Color and Texture

How do we quantify them?

How do we use them to segment an image?

Color

- Used heavily in human vision
- Color is a pixel property, making some recognition problems easy
- Visible spectrum for humans is 400 nm (blue) to 700 nm (red)
- Machines can “see” much more; ex. X-rays, infrared, radio waves

Factors that Affect Perception

- Light: the spectrum of energy that illuminates the object surface
- Reflectance: ratio of reflected light to incoming light
- Specularity: highly specular (shiny) vs. matte surface
- Distance: distance to the light source
- Angle: angle between surface normal and light source
- Sensitivity how sensitive is the sensor

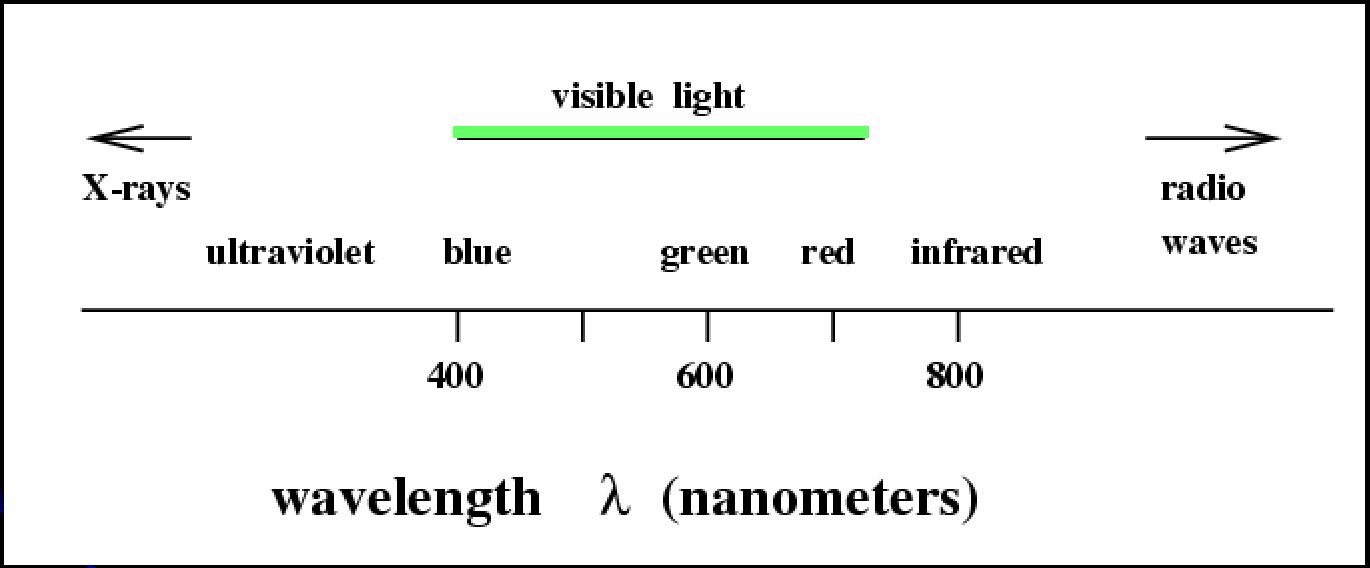
Difference Between Graphics and Vision

- In graphics we are given values for all these parameters, and we create a view of the surface.
- In vision, we are given a view of the surface, and we have to figure out what's going on.

What's going on?

Some physics of color:

Visible part of the electromagnetic spectrum

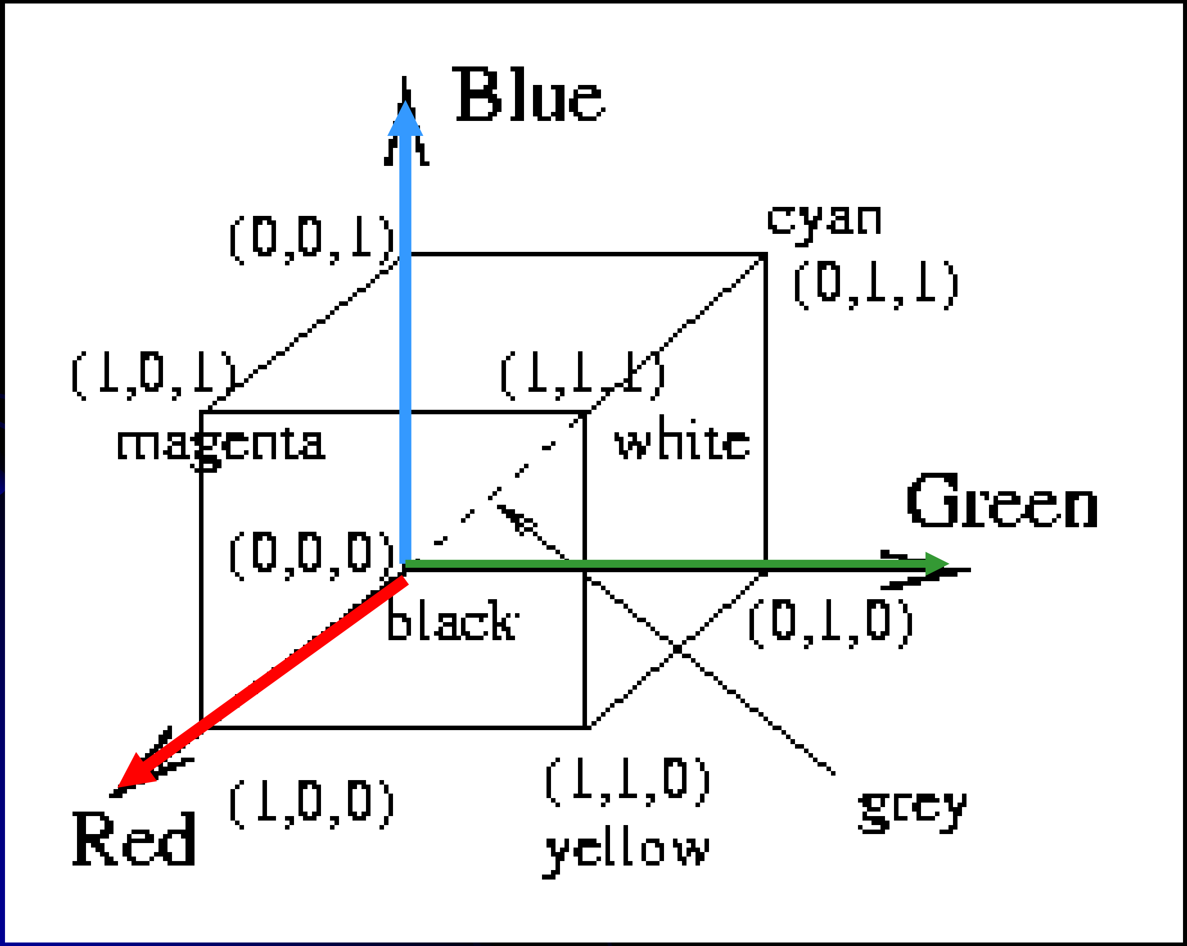


- White light is composed of all visible frequencies (400-700)
- Ultraviolet and X-rays are of much smaller wavelength
- Infrared and radio waves are of much longer wavelength

Coding methods for humans

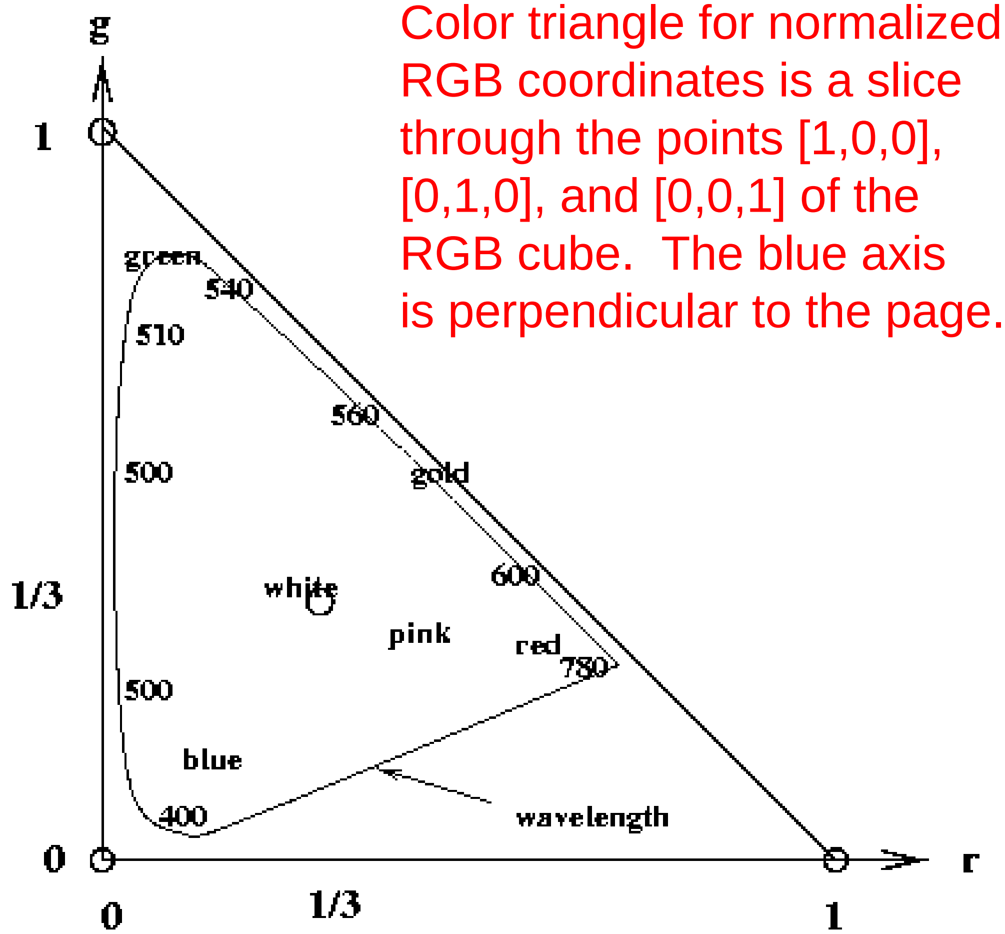
- **RGB** is an additive system (add colors to black) used for displays.
- **CMY** is a subtractive system for printing.
- **HSI** is a good perceptual space for art, psychology, and recognition.
- **YIQ** used for TV is good for compression.

RGB color cube



- R, G, B values normalized to (0, 1) interval
- human perceives gray for triples on the diagonal
- “Pure colors” on corners

Color palette and normalized RGB



Color triangle for normalized RGB coordinates is a slice through the points [1,0,0], [0,1,0], and [0,0,1] of the RGB cube. The blue axis is perpendicular to the page.

Intensity	**I = (R+G+B) / 3**
Normalized red	**r = R/(R+G+B)**
Normalized green	**g = G/(R+G+B)**
Normalized blue	**b = B/(R+G+B)**

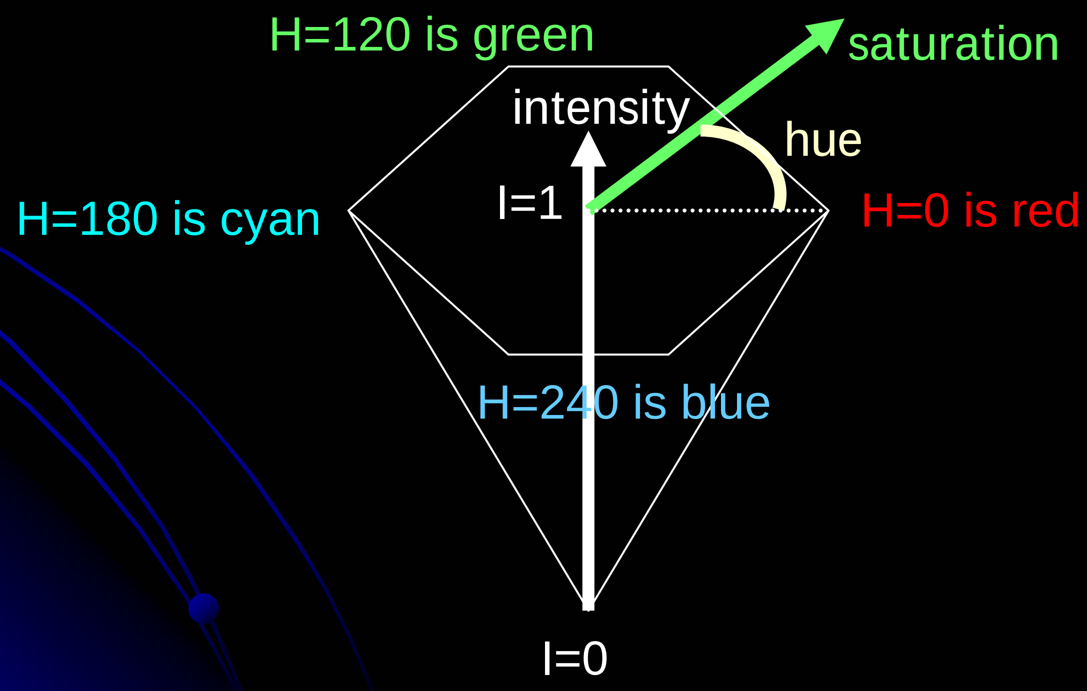
In this normalized representation, **b = 1 – r –g**, so we only need to look at r and g to characterize the color.

Color hexagon for HSI (HSV)

Hue is encoded as an angle (0 to 2π).

Saturation is the distance to the vertical axis (0 to 1).

Intensity is the height along the vertical axis (0 to 1).



Editing saturation of colors

(Left) Image of food originating from a digital camera;

(center) saturation value of each pixel decreased 20%;

(right) saturation value of each pixel increased 40%.

YIQ and YUV for TV signals

- Have better compression properties
- Luminance Y encoded using more bits than chrominance values I and Q; humans more sensitive to Y than I,Q
- Luminance used by black/white TVs
- All 3 values used by color TVs
- YUV encoding used in some digital video and JPEG and MPEG compression

Conversion from RGB to YIQ

An approximate linear transformation from RGB to YIQ:

$$\begin{array}{rrcrcrcr} luminance & Y & = & 0.30R & + & 0.59G & + & 0.11B \\ R-cyan & I & = & 0.60R & - & 0.28G & - & 0.32B \\ magenta-green & Q & = & 0.21R & - & 0.52G & + & 0.31B \end{array}$$

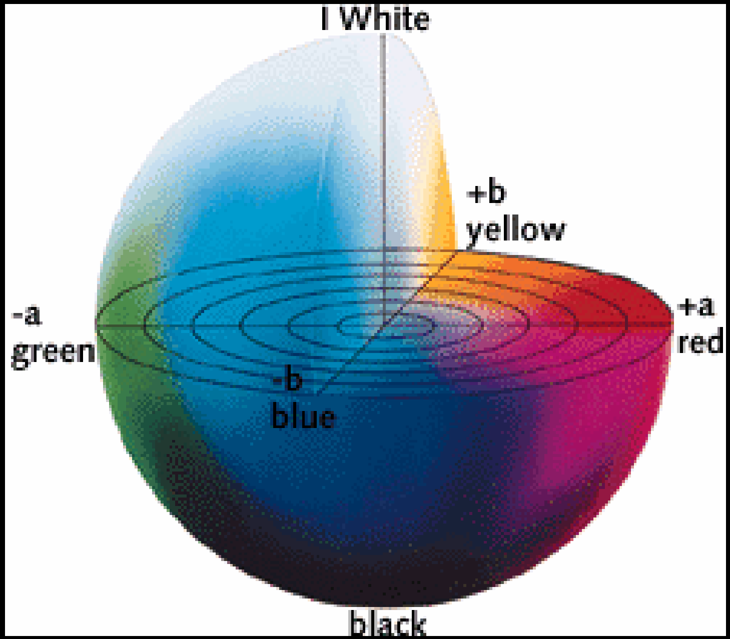
We often use this for color to gray-tone conversion.

CIE, the color system we've been using in recent object recognition work

- Commission Internationale de l'Eclairage - this commission determines standards for color and lighting. It developed the Norm Color system (X,Y,Z) and the Lab Color System (also called the CIELAB Color System).

CIELAB, Lab, L*a*b

- One luminance channel (L)
  and two color channels (a and b).
- In this model, the color differences which you perceive correspond to Euclidian distances in CIELab.
- The a axis extends from green (-a) to red (+a) and the b axis from blue (-b) to yellow (+b). The brightness (L) increases from the bottom to the top of the three-dimensional model.



References

- The text and figures are from
  http://www.sapdesignguild.org/resources/glossary_color/index1.html
- CIELab Color Space
  http://www.fho-emden.de/~hoffmann/cielab03022003.pdf
- Color Spaces Transformations
  http://www.couleur.org/index.php?page=transformations
- 3D Visualization
  http://www.ite.rwth-aachen.de/Inhalt/Forschung/FarbbildRepro/Farbkoerper/Visual3D.html

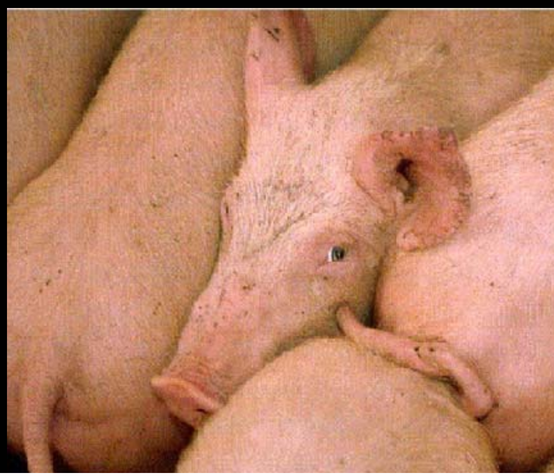
Colors can be used for image segmentation into regions

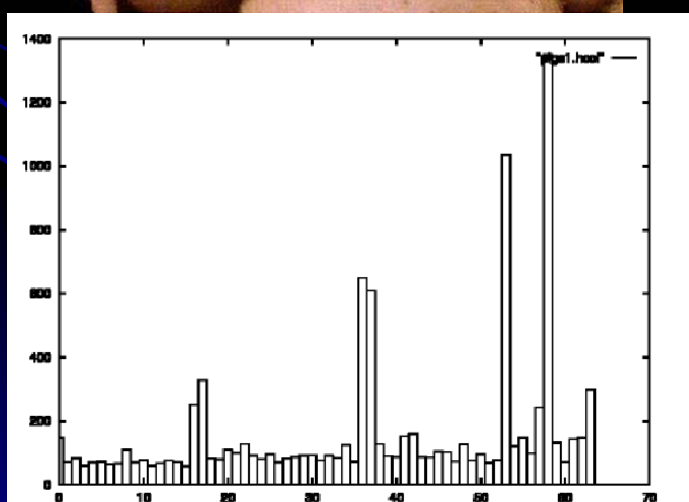
- Can cluster on color values and pixel locations
- Can use connected components and an approximate color criteria to find regions
- Can train an algorithm to look for certain colored regions – for example, skin color

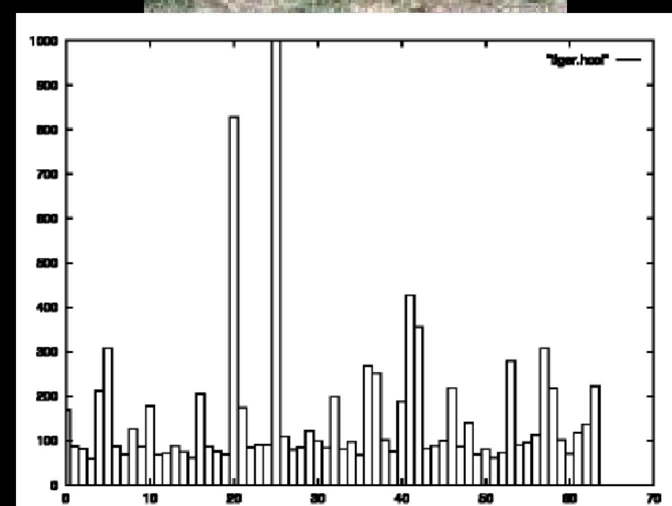
Color histograms can represent an image

- Histogram is fast and easy to compute.
- Size can easily be normalized so that different image histograms can be compared.
- Can match color histograms for database query or classification.

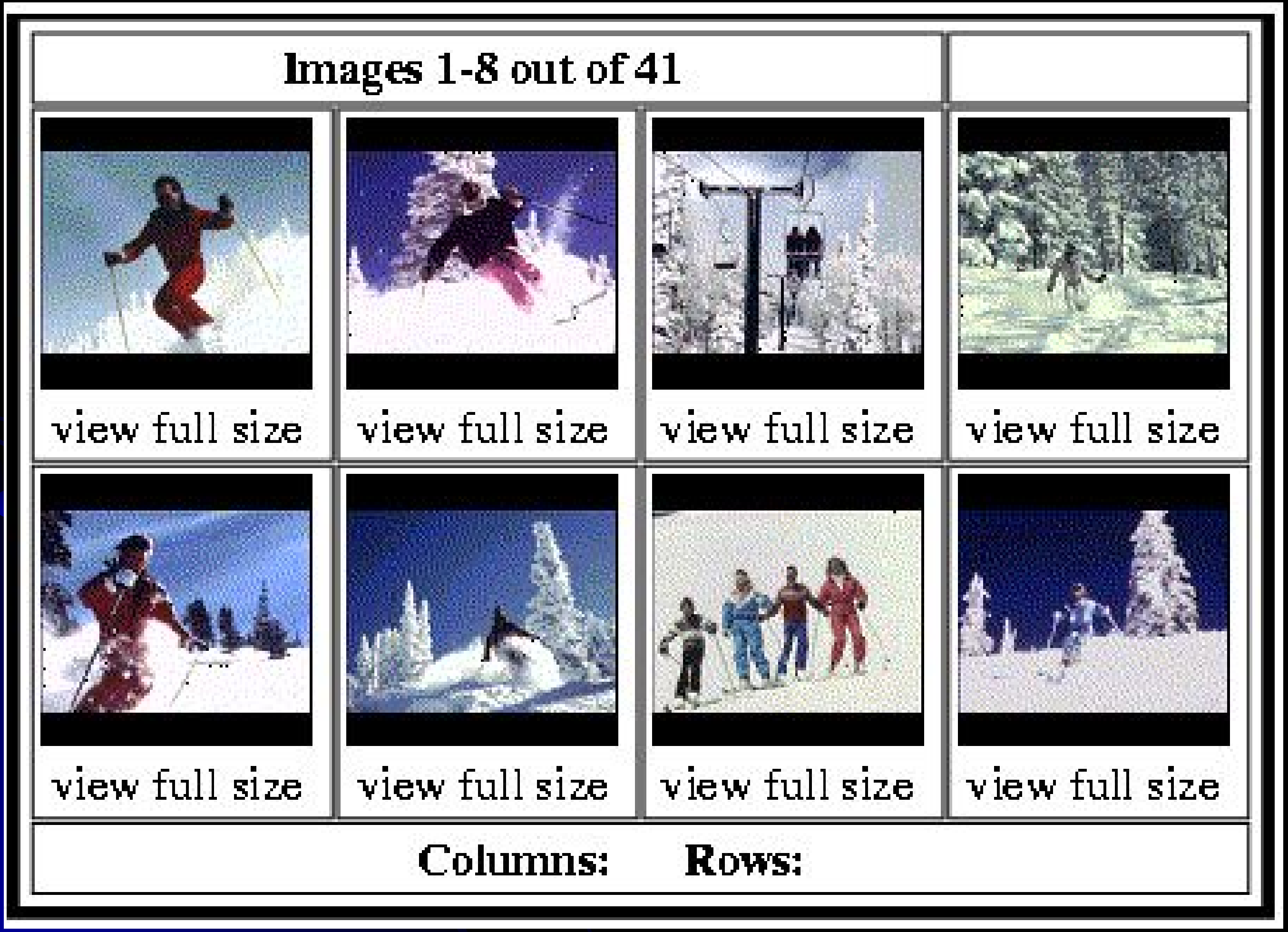
Histograms of two color images







Retrieval from image database

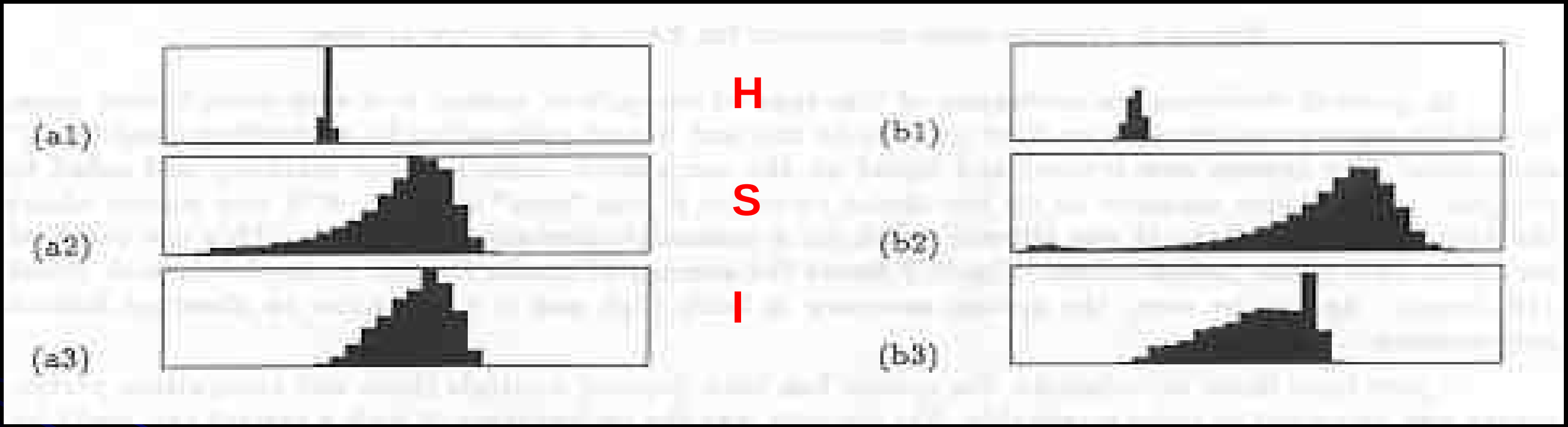


Top left image is query image. The others are retrieved by having similar color histogram (See Ch 8).

How to make a color histogram

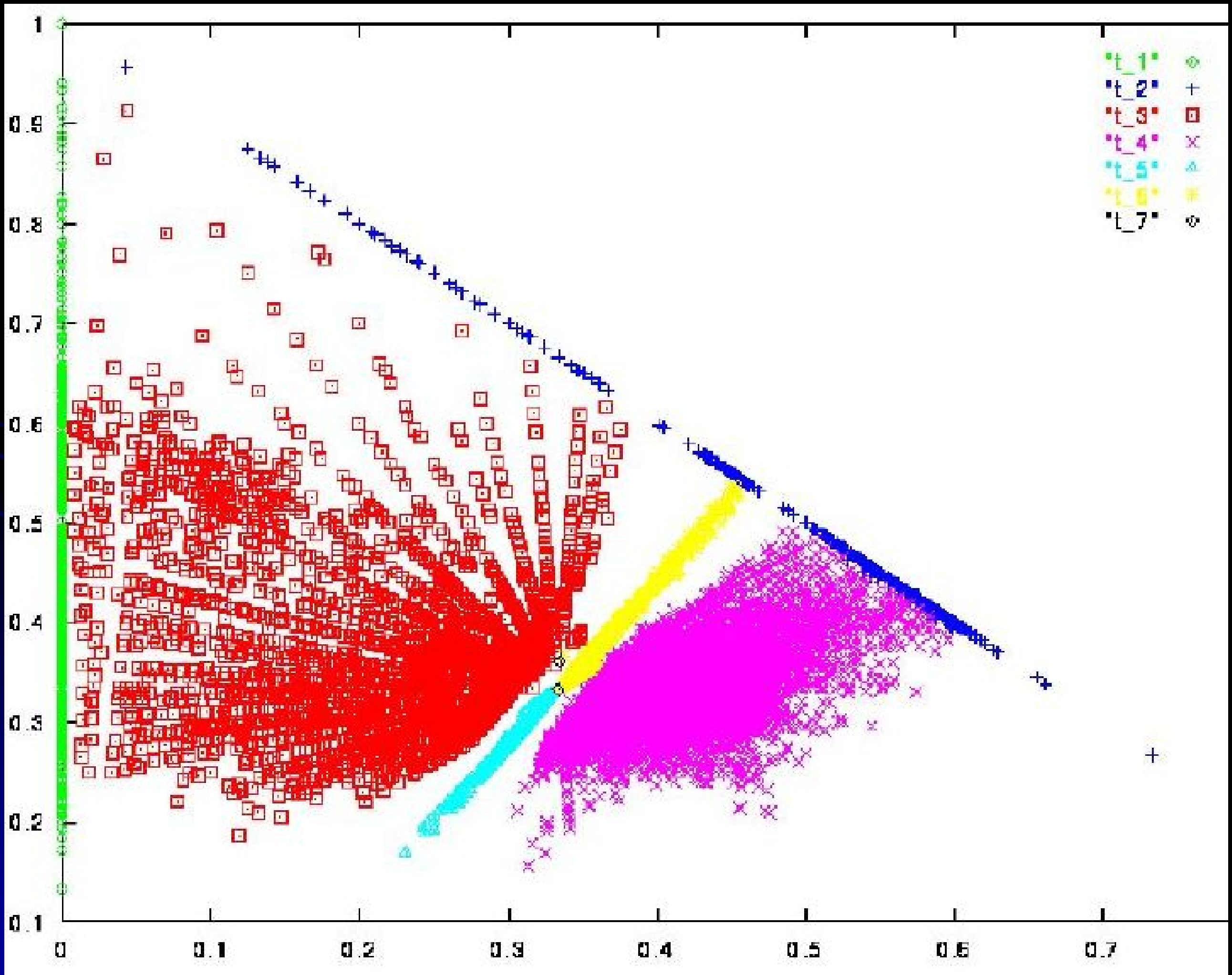
- Make 3 histograms and concatenate them
- Create a single pseudo color between 0 and 255 by using 3 bits of R, 3 bits of G and 2 bits of B (which bits?)
- Use normalized color space and 2D histograms.

Apples versus Oranges



Separate HSI histograms for apples (left) and oranges (right) used by IBM's VeggieVision for recognizing produce at the grocery store checkout station (see Ch 16).

Skin color in RGB space (shown as normalized red vs normalized green)



Purple region shows skin color samples from several people. Blue and yellow regions show skin in shadow or behind a beard.

Finding a face in video frame

- (left) input video frame
- (center) pixels classified according to RGB space
- (right) largest connected component with aspect similar to a face (all work contributed by Vera Bakic)

Swain and Ballard's Histogram Matching for Color Object Recognition (IJCV Vol 7, No. 1, 1991)

Opponent Encoding:

- $wb = R + G + B$
- $rg = R - G$
- $by = 2B - R - G$

Histograms: 8 x 16 x 16 = 2048 bins

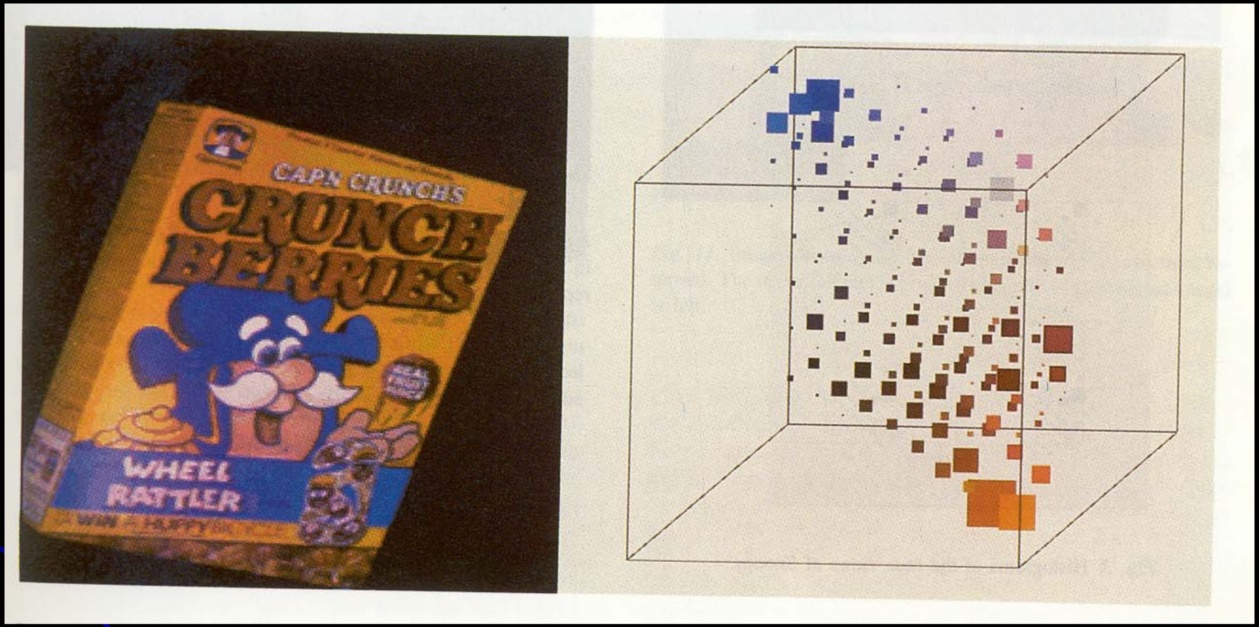
Intersection of image histogram and model histogram:

$$\text{intersection}(h(I),h(M)) = \sum_{j=1}^{\text{numbins}} \min\{h(I)[j],h(M)[j]\}$$

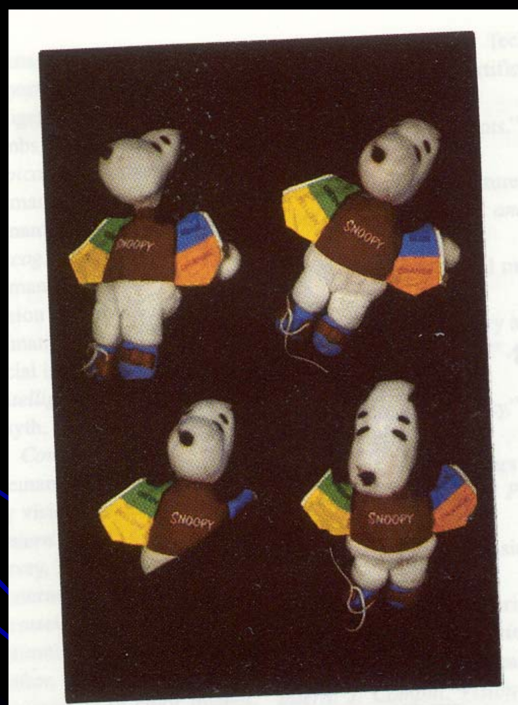
Match score is the normalized intersection:

$$\text{match}(h(I),h(M)) = \text{intersection}(h(I),h(M)) \,/ \sum_{j=1}^{\text{numbins}} h(M)[j]$$

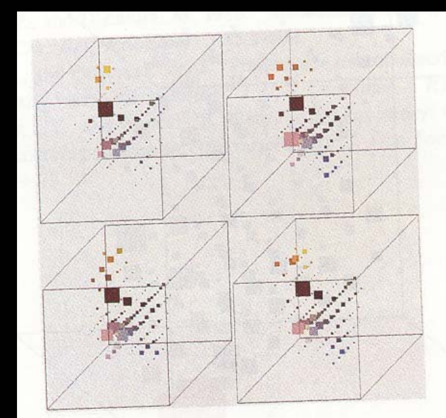
(from Swain and Ballard)



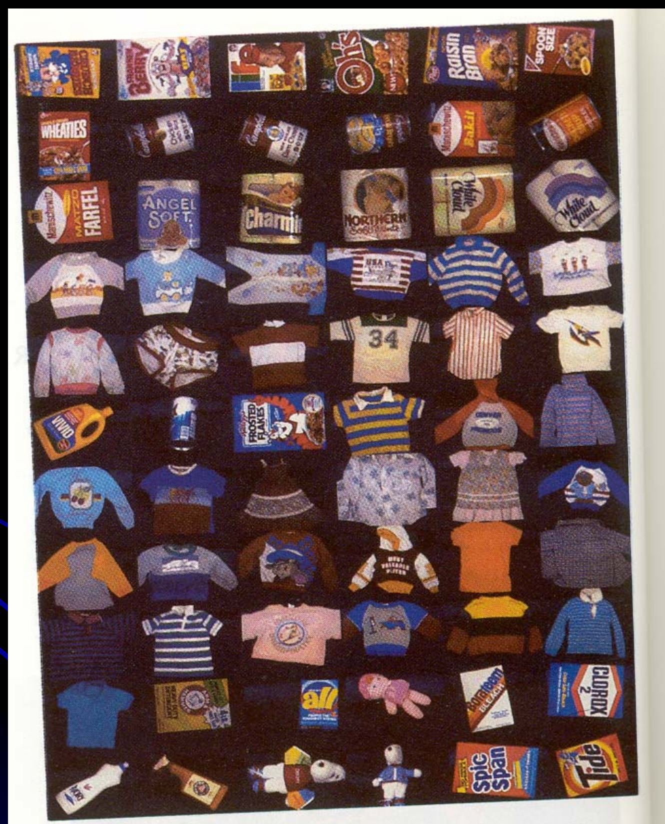
cereal box image 3D color histogram



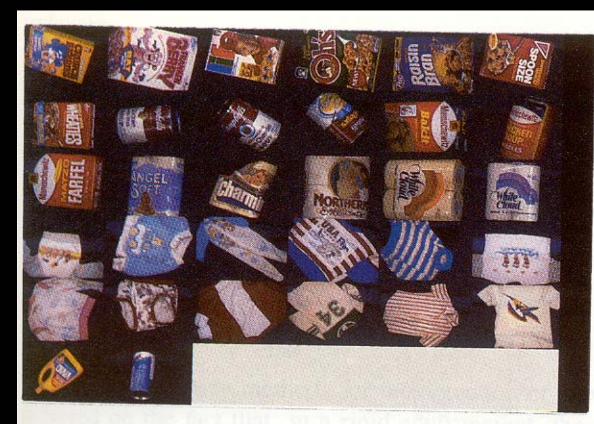
Four views of Snoopy



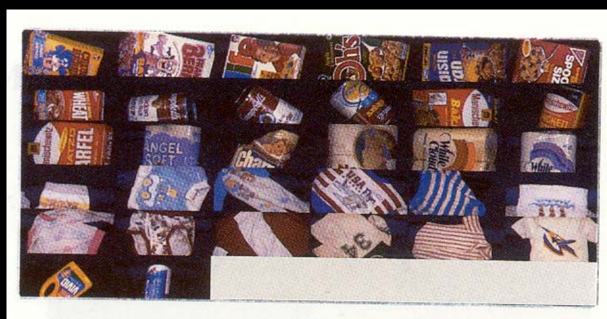
Histograms

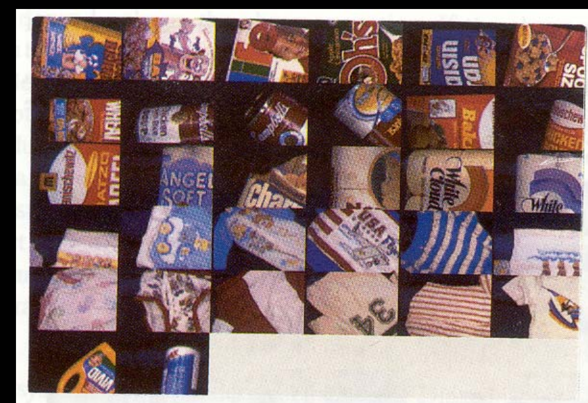


The 66 models objects



Some test objects





More test objects used in occlusion experiments

Results

Results were surprisingly good.

At their highest resolution (128 x 90), average match percentile (with and without occlusion) was 99.9.

This translates to 29 objects matching best with their true models and 3 others matching second best with their true models.

At resolution 16 X 11, they still got decent results (15 6 4) in one experiment; (23 5 3) in another.

Color Clustering by K-means Algorithm
Use for HW 2

Form K-means clusters from a set of n-dimensional vectors

1. Set ic (iteration count) to 1

2. Choose randomly a set of K means m1(1), …, mK(1).

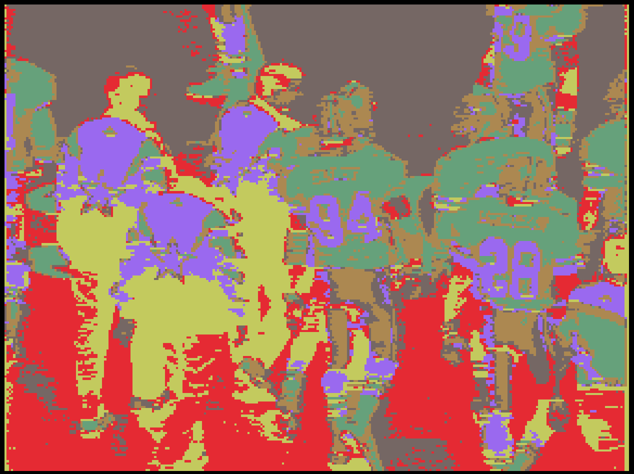
3. For each vector xi, compute D(xi,mk(ic)), k=1,…K
   and assign xi to the cluster Cj with nearest mean.

4. Increment ic by 1, update the means to get m1(ic),…,mK(ic).

5. Repeat steps 3 and 4 until Ck(ic) = Ck(ic+1) for all k.

K-means Clustering Example

Original RGB Image



Color Clusters by K-Means

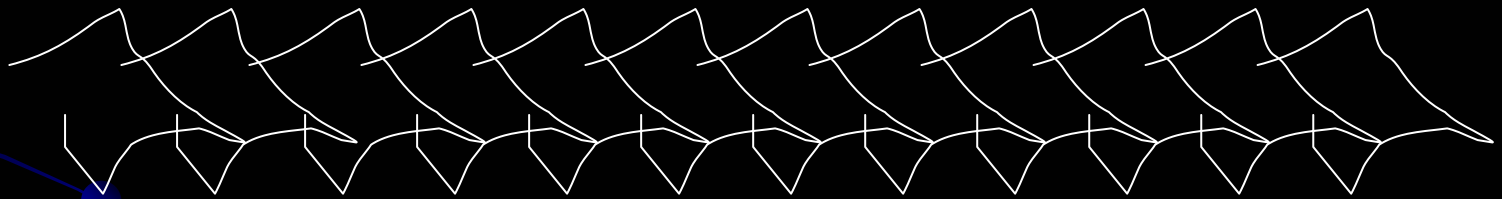
Texture

Texture is a description of the spatial arrangement of color or intensities in an image or a selected region of an image.

Structural approach: a set of texels in some regular or repeated pattern

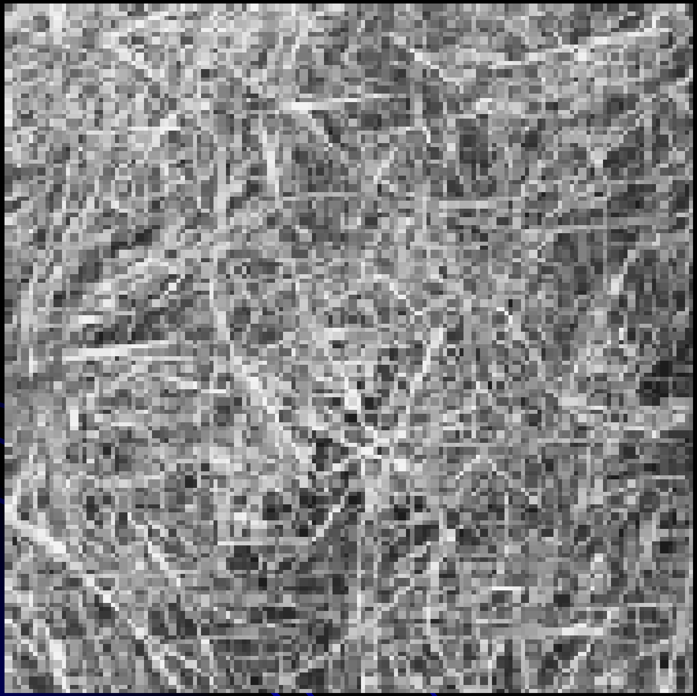
Problem with Structural Approach

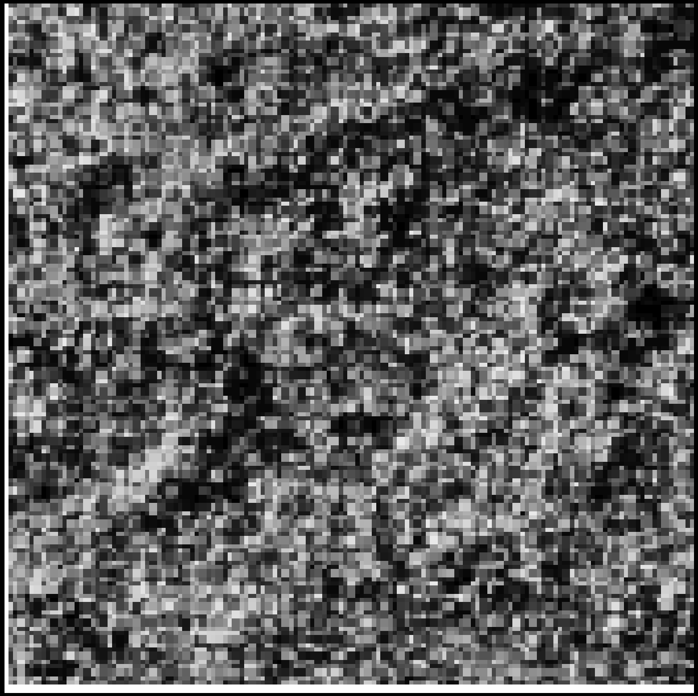
How do you decide what is a texel?



Ideas?

Natural Textures from VisTex


grass


leaves

What/Where are the texels?

The Case for Statistical Texture

- Segmenting out texels is difficult or impossible in real images.
- Numeric quantities or statistics that describe a texture can be computed from the gray tones (or colors) alone.
- This approach is less intuitive, but is computationally efficient.
- It can be used for both classification and segmentation.

Some Simple Statistical Texture Measures

1. Edge Density and Direction

- Use an edge detector as the first step in texture analysis.
- The number of edge pixels in a fixed-size region tells us how busy that region is.
- The directions of the edges also help characterize the texture

Two Edge-based Texture Measures

1. edgeness per unit area

$$\boxed{F_{edgeness} = |\{ p \mid \text{gradient\_magnitude}(p) \geq \text{threshold}\}| / N}$$

where N is the size of the unit area

2. edge magnitude and direction histograms

$$\boxed{F_{magdir} = (H_{magnitude}, H_{direction})}$$

where these are the normalized histograms of gradient magnitudes and gradient directions, respectively.

Original Image

Frei-Chen
Edge Image

Thresholded
Edge Image

Local Binary Pattern Measure

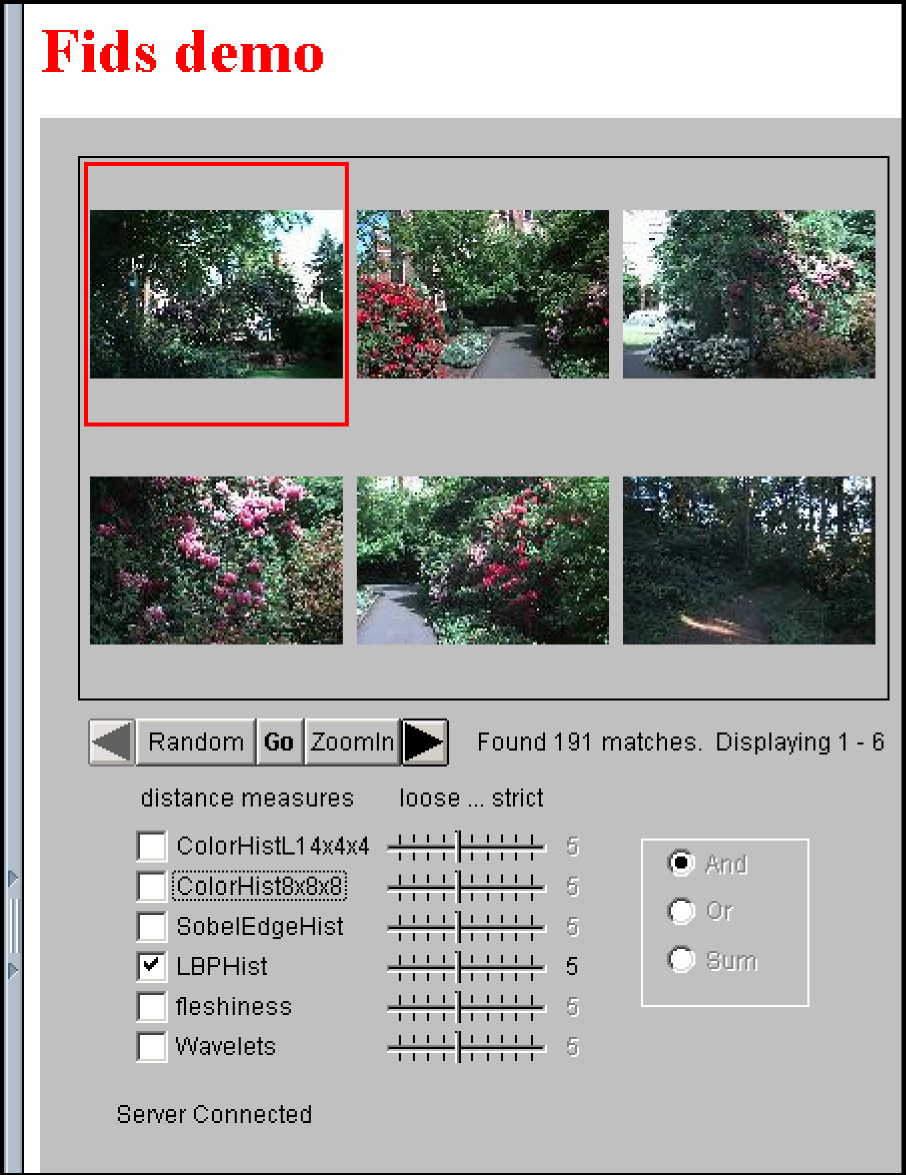
- For each pixel p, create an 8-bit number $b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7\ b_8$, where $b_i = 0$ if neighbor i has value less than or equal to p's value and 1 otherwise.

- Represent the texture in the image (or a region) by the histogram of these numbers.

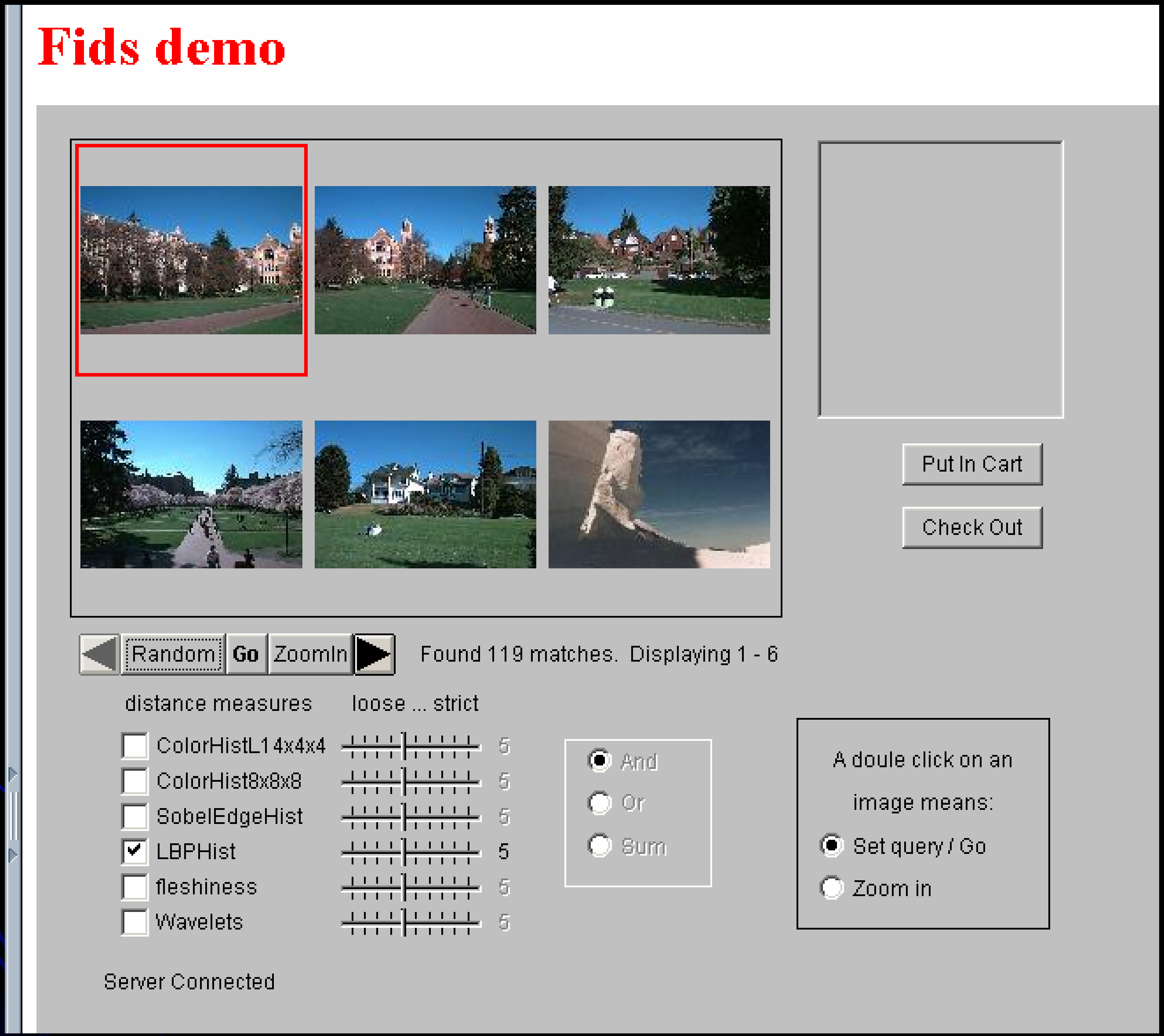
	1	2	3	
	100	101	103	
8	40	50	80	4
	50	60	90	5
	7	6		

→ $\boxed{1\ 1\ 1\ 1\ 1\ 1\ 0\ 0}$

Fids (Flexible Image Database System) is retrieving images similar to the query image using LBP texture as the texture measure and comparing their LBP histograms



Low-level measures don’t always find semantically similar images.



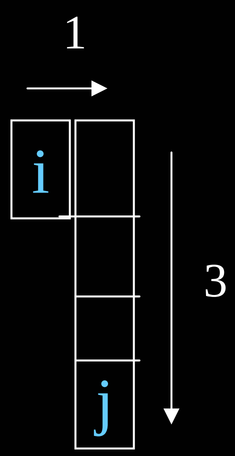
Co-occurrence Matrix Features

A co-occurrence matrix is a 2D array C in which

- Both the rows and columns represent a set of possible image values.
- $C_d(i,j)$ indicates how many times value i co-occurs with value j in a particular spatial relationship d.
- The spatial relationship is specified by a vector d = (dr,dc).

$$\boxed{\begin{array}{cccc} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 2 \end{array}}$$

gray-tone image



$$\boxed{d = (3,1)}$$

$$\begin{array}{cc} & \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \end{array} & \boxed{\begin{array}{ccc} 1 & 0 & 3 \\ 2 & 0 & 2 \\ 0 & 0 & 1 \end{array}} \end{array} \quad C_d$$

co-occurrence matrix

From C_d we can compute N_d, the normalized co-occurrence matrix, where each value is divided by the sum of all the values.

Co-occurrence Features

What do these measure?

$$Energy = \sum_i \sum_j N_d^2(i,j) \quad (7.7)$$

$$Entropy = -\sum_i \sum_j N_d(i,j) log_2 N_d(i,j) \quad (7.8)$$

$$Contrast = \sum_i \sum_j (i-j)^2 N_d(i,j) \quad (7.9)$$

$$Homogeneity = \sum_i \sum_j \frac{N_d(i,j)}{1+ \mid i-j \mid} \quad (7.10)$$

$$Correlation = \frac{\sum_i \sum_j (i-\mu_i)(j-\mu_j) N_d(i,j)}{\sigma_i \sigma_j} \quad (7.11)$$

where μ_i, μ_j are the means and σ_i, σ_j are the standard deviations of the row and column ms.

Energy measures uniformity of the normalized matrix.

But how do you choose d?

- This is actually a critical question with **all** the statistical texture methods.
- Are the "texels" tiny, medium, large, all three …?
- Not really a solved problem.

Zucker and Terzopoulos suggested using a χ^2 statistical test to select the value(s) of d that have the most structure for a given class of images.

Laws' Texture Energy Features

- Signal-processing-based algorithms use texture filters applied to the image to create filtered images from which texture features are computed.

- The Laws Algorithm
  - Filter the input image using texture filters.
  - Compute texture energy by summing the absolute value of filtering results in local neighborhoods around each pixel.
  - Combine features to achieve rotational invariance.

Law's texture masks (1)

L5	(Level)	=	[1 4 6 4 1]
E5	(Edge)	=	[-1 -2 0 2 1]
S5	(Spot)	=	[-1 0 2 0 -1]
R5	(Ripple)	=	[1 -4 6 -4 1]

- (L5) (Gaussian) gives a center-weighted local average
- (E5) (gradient) responds to row or col step edges
- (S5) (LOG) detects spots
- (R5) (Gabor) detects ripples

Law's texture masks (2)

Creation of 2D Masks

- **1D Masks are "multiplied" to construct 2D masks:** mask E5L5 is the "product" of E5 and L5 –

$$\underset{E5}{\begin{bmatrix} -1 \\ -2 \\ 0 \\ 2 \\ 1 \end{bmatrix}} \times \underset{L5}{\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}} = \underset{E5L5}{\begin{bmatrix} -1 & -4 & -6 & -4 & -1 \\ -2 & -8 & -12 & -8 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 8 & 12 & 8 & 2 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}}$$

9D feature vector for pixel

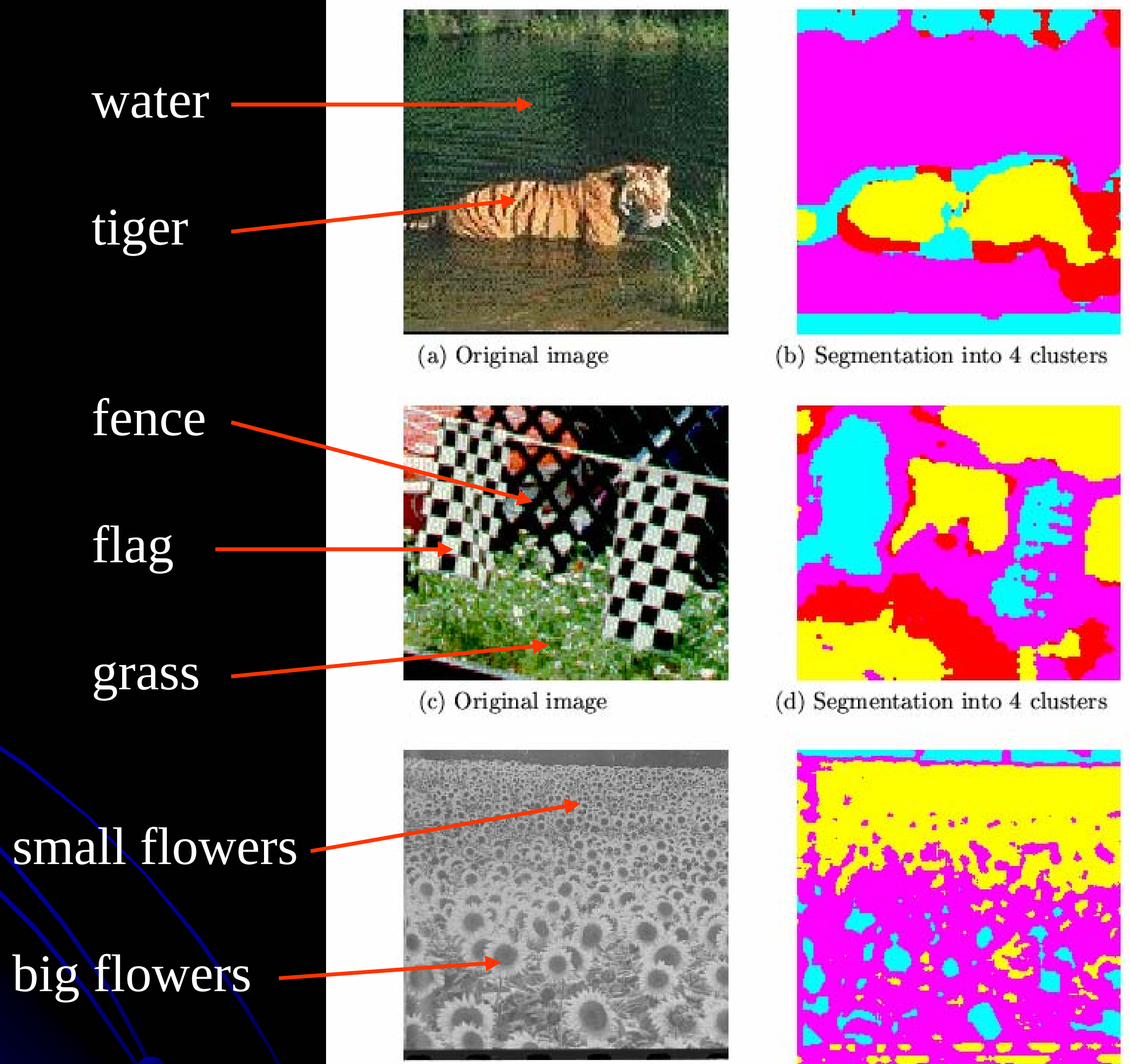
- Subtract mean neighborhood intensity from pixel
- Dot product 16 5x5 masks with neighborhood
- 9 features defined as follows:

L5E5/E5L5	L5S5/S5L5
L5R5/R5L5	E5E5
E5S5/S5E5	E5R5/R5E5
S5S5	S5R5/R5S5
R5R5	

Features from sample images

Table 7.2: Laws texture energy measures for major regions of the images of Figure 7.8.

Region	E5E5	S5S5	R5R5	E5L5	S5L5	R5L5	S5E5	R5E5	R5S5
Tiger	168.1	84.0	807.7	553.7	354.4	910.6	116.3	339.2	257.4
Water	68.5	36.9	366.8	218.7	149.3	459.4	49.6	159.1	117.3
Flags	258.1	113.0	787.7	1057.6	702.2	2056.3	182.4	611.5	350.8
Fence	189.5	80.7	624.3	701.7	377.5	803.1	120.6	297.5	215.0
Grass	206.5	103.6	1031.7	625.2	428.3	1153.6	146.0	427.5	323.6
Small flowers	114.9	48.6	289.1	402.6	241.3	484.3	73.6	158.2	109.3
Big flowers	76.7	28.8	177.1	301.5	158.4	270.0	45.6	89.7	62.9
Borders	15.3	6.4	64.4	92.3	36.3	74.5	9.3	26.1	19.5



(a) Original image (b) Segmentation into 4 clusters

(c) Original image (d) Segmentation into 4 clusters

(e) Original image (f) Segmentation into 3 clusters

Is there a neighborhood size problem with Laws?

A classical texture measure:

Autocorrelation function

- Autocorrelation function can detect repetitive patterns of texels
- Also defines fineness/coarseness of the texture
- Compare the dot product (energy) of non shifted image with a shifted image

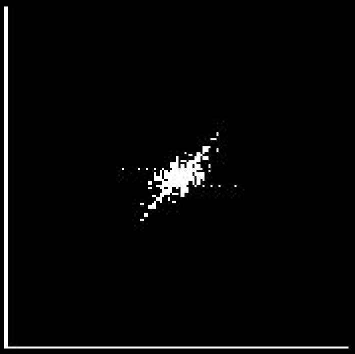
$$\rho(dr, dc) = \frac{\sum_{r=0}^{N} \sum_{c=0}^{N} I[r,c] I(r+dr, c+dc]}{\sum_{r=0}^{N} \sum_{c=0}^{N} I^2[r,c]} = \frac{I[r,c] \circ I_d[r,c]}{I[r,c] \circ I[r,c]}$$

Interpreting autocorrelation

- Coarse texture → function drops off slowly
- Fine texture → function drops off rapidly
- Can drop differently for r and c
- Regular textures → function will have peaks and valleys; peaks can repeat far away from [0, 0]
- Random textures → only peak at [0, 0]; breadth of peak gives the size of the texture

Fourier power spectrum

- High frequency power → fine texture
- Concentrated power → regularity
- Directionality → directional texture



What else?

- Gabor filters (we’ve used a lot)
- 3D textons (Leung and Malik)
- Polarity, anisotropy, and local contrast (Blobworld)