Computer Vision, Lecture 7

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

Filter Pyramids

- An Exercise:
  - Suppose I have $G(\sigma)$ and I perform $G(\sigma) * G(\sigma) * I$
    - Hint: think about the convolution theorem and the FFT

- Suppose I want to subsample images
  - subsampling reduces the highest frequencies
  - averaging reduces noise
  - Can I average and resample and reduce noise while not loosing desirable frequencies?

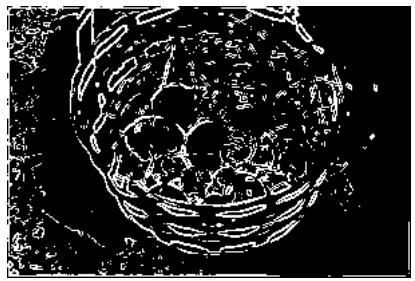
Gaussian Pyramid

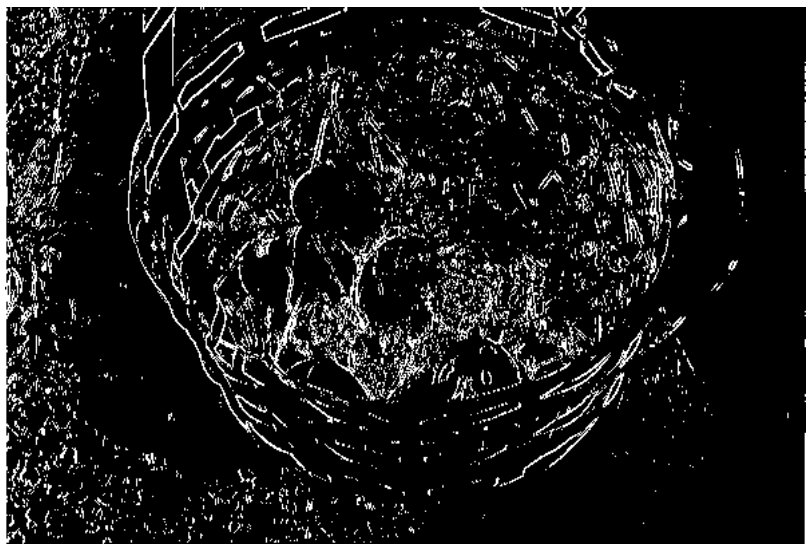
- Algorithm:
  - 1. Filter with G(σ)
  - 2. Resample at every other pixel
  - 3. Repeat
- A common use of this is the Laplacian Pyramid

Laplacian Pyramid Algorithm

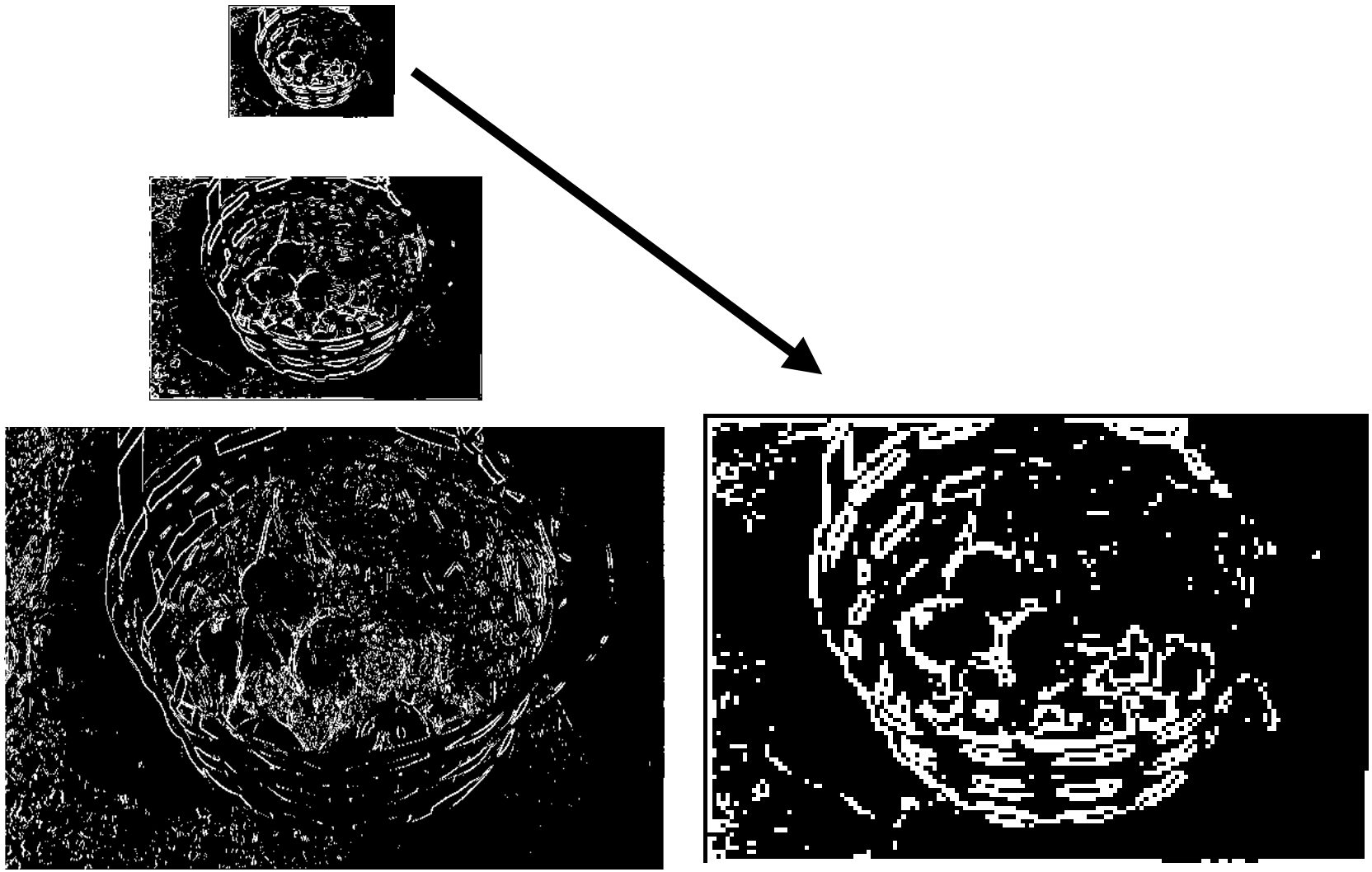
- Check F&S

Laplacian of Gaussian Pyramid





Gaussian Pyramid



Types of Edge Operators

1. Operators approximating derivatives using differences.
   - directional: Roberts, Prewitt, DoG, etc.
   - Rotationally invariant: Laplacian (sum of second derivatives)

2. Operators based on the zero crossing of the second derivative (e.g. Canny).

3. Operators that attempt to match a specific image profile.

From Pixels to Edges

- Various operators can be used to *enhance* rapid contrast changes

- Detecting these contrast changes involves *thresholding* to separate noise from signal

- *Edges* are a result of *grouping* pixels (sometimes called “edgels”) into groups forming continuous curves.

Definitions:

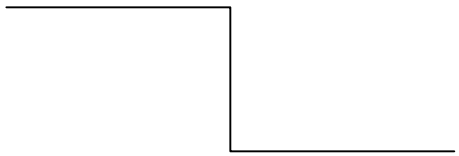
Edge normal: Unit vector in direction of maximum intensity variation

Edge direction: Perpendicular to edge normal

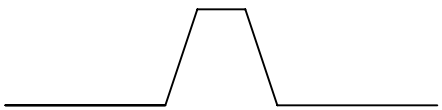
Edge position: Image position of pixels of edge

Edge strength: Change in contrast along normal

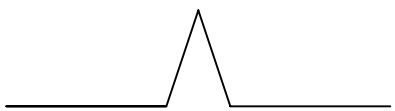
Edge Types



Step



Ridge



Roof

A Simple Edge Detection Algorithm

1. Compute image gradient $\nabla I(u,v) = [I_u(u,v)\ I_v(u,v)]'$
2. Threshold on gradient magnitude: $||\nabla I(u,v)|| > \tau$
3. Compute connected components
    1. maximal sets of pixels whose neighbors exceed threshold
4. Remove components smaller than a certain size
5. Return resulting points as “edge segments”

Problems?

Canny Edge Detector

- The Plan:
  - Formulate an optimization problem for detection on 1-D signals
  - Generalize to 2D signals
  - Apply thresholding with hysteresis
  - Apply this operator at various scales

- The Assumptions:
  - Edge enhancement is linear
  - The edge model is step edges with amplitude A
  - Noise is additive, white and Gaussian

Optimization Criteria

- **Good Detection:** ***Minimize the probability of false positives and false negatives***
  - **Maximize the SNR**

$$\frac{A}{n_0}\sum(f) = \left| \frac{A\int_{-W}^{0} f(t)dt}{n_0\sqrt{\int_{-W}^{W} f^2(t)dt}} \right|$$

- **Good Localization:** ***Edgels detected should lie as close as possible to the true edge***
  - **Maximize 1/distance to edge center which leads to maximizing LOC**

$$\frac{A}{n_0}\Lambda(f') = \left| \frac{A\,|\,f'(0)\,|}{n_0\sqrt{\int_{-W}^{W} f'^2(t)dt}} \right|$$

Optimization Cont'd

- Consider the maximizing the product of both criteria
  - result is itself a step filter
  - step filters are noise amplifying!

- Additional criterion: single response constraint:
  - detector should minimize the number of local maxima about and edge (recall what happens with step filter)
  - ***RESULT1: localization vs. detection***

$$\Sigma(f_w) = \sqrt{w}\Sigma(f) \;\; and \;\; \Lambda(f'_w) = \frac{1}{\sqrt{w}}\Lambda(f') \;\; where \;\; f_w(x) = f(x/w)$$

  - ***RESULT2: optimal detector is very close to the first derivative of a Gaussian.***

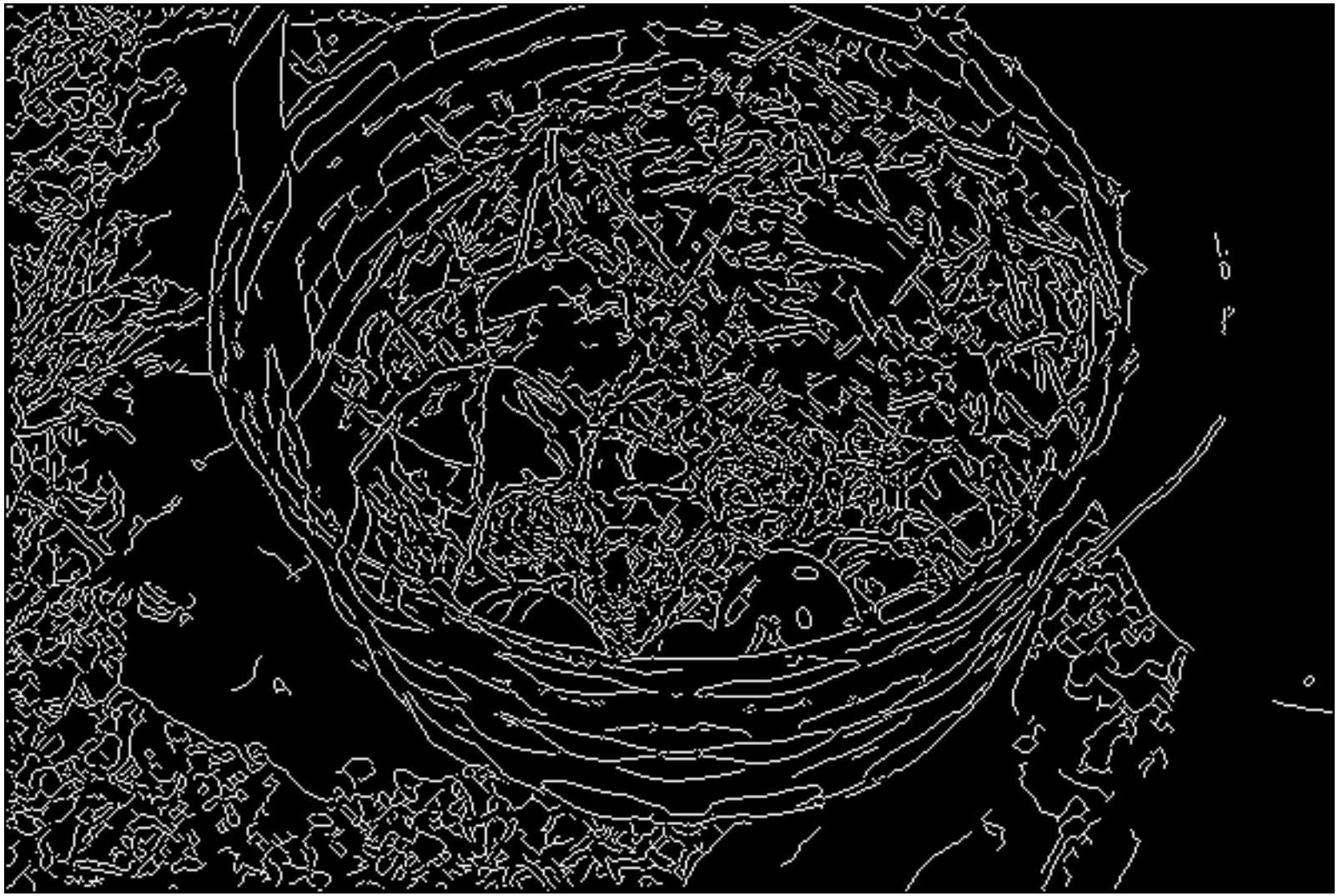
The Procedure

- Enhancement:
  - compute x and y derivatives using DoG's.
  - compute direction and magnitude of gradient (two images)

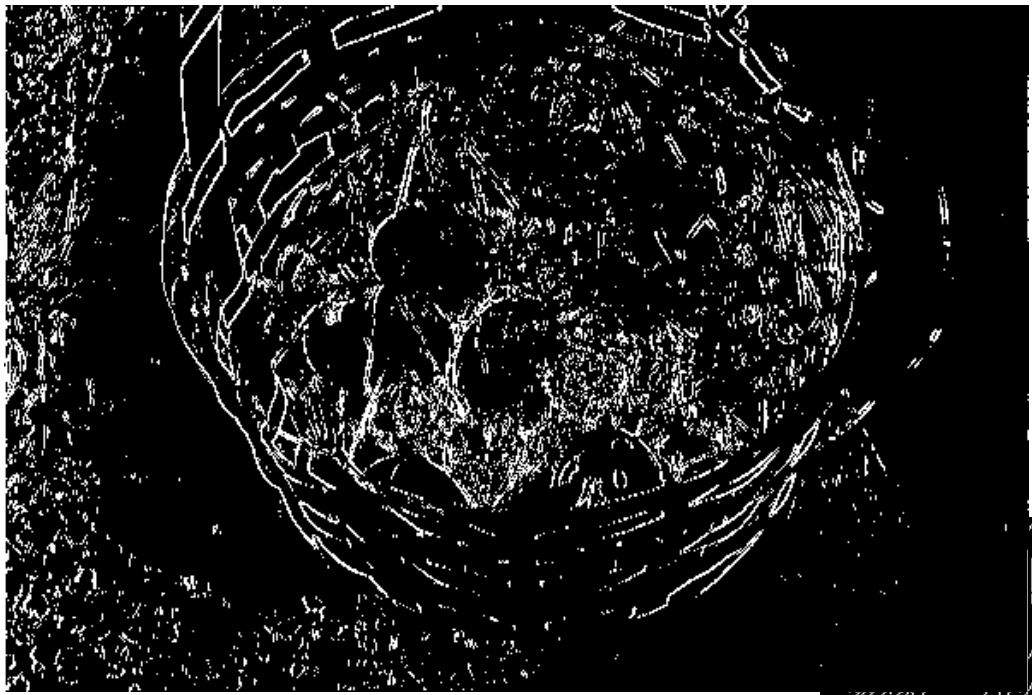
- Nonmaximal Suppression:
  - Sample along the gradient direction
  - If given pixel is smaller than neighbor, set it to zero

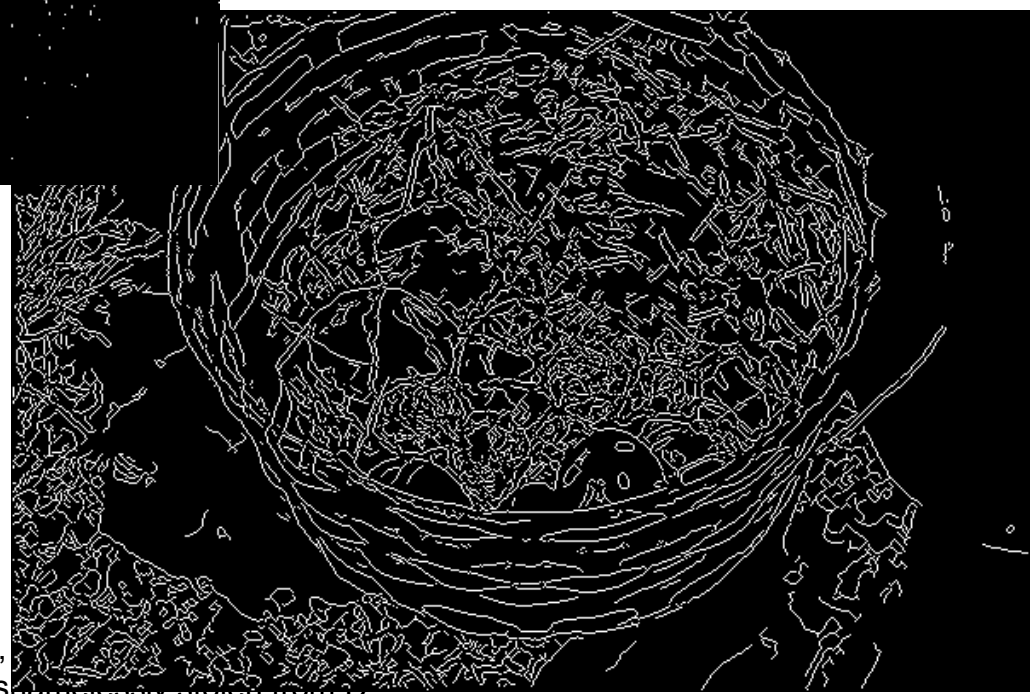
- Hysteresis Thresholding:
  - Starting from upper left, visit pixels until one exceeds t_{upper}
  - Follow chains of maxima in edge direction until value drops below t_{lower}
  - Mark and save all visited values as a connected contour

Canny Output



Laplacian vs. Canny Comparison





SubPixel Precision

- It is often hard to exactly localize the maximum of a function
- Many algorithms need sub-pixel precision
- Thus, it is common to apply a *second derivative* operator locally to locate the edge
  - note we know the edge direction, so we can compute second directional derivatives!

DO IT IN MATLAB GREG!

Edges Summary

- Filtering is a way of removing noise or suppressing/enhancing frequency content
- Typically, we combine some type of image derivative with smoothing
- Image gradients are the basic tool in 2D images
- Derivative of Gaussian is generally the gradient operator of choice
- Canny detector is probably the most widely used algorithm for performing edge detection

Detecting Corners

- Edges can be thought of as “1D” features
- Corners are “2D” features
- To make this precise, we need to think about the span of the gradients

$$C = \sum_N \nabla I(u) (\nabla I(u))' = R D R'$$

$$D = \mathrm{diag}(\lambda_1, \lambda_2)$$

$$c = \min(\lambda_1, \lambda_2)$$

Corners Algorithm

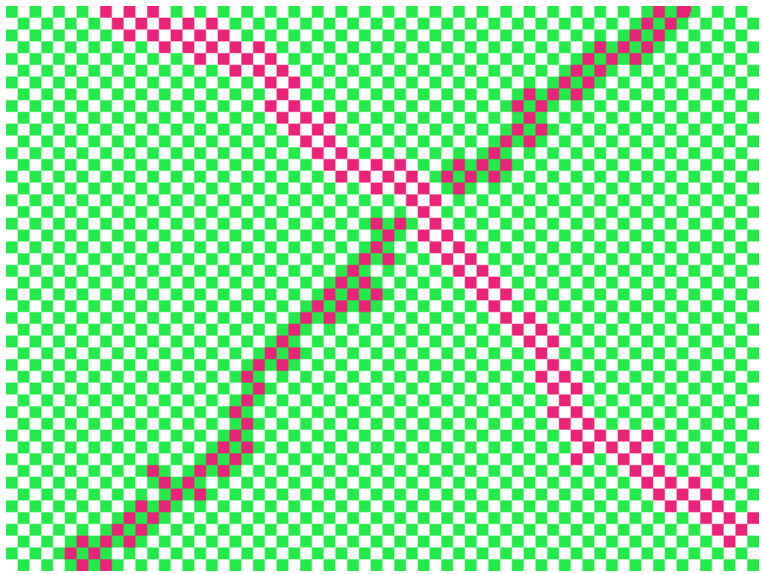
- For every point (u,v)', compute c for a neighborhood about (u,v)
- Threshold all values for which $c(u,v) > \tau$
- Sort the resulting values c(u,v)
- Read off locations starting with highest values and working down until enough locations are found or we run out of locations.

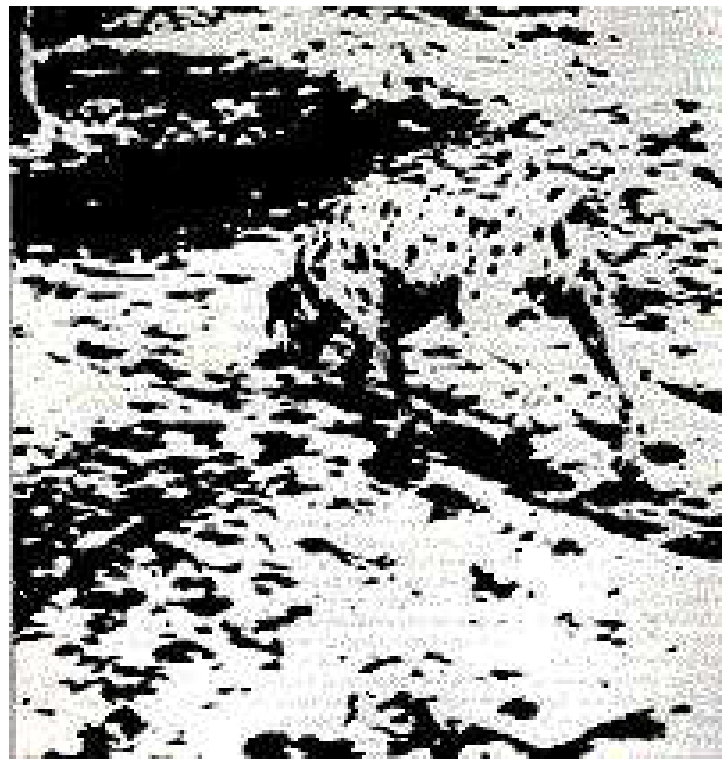
Grouping and Segmentation

- 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



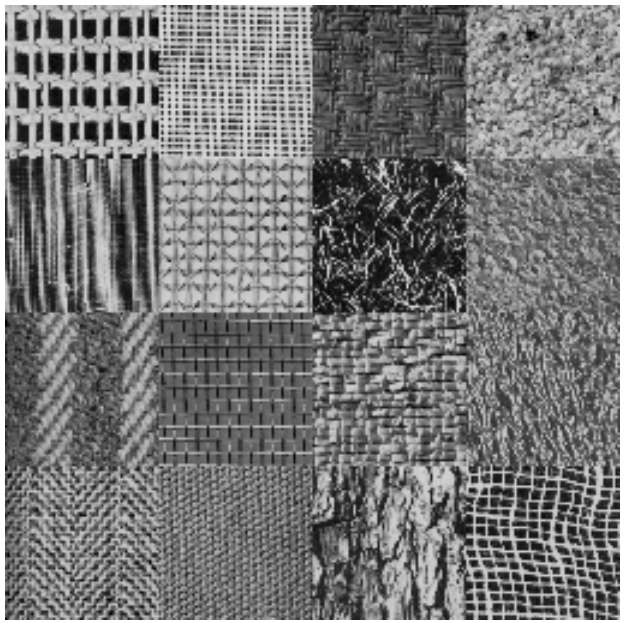


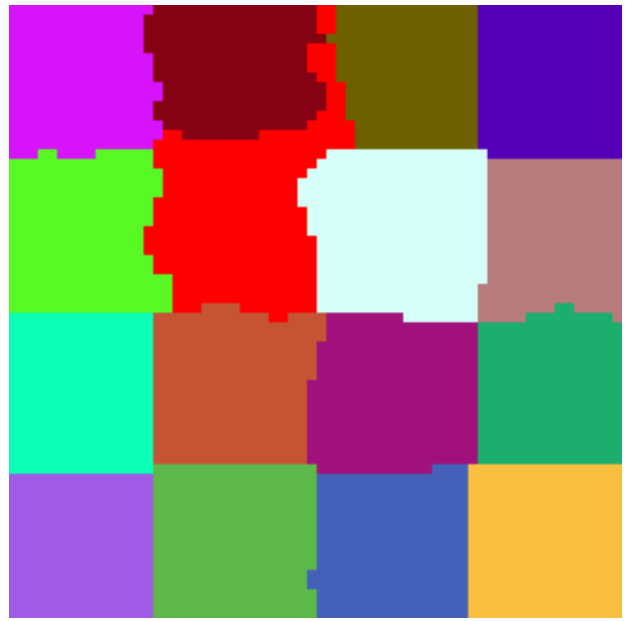
Grouping and Segmentation

- **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

Grouping and Segmentation

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

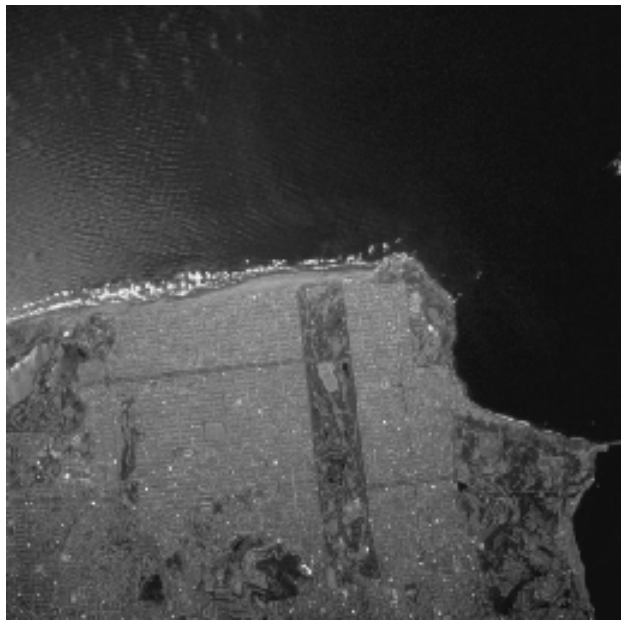


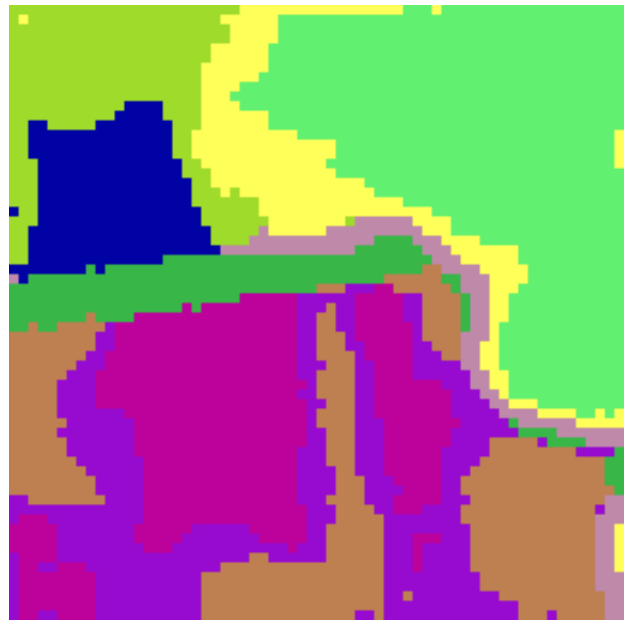


Courtesy Uni Bonn

Grouping and Segmentation

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





Courtesy Uni Bonn

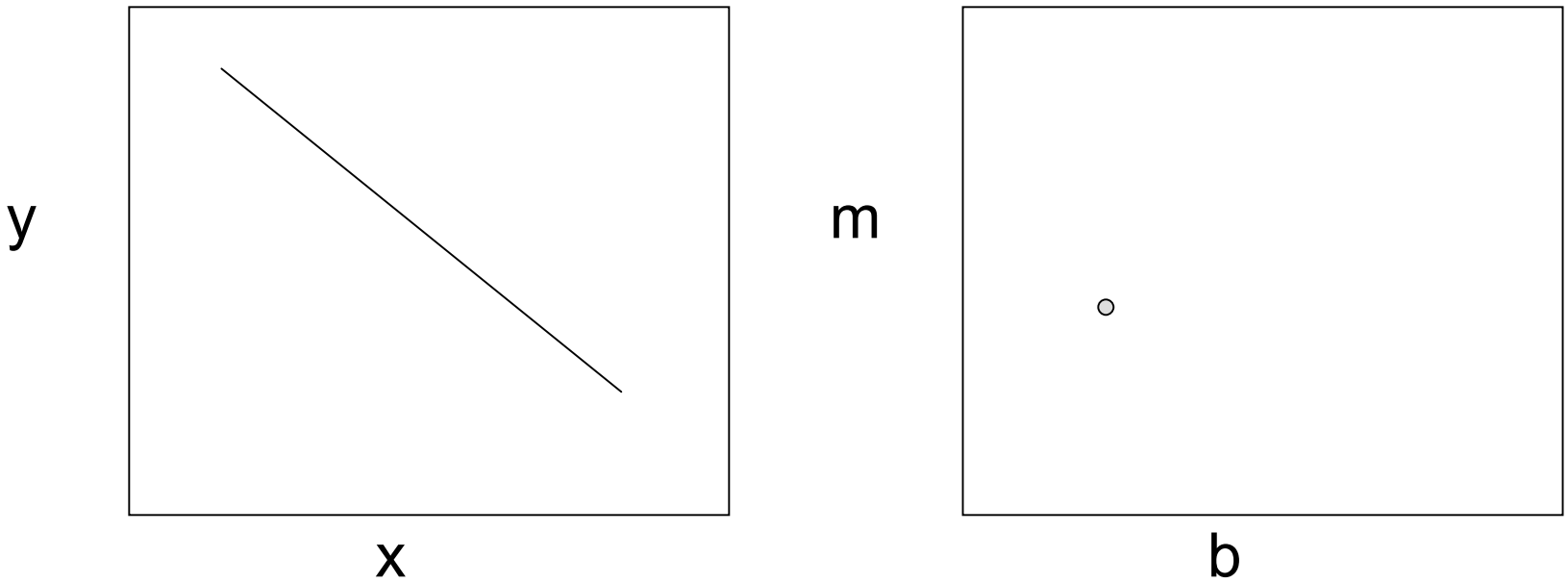
Grouping and Segmentation

- 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.

The Hough Transform

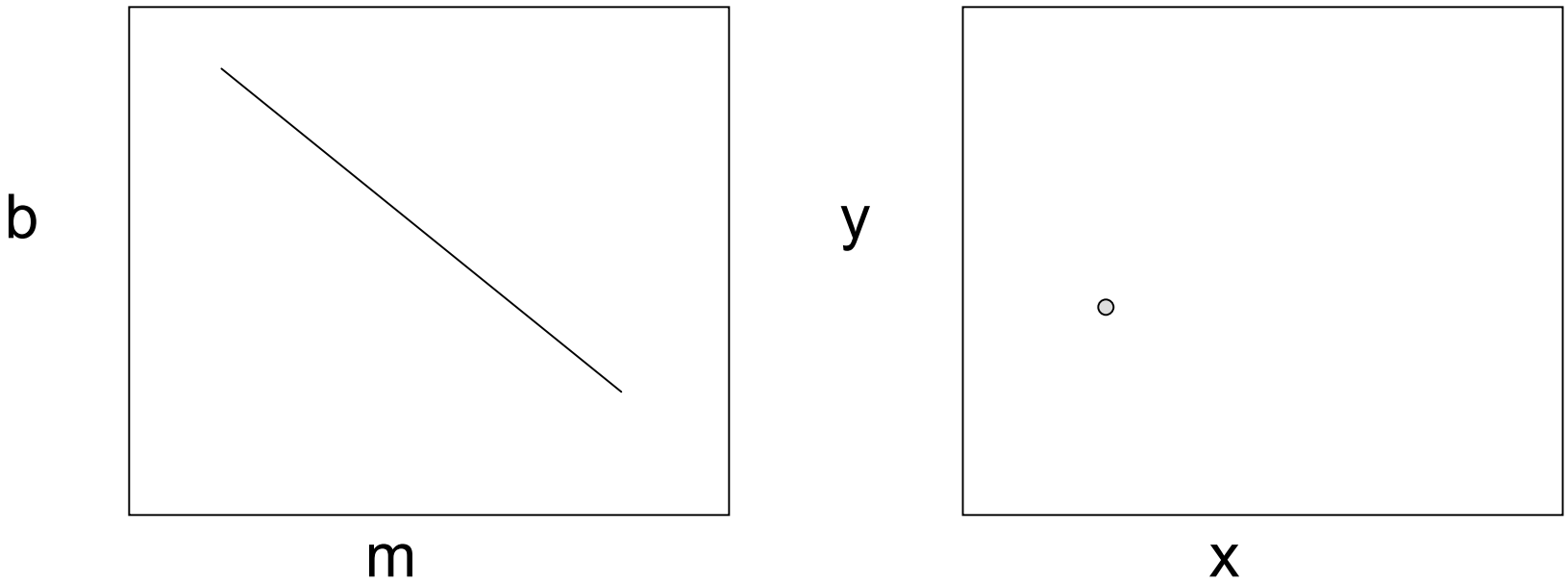
- How can we detect extended line structures in images?



$$y = m\,x + b$$

The Hough Transform

- How can we detect extended line structures in images?



$$b = - x m + y$$

Hough Transform Idea

- Each edge point in an image is a constraint on line parameters:
  - constraint is a line
  - each unique point adds another constraint

- Algorithm:
  - Initialize a 2-D array of counters to zero.
  - For each edge point (x,y), increment any counter which contains a parameter point (b,m) satisfying b = -x m + y
  - Threshold counters
  - Group edgels that belong to above threshold counters into contours
  - Optional: Refit lines to get higher precision.

Hough Transform Idea

- Each edge point in an image is a constraint on line parameters:
  - constraint is a line
  - each unique point adds another constraint

- Algorithm:
  - Initialize a 2-D array of counters to zero.
  - For each edge point (x,y), increment any counter which contains a parameter point (b,m) satisfying b = -x m + y
  - Threshold counters
  - Group edgels that belong to above threshold counters into contours
  - Optional: Refit lines to get higher precision.

Hough Transform Variation

- Problem:
  - m and b are, in principle, unbounded
    - rewrite as sin(t) x + cos(t) y = d
    - For each discrete value of t from 0 to pi, increment counter with nominal d minimal distance from exact value
  - coarse grid leads to grouping of distinct lines
    - in post-fitting stage, re-hough at higher resolution
- Generalizations
  - Any linear in parameters model: e.g a x^2 + b y^2 = 1 can use the same algorithm
  - For f(x,a) = 0, choose any cell a_c s.t. f(x,a_c) < t for some threshold t.
- Limitations:
  - the curse of dimensionality

Grouping on Grouping

- Grouping lines (recall Gestalt discussion)
  - group by proximity
  - $\mu_{pro} = (l_s/g)^2$ where g is the shortest distance between two endpoints
  - under some assumptions you can show $1/\mu_{pro}$ is proportional to the probability the two endpoints are close by accident
  - $\mu_{par} = \{l_s\}^2 / \theta \; s \; l_l$
  - $\mu_{col} = \{l_s\}^2 / \theta \; s \; \{l_s + g\}$

- Algorithm:
  - compute μ_{pro} and either μ_{col} or μ_{par} for all pairs
  - group pairs for which these values exceed threshold

Image Segmentation RoadMap

- Segmentation
  - criteria
  - region group and counting
- Simple Color Segmentation
- Color Histograms and matching
- Selection of Color Regions
- Texture

Segmentation: Definitions

- A distance measure $d(R_1,R_2) \rightarrow \Re$ or a homogeneity measure m(R)
  - note possibly $d(R_1,R_2) = |m(R_1) - M(R_2)|$
- A similarity 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

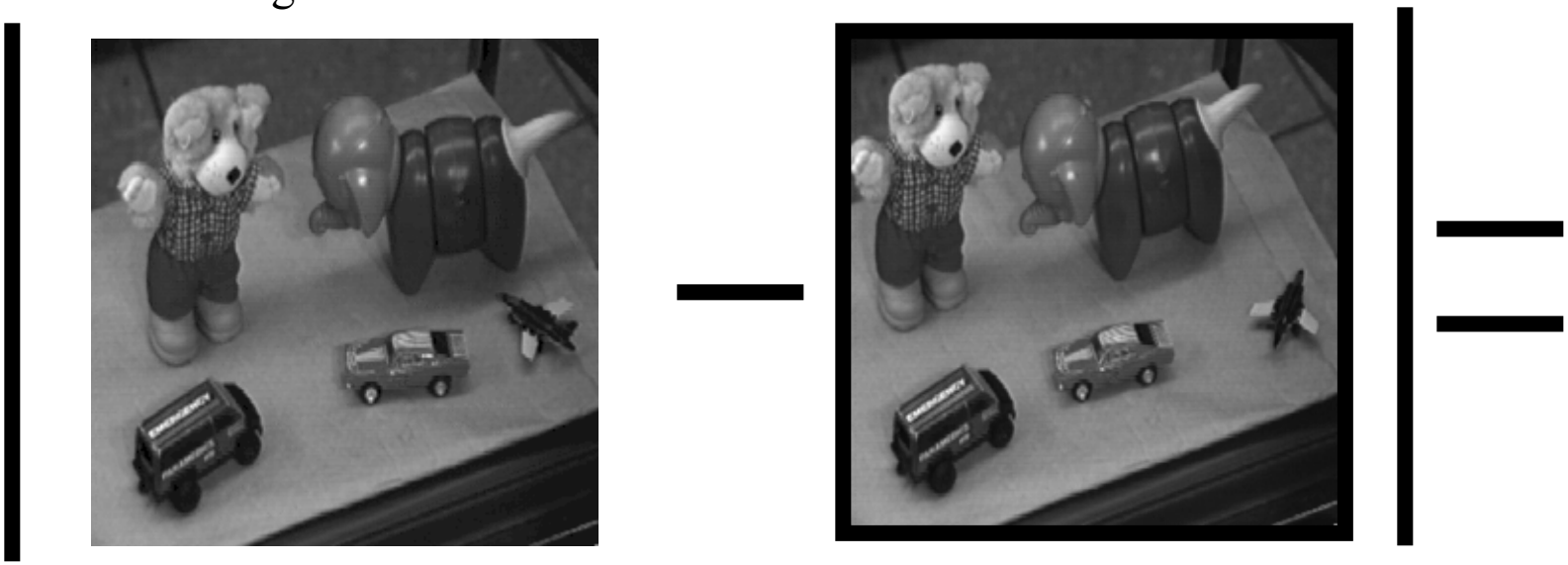
```
x x x
x 0 x
x x x
```

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

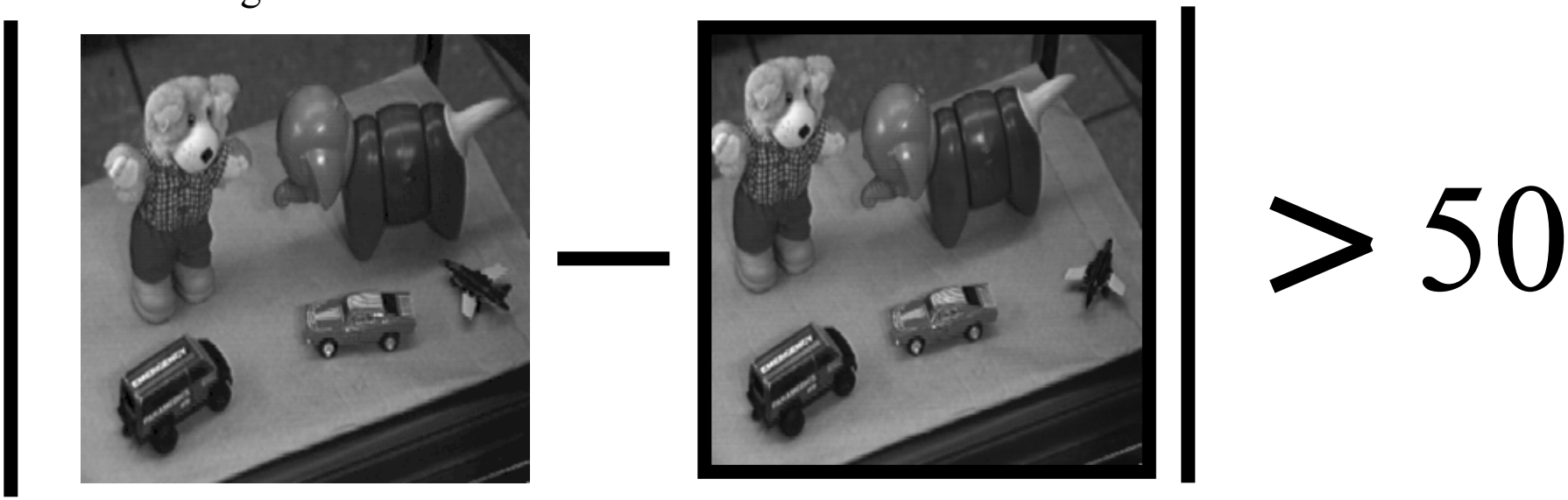
An Example: Motion

Detecting motion:



Thresholded Motion

Detecting motion:



Candidate areas for motion

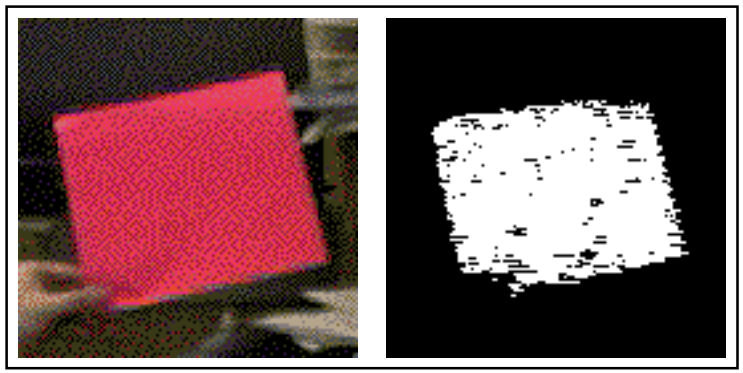
A Closer Look

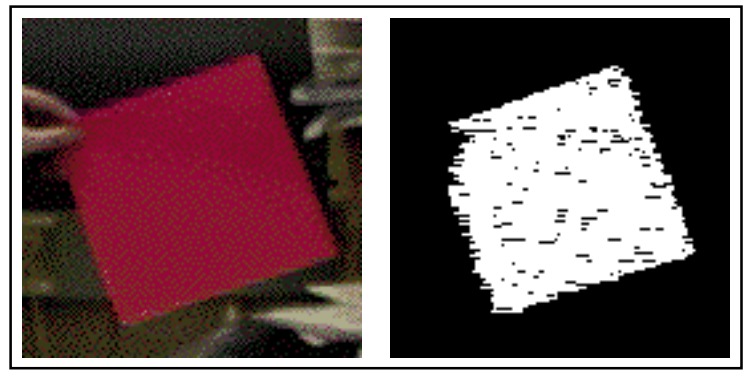
Color: A Second Example

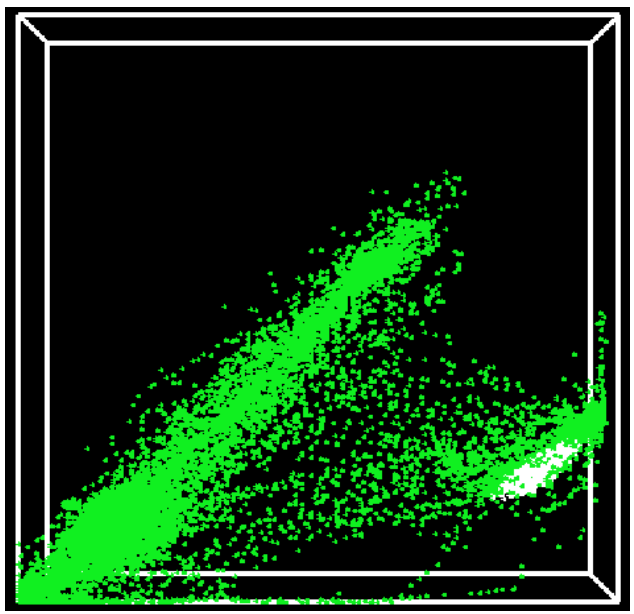
- Color representation
  - DRM [Klinker et al., 1990]: if P is Lambertian, has *matte* line and *highlight* line
  - User selects *matte* pixels in *R*
  - Compute first and second order statistics of cluster
  - Decompose ellipsoid $(\mathbf{S}, \mathbf{R}^T, \mathbf{T})$ of variance of matte cluster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

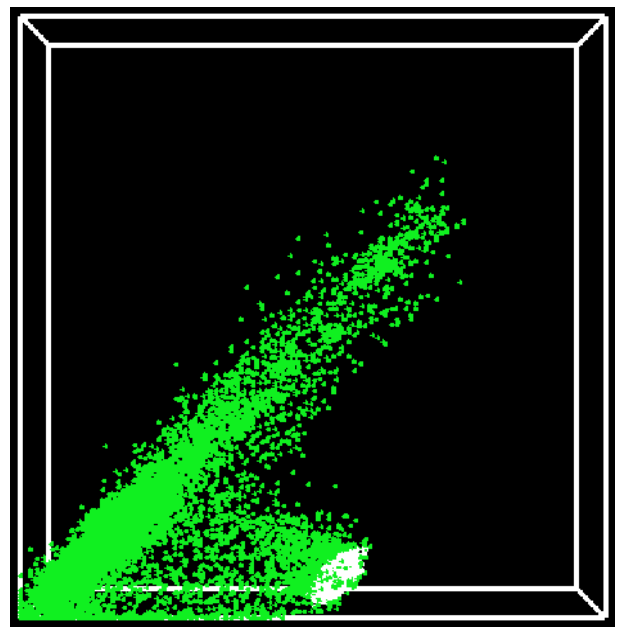
$$|\mathbf{S}^{-1}\mathbf{R}^T(\mathbf{I}(x, y) - \mathbf{T})|$$

Homogeneous Color Region: Photometry

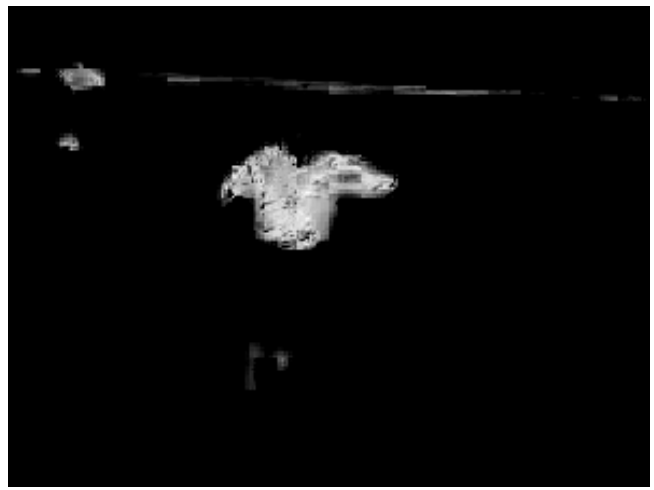




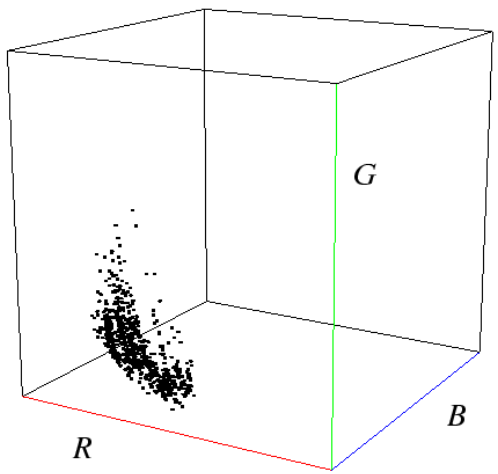


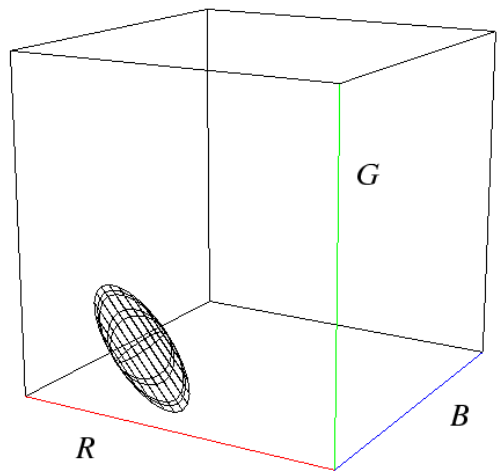


Homogeneous Region: Photometry



Sample





PCA-fitted ellipsoid

D. F
tole

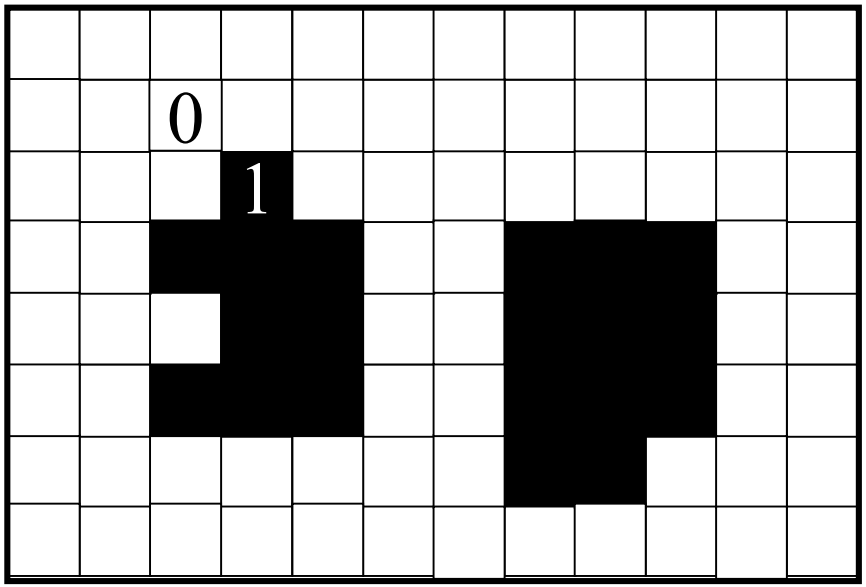
Forsyth

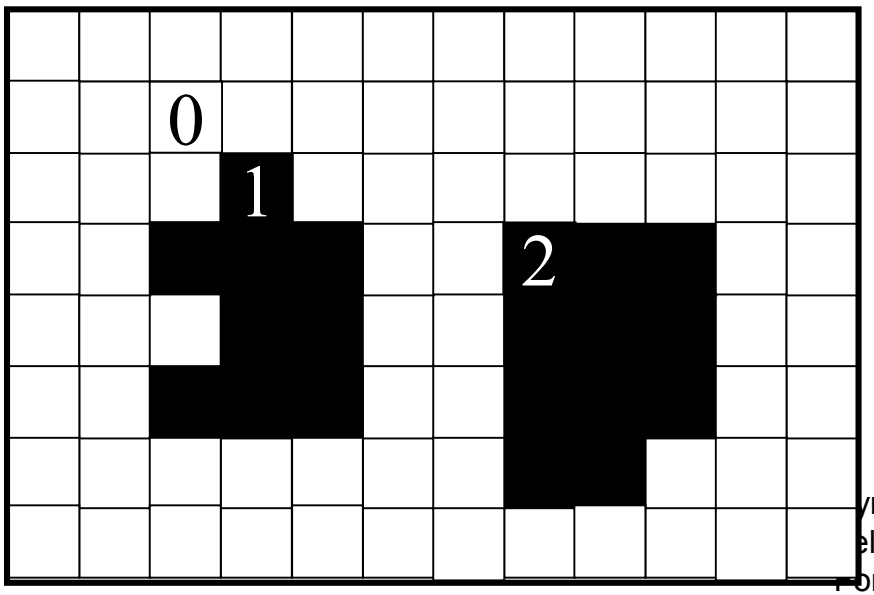
Binary Image Processing

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

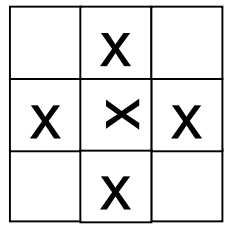
- [x] How many objects are in the image?
- [ ] Where are the distinct “object” components?
- [ ] “Cleaning up” a binary image?

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.

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

structuring element

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

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

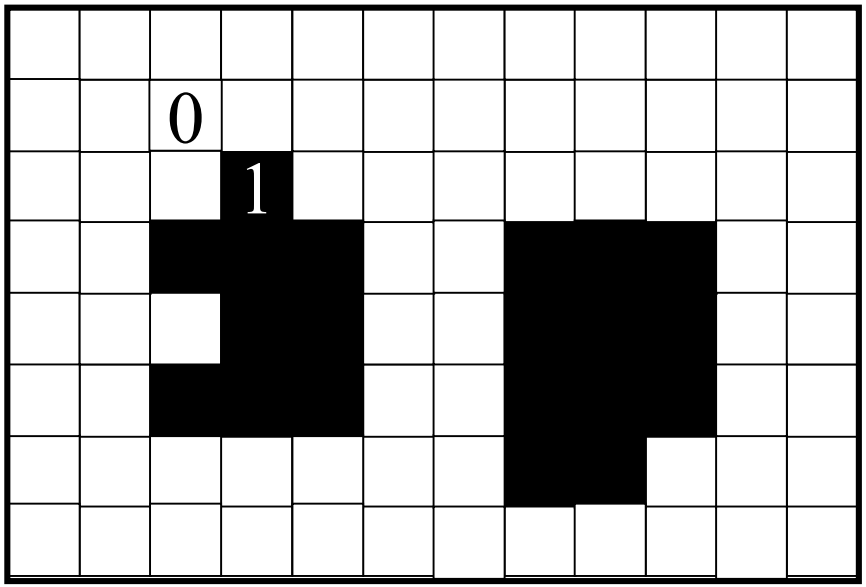
The *opening* of a binary image A by S is

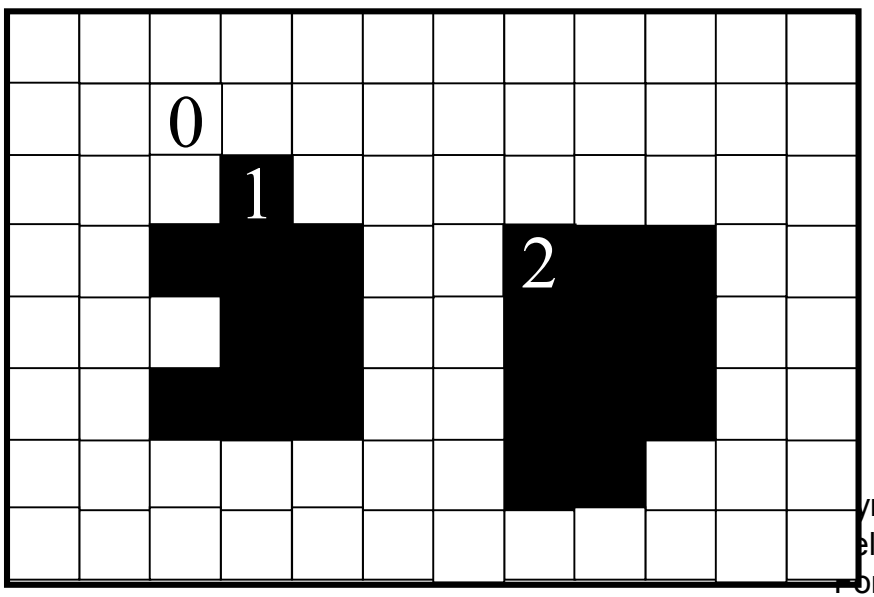
$$A \circ S = (A \ominus S) \oplus S$$

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.

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
    - check a region for tile neighbor similarity
    - If sufficiently similar, stop, otherwise split
    - repeat until no more splitting occurs

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."

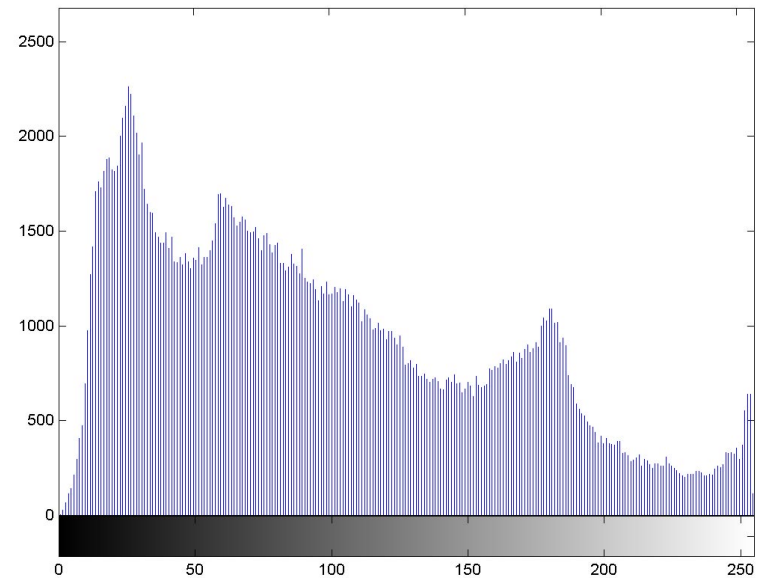
- Possible statistics:
  - mean
  - variance
  - co-occurance matrix

- 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

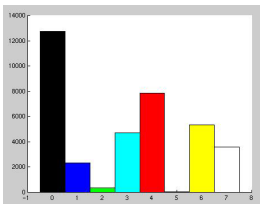
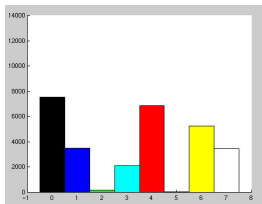
- For each pixel p_i, let h(p_i) = h(p_1) + 1



- EMD

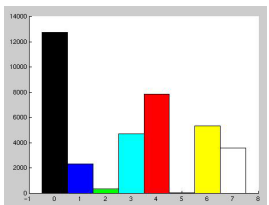
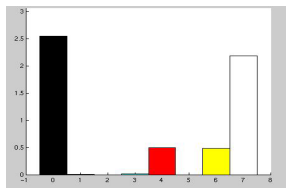
Histogram Metrics

Intersection

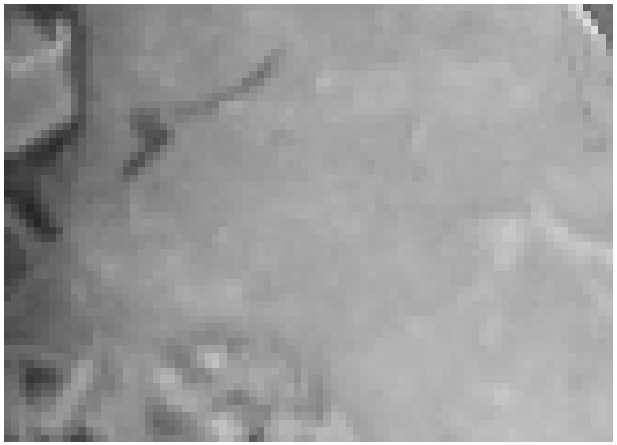
$$= \frac{\sum_i \min(M_i, T_i)}{\sum_i T_i}$$

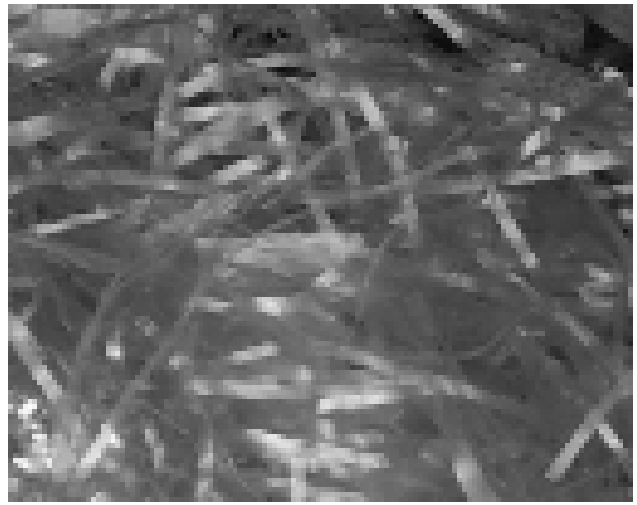
Inner Product

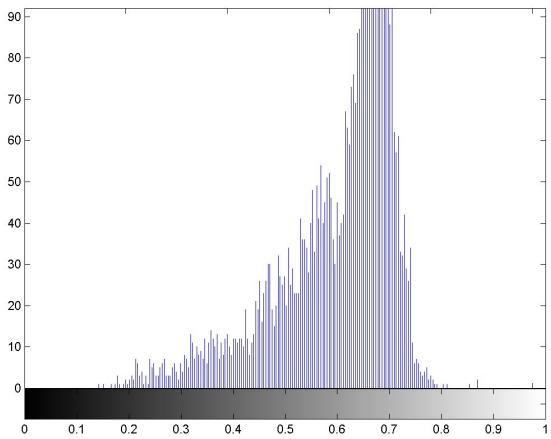



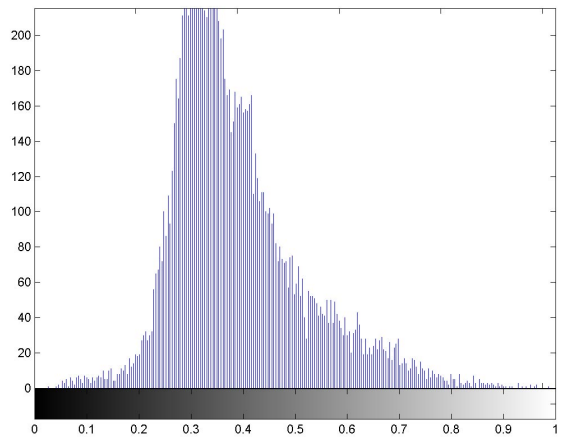
$$= M \cdot T = \frac{\sum M_i T_i}{\sqrt{\sum M_i^2}\sqrt{\sum T_i^2}}$$

Comparing Histograms

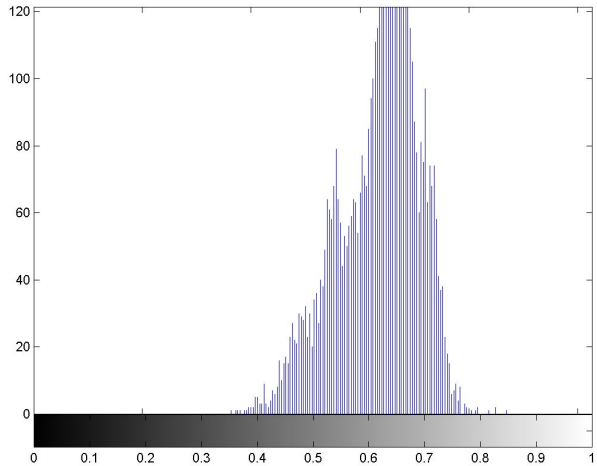


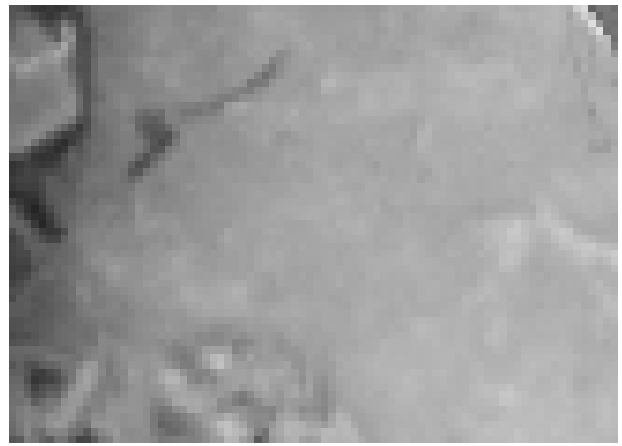




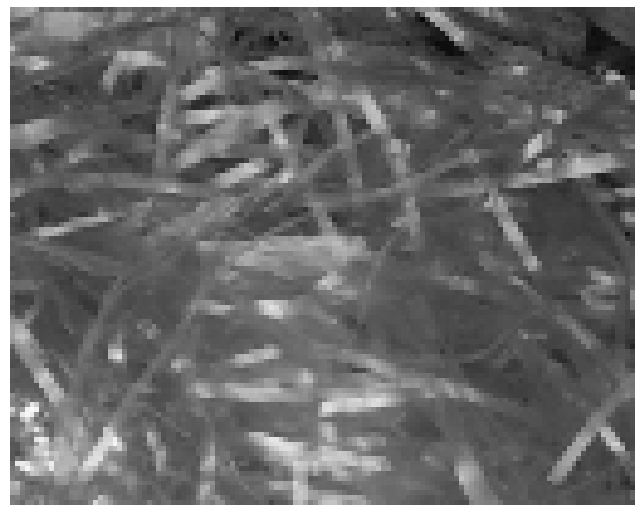


Which is More Similar?





.906

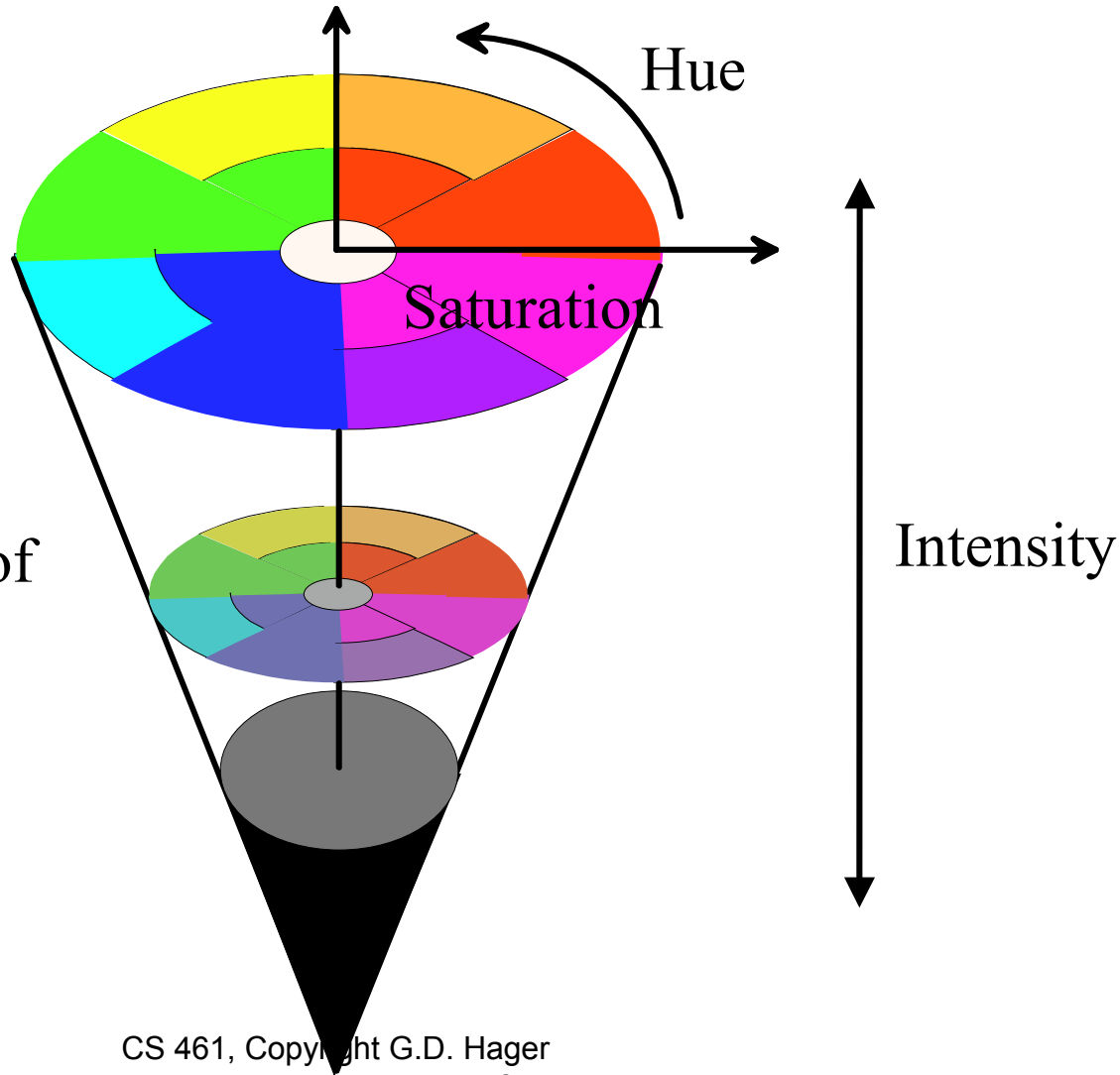


.266

An Example: Color Image Segmentation

- The goal: to choose “unique” regions of an image
- Useful for mobile robot navigation
- Provides a useful example of color histograms
- Recall:
  - a histogram is a representation of the distribution of values in a range of values:

Dividing Up Color Space

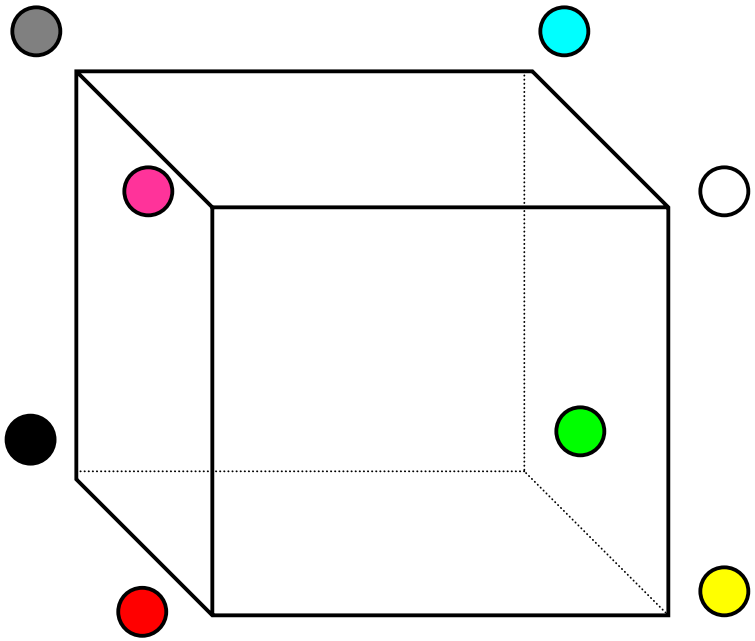


- HSI representation of color space

- Variable resolution gridding of space

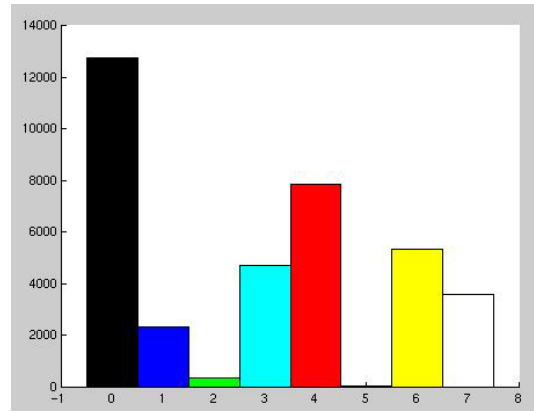
Color Histograms

feature vectors

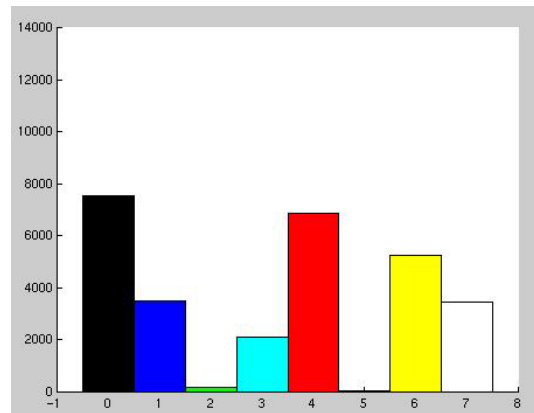


RGB

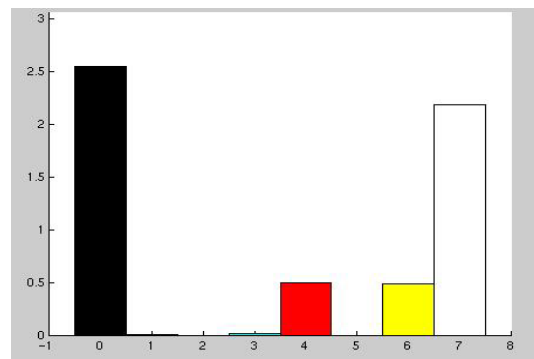
model



test

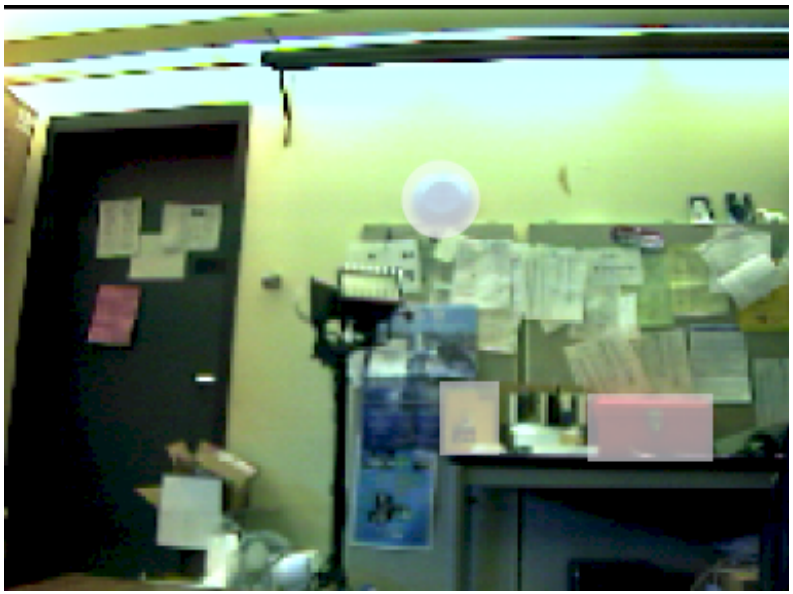


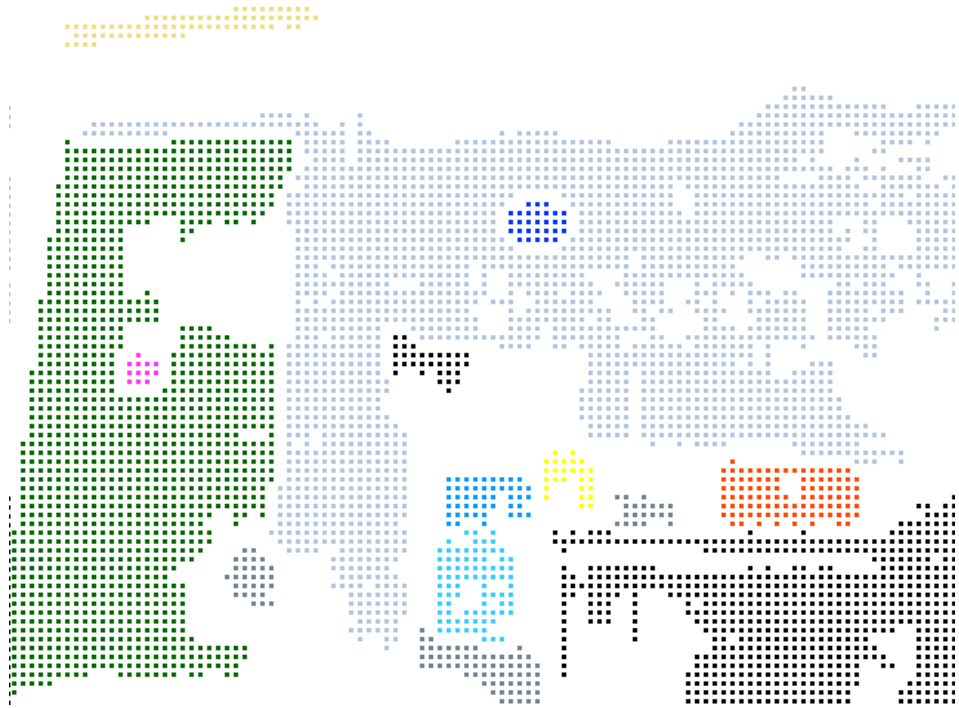
test



A bit of binning...

Results of a Merge Segmentation

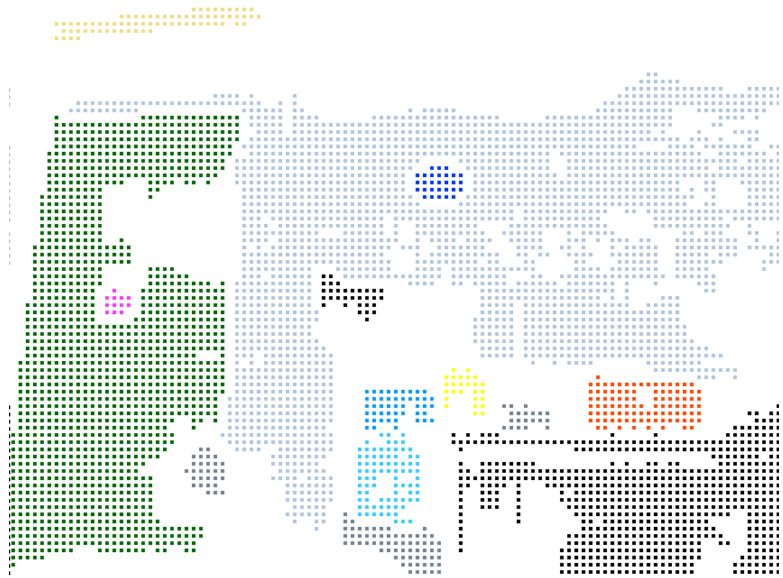




Characterizing Regions

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

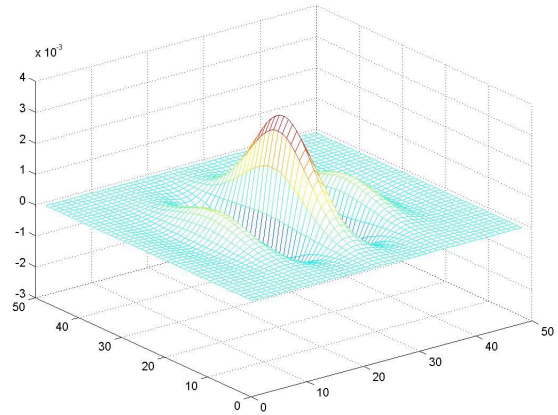
Top Three Features



More Examples

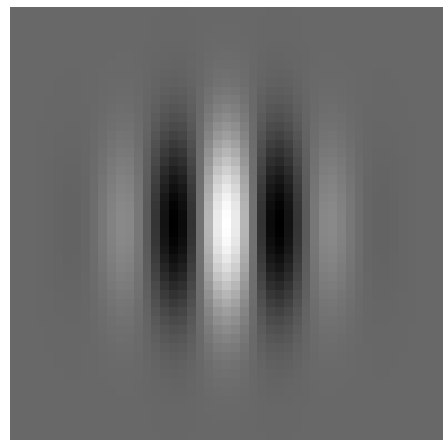
Even More Dimensions

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

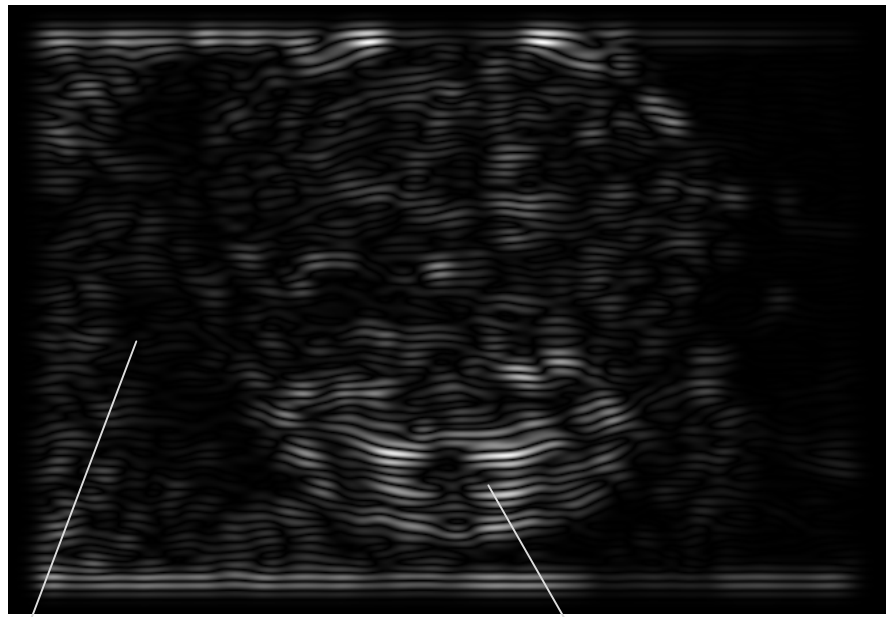


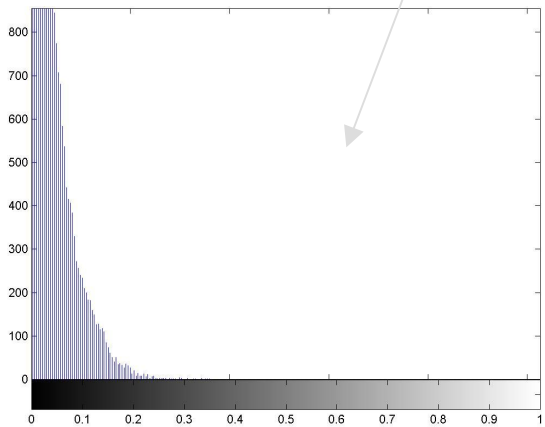
- How about computing something "richer"
  - use derivative measures
  - oriented
  - multiple scales

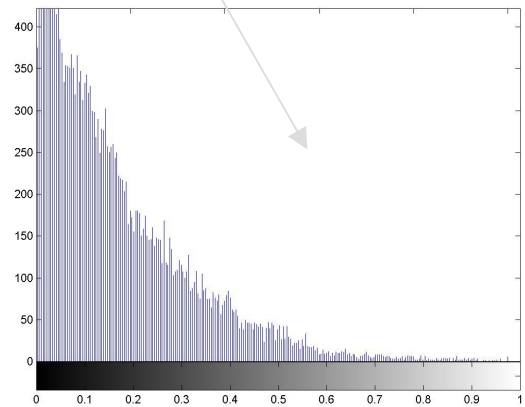
$$\cos(ax + by)\exp(-(x^2 + y^2)/2\,\sigma^2)$$



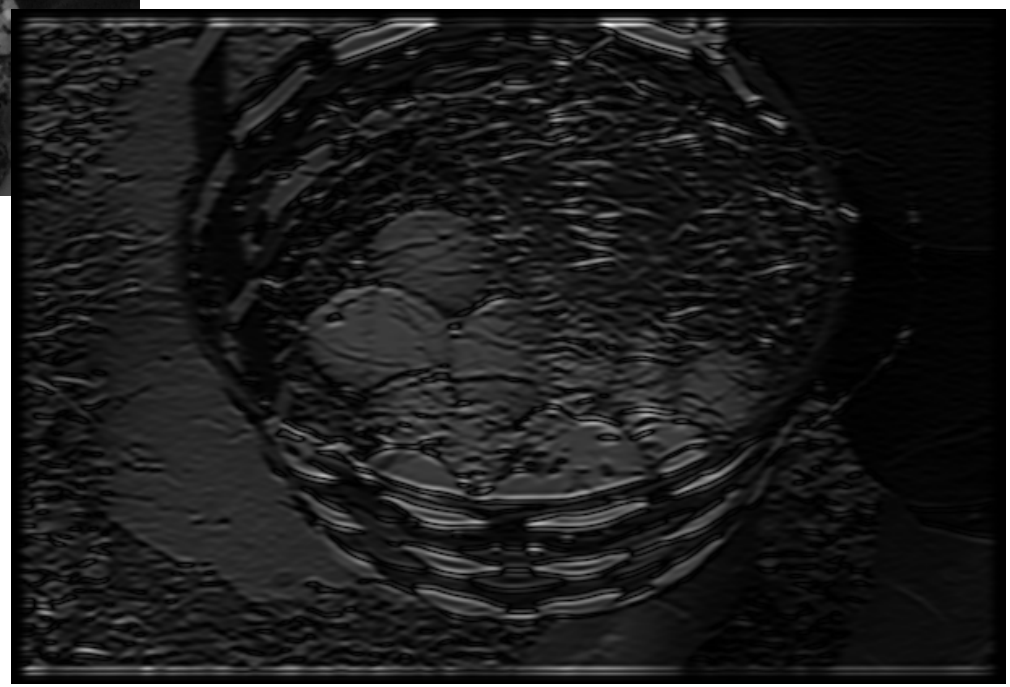
Gabor Response







Gabor Response



An (Example) Plan

- Image Filters:
  - gabor at multiple scales
  - gabor at multiple orientations
- Histograms of results
- Segmentation based on histograms
- Use of texture to
  - segment
  - compute surface properties
  - etc.

Summary

- Hough transform is a useful tool for grouping
- An example of feature grouping from edges.
- Basic thresholding and region labeling is a simple tool for segmentation
- Top-down and bottom up methods allows inter-region comparisons