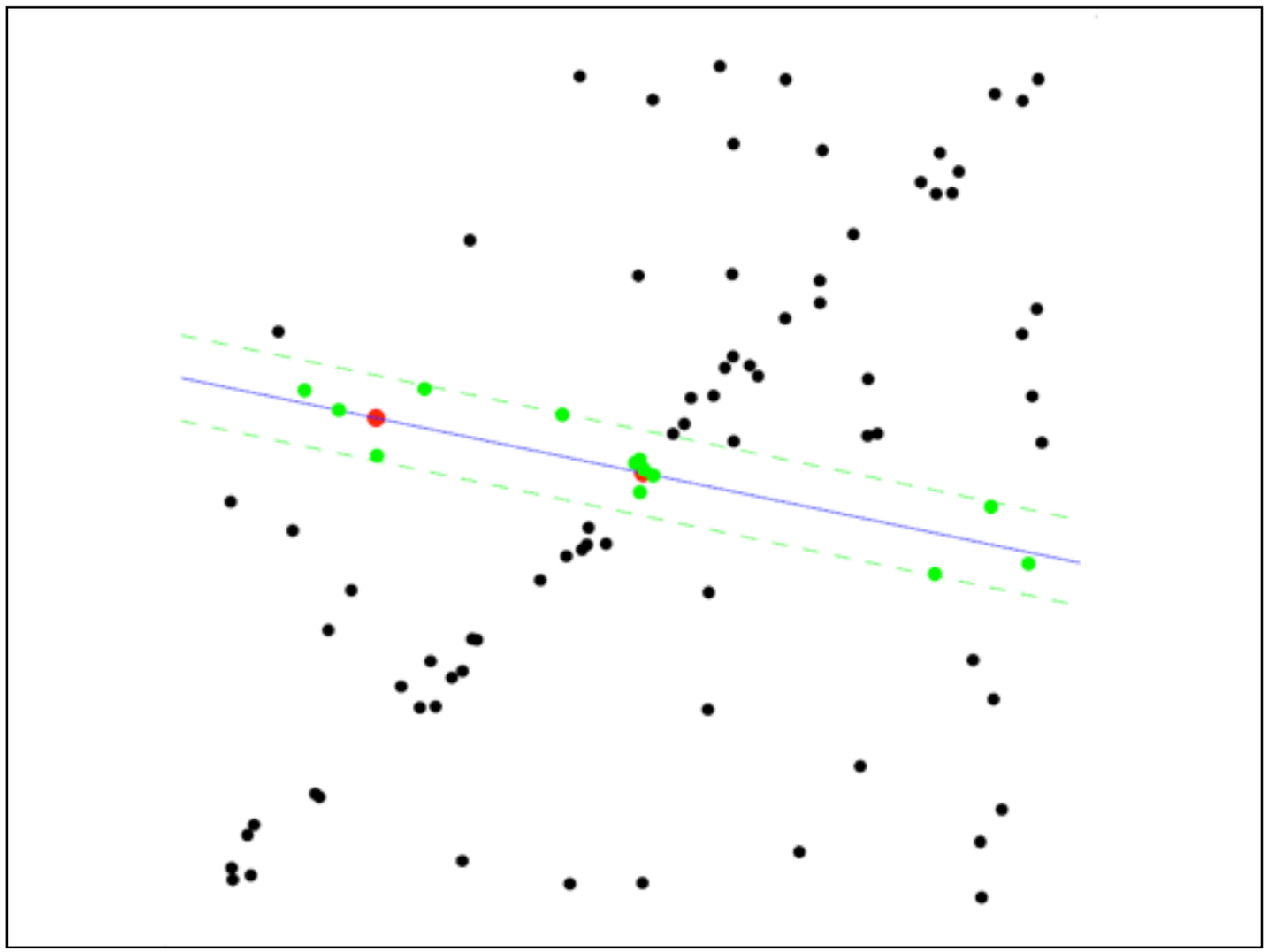
RANSAC for line fitting example



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

Source: R. Raguram

RANSAC for line fitting example

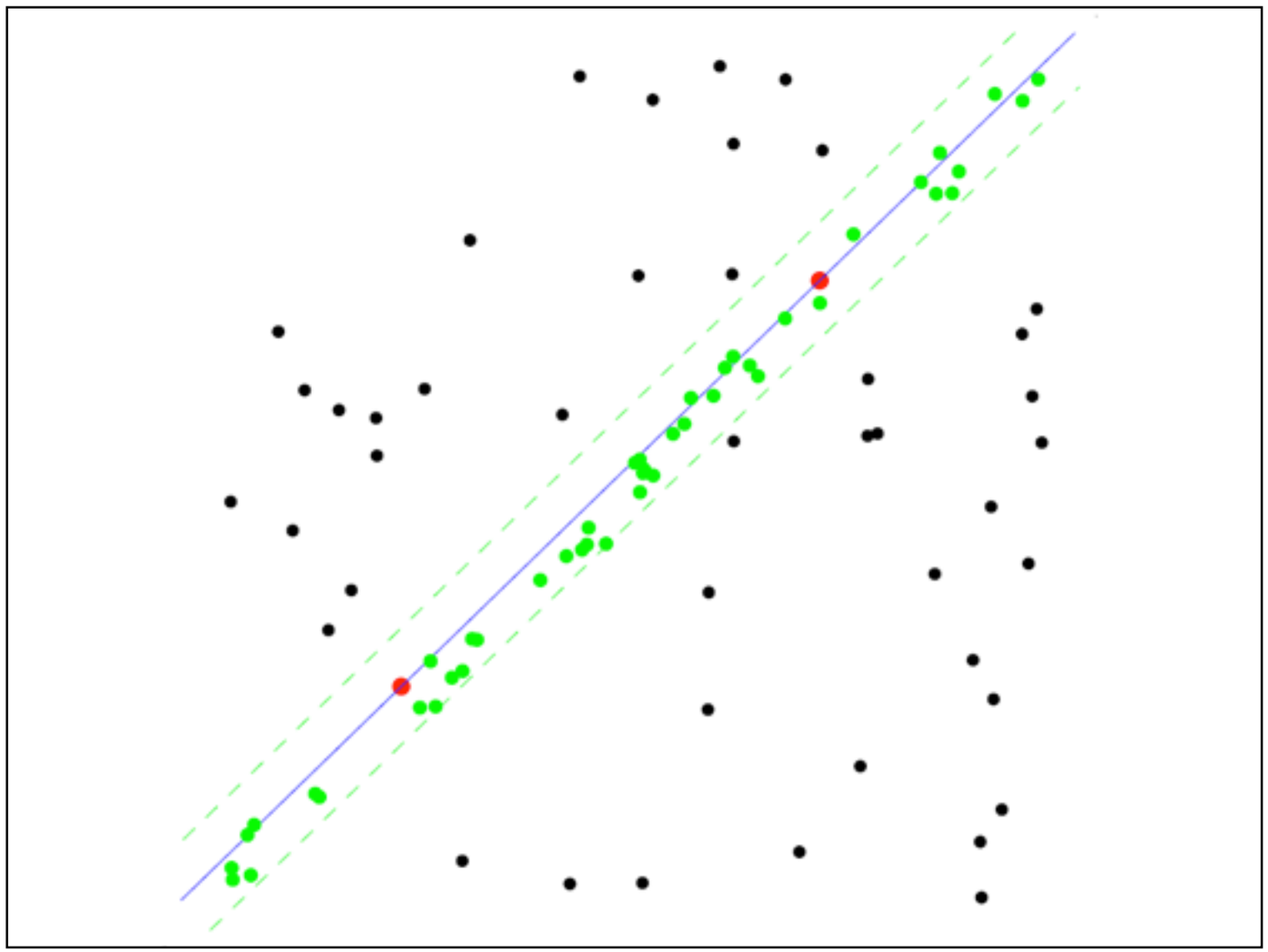


1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat *hypothesize-and-verify* loop

Source: R. Raguram

RANSAC for line fitting example

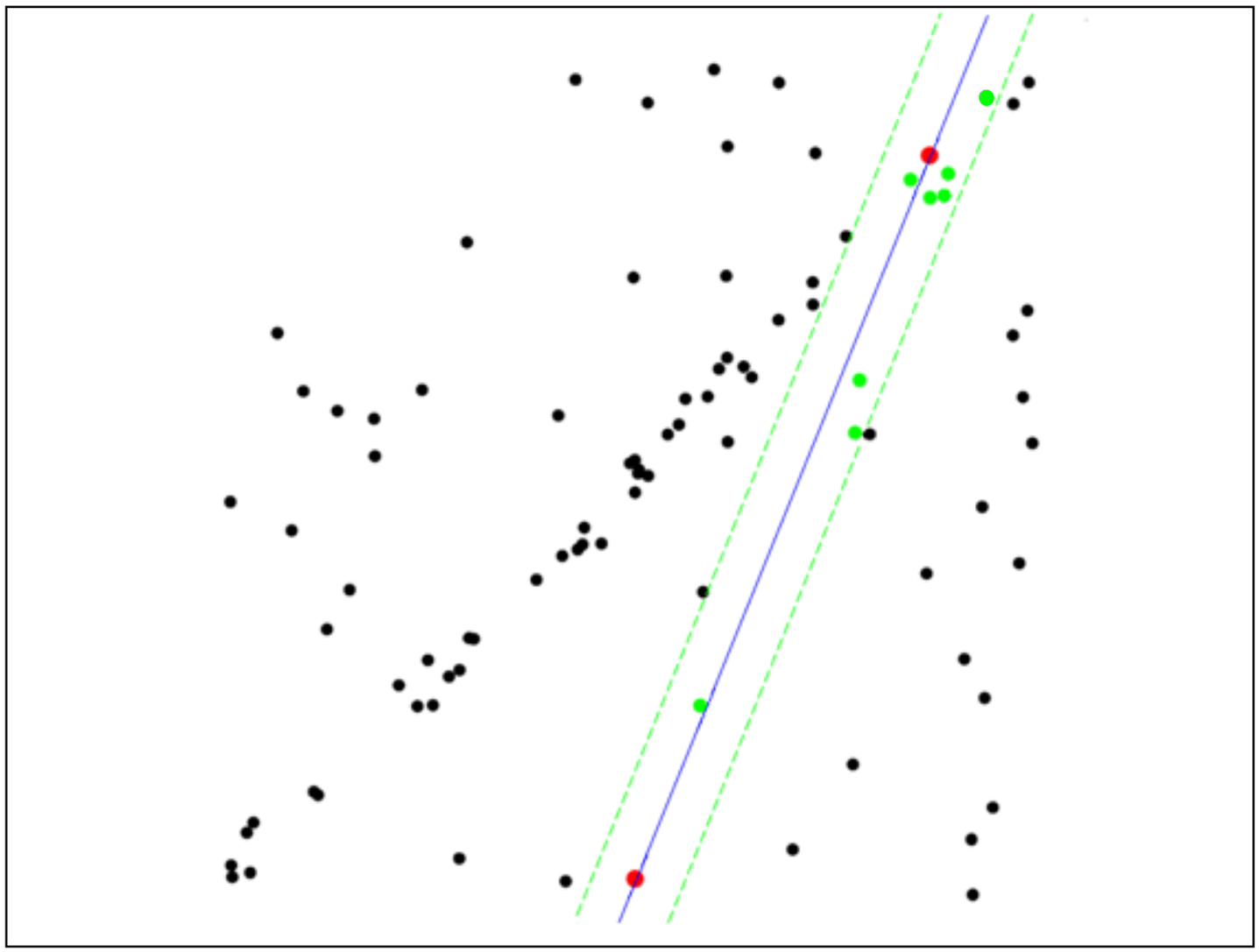
Uncontaminated sample



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

Source: R. Raguram

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

Source: R. Raguram

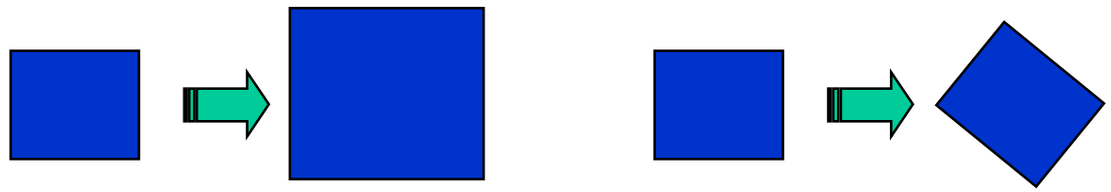
RANSAC for line fitting

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

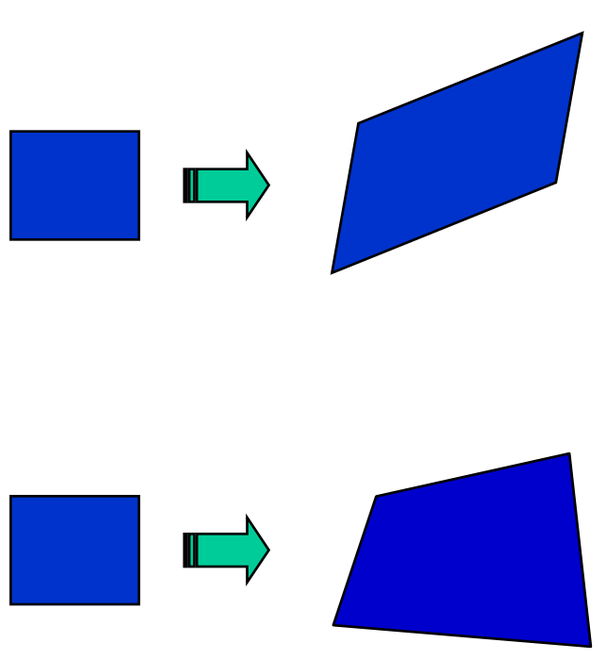
2D transformation models

- Similarity
  (translation, scale, rotation)



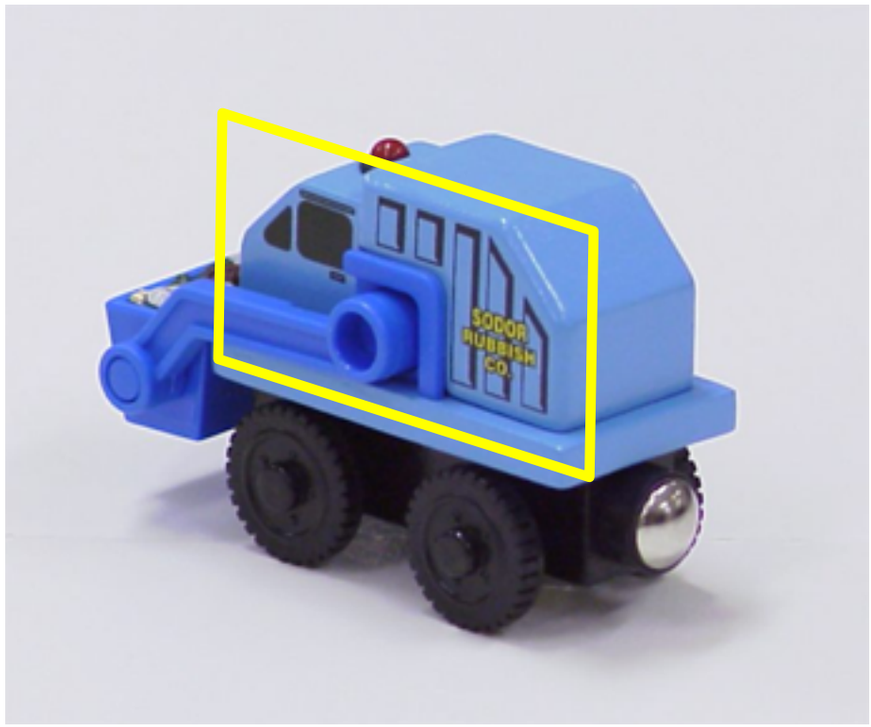
- Affine

- Projective
  (homography)



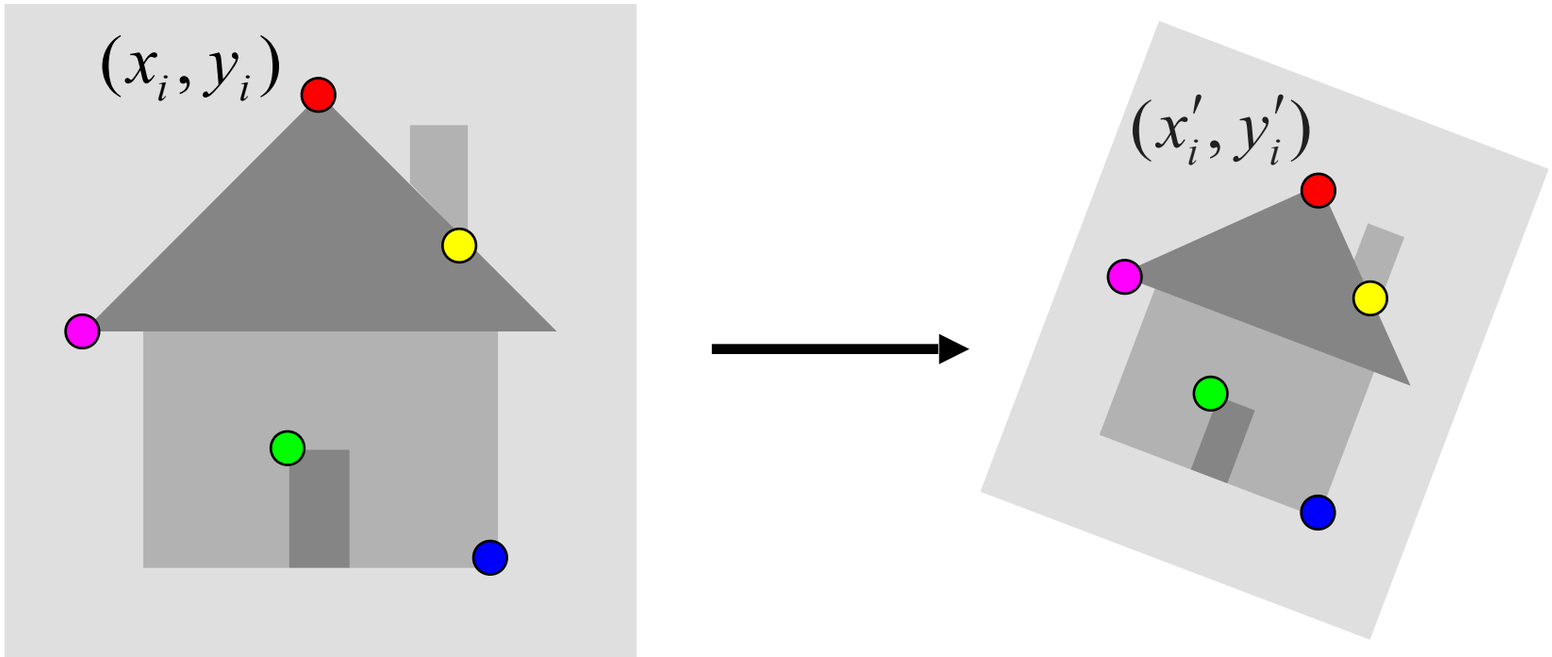
Let's start with affine transformations

- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- Can be used to initialize fitting for more complex models



Fitting an affine transformation

- Assume we know the correspondences, how do we get the transformation?



$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

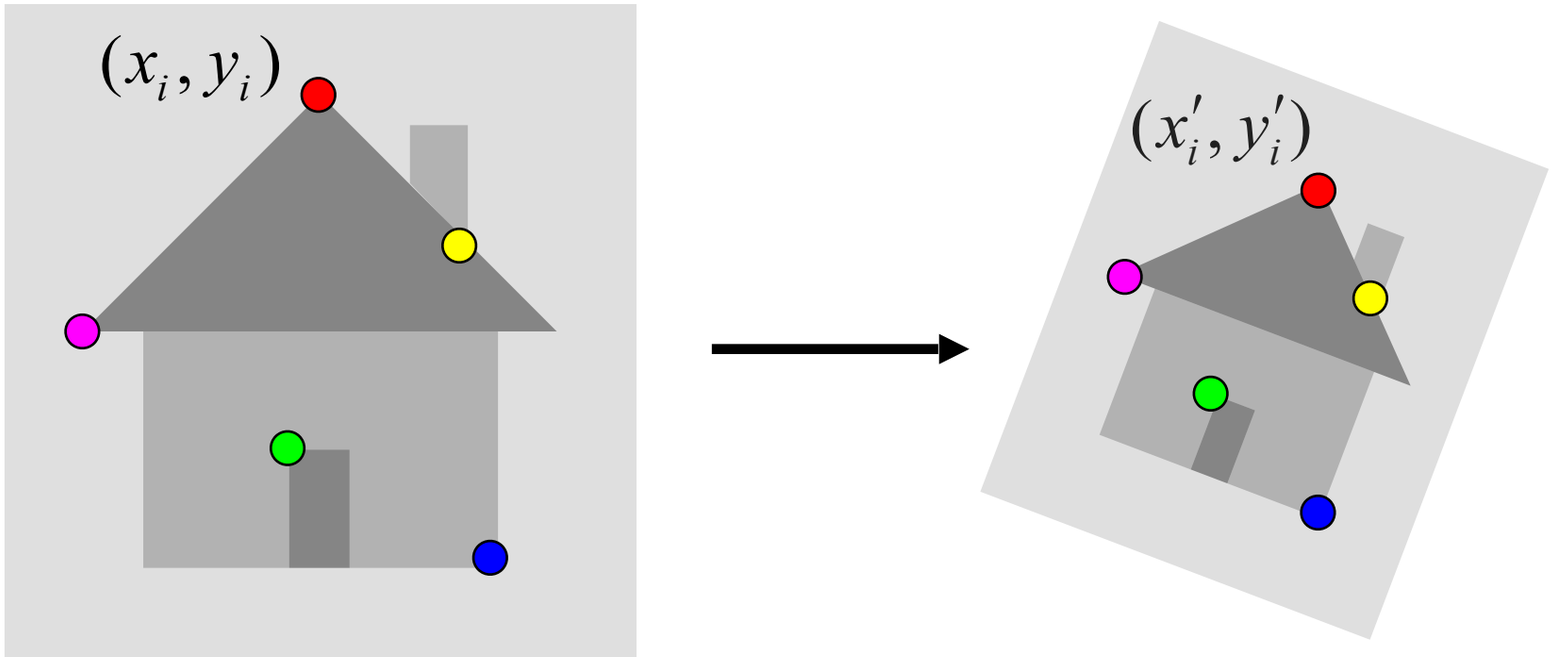
$$\mathbf{x}_i' = \mathbf{M}\mathbf{x}_i + \mathbf{t}$$

Want to find **M**, **t** to minimize

$$\sum_{i=1}^{n} \| \mathbf{x}_i' - \mathbf{M}\mathbf{x}_i - \mathbf{t} \|^2$$

Fitting an affine transformation

- Assume we know the correspondences, how do we get the transformation?



$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

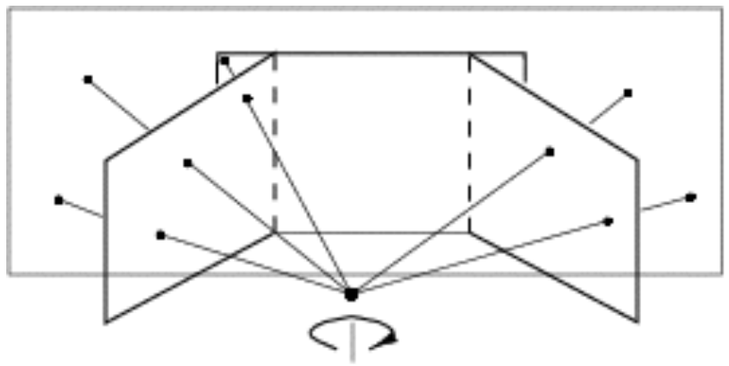
$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

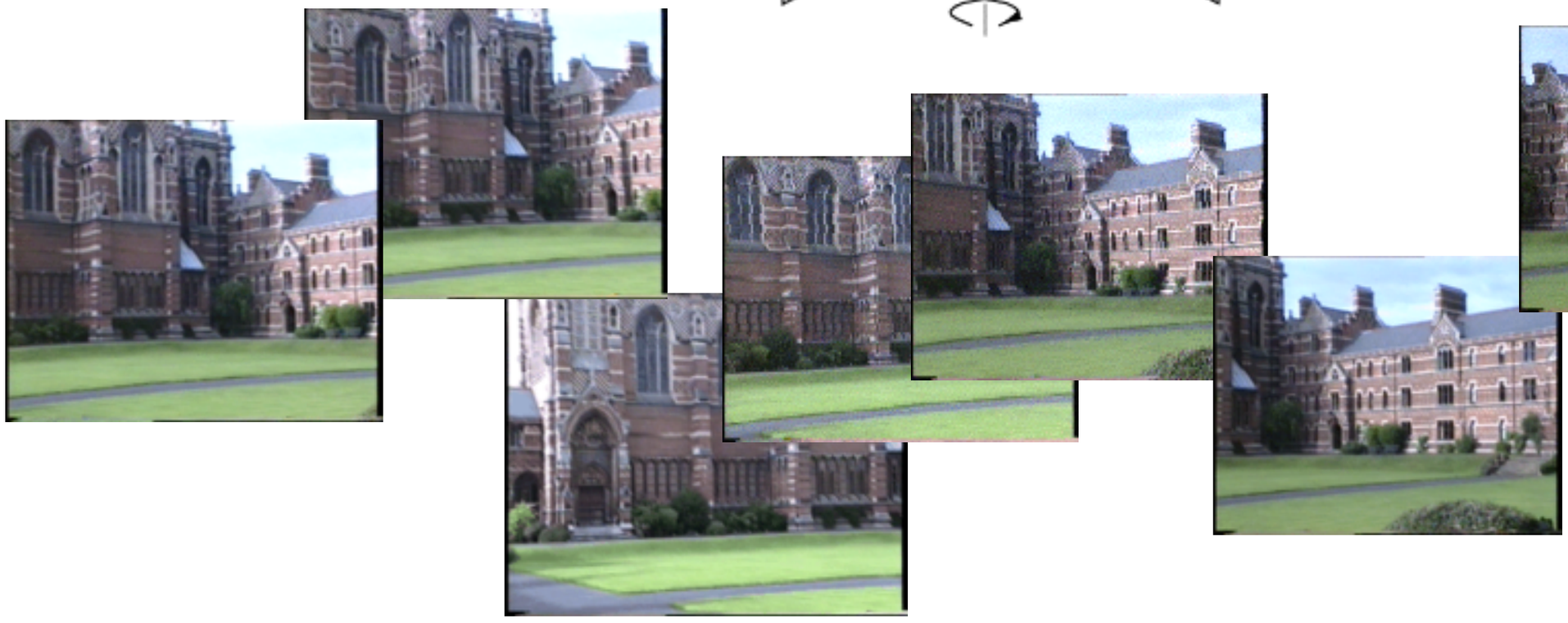
Fitting an affine transformation

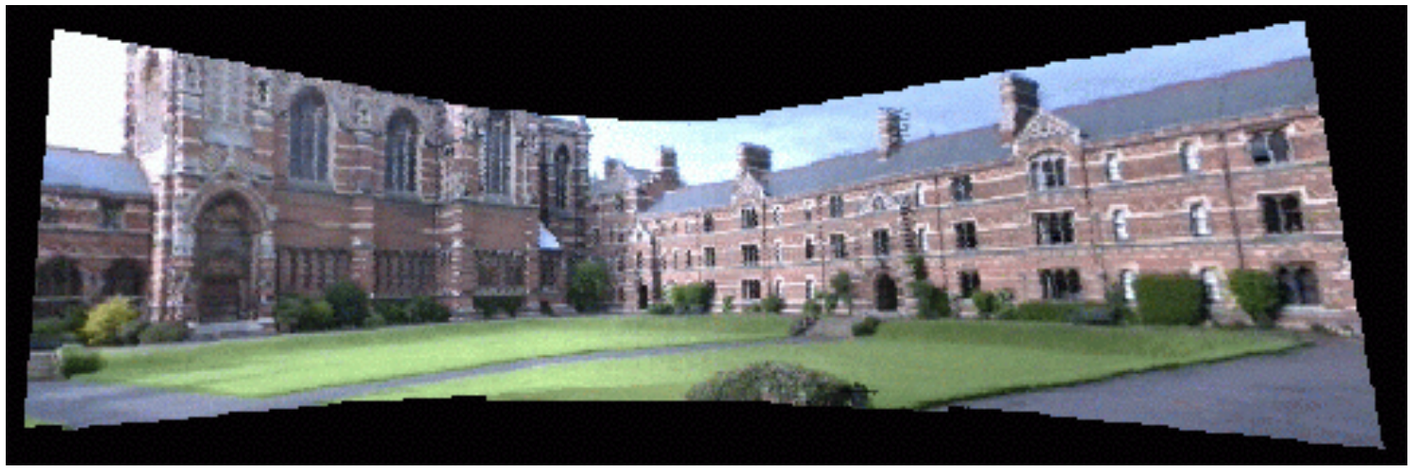
$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

- Linear system with six unknowns
- Each match gives us two linearly independent equations: need at least three to solve for the transformation parameters

Application: Panorama stitching



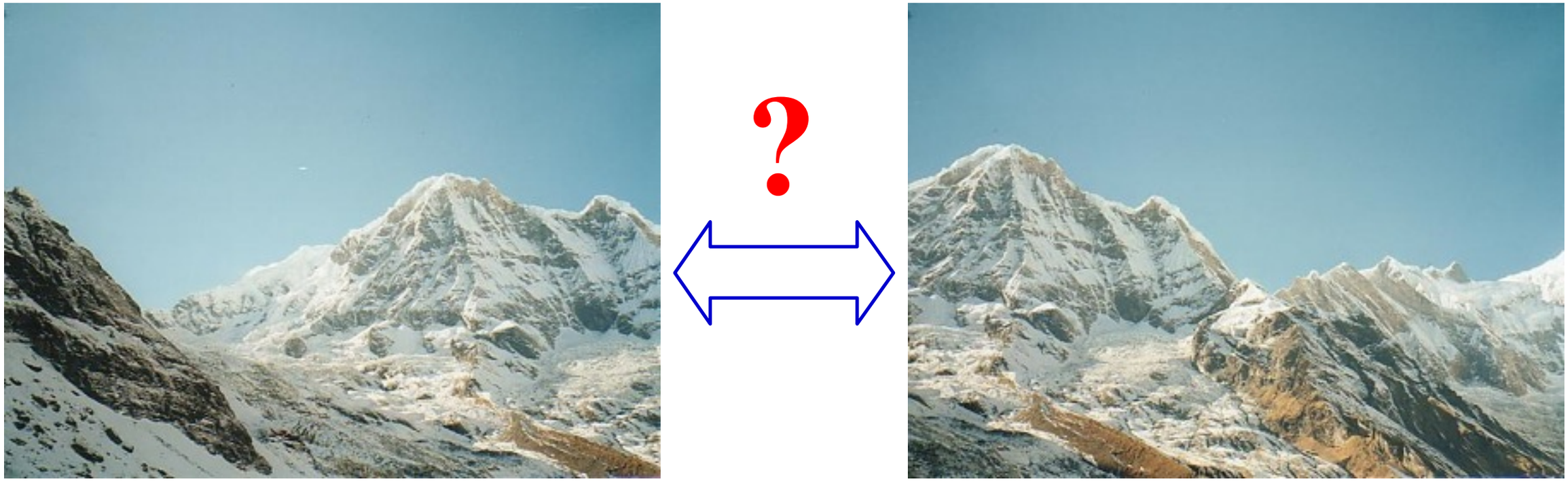




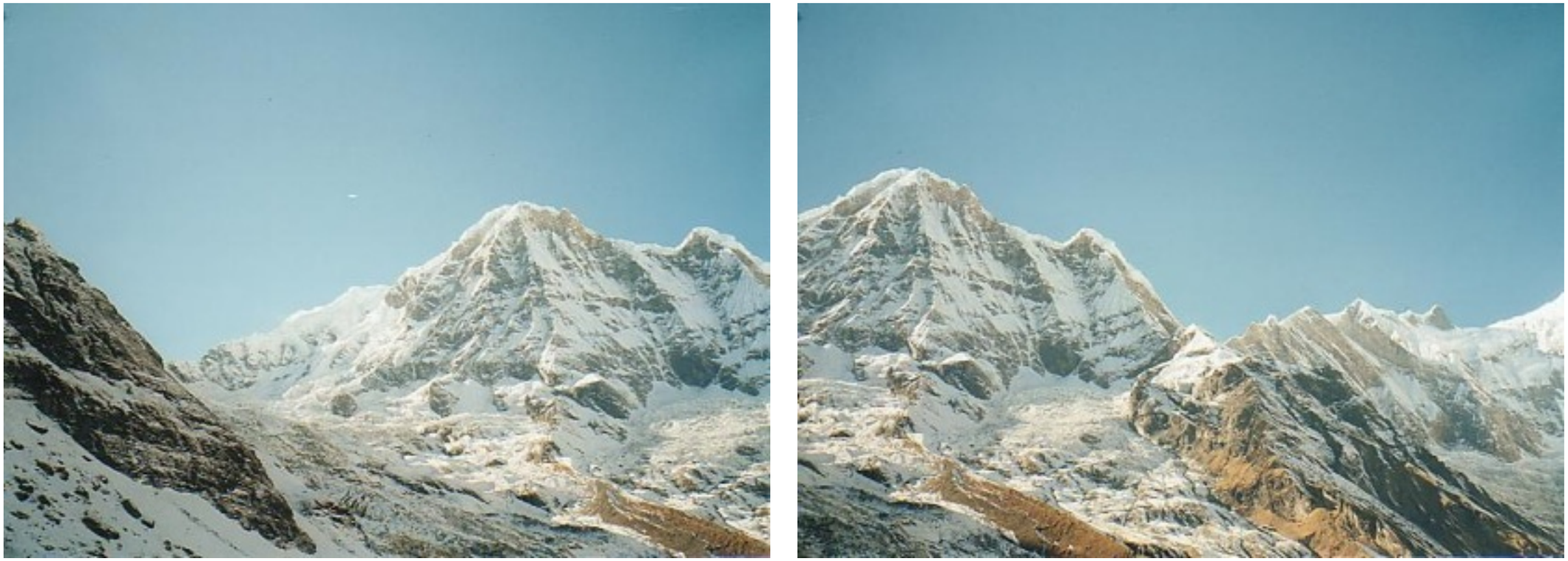
Source: Hartley & Zisserman

Panoramic stitching

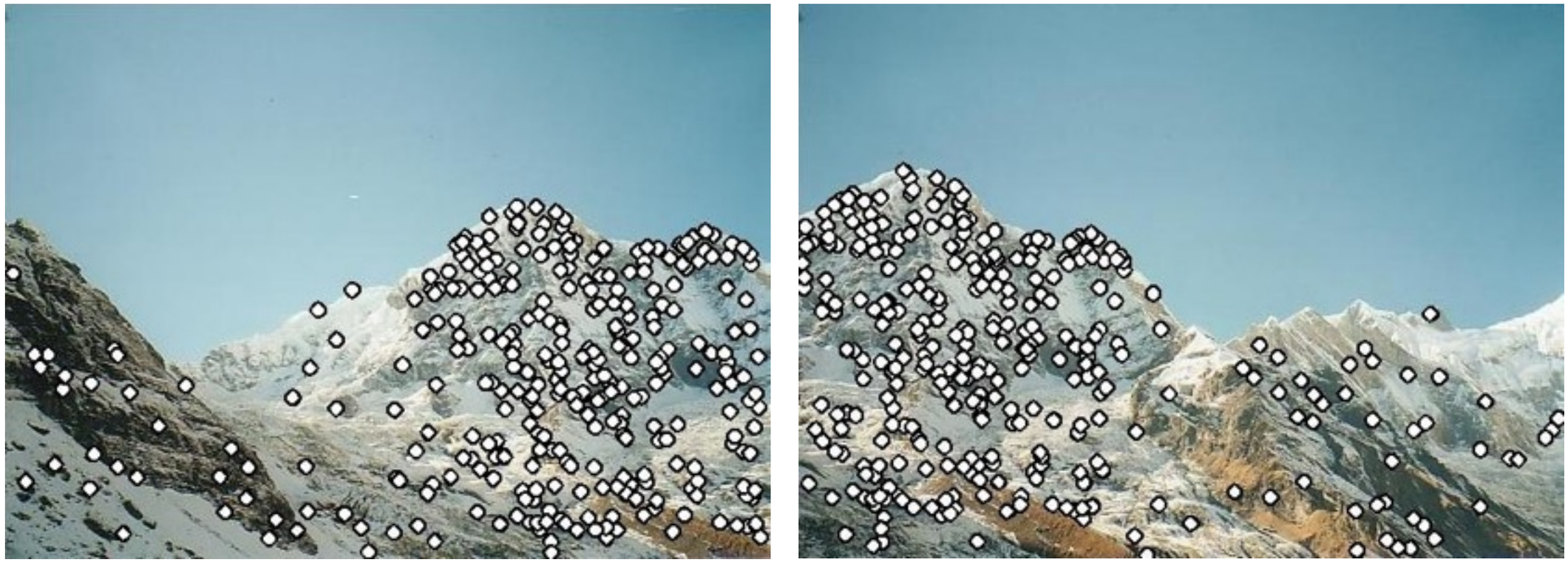
- **Approach**
  - Local feature matching
  - RANSAC for alignment



Panoramic stitching



Panoramic stitching

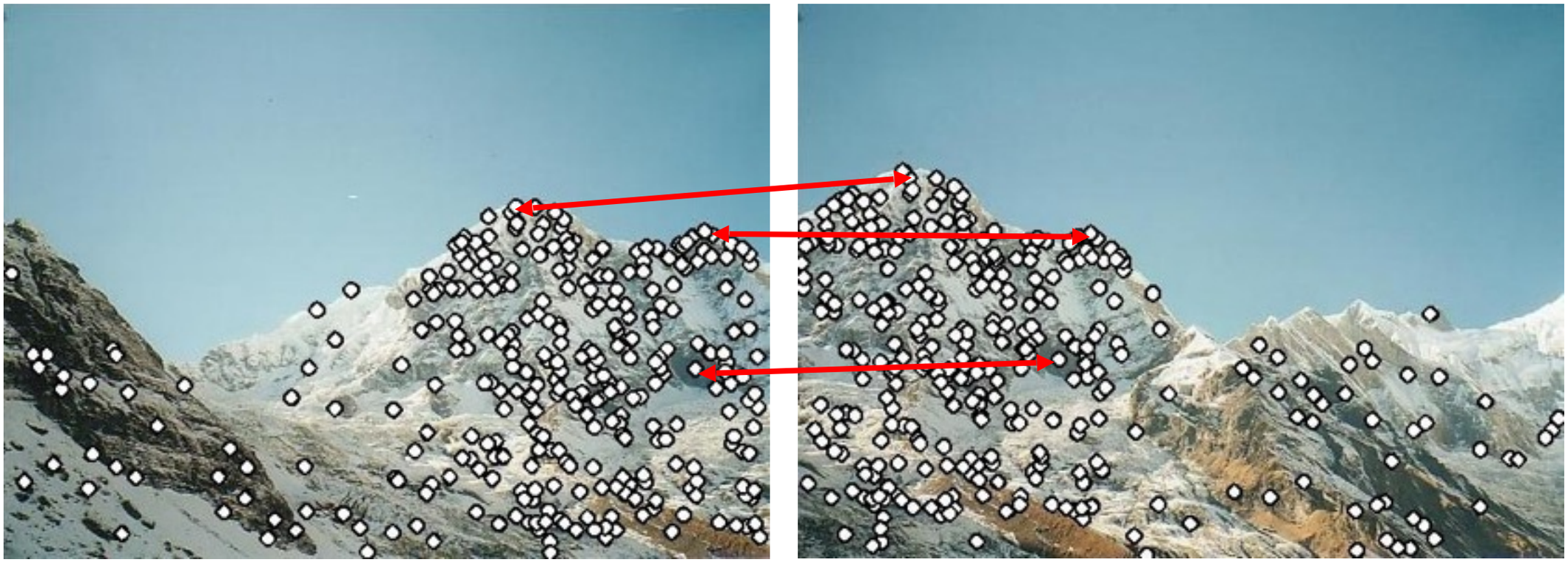


- Extract features
  - corner detector

Panoramic stitching

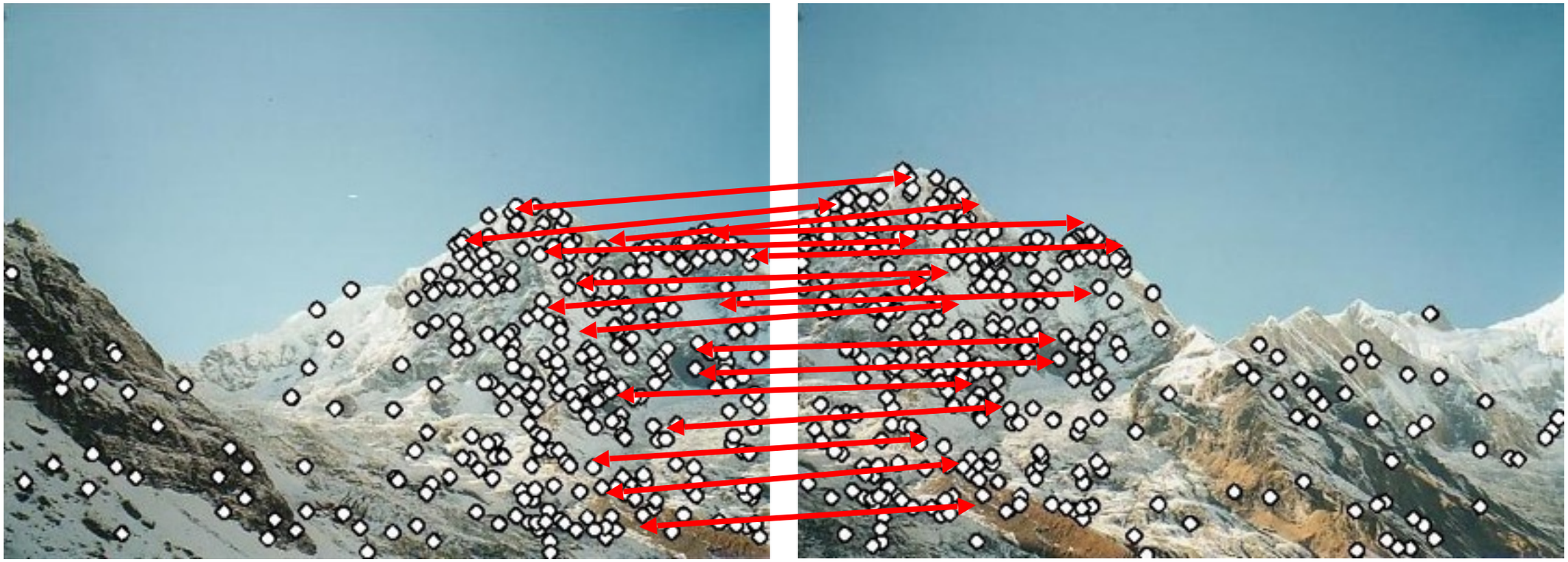
- Extract features
- Compute *putative matches*

Panoramic stitching



- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T*

Panoramic stitching



- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T*
  - *Verify* transformation (search for other matches consistent with *T*)