CS 696i, Spring 2005, part III

Computer Vision

(Making Machines See)

and other miscellaneous stuff

Kobus Barnard

Computer Science, University of Arizona

Computer vision in 100 easy minutes

Be on the lookout for ambiguityand dealing with it.

Bottom up processing---assemble an understanding of images from the pixels upwards: Reason about the pixels in terms of image formation, find edges, regions, estimate distance, group things based on low level inference.

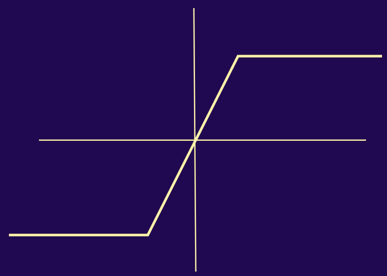
Top down---you have a model of what you are looking for. Can constrain the bottom up activities.

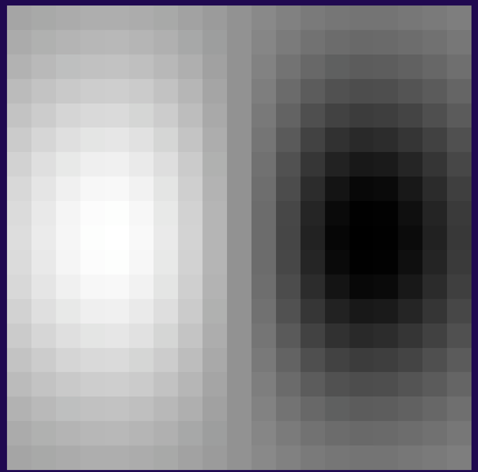
Computer vision in 100 easy minutes

(70 minutes to go)

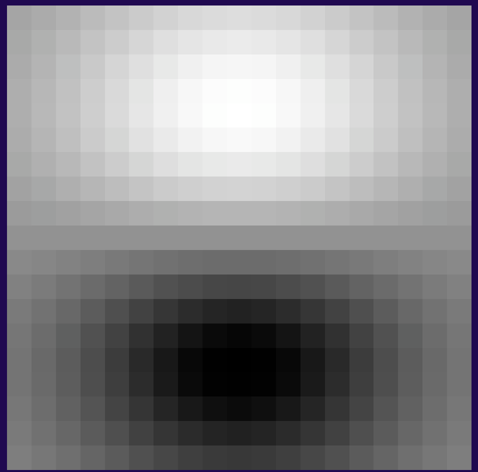
Edge detection

Step edge filter

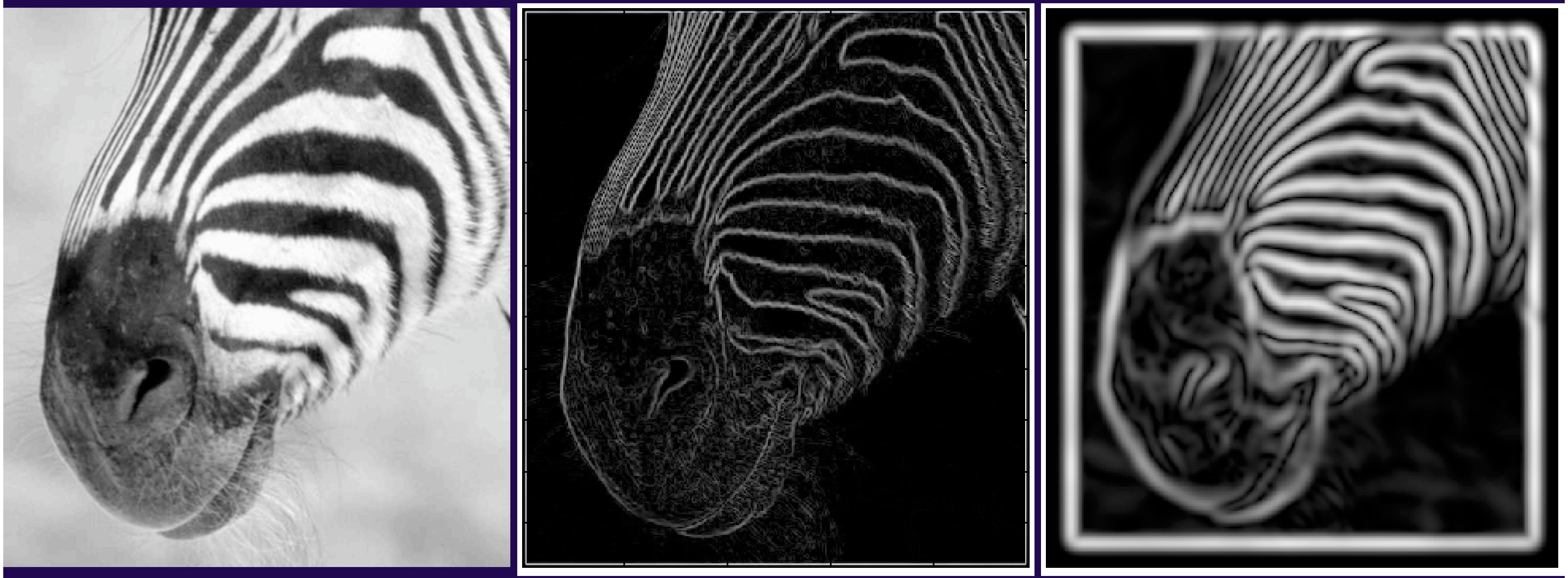


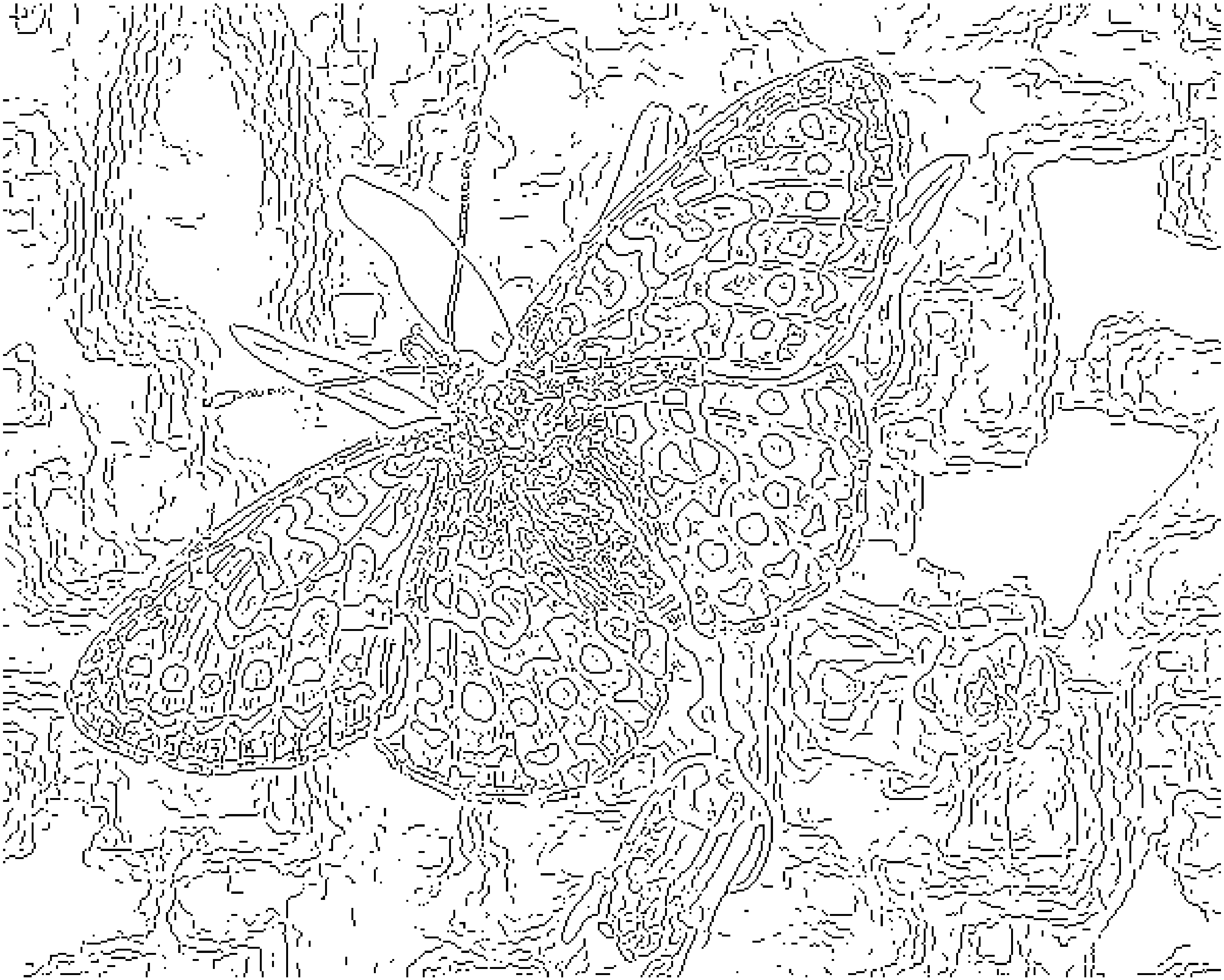


Smoothed
edge filters



Edge detection





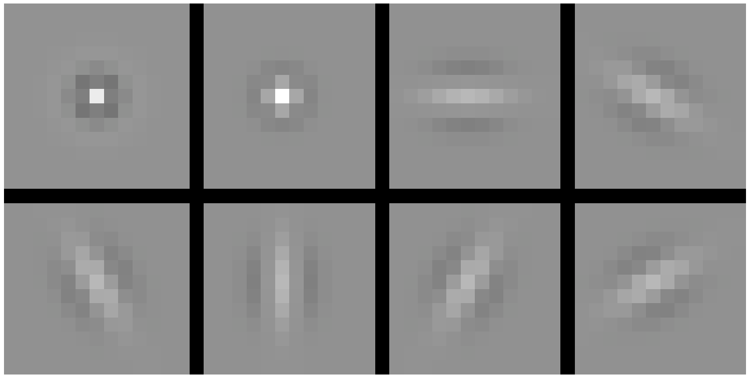
fine scale
high
threshold

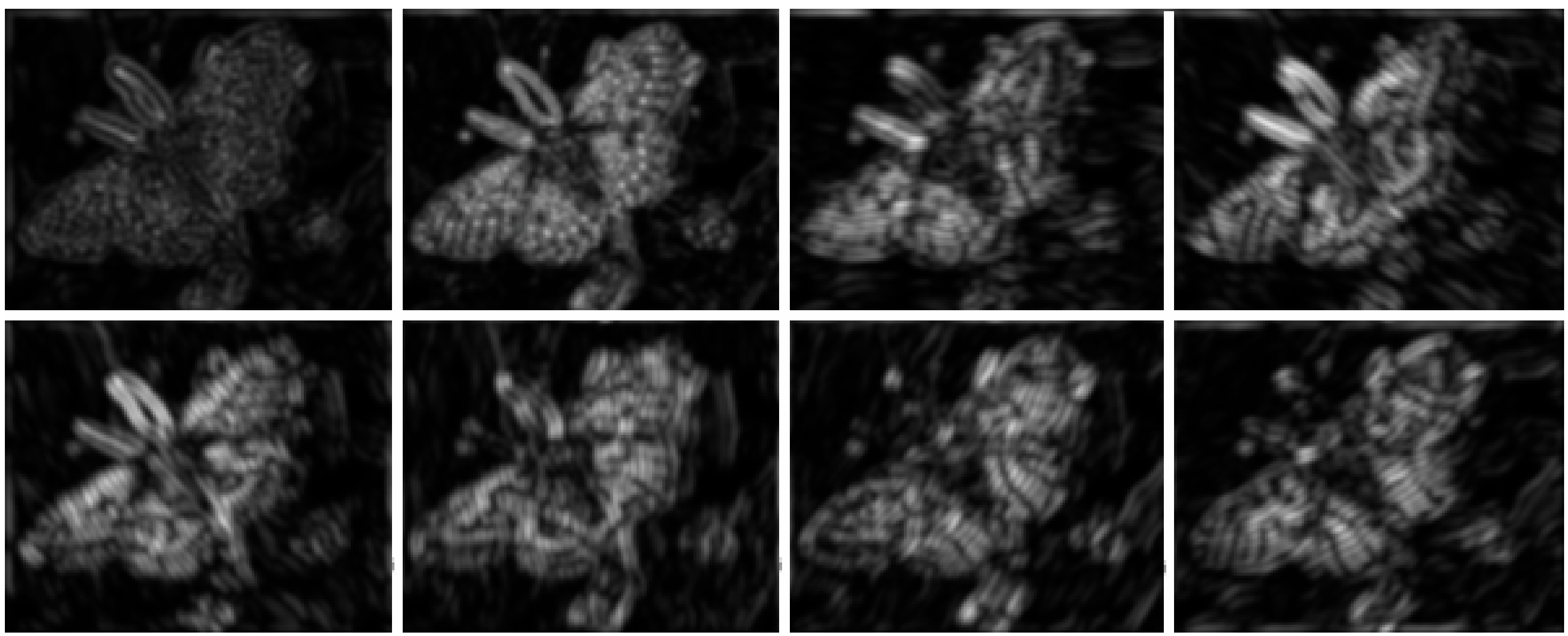
coarse
scale,
high
threshold

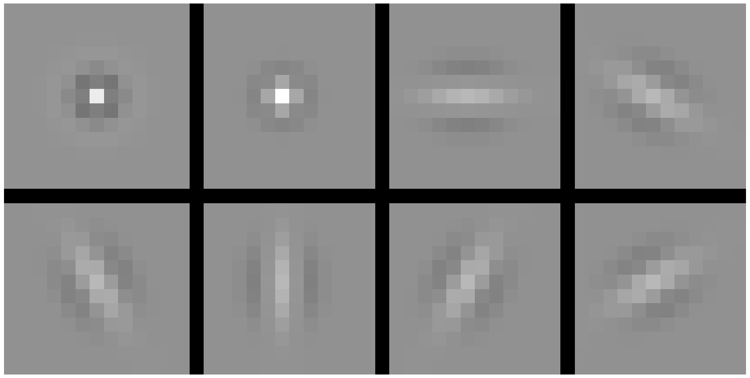
coarse
scale
low
threshold

Representing textures

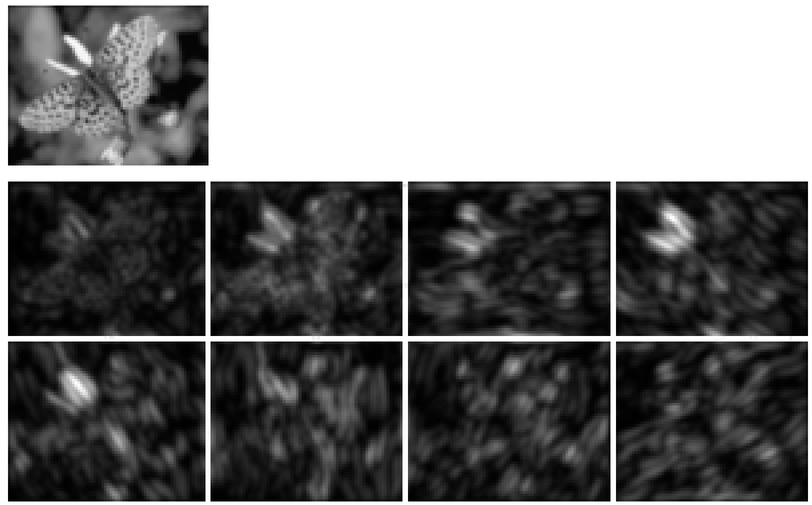
- Standard approach to representing textures is to use the statistics of the responses to a variety of filters --- filter bank.

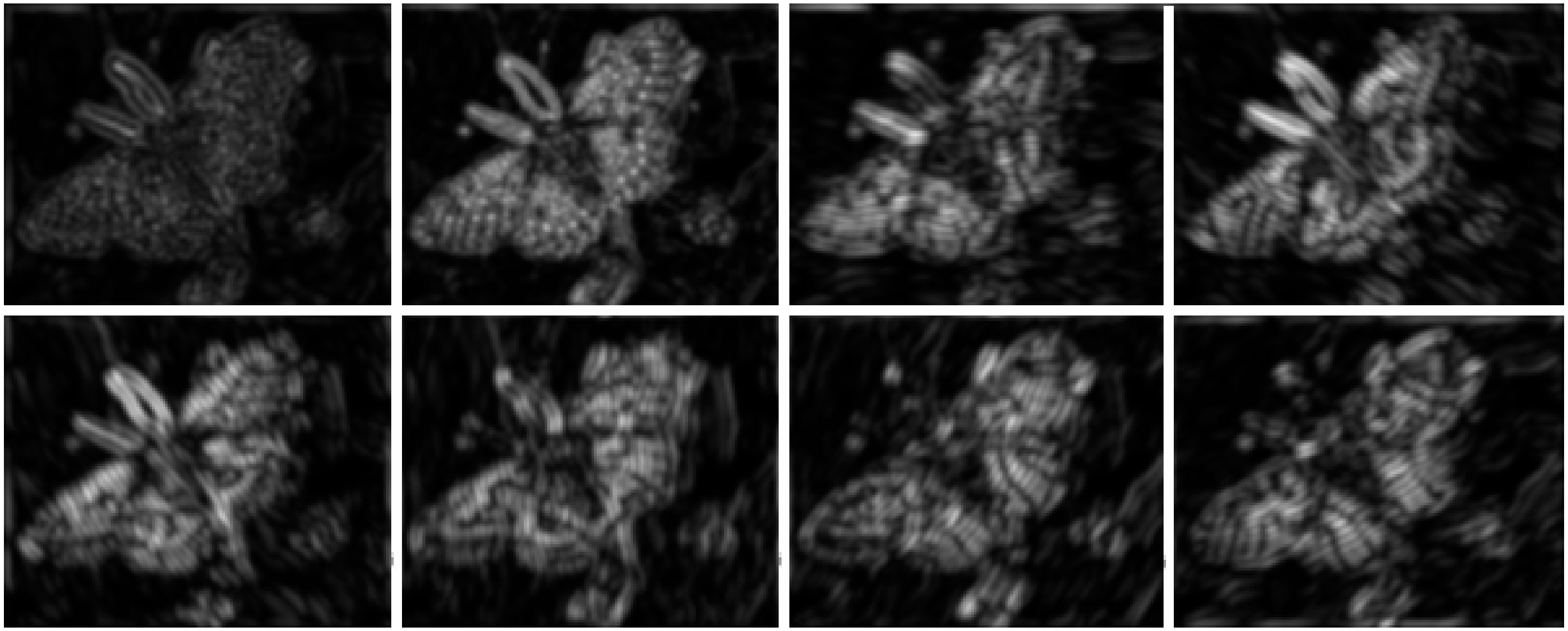




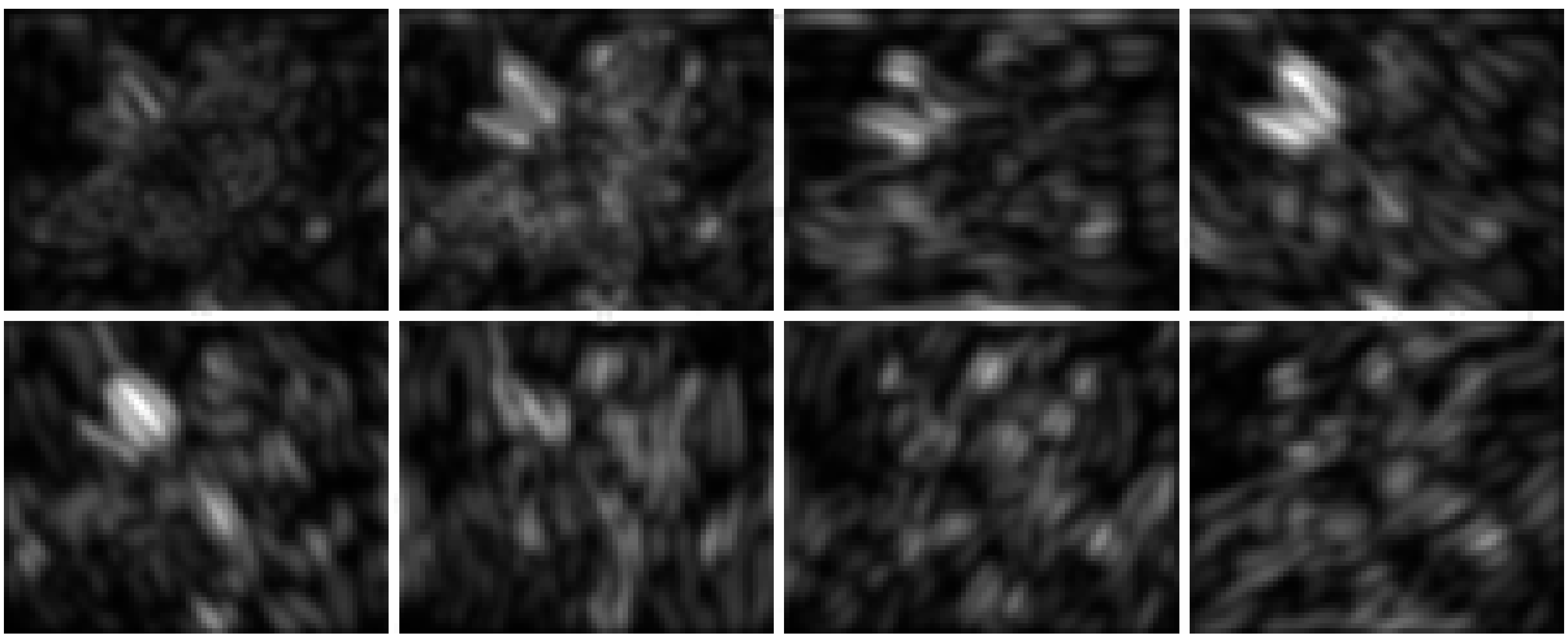


Next scale up





Smaller Scale



Larger Scale (Image from previous slide made larger to compare)

A typical filter bank

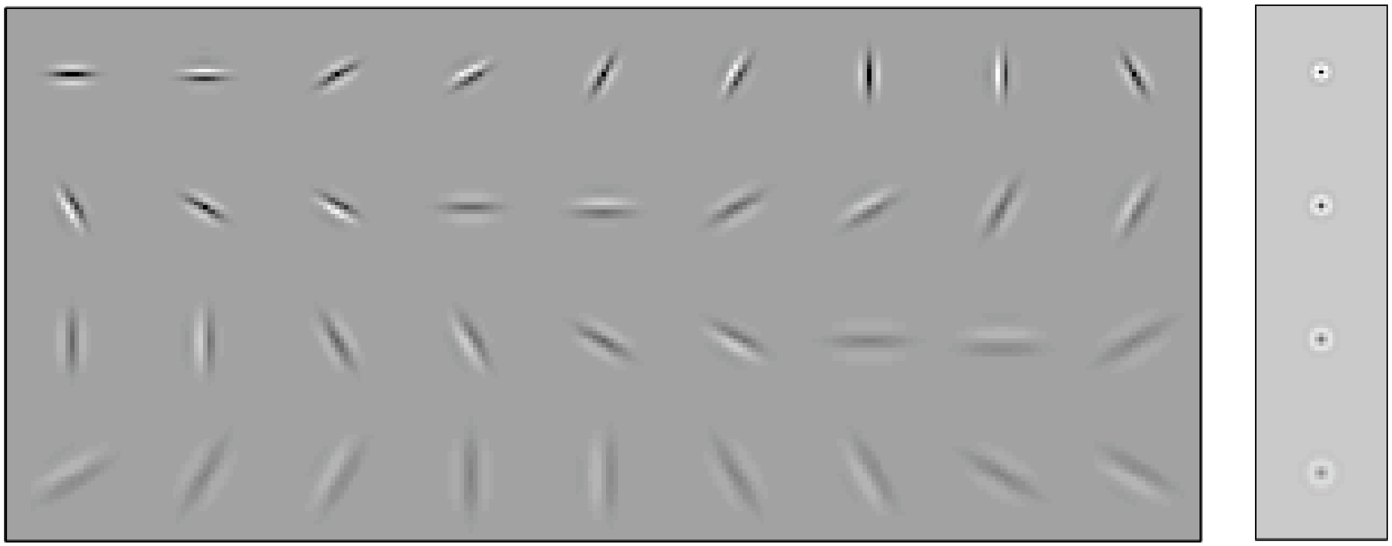


Figure 4. Left: Filter set f_i consisting of 2 phases (even and odd), 3 scales (spaced by half-octaves), and 6 orientations (equally spaced from 0 to π). The basic filter is a difference-of-Gaussian quadrature pair with $3:1$ elongation. Right: 4 scales of center-surround filters. Each filter is L_1-normalized for scale invariance.

From Malik et al., “Contour and texture analysis for image segmentation”

Shape from texture

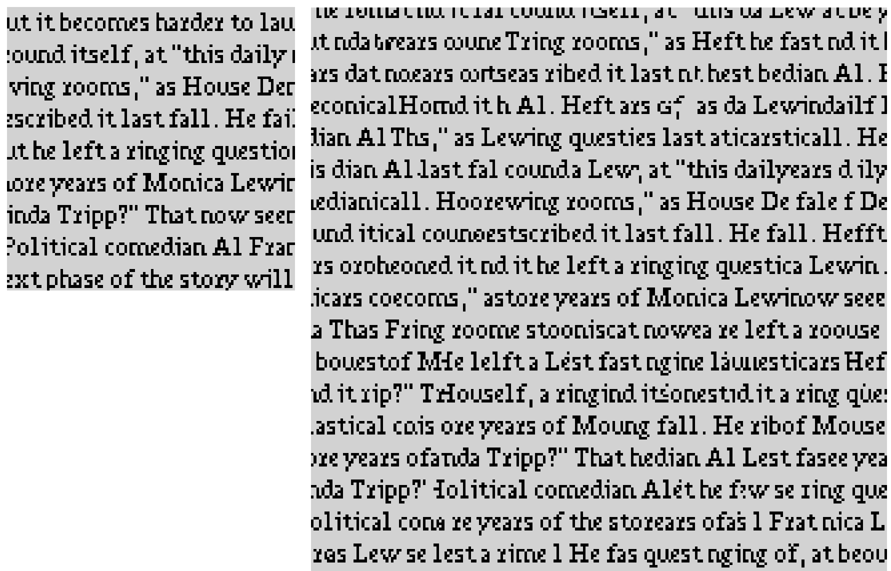


Figure from Texture Synthesis by Non-parametric Sampling, A. Efros and T.K. Leung, Proc. Int. Conf. Computer Vision, 1999 copyright 1999, IEEE

Invariants

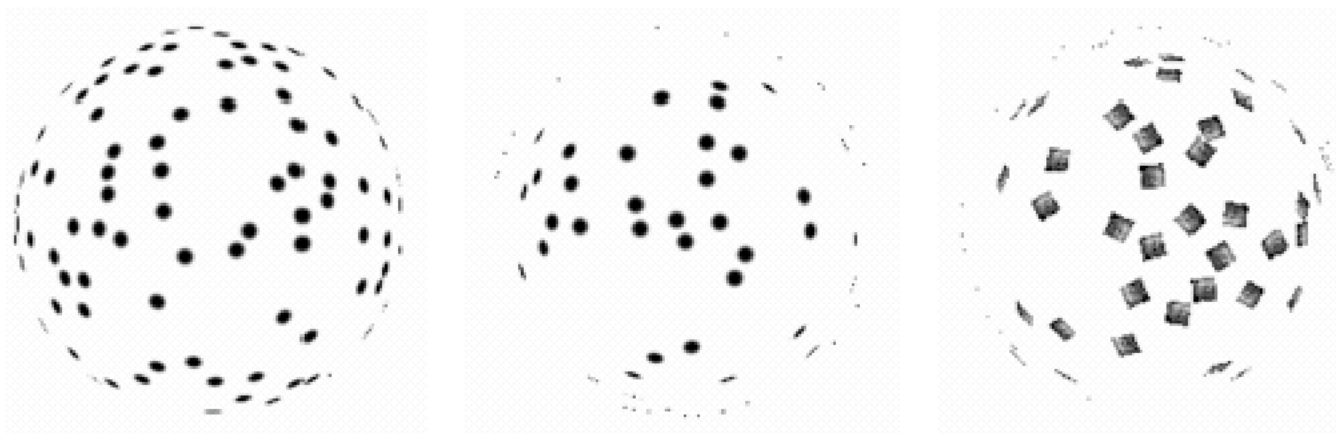
A computed quantity is invariant with respect to X if changing X does not change the result

Can use to match entities under changing X

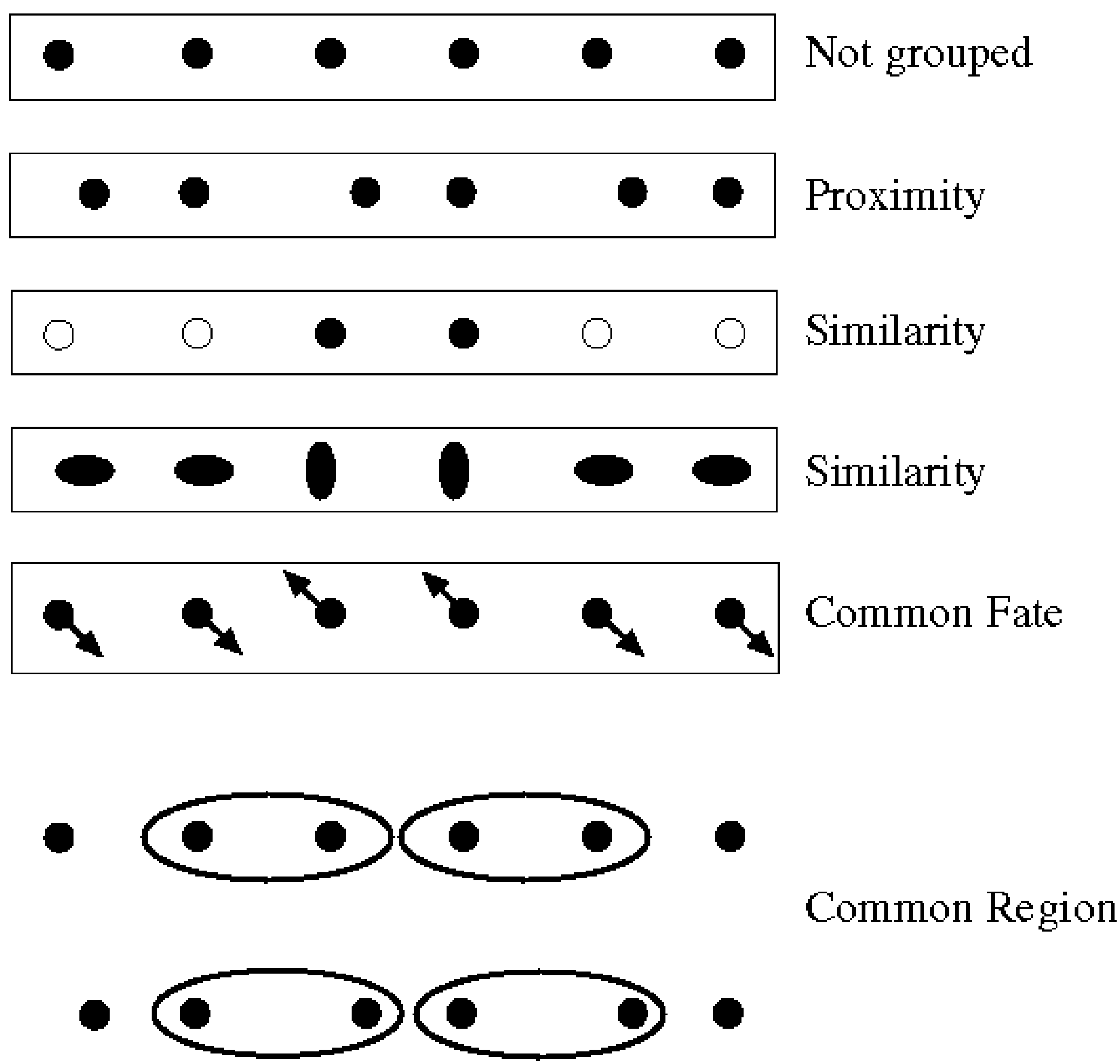
Common values for X:

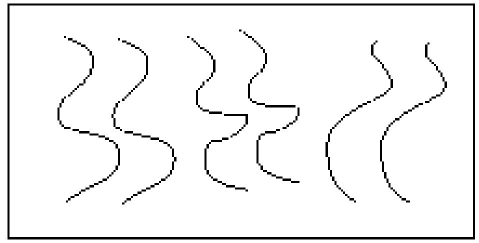
- illumination brightness
- image scale
- affine projection

Grouping / Segmentation

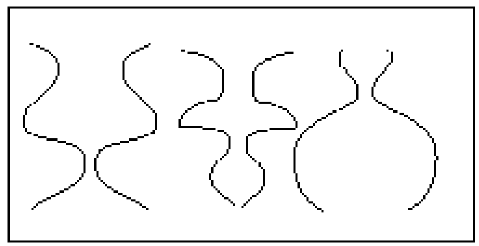


Why do these tokens belong together?

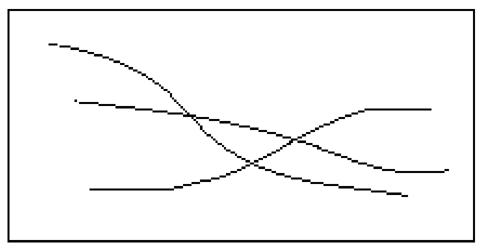

Not grouped
Proximity
Similarity
Similarity
Common Fate
Common Region



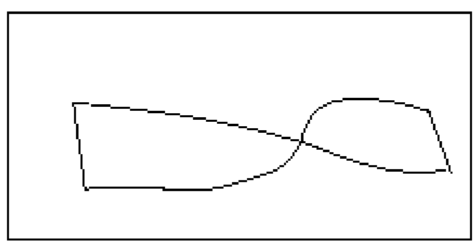
Parallelism



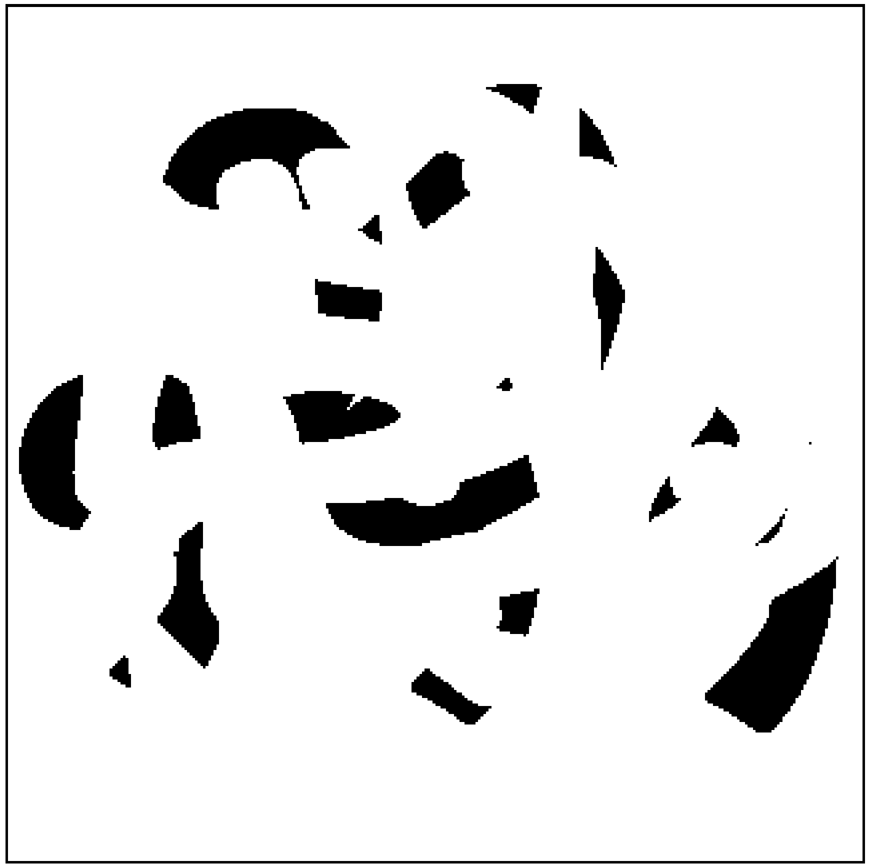
Symmetry



Continuity



Closure



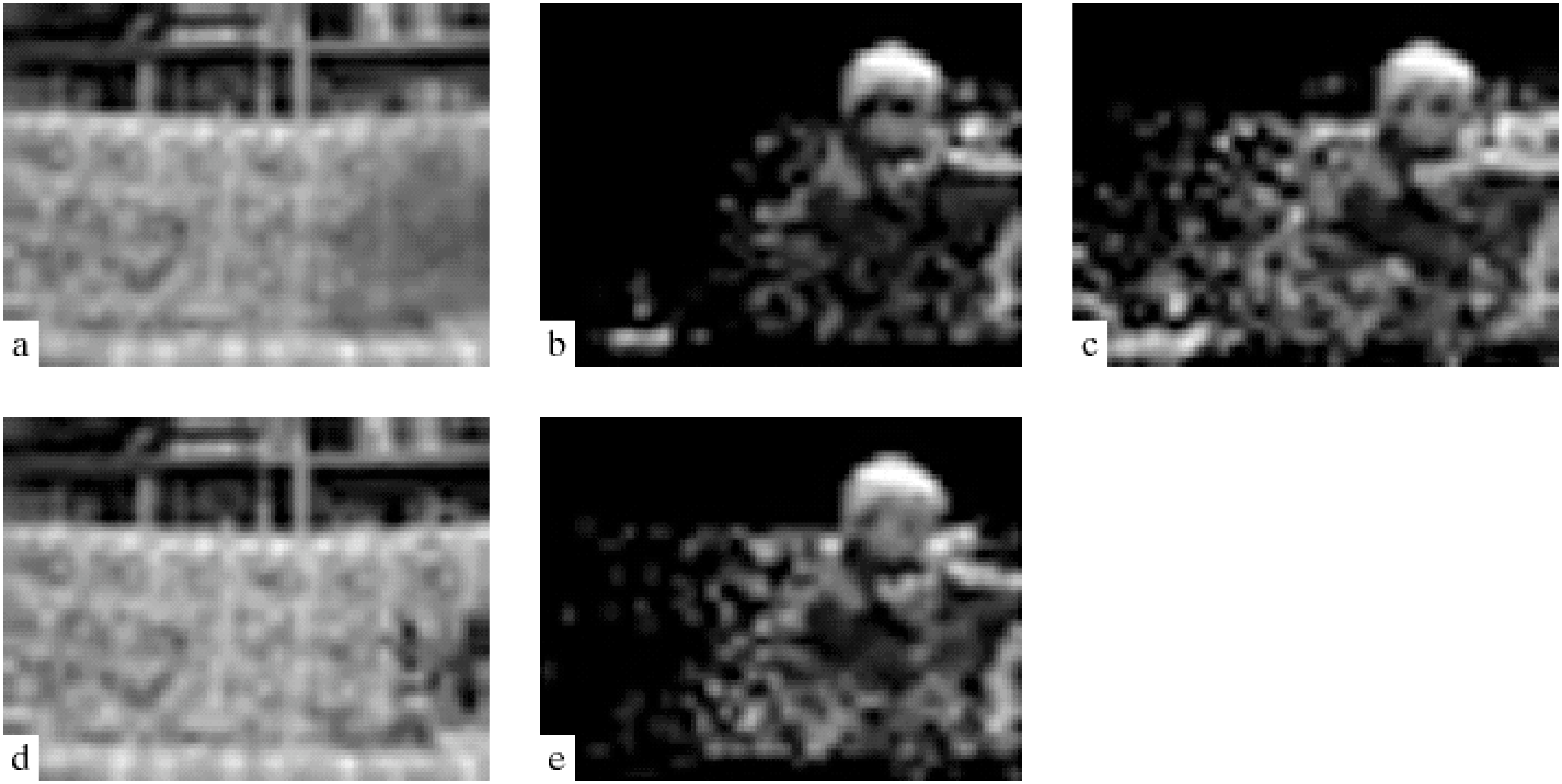
Skipped in class

Using motion: Background Subtraction

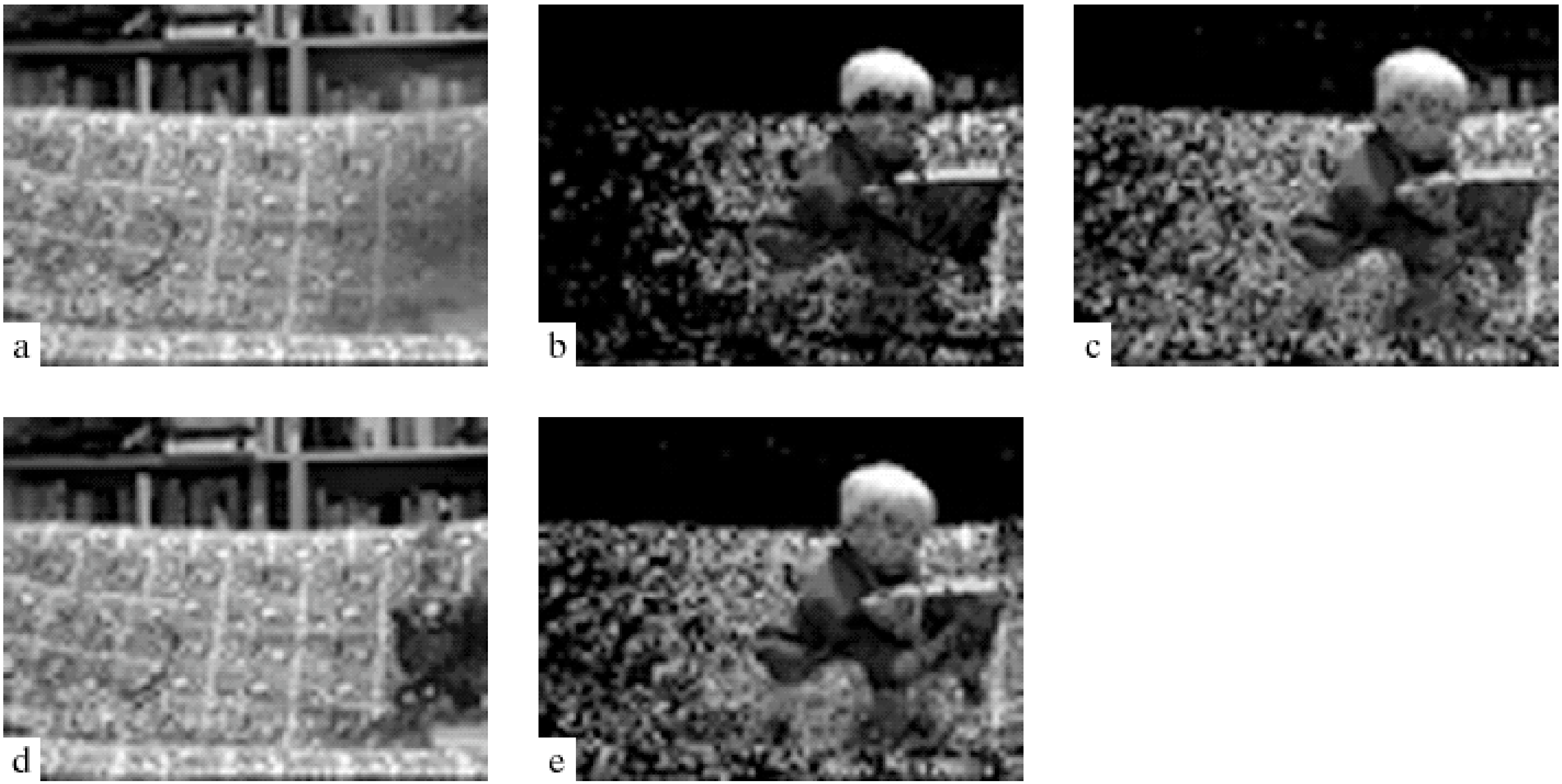
- Grouping/segmentation is hard, but if the interesting bits move together this makes things easier
- Often posited that motion helps "bootstrap" learning to "see"
- **Background subtraction**: If an object moves relative to a background, it is easy to segment it.

Skipped in class

Skipped in class



Skipped in class



Skipped in class

Tracking

- Once an object has been found, tracking it while it movies is (**relatively**) easy
- Typical modern tracker simultaneously builds/updates a probabilistic model of the object/configuration and the track
- If object changes motion, then more computation needs to be expended to evaluate additional hypothesis
- The current motion estimate gives a strong prior on what to associate with the object in the next frame

Segmentation

- Group together (pixels, tokens, etc.) that belong together
- Typically use “segmentation” to refer to a low level process
- Assume that there is a distance function between (pixels, tokens)

Segmentation as clustering

- Agglomerative clustering
  - initialize: every token is a cluster
  - attach closest to cluster it is closest to
  - repeat
- Divisive clustering
  - split cluster along best boundary
  - repeat
- When to stop? (The million dollar question==how many clusters?)

K-Means

- Like the EM clustering process in your assignment, but **not** as clever (K-means is used to initialize EM).
- Choose a fixed number of clusters
- Choose cluster centers and point-cluster allocations to minimize error

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

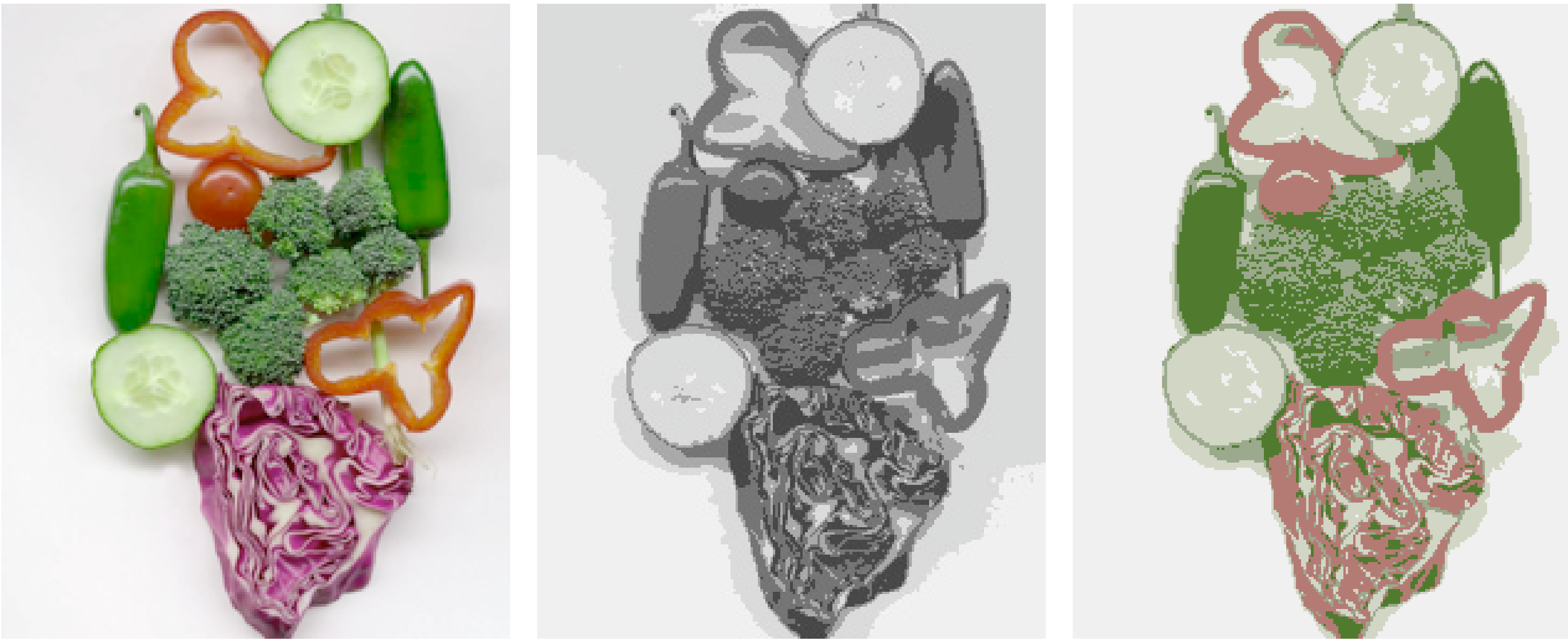
K-Means

$$\sum_{i \in \text{clusters}} \left\{ \sum_{j \in \text{elements of i'th cluster}} \left\| x_j - \mu_i \right\|^2 \right\}$$

- Cannot do this optimization by search, because there are too many possible allocations.
- Standard difficulty which we handle with an iterative process

- Algorithm
  - fix cluster centers; allocate points to closest cluster
  - fix allocation; compute best cluster centers
- x could be any set of features for which we can compute a distance (careful about scaling/normalization)

Image　　Clusters on intensity　　Clusters on color



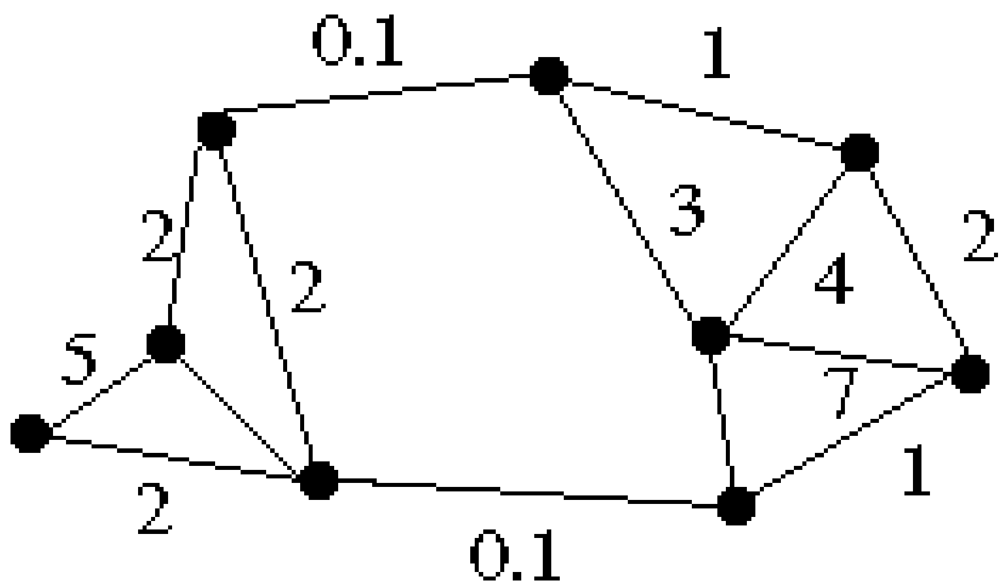
K-means clustering using intensity alone and color alone
(Assuming 5 segments, i.e. k=5)

Skipped in class

Graph theoretic clustering

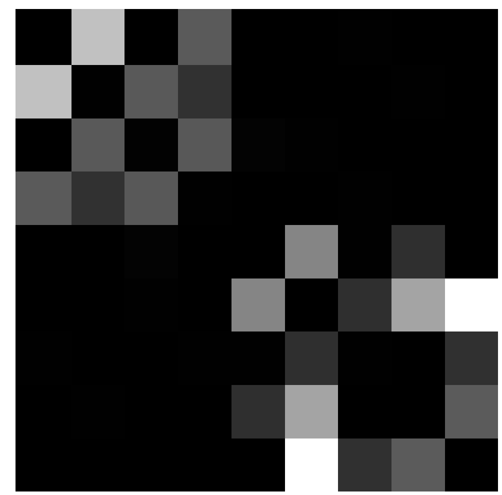
- Represent distance between tokens using a weighted graph.
  - affinity matrix
- Cut up this graph to get subgraphs with strong interior links and weak links between the subgraphs.

Skipped in class

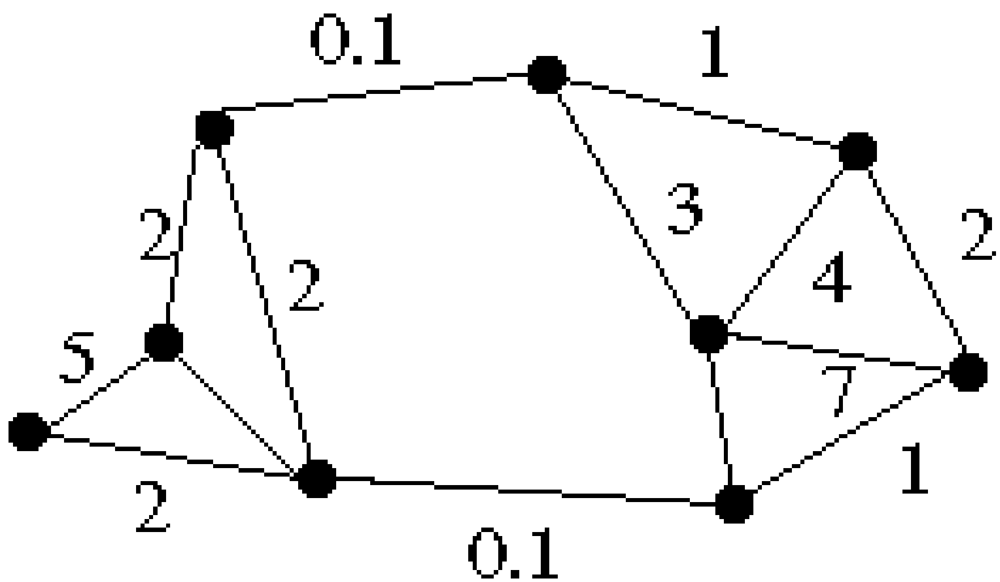


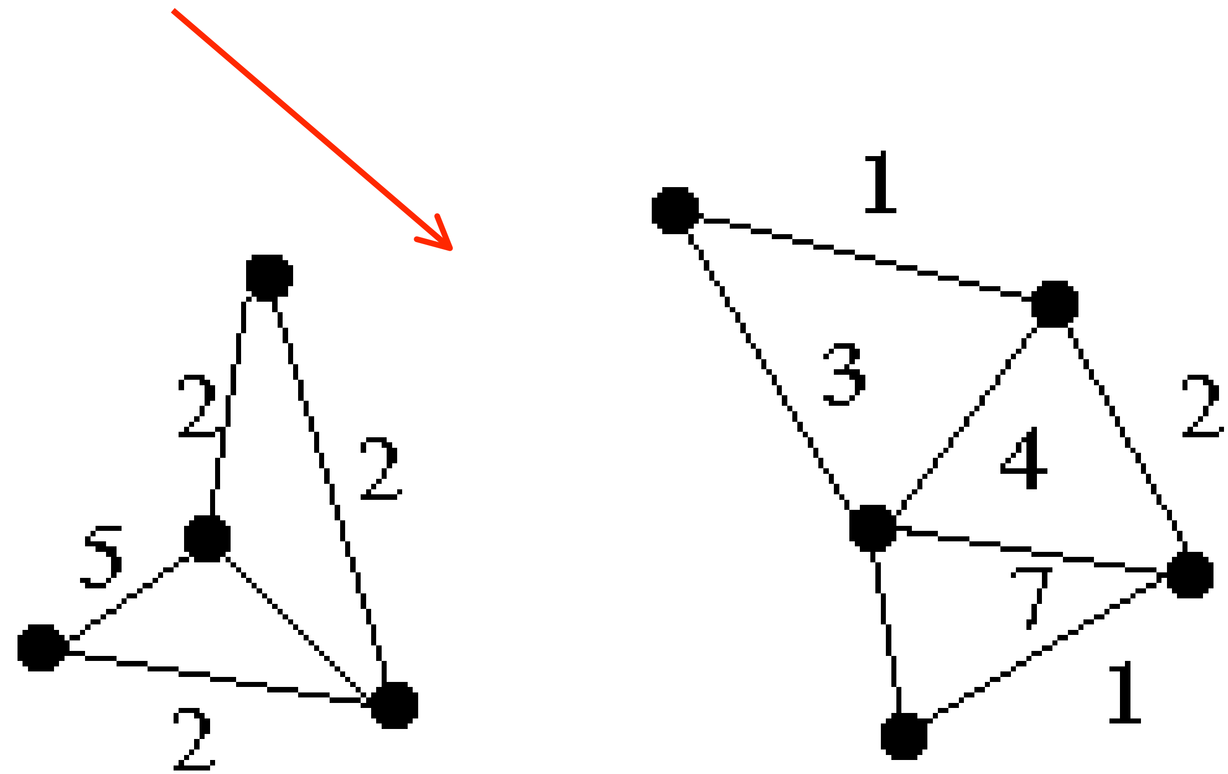
Graph for 9 tokens

Image representation of weight matrix



Skipped in class





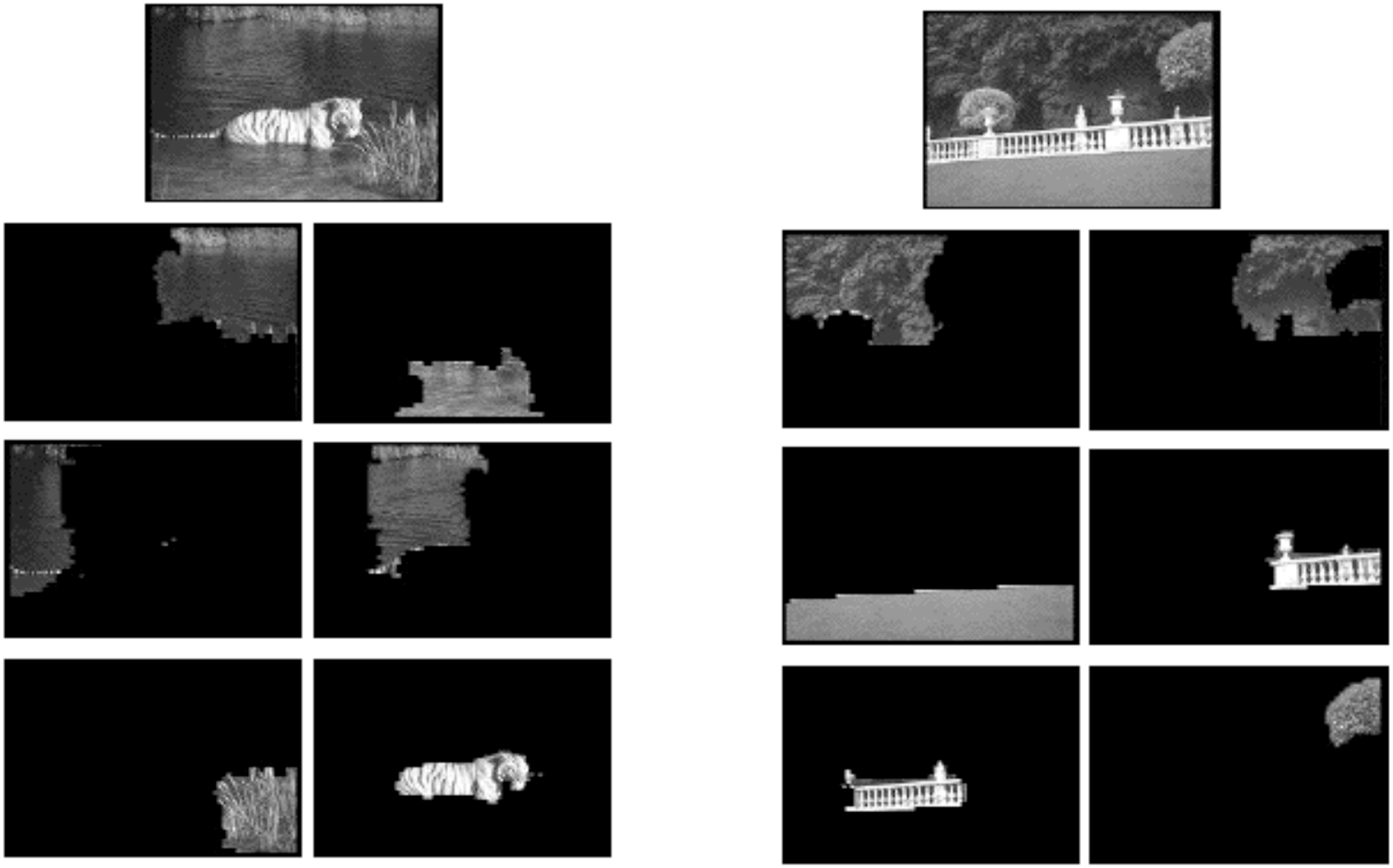


Figure from “Image and video segmentation: the normalised cut framework”, by Shi and Malik, copyright IEEE, 1998

Fitting

- Given some idea of what you are looking for (e.g., straight lines), find them by fitting the data to the "model"

Skipped in class

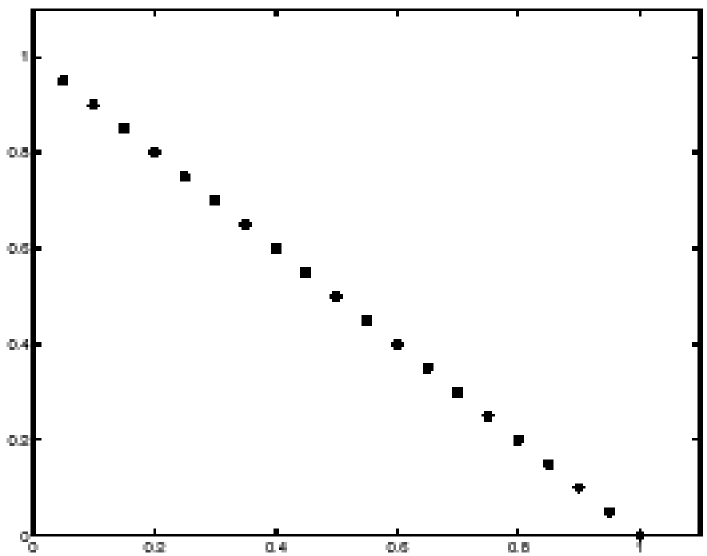
Voting methods / Hough Transform

- Specify what you are looking for by a small number of parameters.
- For example: A line is the set of points (x, y) such that

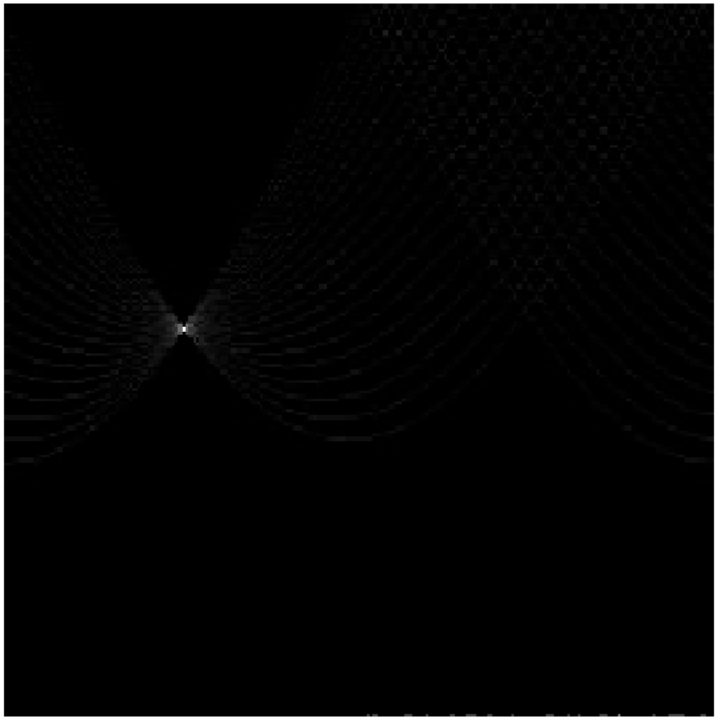
$$(\sin\theta)x + (\cos\theta)y + d = 0$$

- Space of possible lines is described by different θ and d
- Each point (x,y) gets to vote for every θ and d for each line passing through it.
- Record vote in a cell (box) in parameter (θ,d) space.
- If there is a line that has lots of votes, that should be the line passing through the point

Skipped in class

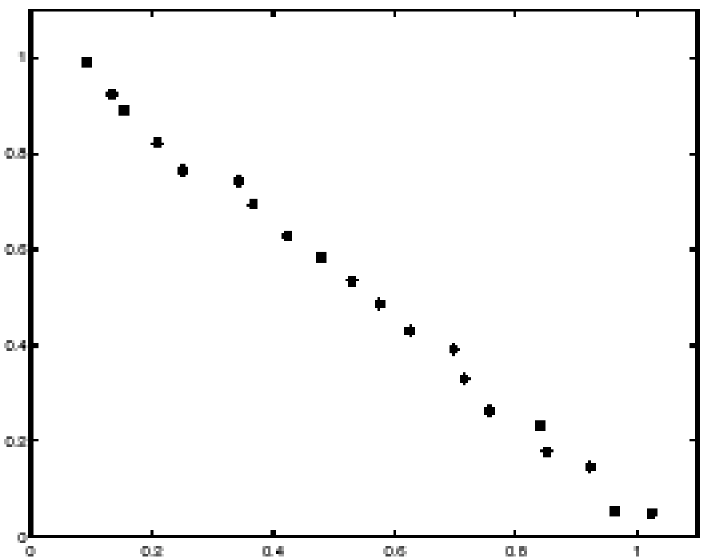


tokens

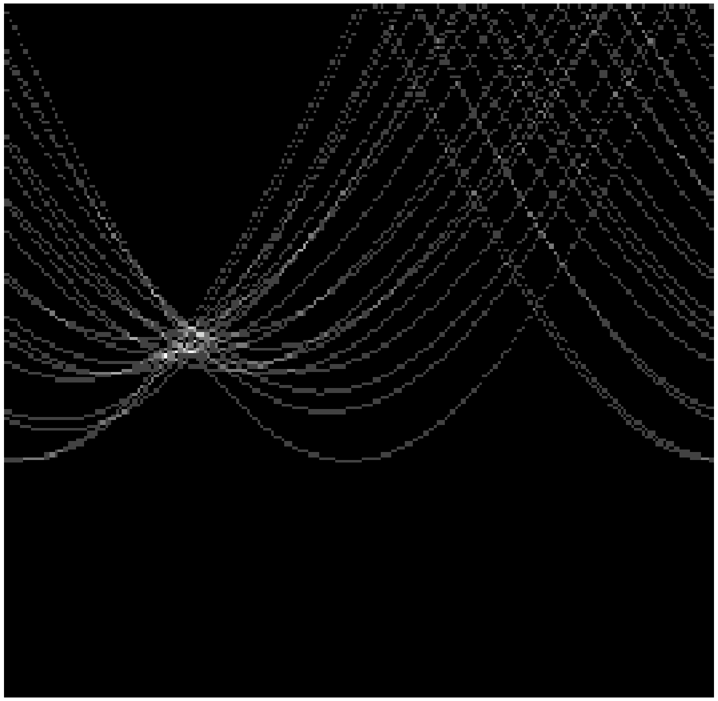


votes

Skipped in class

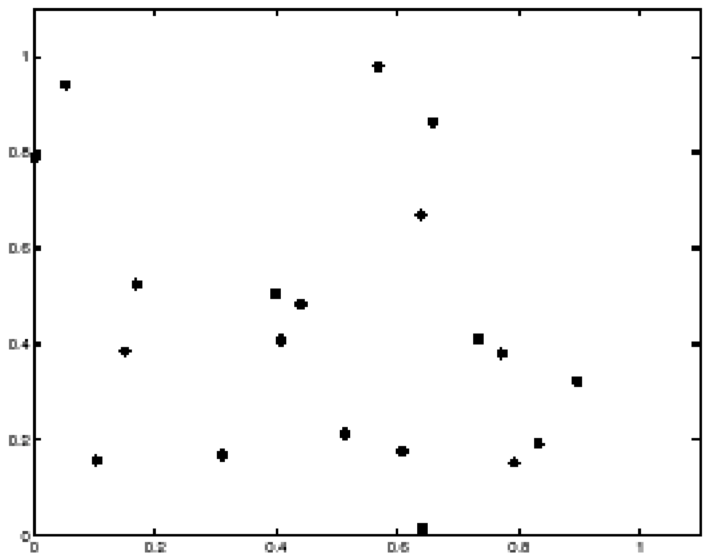


tokens

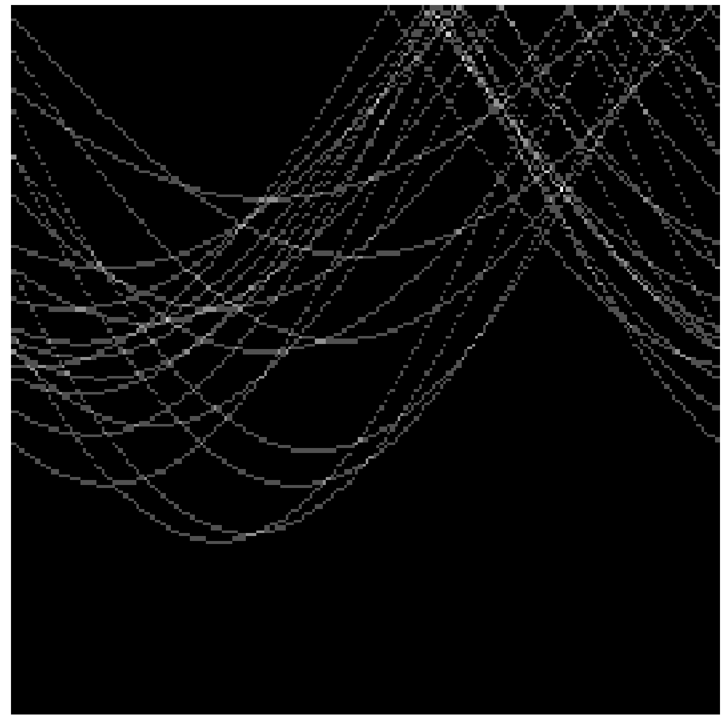


votes

Skipped in class



tokens



votes

Skipped in class

More on the Hough transform

- 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**

Probabilistic Fitting

- Given a model with parameters θ
- Now consider some observations, **x**
- Suppose that the observations are independent
- So, given the model, the probability of observing the data is given by

$$P(\mathbf{x} \mid \theta) = \prod P(x_i \mid \theta)$$

- But what we really want is the probability of the model (parameters) given the data!

Probabilistic Fitting

- Bayes rule: $P(A \mid B) = P(B \mid A)P(A)/P(B)$
- So, $P(\square \mid \mathbf{x}) = P(\mathbf{x} \mid \square)P(\square)/P(\mathbf{x})$
- $P(\square)$ is the prior probability on the parameters (often taken to be uniform
- $P(\mathbf{x})$ is usually not of interest
- Often use $P(\square \mid \mathbf{x}) \quad P(\mathbf{x} \mid \square)$

Probabilistic Fitting

- Now that we have the model posterior, we can use it for additional inferences, or point estimates.

- An example point estimate is the parameters □ such that this *likelihood* is maximum

- See line fitting example from two lectures ago.

RANSAC

- Choose a small subset uniformly at random
- Fit to that
- Anything that is close to result is signal; all others are noise
- Refit
- Do this many times and choose the best

Missing variable problems

- In many vision problems, if some variables were known the maximum likelihood inference problem would be easy
  - fitting; if we knew which line each token came from, it would be easy to determine line parameters
  - segmentation; if we knew the segment each pixel came from, it would be easy to determine the segment parameters
  - **Assignment two**
  - many, many, others!

Missing variable problems

- Strategy
  - estimate appropriate values for the missing variables
  - plug these in and now estimate parameters
  - re-estimate appropriate values for missing variables, continue
- Example with lines
  - guess which line gets which point
  - now fit the lines
  - now reallocate points to lines, using our knowledge of the lines
  - now refit, etc.
- We’ve seen this line of thought before (k means)

Missing variables - strategy

- In the Expectation-Maximization algorithm, we use the expected values of the missing values as the estimate.
- Thus iterate until convergence
  - replace missing variable with **expected** values, given **fixed** values of parameters
  - fix missing variables, choose parameters to maximize likelihood given fixed values of missing variables
- EM is basically gradient descent on the log likelihood.

Segmentation with EM

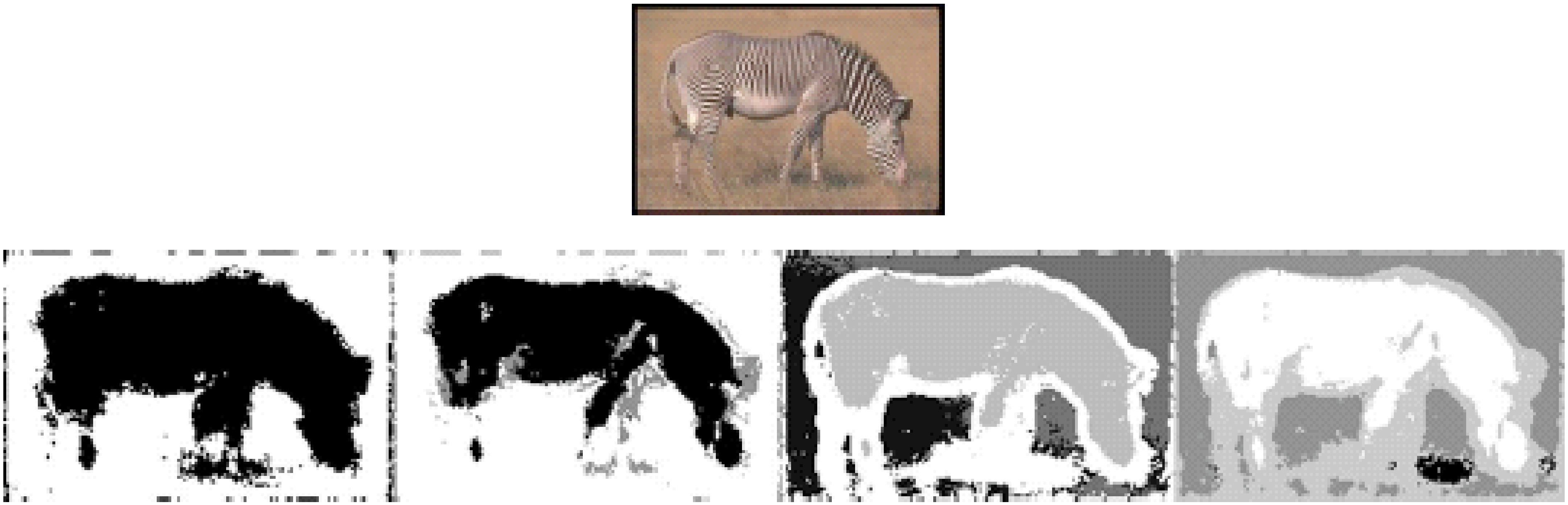


Figure from “Color and Texture Based Image Segmentation Using EM and Its Application to Content Based Image Retrieval”,S.J. Belongie et al., Proc. Int. Conf. Computer Vision, 1998, c1998, IEEE