Normalized cuts

- Previous criterion evaluates **within** cluster similarity, but does not promote large differences **between** clusters.
- N-cuts proposes maximizing the within cluster similarity **compared** to the across cluster difference
- Write graph nodes as V, part is cluster A, and the other is B.
- We have edges within A, B, and across A and B.

Normalized cuts

- N-cuts proposes maximizing the within cluster similarity **compared** to the across cluster difference.
- Define cut(A,B) to be the sum of the weights of the edges that you remove to split up the image.
- Define assoc(A,V) to be the sum of all the weights between elements in A and elements in V.

Normalized cuts

- Two equivalent formulations
- Minimize

$$\left(\frac{cut(A,B)}{assoc(A,V)}\right)+\left(\frac{cut(A,B)}{assoc(B,V)}\right)$$

- Maximize

$$\left(\frac{assoc(A,A)}{assoc(A,V)}\right)+\left(\frac{assoc(B,B)}{assoc(B,V)}\right)$$

Optional

Normalized cuts

- Let **y** be a vector whose elements are (ideally) 1 if the element is in A, and -b if it's in B.
  - b is theoretically defined for the derivation, but **y** is going to be estimated.
- Write the matrix of the graph as W, and the matrix which has the row sums of W on its diagonal as D. Let **1** be a vector with all ones.
- With some algebra, the criterion becomes $\min_y\left(\frac{y^T(D-W)y}{y^T Dy}\right)$
- And we have a constraint $y^T D1 = 0$
- This is hard to do, because y's values are quantized

Normalized cuts

Optional

- Instead, solve the generalized eigenvalue problem

$$\max_y \left(y^T (D-W) y \right) \text{ subject to } \left(y^T D y = 1 \right)$$

- which gives

$$(D-W)y = \lambda D y$$

- Now look for a quantization threshold that maximizes the criterion --- i.e all components of **y** above that threshold go to one, all below go to -b

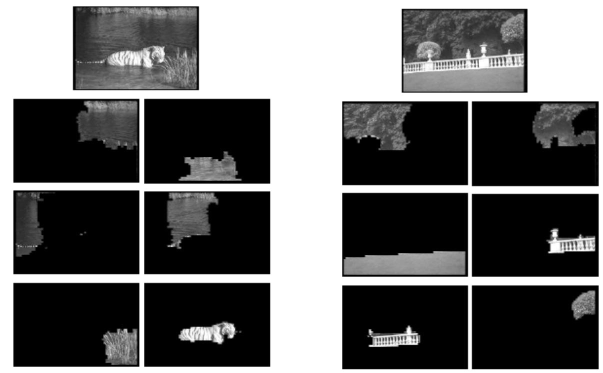


Figure from "Image and video segmentation: the normalised cut framework", by Shi and Malik, copyright IEEE, 1998

Grouping by Fitting to a Model

- Work with a parametric representation for "objects"
  - (e.g "line", "ellipse").
- 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 a given set of tokens best?
  - which of several objects gets which token? (**correspondence**!)
  - how many objects are there?

Example: Hough Transform for lines

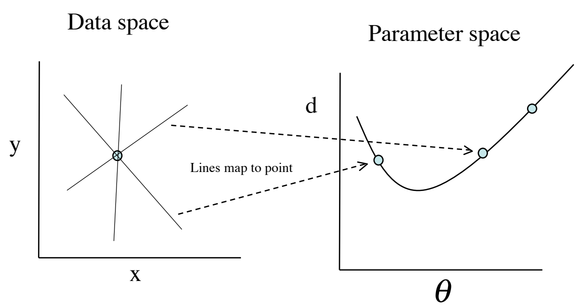
- A line is the set of points (x, y) such that

$$(\sin\theta)x + (\cos\theta)y + d = 0$$

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

$$(\sin\theta)x + (\cos\theta)y + d = 0$$

- The choice of θ fixes d. The family of lines has **one** parameter.



$$(\sin\theta)x + (\cos\theta)y + d = 0$$

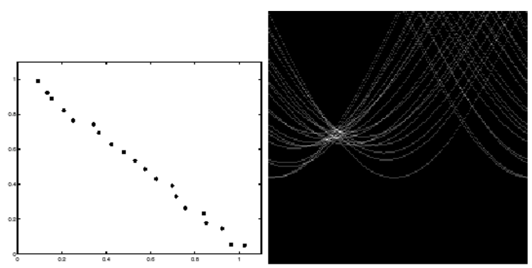
(Note: Curve is **not** accurate)

Example: Hough Transform for lines

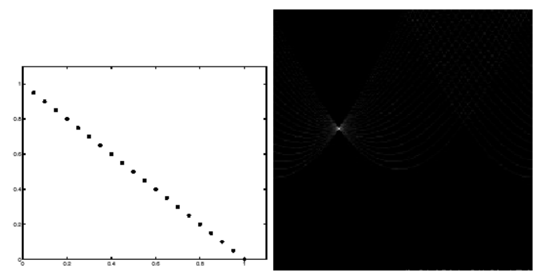
- Main idea: Each observed (x,y) votes for all (θ, d) satisfying

$$(\sin\theta)x + (\cos\theta)y + d = 0$$

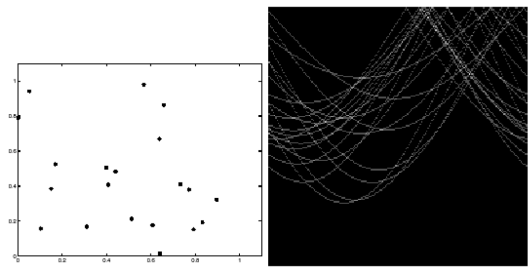
- Discretize the parameter space (θ, d) by an array
- Now each (x,y) leads to a bunch of votes (counts) in a (θ, d) grid (along the curve in the preceding slide).
- To find lines, let all edge points (x,y) vote, and look for (θ, d) cells with lots of votes.



tokens votes



tokens votes



tokens votes