Announcements

- Homework 3 is due Tue, Nov 24, 11:59 PM
- Homework 4 will be assigned today
  – Due Sat, Dec 5, 11:59 PM
- No class meetings next week
  – Happy Thanksgiving
- Reading:
  – Chapter 15: Learning to Classify
  – Chapter 16: Classifying Images
  – Chapter 17: Detecting Objects in Images

A Rough Recognition Spectrum

Model-Based Vision

- Given 3-D models of each object
- Detect image features (often edges, line segments, conic sections)
- Establish correspondence between model & image features
- Estimate pose
- Consistency of projected model with image

Recognition by Hypothesize and Test

- General idea
  – Hypthesize object identity and pose
  – Recover camera parameters
  – Render object using camera parameters
  – Compare to image
- Issues
  – Where do the hypotheses come from?
  – How do we compare to image (verification)?
- Simplest approach
  – Construct a correspondence for all object features to every correctly sized subset of image points
  – These are the hypotheses
  – Expensive search, which is also redundant

Pose consistency

- Correspondences between image features and model features are not independent
- A small number of correspondences yields a camera matrix
  – The others correspondences must be consistent with this
- Strategy:
  – Generate hypotheses using small numbers of correspondences (e.g., triples of points for a calibrated perspective camera)
  – Recover camera parameters (e.g., calibrated camera rotation and translation) and verify
Voting on Pose

- Each model leads to many correct sets of correspondences, each of which has the same pose
  - Vote on pose, in an accumulator array (similar to a Hough transform)
Invariance

- Properties or measures that are independent of some group of transformation (e.g., rigid, affine, projective, etc.)
- For example, under affine transformations:
  - Collinearity
  - Parallelism
  - Intersection
  - Distance ratio along a line
  - Angle ratios of three intersecting lines
  - Affine coordinates

Geometric hashing

- Vote on identity and correspondence using invariants
  - Take hypotheses with large enough votes
- Building a table:
  - Take all triplets of points on model image to be base points $P_1$, $P_2$, and $P_3$
  - Take every fourth point and compute $\mu_{ka}$ and $\mu_{kb}$
  - Fill up a table, indexed by $\mu_{ka}$ and $\mu_{kb}$, with
    - The base points and fourth point that yielded $\mu_{ka}$ and $\mu_{kb}$
    - The object identity

Algorithm 18.3: Geometric hashing: voting on identity and point labels

```
For all groups of three image points $T(I)$
  For every other image point $p$
    Compute the $\mu_i$'s from $p$ and $T(I)$
    Obtain the table entry at those values
    if there is one, it will label the three points in $T(I)$
    with the name of the object
    and the names of those particular points.
    Cluster those labels;
    if there are enough labels, backproject and verify
end
end
```
Verification

- Edge score
  - Are there image edges near predicted object edges?
  - Very unreliable; in texture, answer is usually yes

- Oriented edge score
  - Are there image edges near predicted object edges with the right orientation?
  - Better, but still hard to do well

- Texture
  - For example, does the spanner have the same texture as the wood?

Application: Surgery

- To minimize damage by operation planning
- To reduce number of operations by planning surgery
- To remove only affected tissue

Problem
- Ensure that the model with the operations planned on it and the information about the affected tissue lines up with the patient
- Display model information supervised on view of patient

**Big Issue**: coordinate alignment, as above
Matching using Local Image features

Simple approach

- Detect corners in image (e.g., Harris corner detector)
- Represent neighborhood of corner by a feature vector (produced by Gabor Filters, K-jets, affine-invariant features, etc.)
- Modeling: Given a training image of an object without clutter, detect corners, and compute and store feature descriptors
- Recognition time: Given test image with possible clutter, detect corners and compute features. Find models with same feature descriptors (hashing) and vote

Probabilistic interpretation

- Write
  \[ P(\text{patch of type } i \text{ appears in image | pattern is present}) = p_{ij} \]
  \[ P(\text{patch of type } i \text{ is pattern is present}) = p_i \]
- Assume
  \[ p_{ij} = \mu \text{ if the pattern can produce this patch and 0 otherwise } \]
  \( p_{ij} = \lambda < \mu \text{ for all } i \)
- Likelihood of image given pattern
  that \( n_p \) patches come from that pattern and \( n_n \) patches come from noise, is
  \[ P(\text{interpretation | pattern}) = \lambda^{n_p} \mu^{n_n} \]

Employ spatial relations

Figure from “Local grayvalue invariants for image retrieval,” by C. Schmid and R. Mohr, IEEE Trans. Pattern Analysis and Machine Intelligence, 1997 copyright 1997, IEEE
Finding faces using relations

- **Strategy:**
  - Face is eyes, nose, mouth, etc. with appropriate relations between them
  - Build a specialized detector for each of these (template matching) and look for groups with the right internal structure
  - Once a face is detected, there is little uncertainty about where the other parts could be

Finding faces using relations

- **Strategy:** compare

\[ P(\text{true face at } X_0 = x_0, X_{10} = x_1, X_{1} = x_2, X_{11} = x_3, \text{all other responses}) \]

\[ P(\text{false face at } X_0 = x_0, X_{10} = x_1, X_{1} = x_2, X_{11} = x_3, \text{all other responses}) \]

Notice that once some facial features have been found, the position of the rest is quite strongly constrained.

Next Lecture

- Color
- Light field camera
- Reading:
  - Chapter 3: Color
  - Section 19.3: The Light Field