CAP5415 Computer Vision
Spring 2003

Khurram Hassan-Shafique

Finding Connected Components

- Scan the binary image left to right top to bottom
- If there is an unlabeled pixel *p* with a value of '1'
  - assign a new label to it
  - Recursively check the neighbors of pixel *p* and assign the same label if they are unlabeled with a value of '1'.
- Stop when all the pixels with value '1' have been labeled.

Finding Connected Components (Sequential Algorithm 4-connectivity)

- Scan the binary image left to right top to bottom
- If an unlabelled pixel has a value of 1, assign a new label to it according to the following rules.

$$\begin{matrix} & 0 \\ 0 & 1 \end{matrix} \rightarrow \begin{matrix} 0 & 0 \\ 0 & L \end{matrix} \qquad \begin{matrix} & 0 \\ L & 1 \end{matrix} \rightarrow \begin{matrix} 0 & 0 \\ L & L \end{matrix}$$

$$\begin{matrix} & L \\ 0 & 1 \end{matrix} \rightarrow \begin{matrix} L & L \\ 0 & L \end{matrix} \qquad \begin{matrix} & L \\ M & 1 \end{matrix} \rightarrow \begin{matrix} L & L \\ M & L \end{matrix} \quad \text{(Set } L=M\text{)}$$

- Determine equivalence classes of labels.
- In the second pass, assign the same label to all elements in an equivalence class.

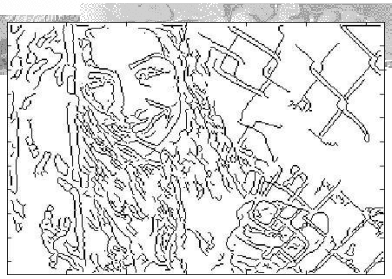
Rules for 8-connectivity

$$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & \end{matrix} \rightarrow \begin{matrix} 0 & 0 & 0 \\ & 0 & L \end{matrix} \qquad \begin{matrix} L & 0 & 0 \\ 0 & 1 & \end{matrix} \rightarrow \begin{matrix} L & 0 & 0 \\ & 0 & L \end{matrix}$$

$$\begin{matrix} 0 & L & 0 \\ 0 & 1 & \end{matrix} \rightarrow \begin{matrix} 0 & L & 0 \\ & 0 & L \end{matrix} \qquad \begin{matrix} 0 & 0 & L \\ 0 & 1 & \end{matrix} \rightarrow \begin{matrix} 0 & 0 & L \\ & 0 & L \end{matrix}$$

$$\begin{matrix} * & * & 0 \\ L & 1 & \end{matrix} \rightarrow \begin{matrix} 0 & 0 & 0 \\ & L & L \end{matrix} \qquad \begin{matrix} * & * & L \\ M & 1 & \end{matrix} \rightarrow \begin{matrix} * & * & L \\ & M & L \end{matrix} \quad \text{(Set } L=M\text{)}$$

Fitting



- Choose a parametric object/some objects to represent a set of tokens
- Most interesting case is when criterion is not local
  - can't tell whether a set of points lies on a line by looking only at each point and the next.
- Three main questions:
  - what object represents this set of tokens best?
  - which of several objects gets which token?
  - how many objects are there?

(you could read line for object here, or circle, or ellipse or...)

Fitting and the Hough Transform

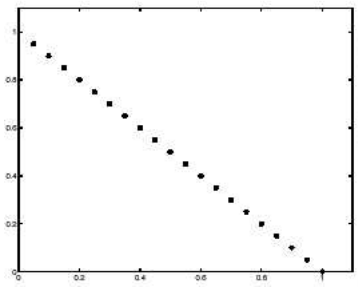
- Purports to answer all three questions
  - in practice, answer isn't usually all that much help
- We do for lines only
- A line is the set of points (x, y) such that

$$d = x\cos\theta + y\sin\theta$$

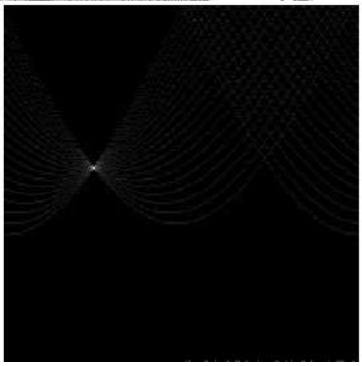
- Different choices of θ, d>0 give different lines
- For any (x, y) there is a one parameter family of lines through this point, given by

$$d = x\cos\theta + y\sin\theta$$

- Each point gets to vote for each line in the family; if there is a line that has lots of votes, that should be the line passing through the points



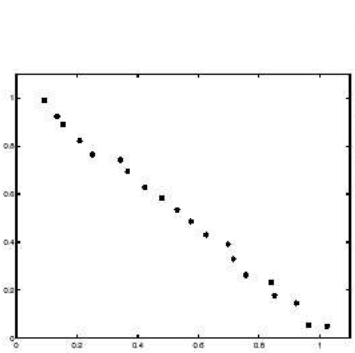
tokens

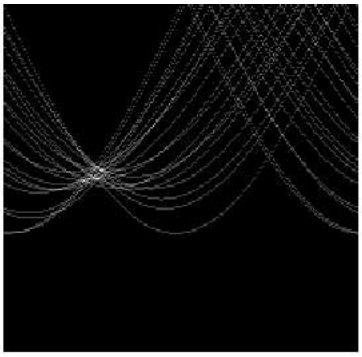


votes

Mechanics of the Hough transform

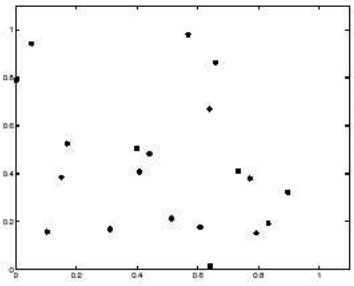
- Construct an array representing θ, d
- For each point, render the curve (θ, d) into this array, adding one at each cell
- Difficulties
  - how big should the cells be? (too big, and we cannot distinguish between quite different lines; too small, and noise causes lines to be missed)
- How many lines?
  - count the peaks in the Hough array
- Who belongs to which line?
  - tag the votes
- Hardly ever satisfactory in practice, because problems with noise and cell size defeat it

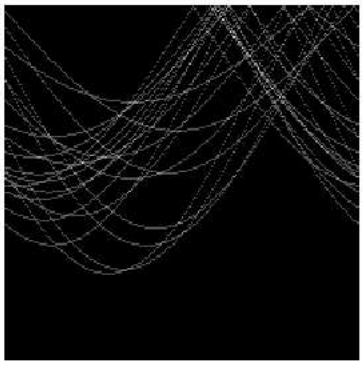




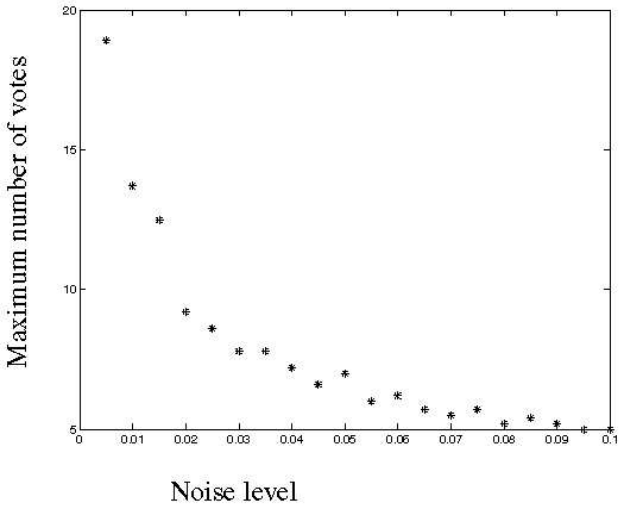
tokens

votes

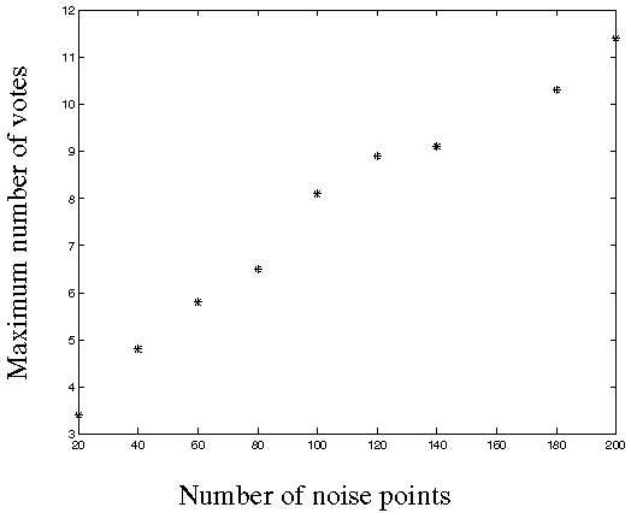




lots of noise can lead to large peaks in the array



This is the number of votes that the real line of 20 points gets with increasing noise



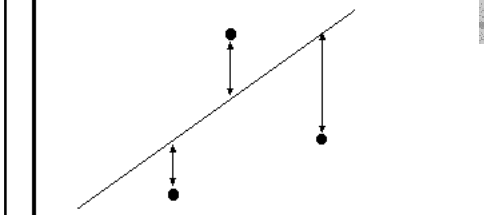
as the noise increases in a picture without a line, the number of points in the max cell goes up, too

Least Squares Fit

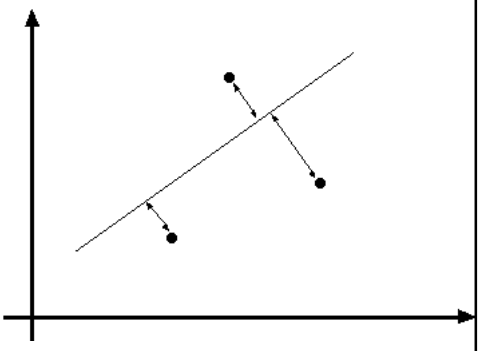
- Standard linear solution to a classical problem.
- Poor Model for vision applications.

$$y = ax + b = f(x, a, b)$$

Minimize $\sum_i [y_i - f(x_i, a, b)]^2$



Line fitting can be max. likelihood - but choice of model is important



Maximum Likelihood

Maximize the Log likelihood function L

$$L = -\frac{\sum_i (ax_i + by_i + c)^2}{2\sigma^2}$$

Given constraint

$$a^2 + b^2 = 1$$

Suggested Reading

- Chapter 15, David A. Forsyth and Jean Ponce, "Computer Vision: A Modern Approach“
- Chapter 5, Emanuele Trucco, Alessandro Verri, "Introductory Techniques for 3-D Computer Vision"
- Chapter 4, Mubarak Shah, “Fundamentals of Computer Vision”
- “Use of the Hough Transformation to Detect Lines & Curves in Pictures,” R.O. Duda & P.E. Hart, Computer methods in image analysis (On Reserve)