Computer Vision
Segmentation

Professor Hager

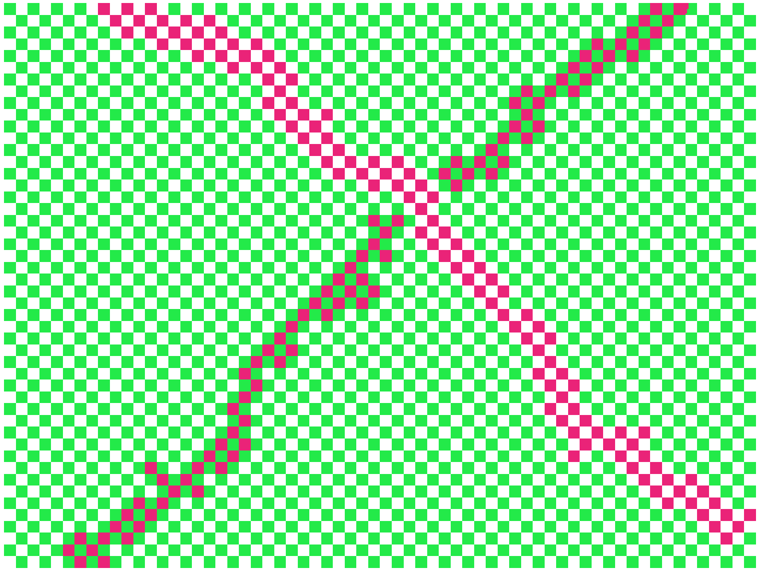
http://www.cs.jhu.edu/~hager

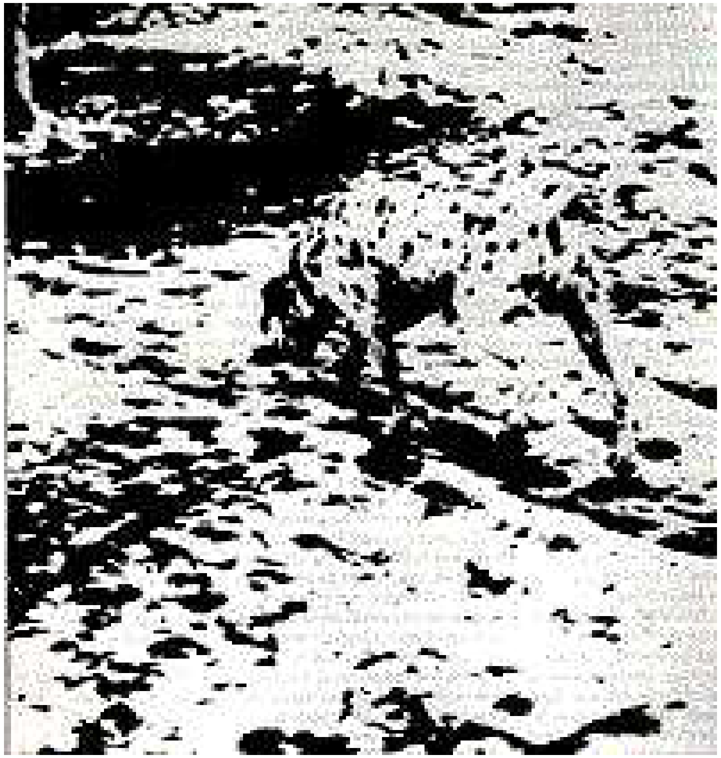
Grouping and Segmentation

- G&S appear to be one of the early processes in human vision
- They are a way of *organizing* image content into “semantically related” groups
- In some applications, segmentation is the crucial step (e.g. some types of aerial image interpretation).

Grouping and Segmentation

- **Grouping** is the process of associating similar image features together



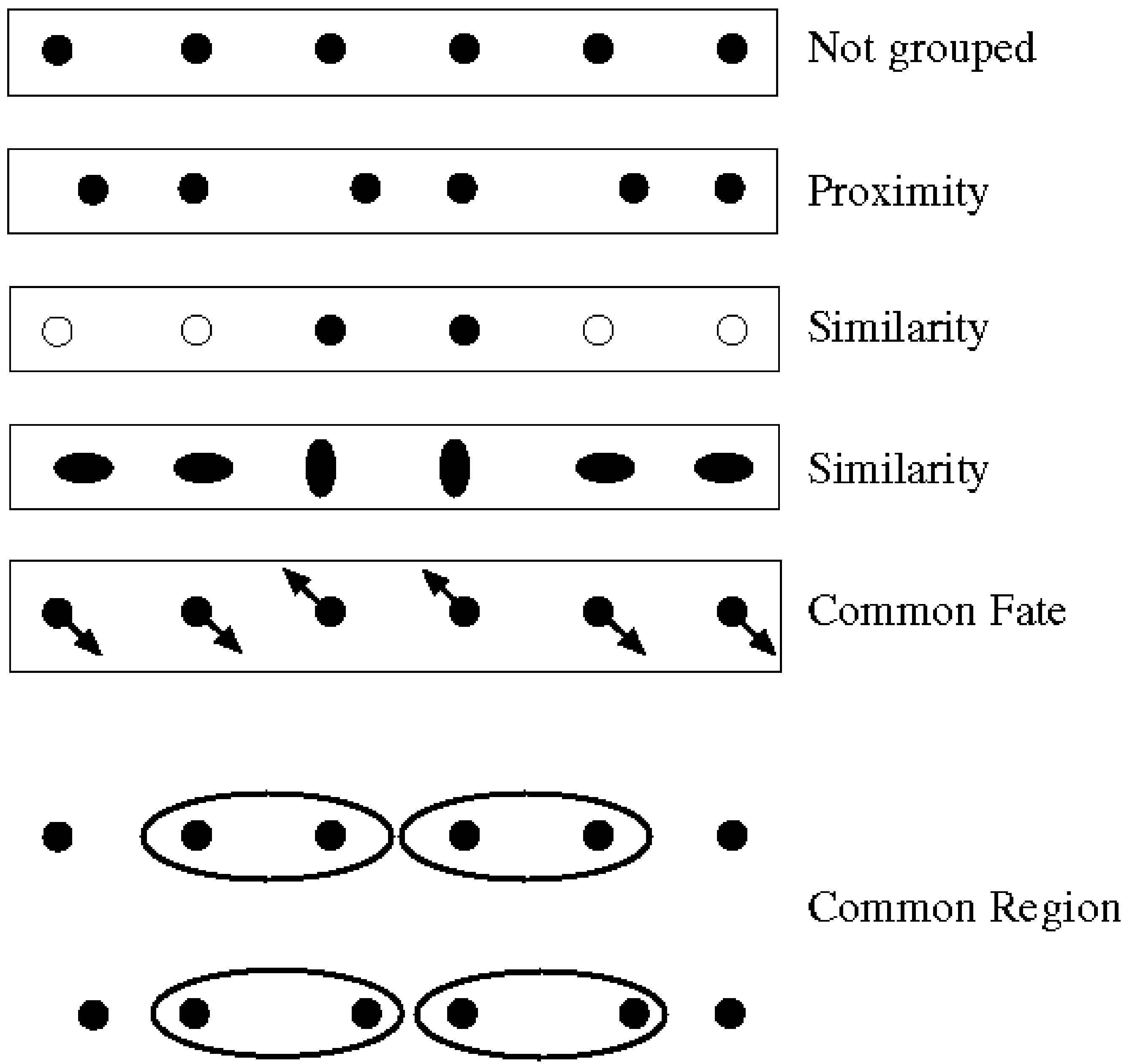


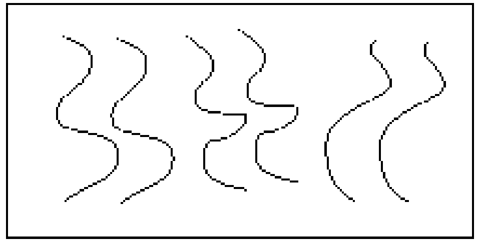
Grouping and Segmentation

- **Grouping** is the process of associating similar image features together

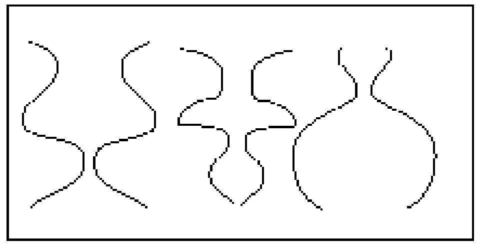
- The Gestalt School:

  – **Proximity**:tokens that are nearby tend to be grouped.
  – **Similarity:**similar tokens tend to be grouped together.
  – **Common fate**:tokens that have coherent motion tend to be grouped together.
  – **Common region:**tokens that lie inside the same closed region tend to be grouped together.
  – **Parallelism:**parallel curves or tokens tend to be grouped together.
  – **Closure:**tokens or curves that tend to lead to closed curves tend to be grouped together.
  – **Symmetry:**curves that lead to symmetric groups are grouped together.
  – **Continuity:**tokens that lead to “continuous ” (as in “joining up nicely ”, rather than in the formal sense): curves tend to be grouped.
  – **Familiar Conguration:**tokens that, when grouped, lead to a familiar object,tend to be grouped together

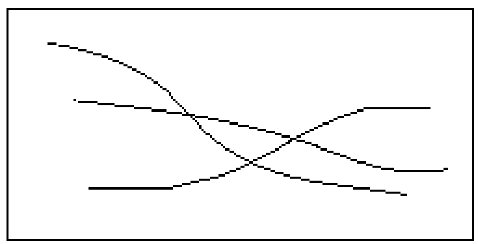

Not grouped
Proximity
Similarity
Similarity
Common Fate
Common Region



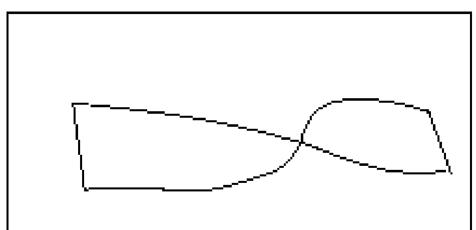
Parallelism



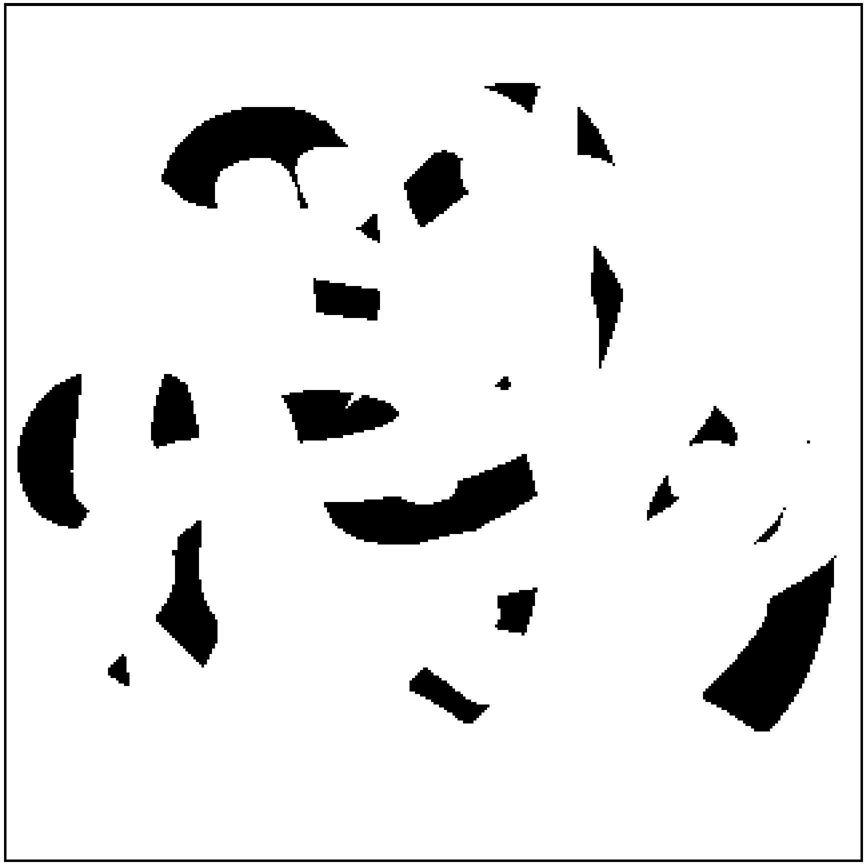
Symmetry



Continuity

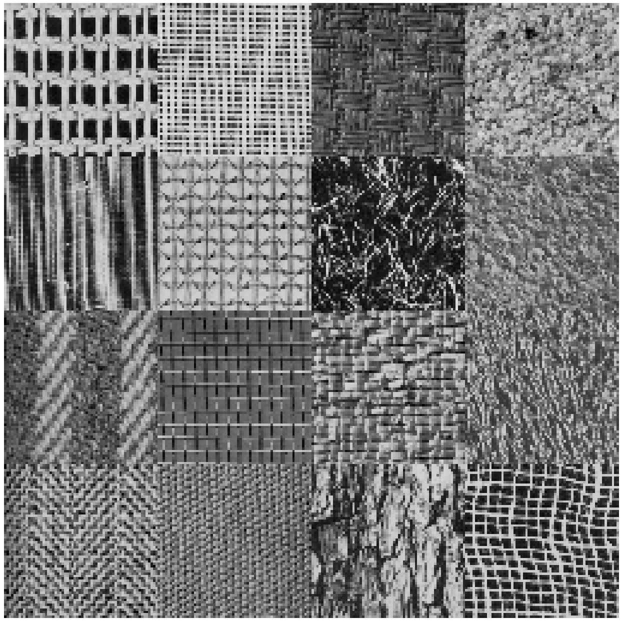


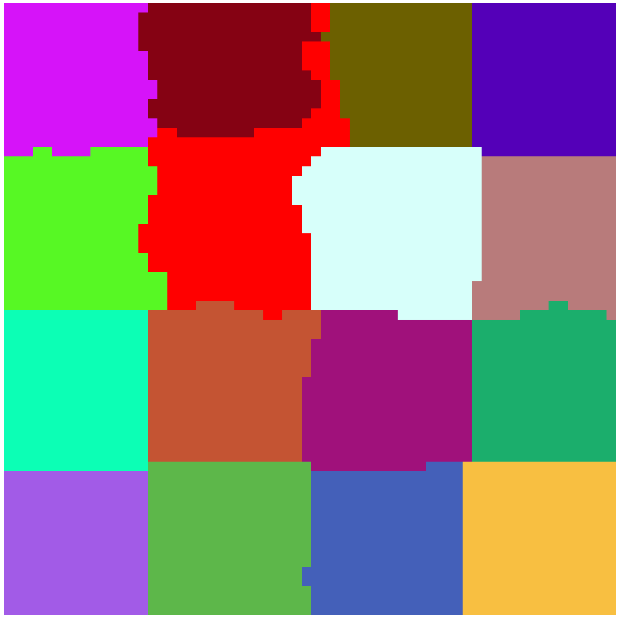
Closure



Grouping and Segmentation

- **Segmentation** is the process of dividing an image into regions of "related content"

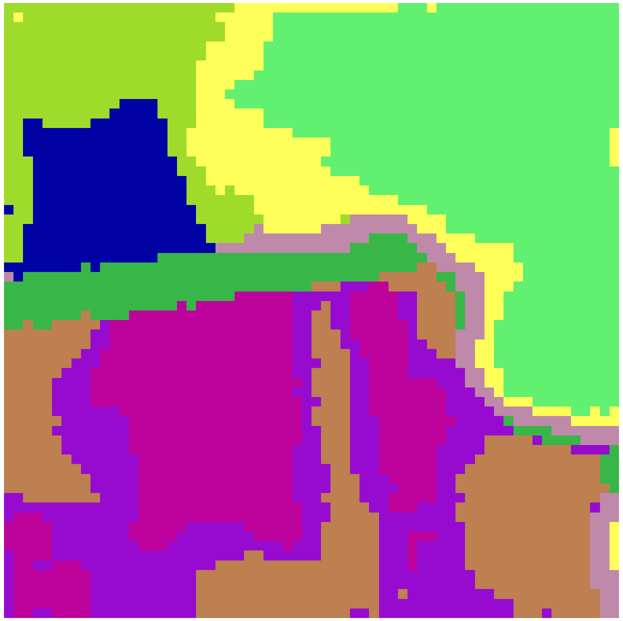




Courtesy Uni Bonn

Grouping and Segmentation

- **Segmentation** is the process of dividing an image into regions of "related content"

Courtesy Uni Bonn

Grouping and Segmentation

- Both are an ill defined problem --- related or similar is often a high-level, cognitive notion
  - an unclear role --- is this an “early” process that drives later processes?
- The literature on segmentation and grouping is large and generally inconclusive --- we’ll discuss a couple of algorithms and an example.

Image Segmentation RoadMap

- Segmentation
  - criteria
  - region group and counting
- Simple Color Segmentation
- Color Histograms and matching
- Selection of Color Regions
- K means
- Graph Cuts

Segmentation: Definitions

- An affinity measure $d(R_1,R_2) \rightarrow \Re$ or a homogeneity measure $m(R)$
  - note possibly $d(R_1,R_2) = |m(R_1) – M(R_2)|$
- An threshold τ (could operate on distance or homogeneity)
- A region definition (e.g. square tiles)
- A neighborhood definition
  - 4 neighbors

```
  x
x 0 x
  x
```

  - 8 neighbors

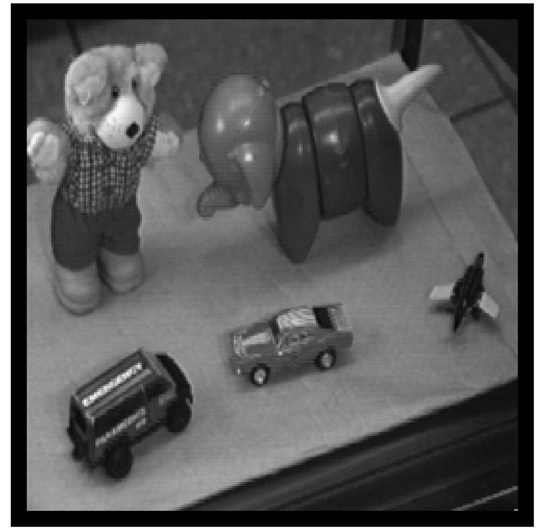
```
x x x
x 0 x
x x x
```

Simple Thresholding

- Choose an image criterion c
- Compute a binary image by $b(i,j) = 1$ if $c(I(i,j)) > t$; 0 otherwise
- Perform “cleanup operations” (image morphology)
- Perform grouping
  - Compute connected components and/or statistics thereof

An Example: Motion

Detecting motion:

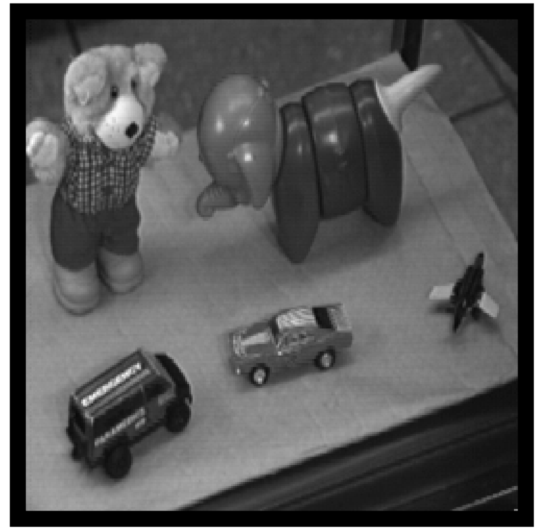


Thresholded Motion

Detecting motion:

|

−



|

> 50

Candidate areas for motion

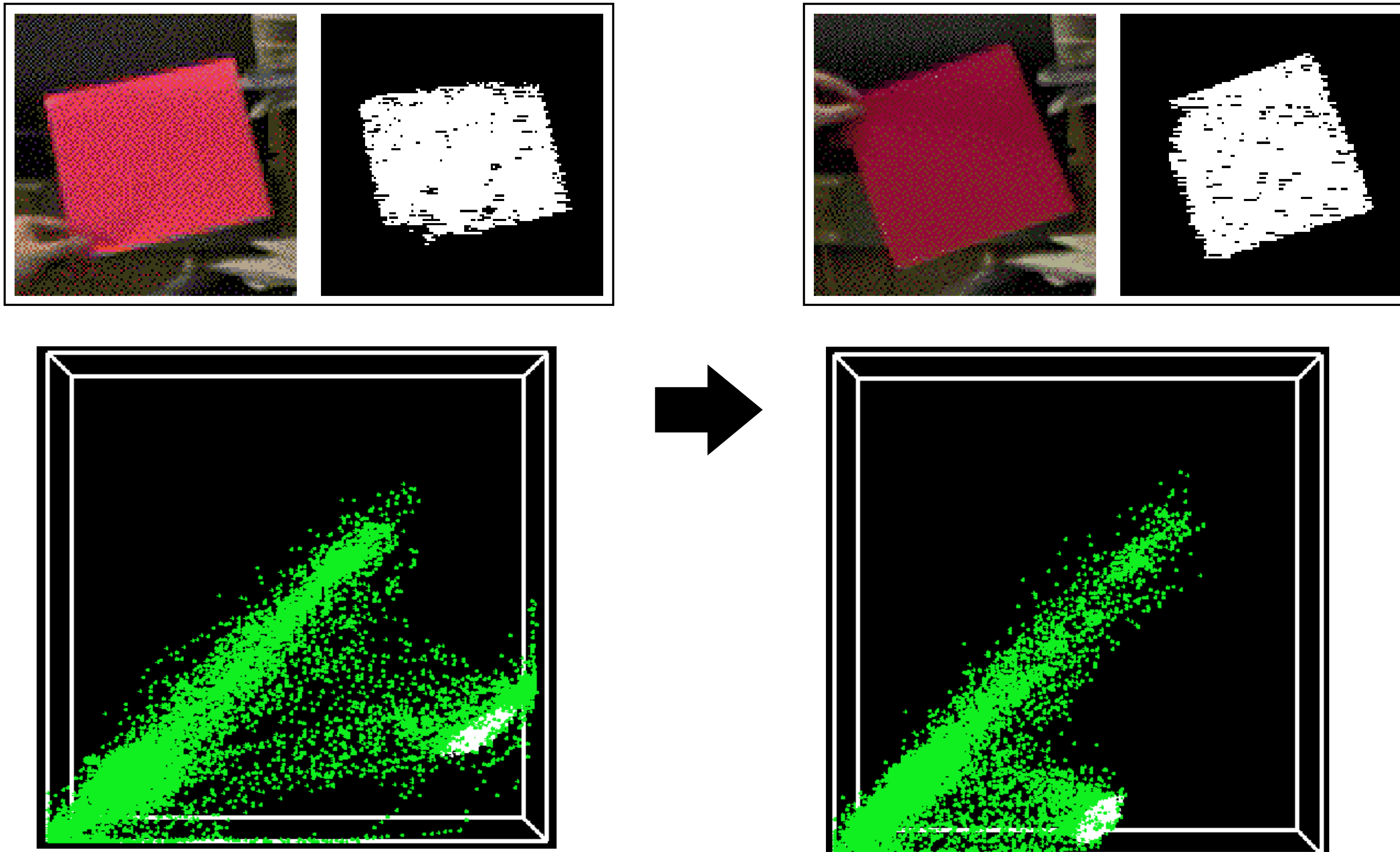
A Closer Look

Color: A Second Example

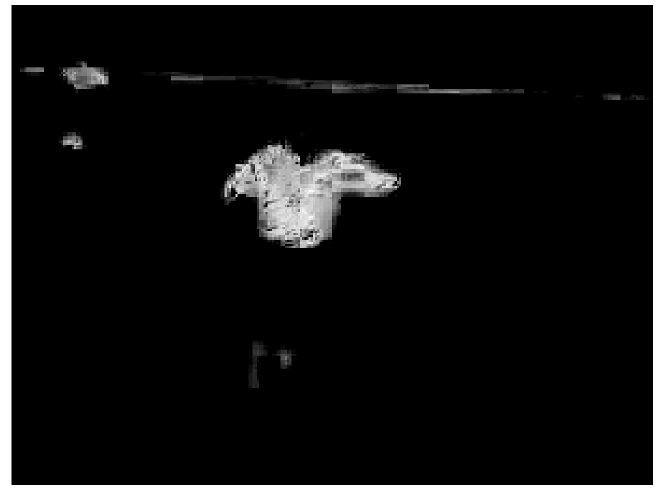
- Color representation
  - DRM [Klinker et al., 1990]: if P is Lambertian, has *matte* line and *highlight* line
  - User selects *matte* pixels in *R*
  - Compute first and second order statistics of cluster
  - Decompose ellipsoid $(\mathbf{S}, \mathbf{R}^T, \mathbf{T})$ of variance of matte cluster
  - Color similarity $\gamma(\mathbf{I}(x, y))$ is defined by Mahalanobis distance

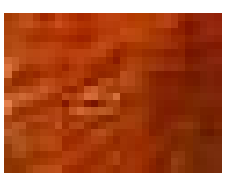
$$|\mathbf{S}^{-1}\mathbf{R}^T(\mathbf{I}(x, y) - \mathbf{T})|$$

Homogeneous Color Region: Photometry

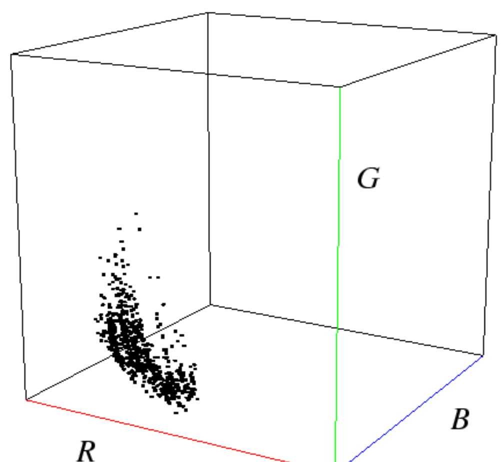


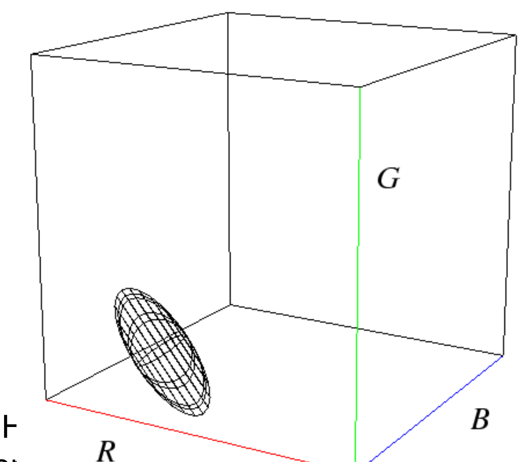
Homogeneous Region: Photometry





Sample





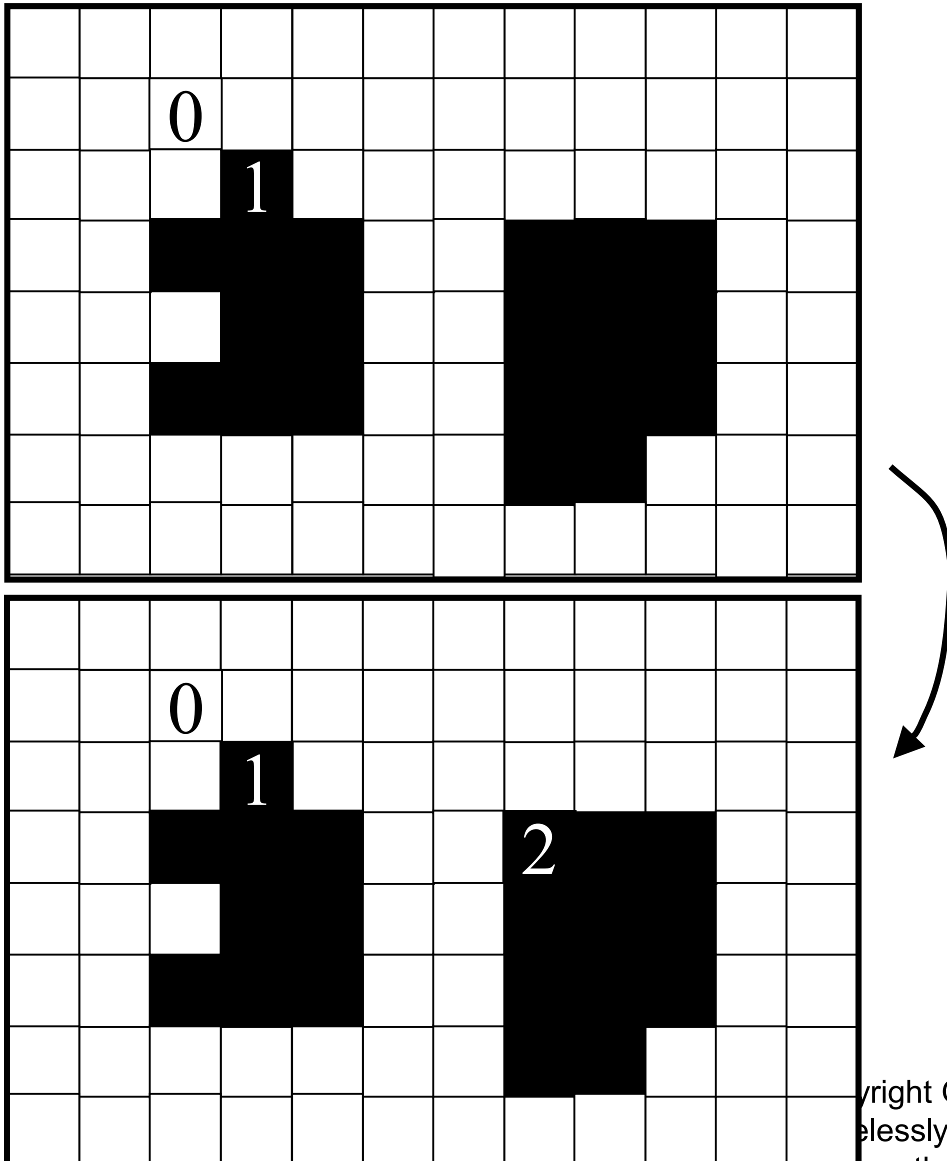
PCA-fitted ellipsoid

Binary Image Processing

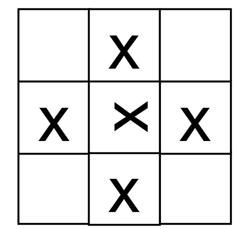
After thresholding an image, we want to know something about the regions found ...

- [x] How many objects are in the image?
- [ ] Where are the distinct “object” components?
- [ ] “Cleaning up” a binary image?

Connected Component Labeling



Goal: Label contiguous areas of a segmented image with unique labels



4 neighbors vs. 8 neighbors

Morphological Operators summarized

Let S_t be the *translation* of a set of pixels S by t.

$$S_t = \{ x + t \mid x \in S \}$$

The *dilation* of a binary image A by a mask S is then

$$A \oplus S = \bigcup_{b \in S} A_b$$

The *erosion* of a binary image A by a mask S is

$$A \ominus S = \{ x \mid x + b \in A, \forall\, b \in S \}$$

The *closing* of a binary image A by S is

$$A \bullet S = (A \oplus S) \ominus S$$

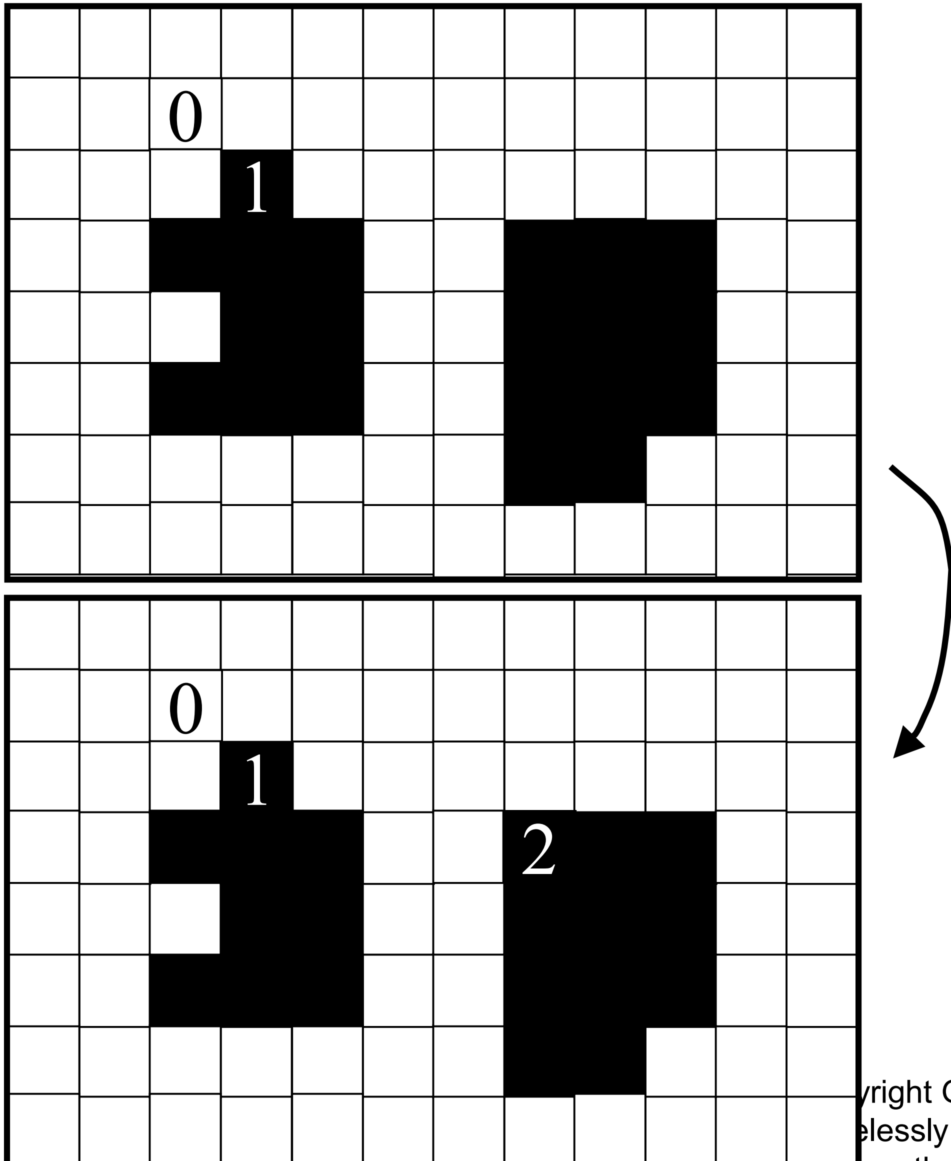
The *opening* of a binary image A by S is

$$A \circ S = (A \ominus S) \oplus S$$

The “Poor Man’s” Closing

- Note that median (or more generally any order statistic) filtering is one way of achieving similar effects. On binary images this can be also implemented using the averaging filter

Connected Component Labeling



Algorithm

1. Image is A. Let A = -A;

2. Start in upper left and work L to R, Top to Bottom, looking for an unprocessed (-1) pixel.

3. When one is found, change its label to the next unused integer. Relabel all of that pixel’s unprocessed neighbors and their neighbors recursively.

4. When there are no more unprocessed neighbors, resume searching at step 2 -- but do so where you left off the last time.

Limitations of Thresholding

- A uniform threshold may not apply across the image
- It measures the uniformity of regions (in some sense), but doesn’t examine the inter-relationship between regions.
- Local “disturbances” can break up nominally consistent regions

More General Segmentation

- Region Growing:
  - Tile the image
  - Start a region with a seed tile
  - Merge similar neighboring tiles in the region body
  - When threshold exceeded, start a new region

- Region Splitting
  - Start with one large region
  - Recursively
    - Choose the region with highest dissimilarity
    - If sufficiently similar, stop, otherwise split
    - repeat until no more splitting occurs

Greedy Segmentation: Bottom Up (Grouping)

- 1. Divide the image into the smallest region of interest
- 2. Compute the distance from each region to its neighbors and create a sorted list of distance/neighbor pairs: (measure/region pairs)
  $L = (d_1,p_1), (d_2,p_2), ... (d_N,p_N)$
- 3. While the smallest distance $d_{min} < \tau$ (smallest measure $< \tau$)
  a. merge the region pair (r_a,r_b) associated with d_{min} creating a new region r'
  b. remove all pairs containing r_a or r_b from the sorted list
  c. compute the distance between r' and its neighbors and add these new pairs to the list (add new measure to the list)

Example metrics:
mean gray value
gray value variance
color distance
edge direction

Note that we can replace "region" with "feature" and perform feature grouping with the same algorithm

Segmentation: Top Down (Partitioning)

- Start with one large region and compute homogeneity m. Call it R* and add it to a list L
- While m(first(L)) > τ
  - 1. consider splits of R* into R_1 and R_2 and choose that which minimizes $m(R_1) + m(R_2)$ (note this is effectively a distance!).
  - 2. remove R* and put R_1 and R_2 onto the list in decreasing order

- Note that the resulting segmentation is not guaranteed to be optimal or even connected. It often makes sense to first do a top down segmentation, followed by a bottom-up merge.

An Example: Image Segmentation

- The goal: to choose regions of the image that have similar "statistics."

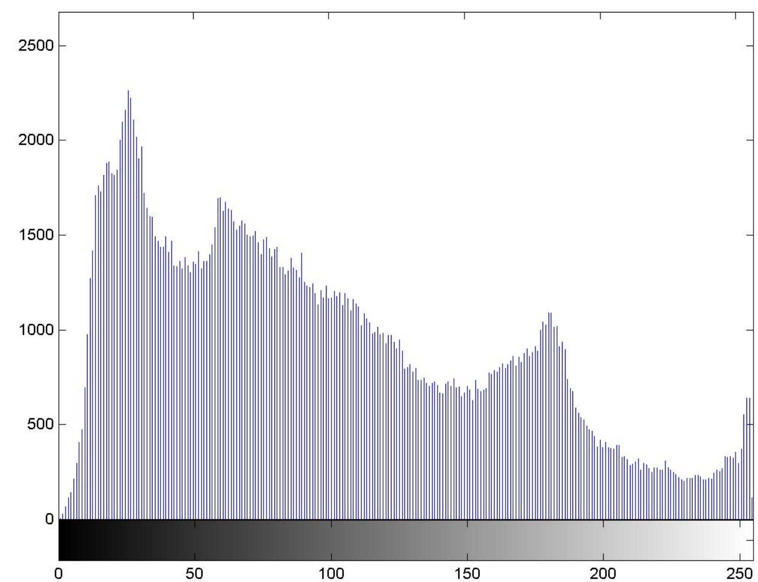
- Possible statistics:
  - mean
  - variance
  - co-occurance matrix

- Recall:
  - a histogram is a representation of the distribution of values in a range of values

- Idea: merge regions with similar histograms

An Image Histogram

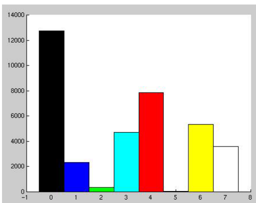
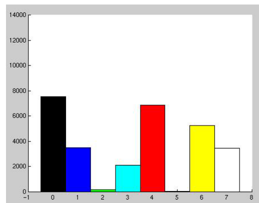
- For each pixel p_i, let h(p_i) = h(p_1) + 1



• EMD

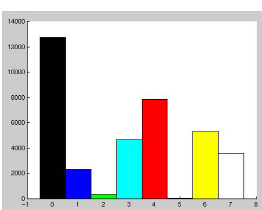
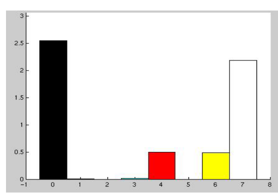
Histogram Metrics

Intersection

 $\cap$  $=$

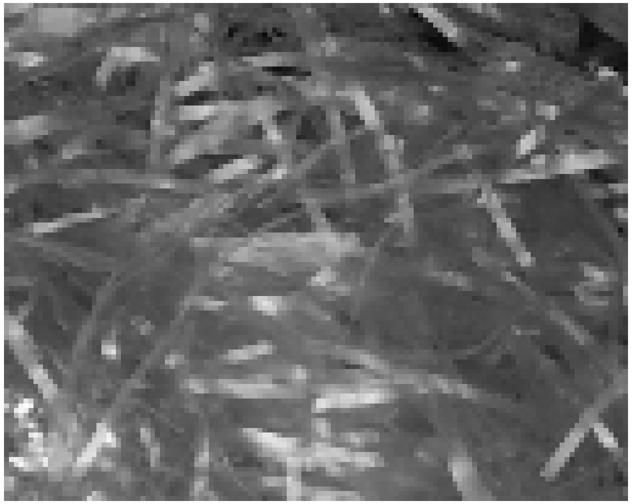
$$\frac{\sum_i \min(M_i\ ,\ T_i)}{\sum_i\ T_i}$$

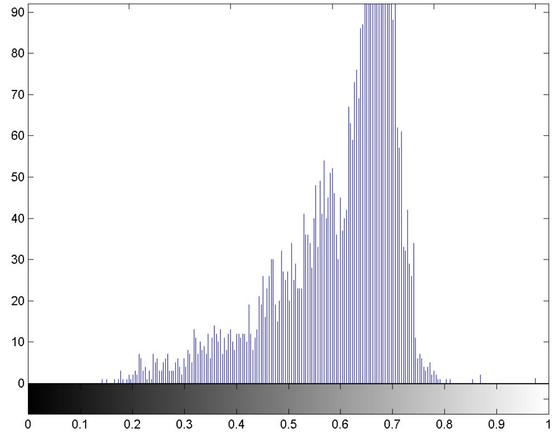
Inner Product

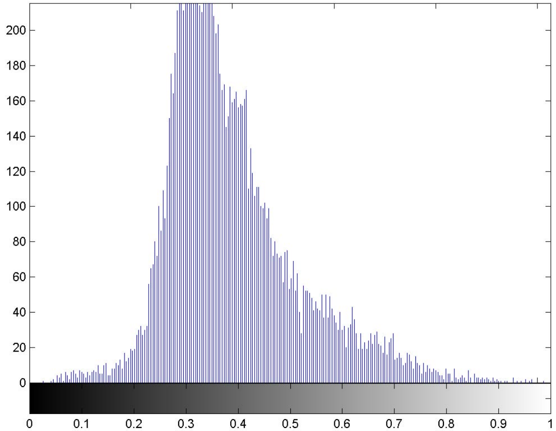
 $\cap$  $=$

$$M \cdot T = \frac{\sum M_i T_i}{\sqrt{\sum M_i^2}\sqrt{\sum T_i^2}}$$

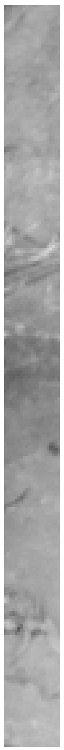
Comparing Histograms

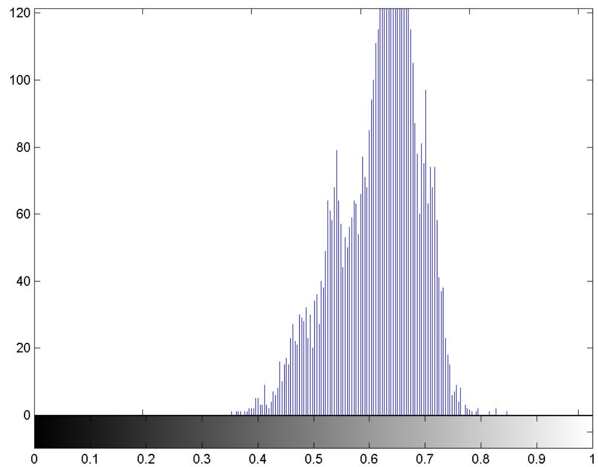


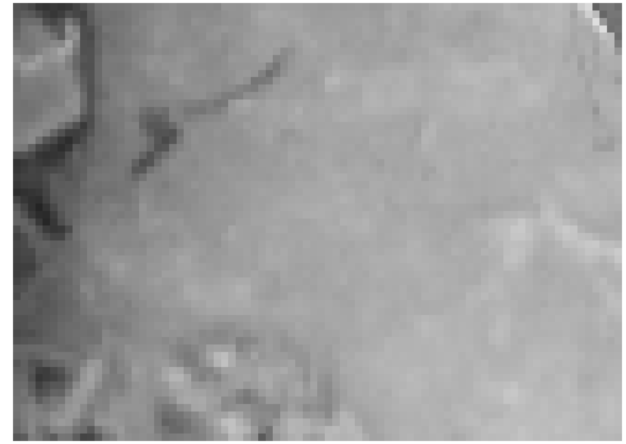




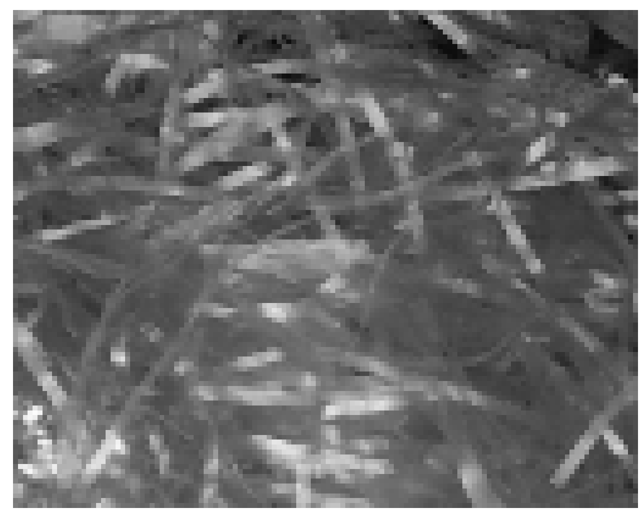
Which is More Similar?





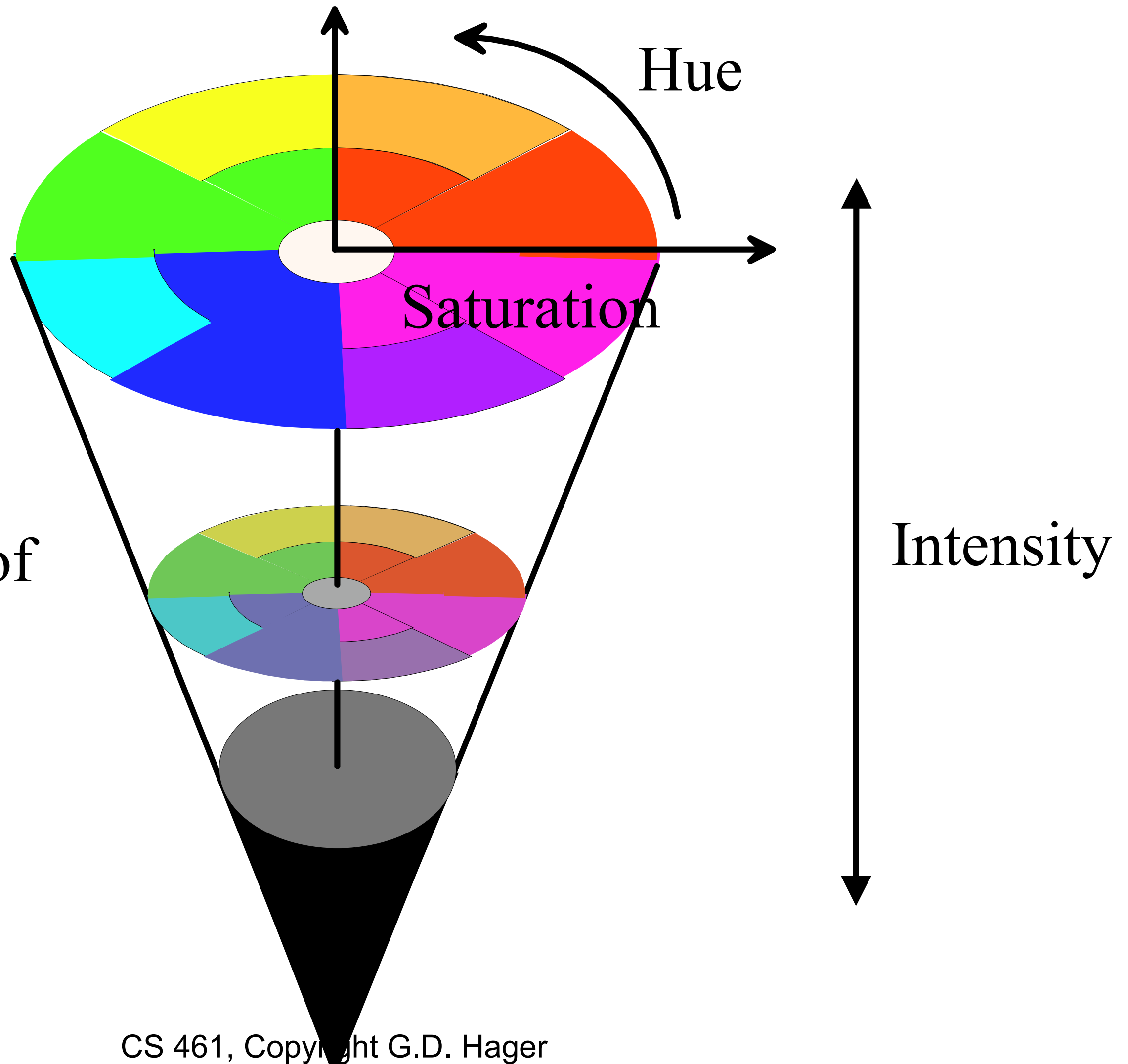


.906



.266

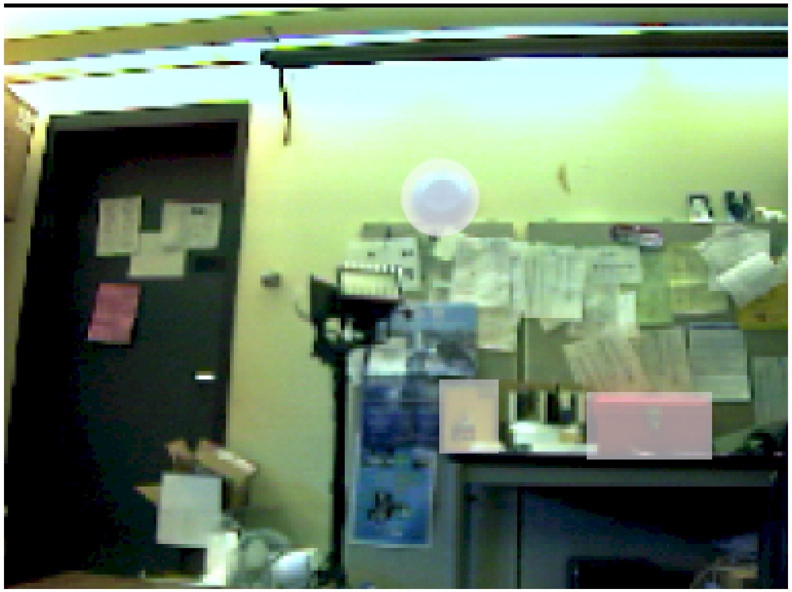
Dividing Up Color Space

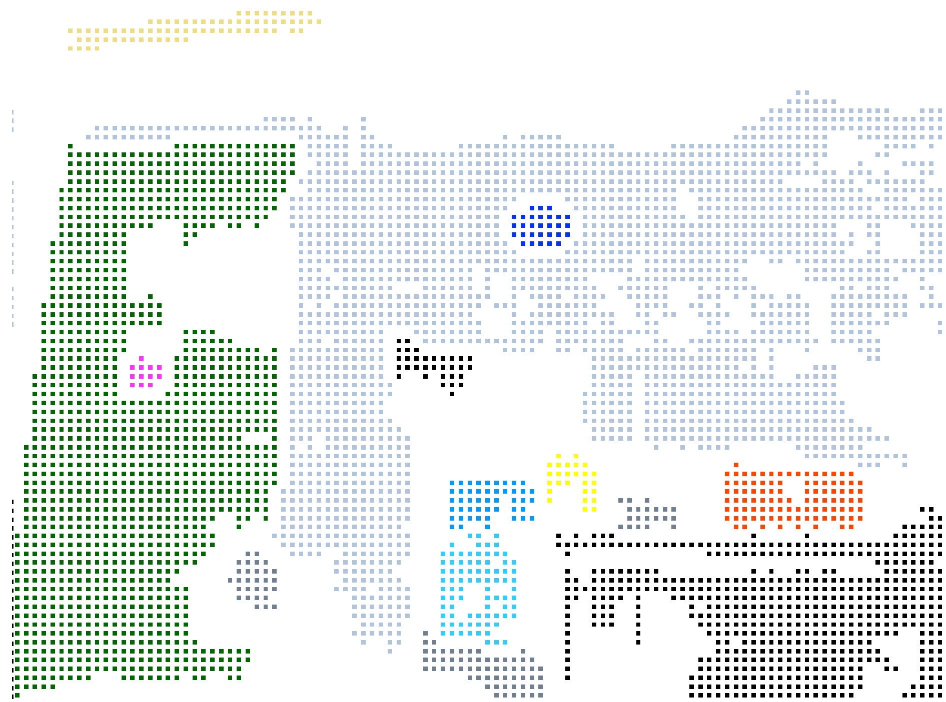


- HSI representation of color space

- Variable resolution gridding of space

Results of a Merge Segmentation

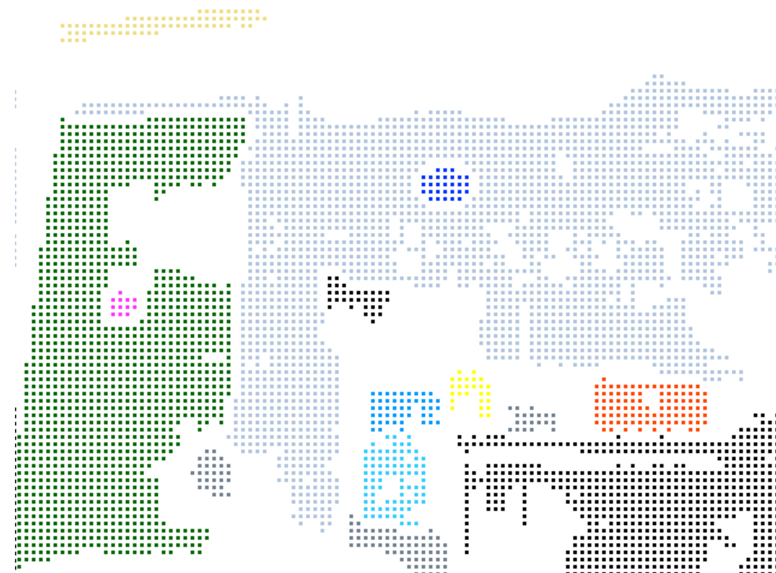




Characterizing Regions

- Distinctiveness
  - distractors (image distance to nearest similar region) 3.0
  - uniqueness (whole-image correlation) 1.0
- Scale changes
  - size weighting 0.5
- Overlapping objects
  - circularity (ratio of the area to the perimeter^2) 0.8
- Lighting effects
  - frame correlation 1.0
  - average saturation 2.5

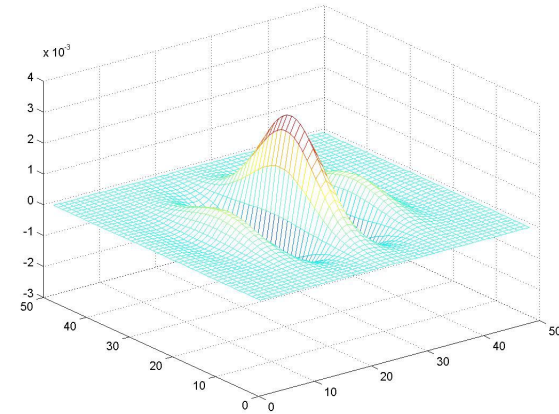
Top Three Features



More Examples

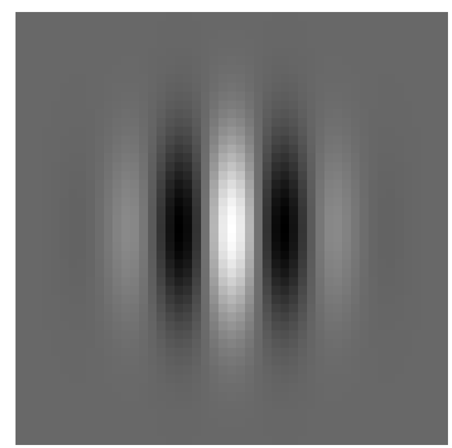
Even More Dimensions

- Are gray value histograms all we need?
  - not brightness invariant (what is?)
  - don't capture orientation
  - don't capture other pattern regularity

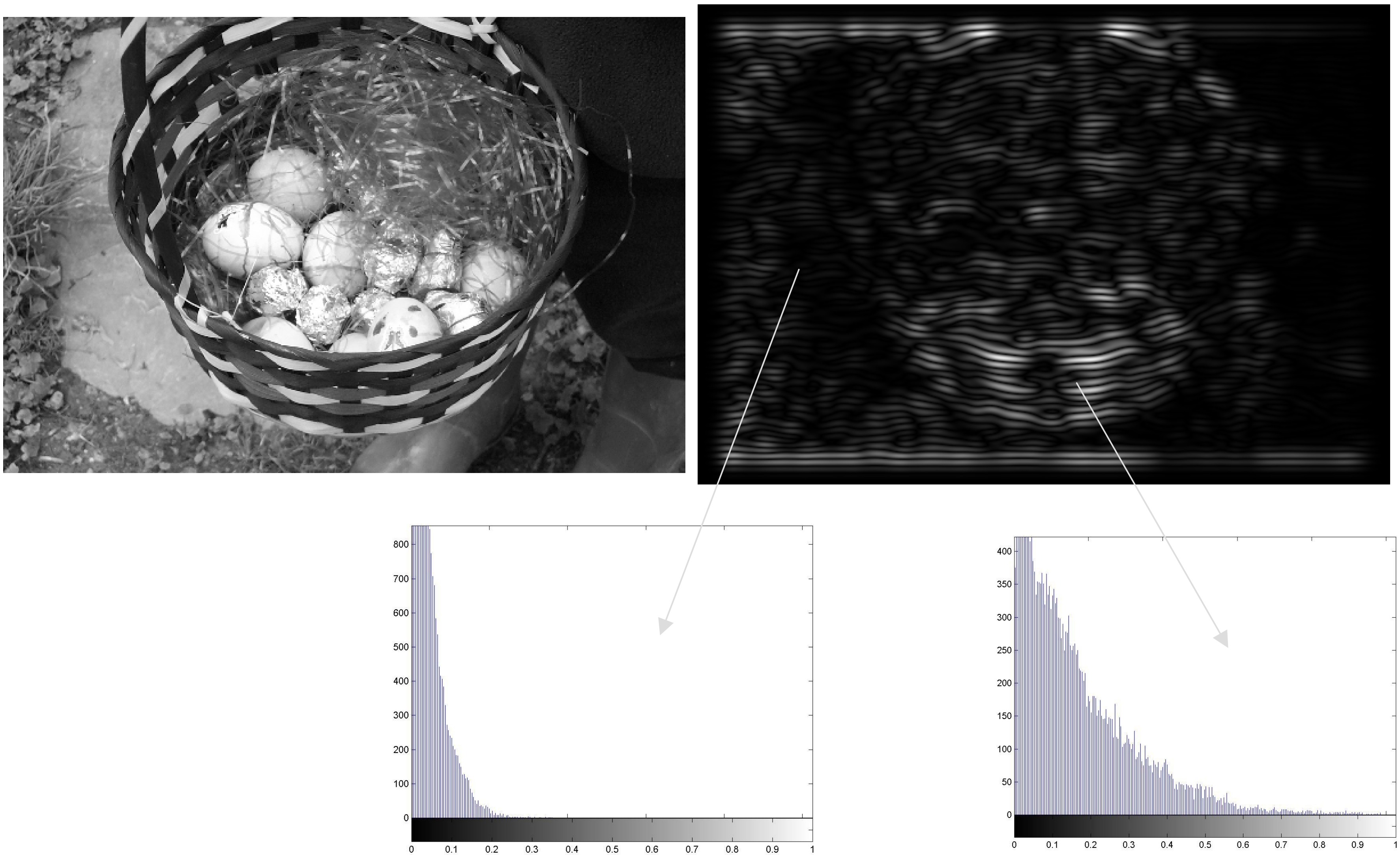


- How about computing something "richer"
  - use derivative measures
  - oriented
  - multiple scales

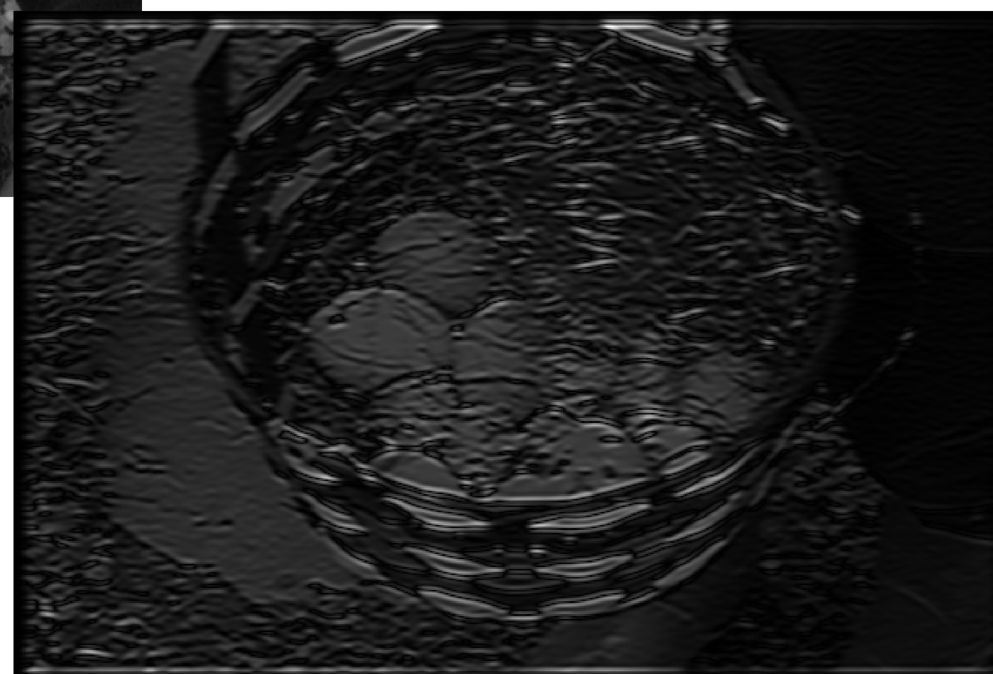
$$\cos(ax + by)\exp(-(x^2 + y^2)/2\sigma^2)$$



Gabor Response



Gabor Response



K-Means

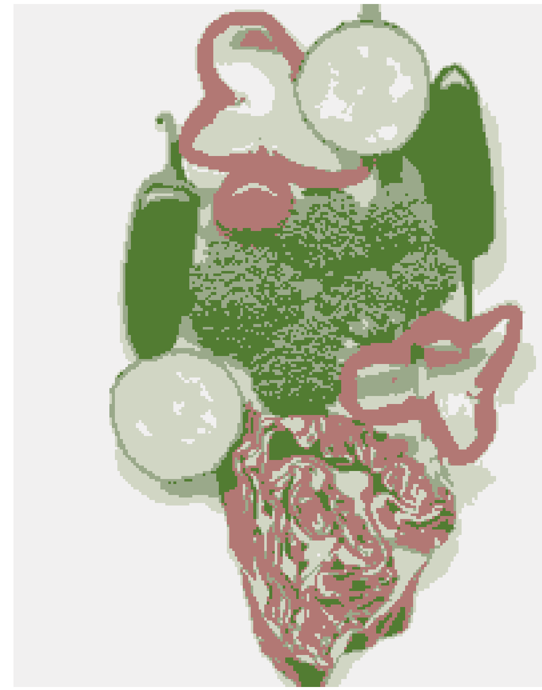
- Choose a fixed number of clusters
- Choose cluster centers and point-cluster allocations to minimize error
- can't do this by search, because there are too many possible allocations.
- Algorithm
  - fix cluster centers; allocate points to closest cluster
  - fix allocation; compute best cluster centers
- x could be any set of features for which we can compute a distance (careful about scaling)

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

Image

Clusters on intensity

Clusters on color



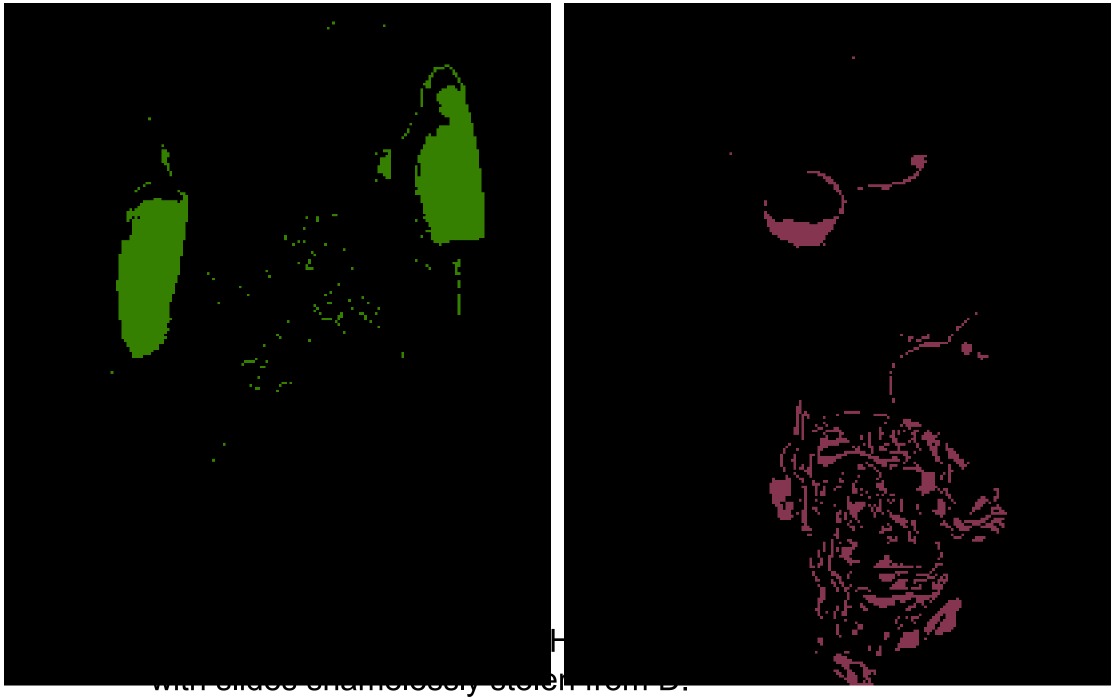
K-means clustering using intensity alone and color alone

Image

Clusters on color

K-means using color alone, 11 segments

K-means using
color alone,
11 segments.



K-means using colour and position, 20 segments

More Elaborate Approaches

Intensity

$$aff(x,y) = \exp\left\{ -\left(\frac{1}{2\sigma_i^2} \right) \left(\left\| I(x) - I(y) \right\|^2 \right) \right\}$$

Distance

$$aff(x,y) = \exp\left\{ -\left(\frac{1}{2\sigma_d^2} \right) \left(\left\| x - y \right\|^2 \right) \right\}$$

Texture

$$aff(x,y) = \exp\left\{ -\left(\frac{1}{2\sigma_t^2} \right) \left(\left\| c(x) - c(y) \right\|^2 \right) \right\}$$

Note c could be any histogram of image-derived values

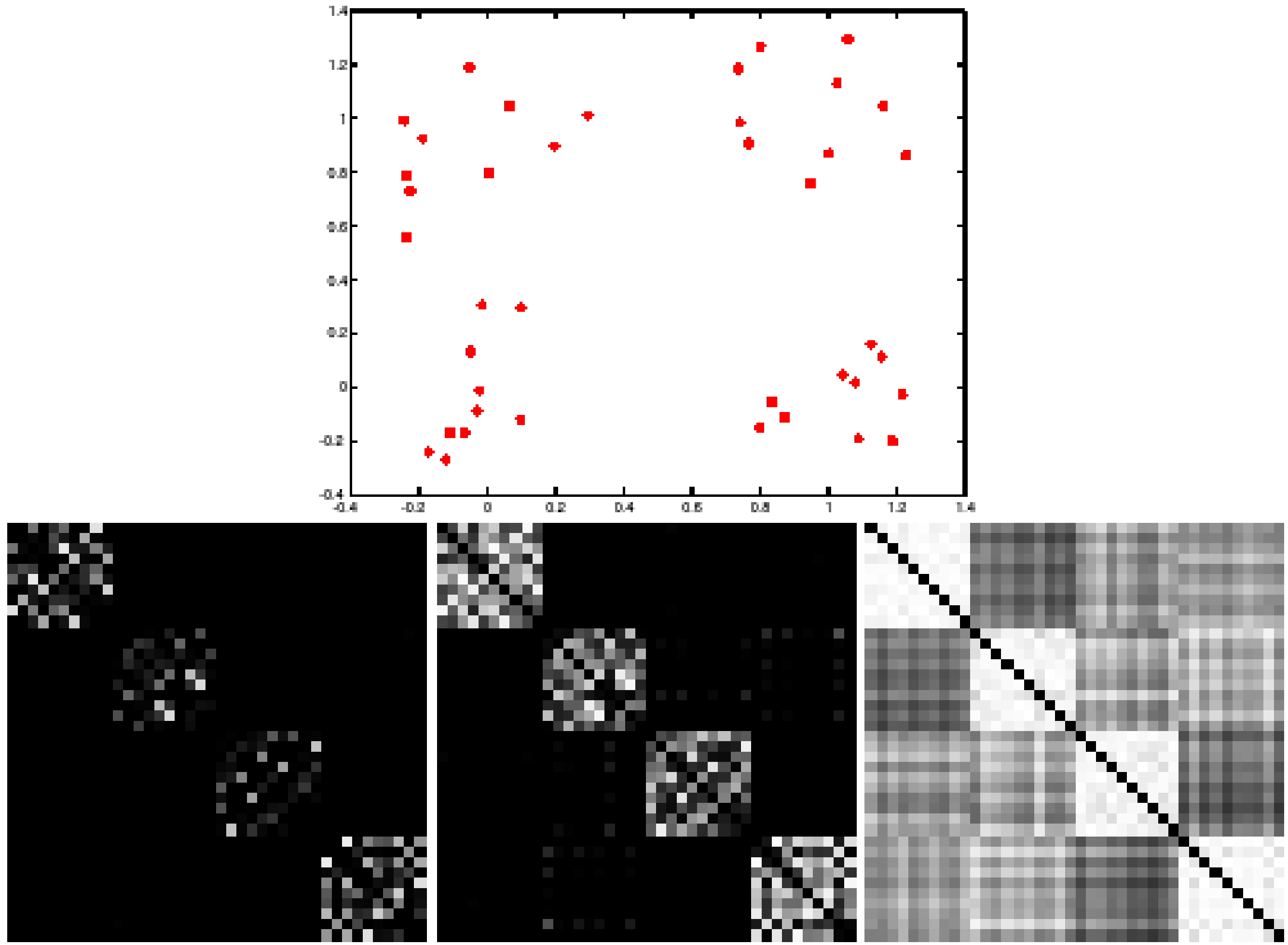
Combine by product to "and"
Combine by max for "or"

More Elaborate Approaches

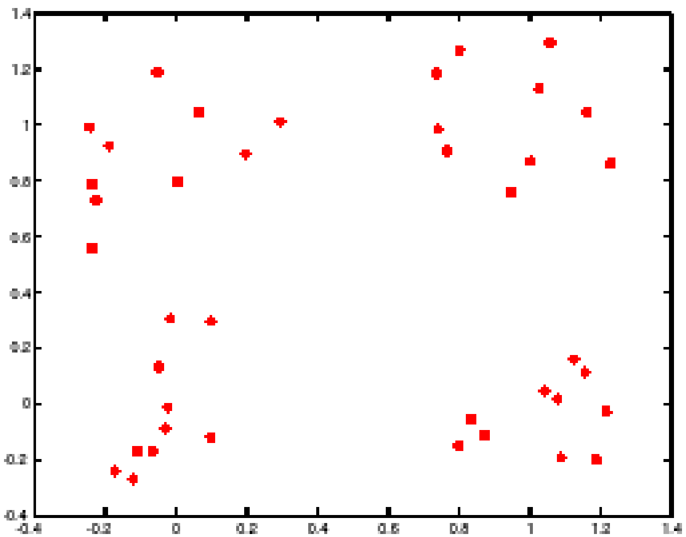
- Let A be a matrix of affinity between i's and j's
- Let w_n be a vector of weights of each element to the n'th cluster
- Make sure w_n are unit vectors
- maximize w_n^t A w_n
- Solution is by finding eigenvector with maximum eigenvalue

- More generally, if we can arrange A to be block diagonal (which spatially localized things generally are, then we can extract more eigenvectors
  - zero elements mean not part of the cluster
  - threshold to get elements that belong

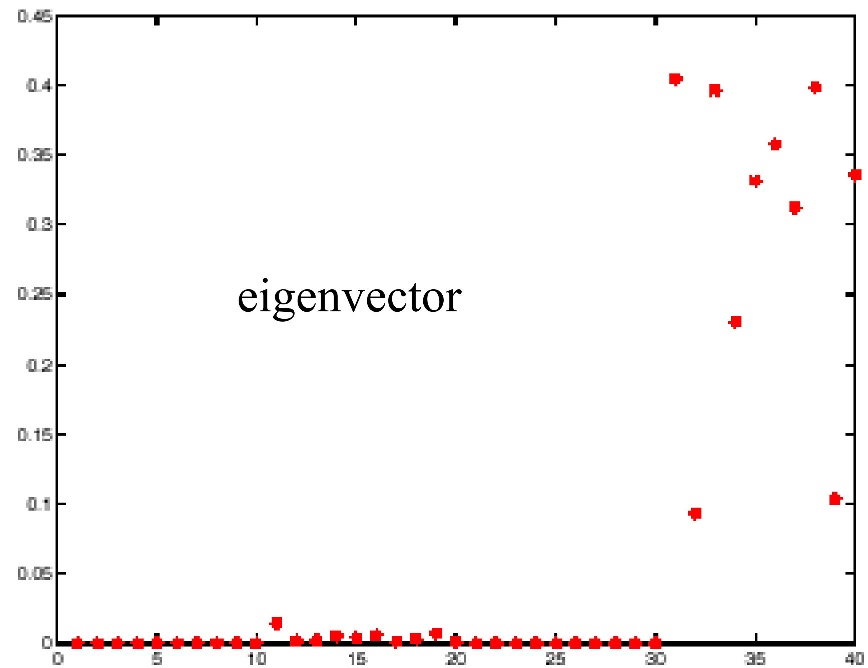
Scale affects affinity



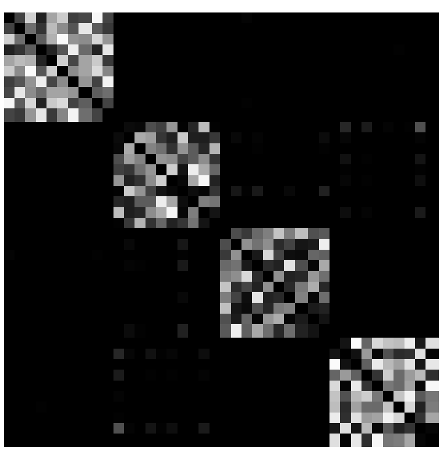
Example eigenvector



points



eigenvector



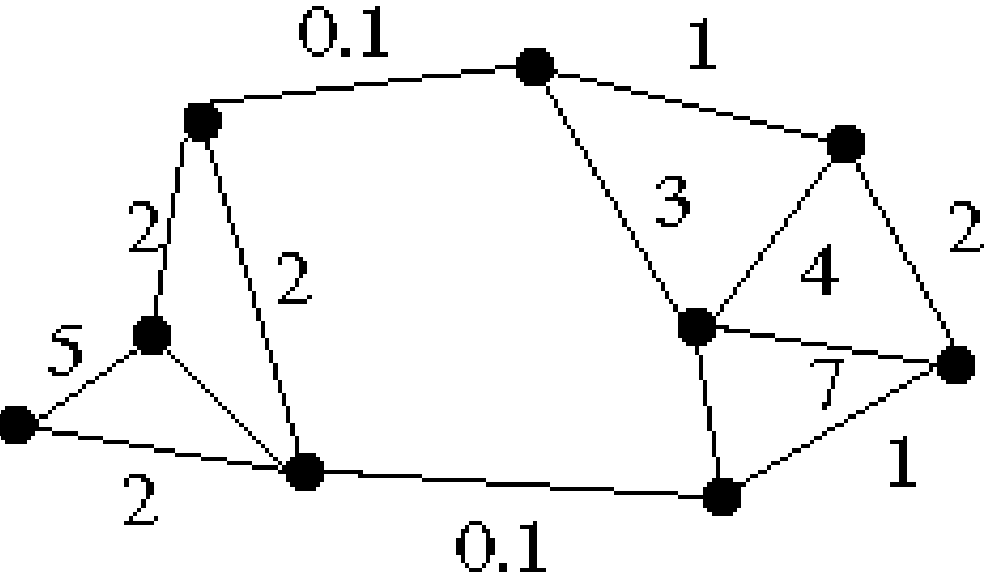
matrix

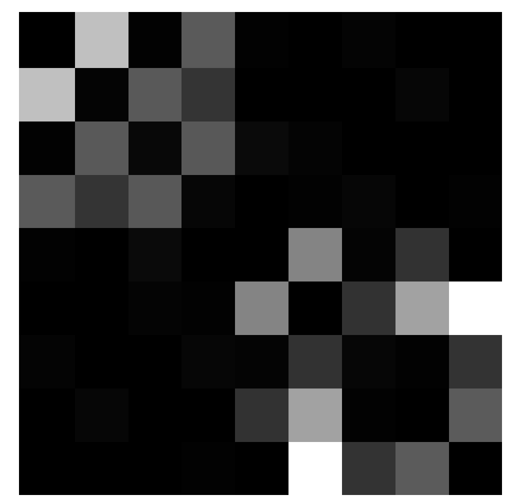
Graph Cuts

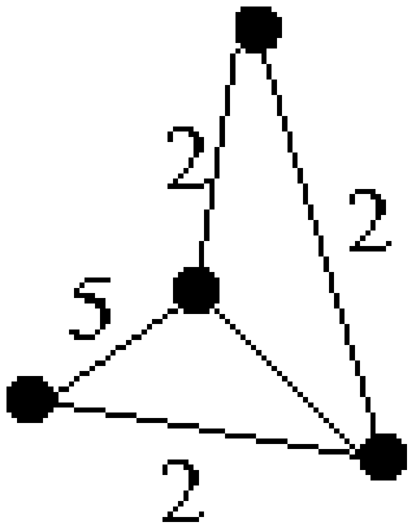
- Problem is that eigenvectors are unique only up to a linear transformation

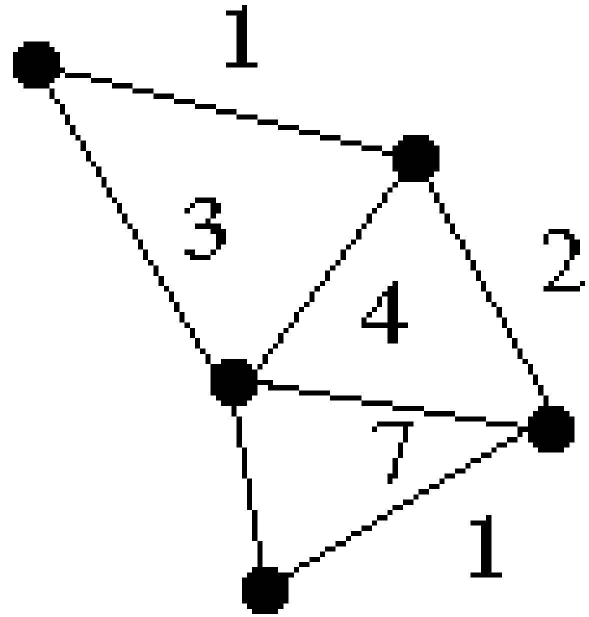
- Think of A as connectivity matrix of a graph
- Goal is to choose a cut of the graph that removes "weak" links and preserves strong ones.
  - cut(A,B)/assoc(A,V) + cut(A,b)/assoc(B,V)
  - minimize this value

- Let $D_{i,i} = \sum_j A_{i,j}$
- Let y be n vector of 1 or –b (in or out)
- Minimize $(y^t(D - A)\, y)/y^tDy$
- The exact problem is integer programming --- combinatorial optimization!


0.1
1
2
2
3
4
2
5
7
1
2
0.1




2
2
5
2


1
3
4
2
7
1

Normalized cuts

- Current criterion evaluates within cluster similarity, but not across cluster difference
- Instead, we'd like to maximize the within cluster similarity compared to the across cluster difference
- Write graph as V, one cluster as A and the other as B

- Maximize

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

- i.e. construct A, B such that their within cluster similarity is high compared to their association with the rest of the graph

Normalized cuts

- Write a vector y whose elements are 1 if item is in A, -b if it's in B
- Write the matrix of the graph as W, and the matrix which has the row sums of W on its diagonal as D, 1 is the vector with all ones.
- Criterion becomes

$$\min_{\mathrm{y}}\left(\frac{y^T(D-W)y}{y^T D y}\right)$$

- and we have a constraint

$$y^T D 1 = 0$$

- This is hard to do, because y's values are quantized

Normalized cuts

- 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 maximises the criterion --- i.e all components of y above that threshold go to one, all below go to -b

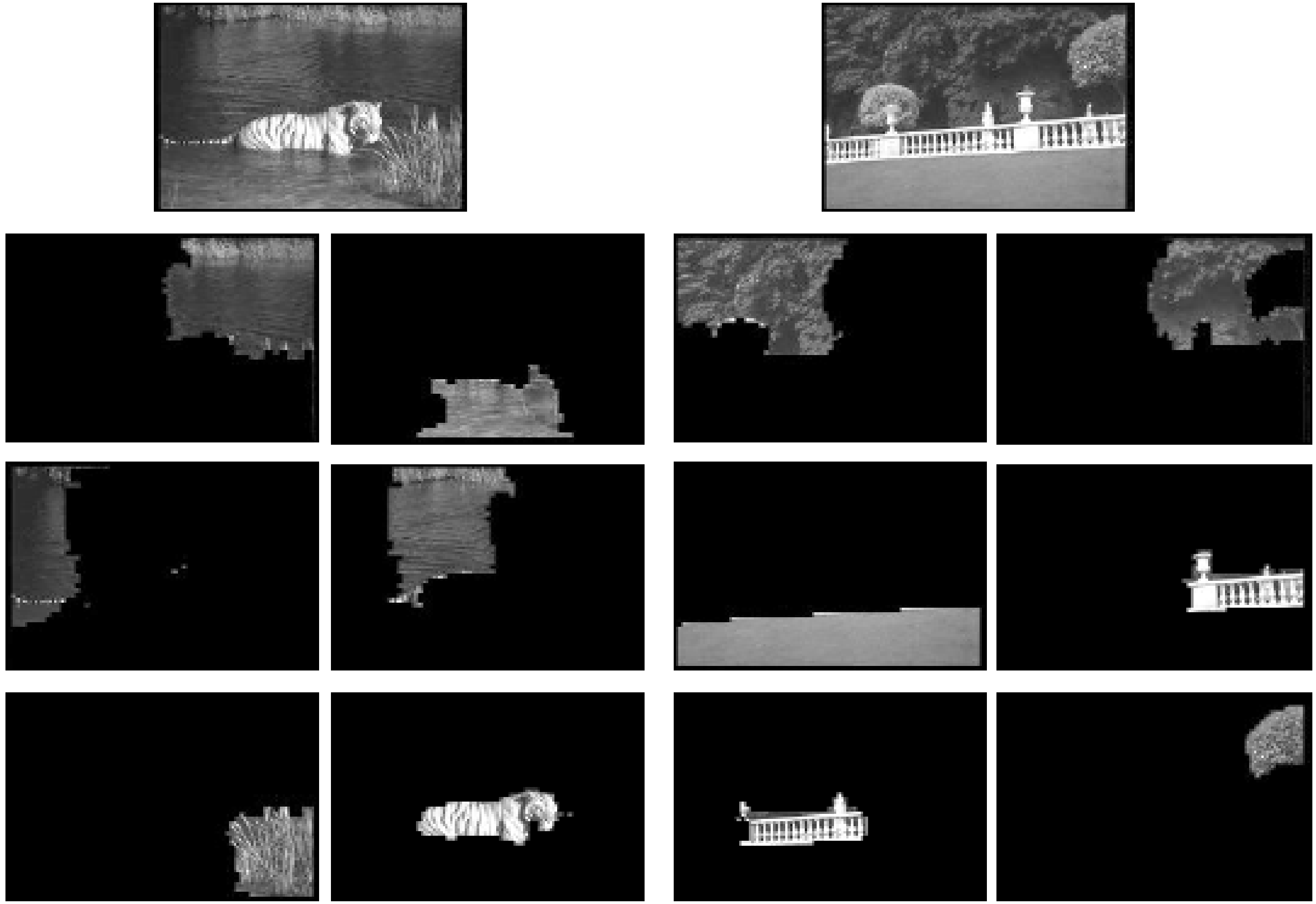


Figure from “Image and video segmentation: the normalised cut framework”, by Shi and Malik, copyright IEEE, 1998

F igure from “Normalized cuts and image segmentation,” Shi and Malik, copyright IEEE, 2000

An Example: BlobWorld

(Carson, Belongie, Greenspan, Malik)

The problem: query images (e.g. from the WEB) using image information

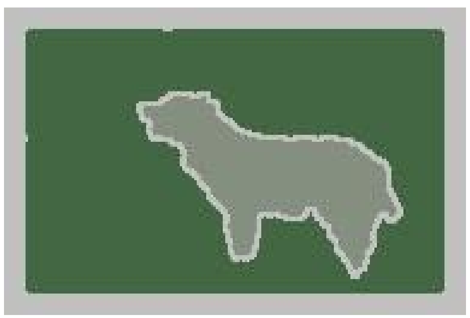
The solution: segment images into roughly uniform regions and search based on feature vectors

The features:

- color
- texture
- location (i.e. spatial compactness)

The segmentation algorithm: Expectation Maximization (similar in conception to K-mean clustering algorithm

Example Segmentations



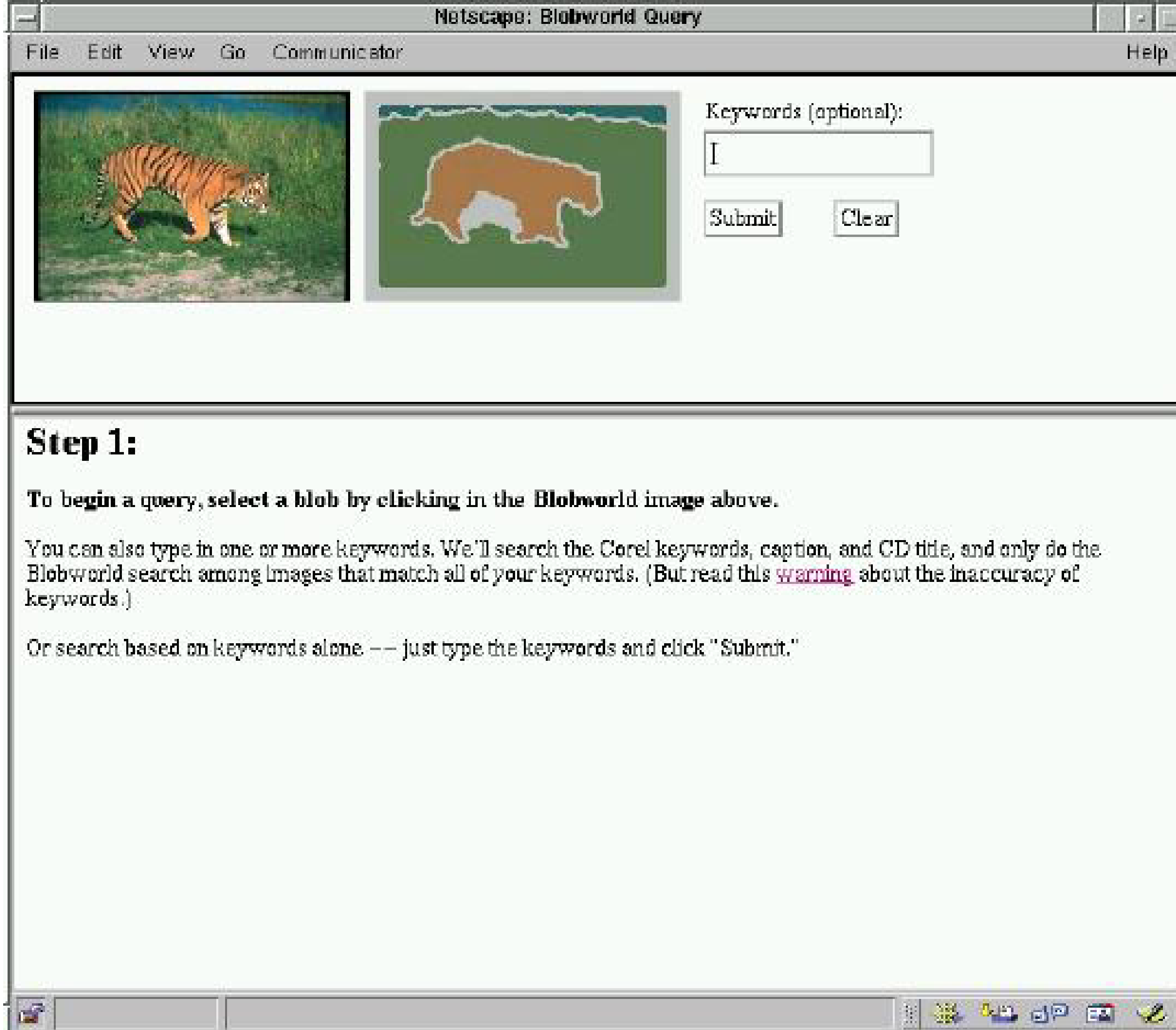
Querying

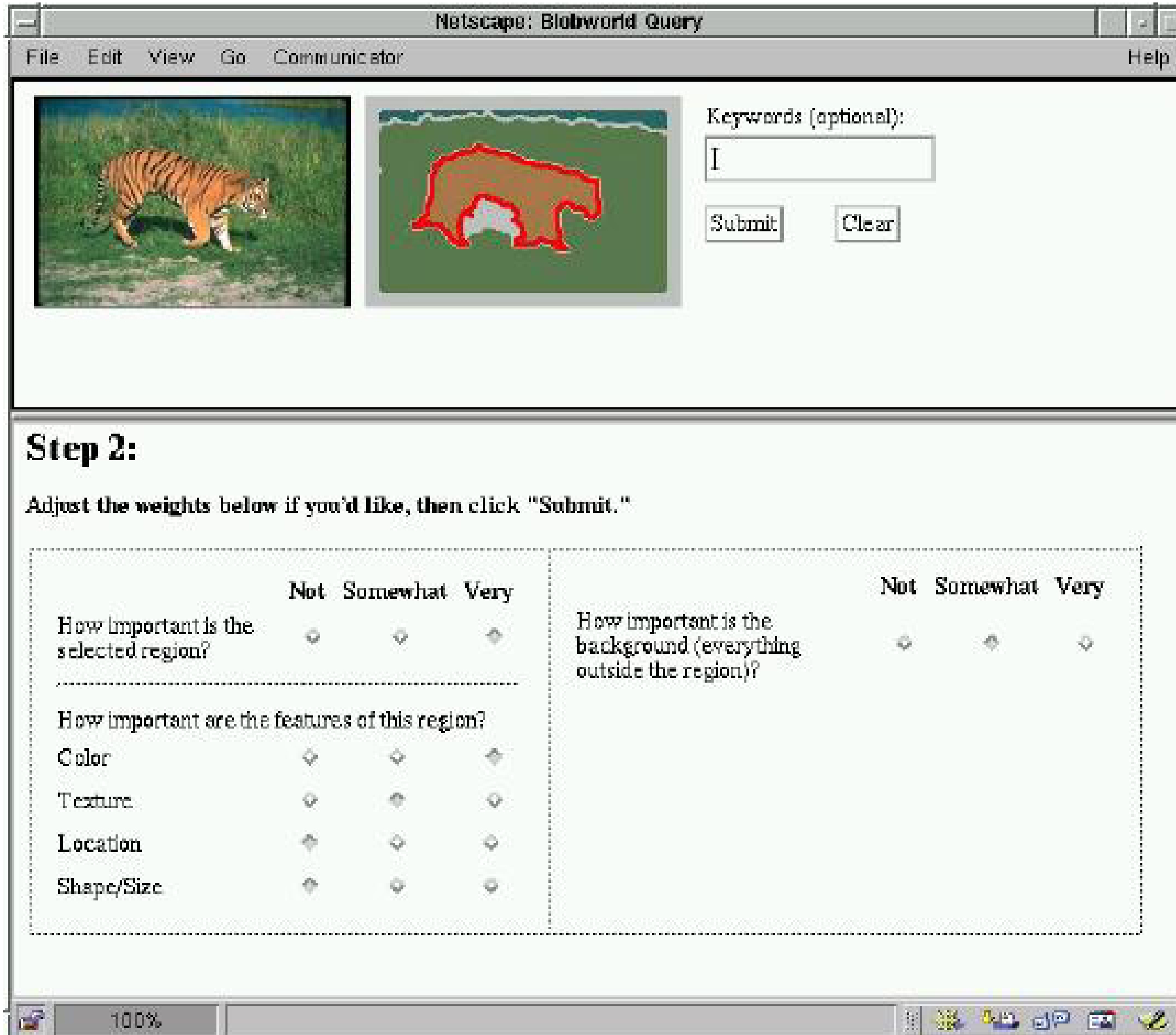
User selects a weighting of color vs. texture giving a diagonal weight matrix Σ

Given a blob with feature vector v_i, compared to another vector v_j using Mahalanobis distance:

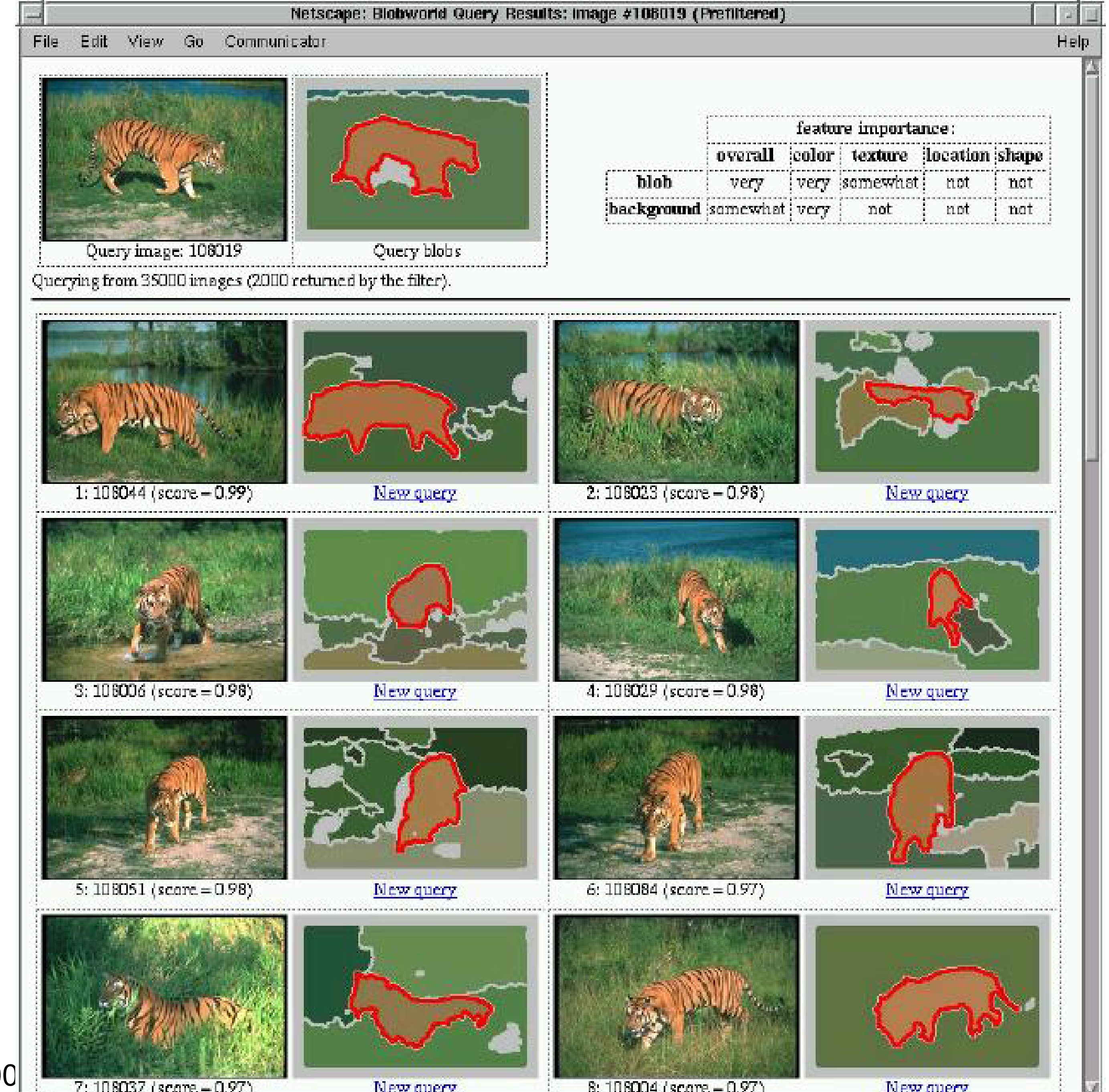
$$d_{i,j} = (v_i - v_j)^t \, \Sigma \, (v_i - v_j)$$

Compound queries using min and max for and and or


Netscape: Blobworld Query
File Edit View Go Communicator Help
Keywords (optional):
Submit
Clear
Step 1:
To begin a query, select a blob by clicking in the Blobworld image above.
You can also type in one or more keywords. We'll search the Corel keywords, caption, and CD title, and only do the Blobworld search among images that match all of your keywords. (But read this warning about the inaccuracy of keywords.)
Or search based on keywords alone -- just type the keywords and click "Submit."



Netscape: Blobworld Query
File Edit View Go Communicator Help
Keywords (optional):
Submit
Clear
Step 2:
Adjust the weights below if you'd like, then click "Submit."
Not Somewhat Very
How important is the selected region?
How important are the features of this region?
Color
Texture
Location
Shape/Size
Not Somewhat Very
How important is the background (everything outside the region)?
100%


Netscape: Blobworld Query Results: image #108019 (Prefiltered)
File Edit View Go Communicator Help
Query image: 108019
Query blobs
feature importance:
overall color texture location shape
blob very very somewhat not not
background somewhat very not not not
Querying from 35000 images (2000 returned by the filter).
1: 108044 (score = 0.99)
New query
2: 108023 (score = 0.98)
New query
3: 108006 (score = 0.98)
New query
4: 108029 (score = 0.98)
New query
5: 108051 (score = 0.98)
New query
6: 108084 (score = 0.97)
New query
7: 108037 (score = 0.97)
New query
8: 108004 (score = 0.97)
New query

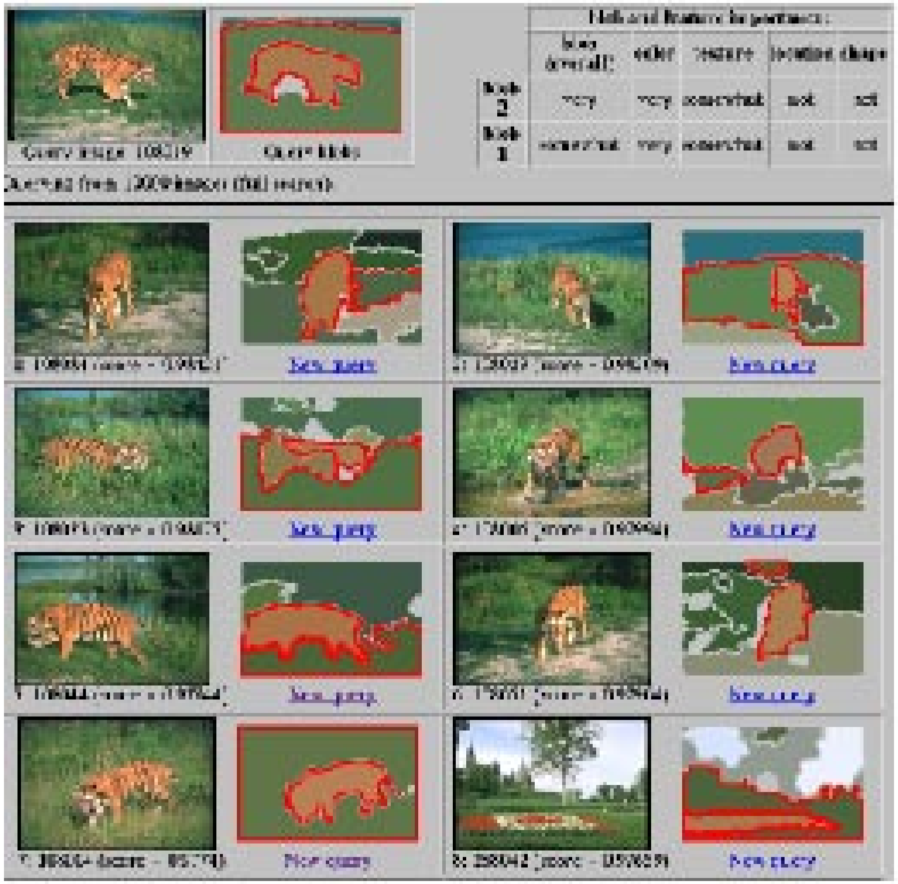


Figure 7. Blobworld query for tiger images using two blobs. The overall weights are 1.0 for the tiger blob and 0.5 for the grass blob. For both blobs, the color weight is 1.0 and the texture weight is 0.5.

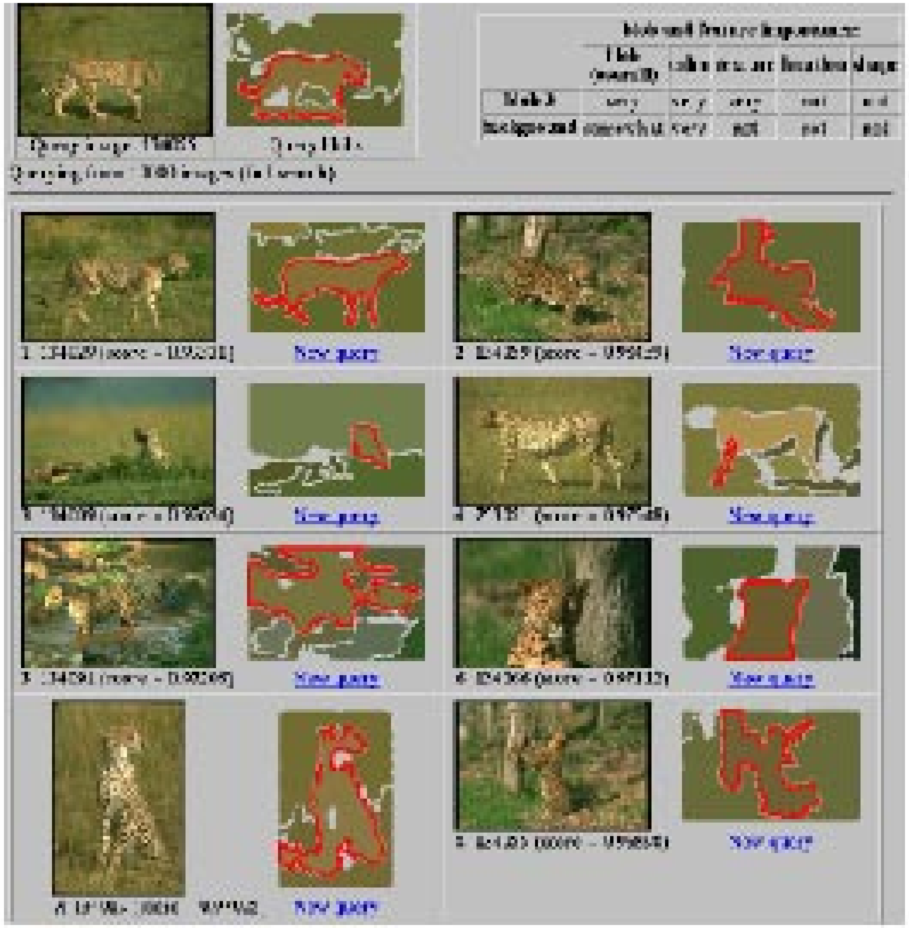


Figure 9. Blobworld query for cheetah images using one blob plus the background. The overall weights are 1.0 for the cheetah blob and 0.5 for the background. For the cheetah blob, both color and texture weights are 1.0. (Only color is used for the background score.)

Finale

- Blobworld is one example of an increasing trend: computer vision technology applied to real world problems
- In the long term, computer vision is on the same trend as graphics was 25 yrs ago, and speech in the last decade
- You will probably see the “seeing computer” in your lifetimes.