Point Matching

CS510
Lecture #21
April 10, 2002

Introduction to Feature Matching

- Up to now, we have:
  - Matched images
  - Described images (by extracting features)
- Now we will look at feature matching
  - Occasionally: match features from one image to features of another
  - More Often: match image features to model features

Why Match Features?

- To ignore irrelevant data
  - Backgrounds
  - Meaningless variation among instances
    - E.g. color of car, whether pickup is covered
  - Variations in illumination
- To accommodate changes in viewpoint
  - Insensitivity to classes of transformations
    - Depends on the feature extraction and matching techniques
  - Defining the class of transformation is critical to the success of any feature matching system

Simple Example: Matching Points

- Edges & Corners can both be viewed as interesting 2D points in an image
- It might be useful to match a set of edges/corners to points in an object model
- Similar - but different - from correlating Prewitt edge scores

The Directed Hausdorff Metric

- Given two sets of points:
  $$A = \{a_1, a_2, \ldots, a_m\}, B = \{b_1, b_2, \ldots, b_n\}$$
- The Hausdorff metric is:
  $$h(A,B) = \underset{a \in A}{MAX}\ \underset{b \in B}{MIN} \|a - b\|$$
- In other words, it is the maximum distance from any point in A to its nearest point in B
  - What class of transformations does this support?

Hausdorff (cont.)

- Note: the directed Hausdorff metric is non-symmetric.
  - Good for comparing models to images
  - Good for comparing subimages to images
  - Less good for comparing two images
- A symmetric version is defined as:
  $$H(A,B) = MAX(h(A,B), h(B,A))$$

Robust Hausdorff

- The Hausdorff metric is highly sensitive to noise
  - One "outlier" point in set A will ruin the match
  - A more robust version is:

$$h(A,B) = \underset{a\in A}{Kth}\ \underset{b\in B}{MIN}\|a-b\|$$

- Typical values of K are 10%, 20% or 50% of m.

Simplified Hausdorff Matching

- To match a set of points A to a set of points in an image B:
  - for every pixel in B, compute distance to nearest edge/corner (call this image D).
  - Make binary template from points in A
  - Slide template over image D:
    - compute only at points in A
    - find max value in D under point in A
- How does this compare to cross-correlation?

a.k.a. geometric hashing

Combinatorial Point Matching

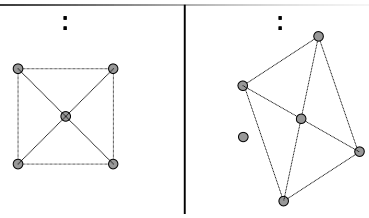
- What if we want to match points under arbitrary 8-DOF transformations? (We couldn't do this with images!)
  - Four point correspondences are enough to define an 8-DOF "perspective" transformation
  - General idea:
    - Hypothesize four point correspondences
    - Solve for transformation
    - Evaluate hypothesis by the Hausdorff metric of the transformed data

Geometric Hashing (II)

More Formally:

- Given two sets of points, A & B, such that the points in A are a transformed subset of the points in B, with noise added, then:
  - Pick four non-colinear points in A at random: $\{a_1,a_2,a_3,a_4\}$
  - For all possible correspondences of points $a_{1\text{-}4}$ to points in B, compute the perspective transformation matrix.
  - For each transformation, compute the Hausdorff distance; select the minimum Hausdorff value.
- How expensive is this?

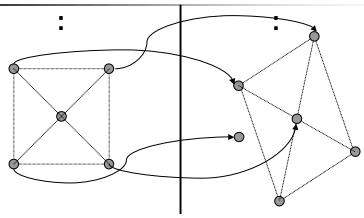
Example:



- B is non-uniformly scaled, sheared and rotated form of A, with one extra point.

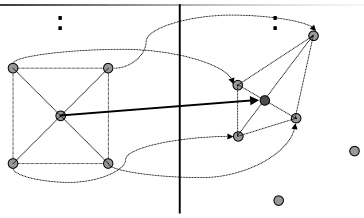
(Dotted lines are for visualization only, and not part of the problem defintion)

Example (II):



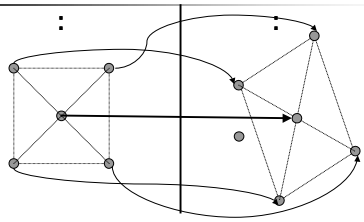
- Assume we choose the wrong pairing of four points to four points (as shown)
  - Where does the middle point project to?

Example (III):



- Under this transformation, the middle point should map to the red spot in B, which is far from any green point in B.

Example (IV):



- :::::::::
  :::::::::
  ::::

Notes

- The example shown was an affine transformation, but it would have worked just as well for a perspective transformation
- Some transformations can be dismissed a-priori
  - Example: if points at infinity (in A) map to the image (in B)
  - If transformation exceeds domain criteria
- If distance image D_B is computed, checking a set of point correspondences is very fast
- Overall cost is still $O(N^4)$
  - $O(N^4)$ for affine transformations