Introduction to Features

CS 510
Lecture #19
3/25/02

Where are we?

- We know how to generate images:
  - ray tracing
  - radiosity.
- We know how to manipulate images:
  - frequency-based filtering
  - nn/bilinear/cubic interpolation
  - morphing
- We know how to match images
  - correlation
  - eigenspaces
  - Fourier

Limitations on Image Matching

- We know how to compare images *pixel by pixel*
- What if I want to match a picture of a black car to a picture of a white one?
  - Will they correlate?
  - Match in eigenspace?
  - Match in the frequency domain?
- What are the limitations on pixel matching?

Pixel Matching Example

- In black car vs. white car, the *structure* is the same, but the pixel *values* are different.
- Transform the image by replacing each pixel with the (unsigned) difference between the pixel and its neighbor
  - A simple form of edge detection
- Now correlate the edge maps! (or Fourier, or eigenspace…)

Mutual Information

- In general, to match two images whose intensities have been remapped…
  - How? Sensor switch: match MRI to PET
  - Object instance variation: color schemes
- Mutual Information
  - Count how many times each pixel value occurs (histogram)
  - Convert count to probability (count / total pixels)

$$\sum_{v_1 \in I_1(x,y)} \sum_{v_2 \in I_2(x,y)} P_{v_1 v_2} \log_2 \left[\frac{P_{v_1 v_2}}{P_{v_1} P_{v_2}} \right]$$

Image Matching (cont.)

- Anytime you can draw a pixel-to-pixel correspondence between two objects, image matching is possible.
- It is a matter of debate whether image matching is always preferable.
- “Recognition” often defies pixel-to-pixel correspondence
  - Broad classes: “chair”, “house”, “car”...
  - Mutable objects: people, trees, rivers …
  - Unknown viewpoint variations?

Structural/Feature Matching

- If an object has windows, doors, a mailbox and sufficient volume, we might call it a house.
- If an object has windows, wheels and a fender we might call it a car.
- Structural or Feature matching looks for specific substructures, ignoring the rest of the image
  - Feature Extraction & Segmentation
  - Matching

What is a Feature?

- According to your text (p. 68) a feature is anything that is:
  - Localized
  - Meaningful
  - Detectable
- Features are also *intermediate*; a means, not an end
- Q: Are features subimages or structures?

Examples of Features

- Edges
- Corners
- Chains
- Line segments
- Parameterized curves
- Regions
- Surface patches (3D)
- Closed Polygons

What is an Edge?

- Is an edge a description of a localized image pattern?
  - Then we need to know what aspect of the pattern we are measuring → Facet Model
- Is an edge a symbolic feature?
  - Then we need to know what it denotes:
    - surface marking, or
    - surface discontinuity, or
    - shadow (illumination discontinuity)
  - These things have precise positions

The Facet Model

Haralick & Shapiro, Vol. 1, Chapter 8

- The image can be thought of as a gray level intensity surface
  - piecewise flat (flat facet model)
  - piecewise linear (sloped facet model)
  - piecewise quadratic
  - piecewise cubic
- Processing implicitly or explicitly estimates the free parameters.

Facet Edge Detection

- Facet edge detectors assume a piecewise linear model, and calculate the slope of the planar facet (1st derivative).
  - If we assume that the noise is zero mean, and increases with the square of distance, then the Sobel Edge Operator is optimal:

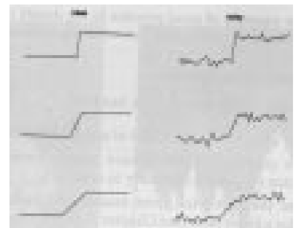
$$\left.\begin{array}{l} H = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, V = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \\ Mag = \sqrt{H^2 + V^2},\ \tan\theta = H/V \end{array}\right\} \text{H\&S, pp. 337-342}$$

Symbolic Edge Detection

- Although Sobel edges are optimal estimators for the slope of a planar facet in the presence of noise, for symbolic purposes they:
  - Are continuous; need to be thresholded
  - May be “thick”; need to be localized
  - Are isolated; need to be grouped into longer lines
- If they correspond to scene structure (e.g. discontinuities), we need a model of how scene structures map to images.

Edge Models

- The most common models are the step and ramp edges:



Canny Edge Detection (Step 1)

- To maximize the likelihood of finding step-edges, Canny argues (based on SNR & Gaussian noise models) that the optimal edge detector is

  1) Smooth image with a Gaussian filter
     - Size is determined by noise model

  2) Compute image gradients over the same size mask
     - We discussed this earlier in the context of rotation-free correlation...
- The bigger the mask, the better detection is but the worse localization is...

Canny Edge Detection (step 2)

- Non-maximal suppression
  - So far, edges are still “thick”
- For every edge pixel:

  1) Calculate direction of edge (gradient)

  2) Check neighbors in edge direction

     If either neighbor is “stronger”, set edge to zero.

Canny Edge Detection (step 3)

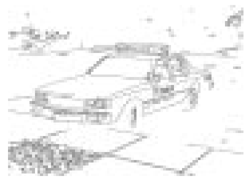
- We still have continuous values that we need to threshold
- Algorithm takes <u>two</u> thresholds: high & low
  - Any pixel with edge strength above the high threshold is an edge
  - Any pixel above the low threshold and next to an edge is an edge
- Iteratively label edges (they “grow out” from high points). This is called *hysteresis*.

Canny Example

Source image

Canny: sigma = 2.0, low = 0.40, high = 0.90

Canny Example (cont.)



Sigma = 3.0
low = 0.4, high = 0.9

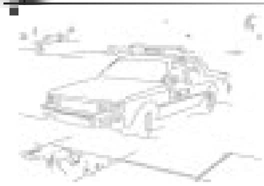
Sigma = 1.0
low = 0.4, high = 0.9

Canny Example (III)

Sigma = 2.0
low = 0.4, high = 0.99

Sigma = 2.0
low = 0.4, high = 0.99

Canny Example (IV)



Sigma = 2.0
low = 0.2, high = 0.9

Sigma = 2.0
low = 0.6, high = 0.9