# Lecture 5 : Pattern Recognition(1)

**Special Topics in Image Processing**

# Contents

- Reference:
  - Shapiro Chap 4
  - Duda & Hart (2nd edition) Chap 2,3,10
- Features
- Principle Components Analysis (PCA)
- Bayes Decision Theory
- Gaussian Normal Density

# Pattern Recognition Basic Steps

**input image/s**

↓

**feature extraction**

↓

**feature classification**

↓

**pattern recognition results**

# Feature Extraction

- Linear features
  eg. Lines, curves

- 2D features
  eg. Texture

- Feature of features

# Feature Extraction

- A good feature should be invariant to changes in

  1. scale
  2. rotation
  3. translation
  4. illumination

# 1. Spatial Feature Extraction

- Feature Vectors & Feature Spaces
- Amplitude Features
- Binary Object Features
- Histogram Features

# 1.1 Feature Vectors and Feature Space

- Feature vectors
  - one method to represent an image, or part of an image, by finding measurements on a set of features.
  - An n-dimensional vector that contain these measurements.

- Feature spaces
  - a mathematical abstraction associated with the feature vector.
  - Also n-dimensional.

- Compare two feature vectors
  - difference or similarity

# 1.2 Amplitude Features

- The simplest and perhaps the most useful.
- Reflectivity, Transmissitivity, color, Multispectral response.
- Can be extracted easily by intensity window slicing or by the more general point transformation
- Ex)
  - X-ray image

    represent the absorption characteristics of the body masses.
  - Infrared image

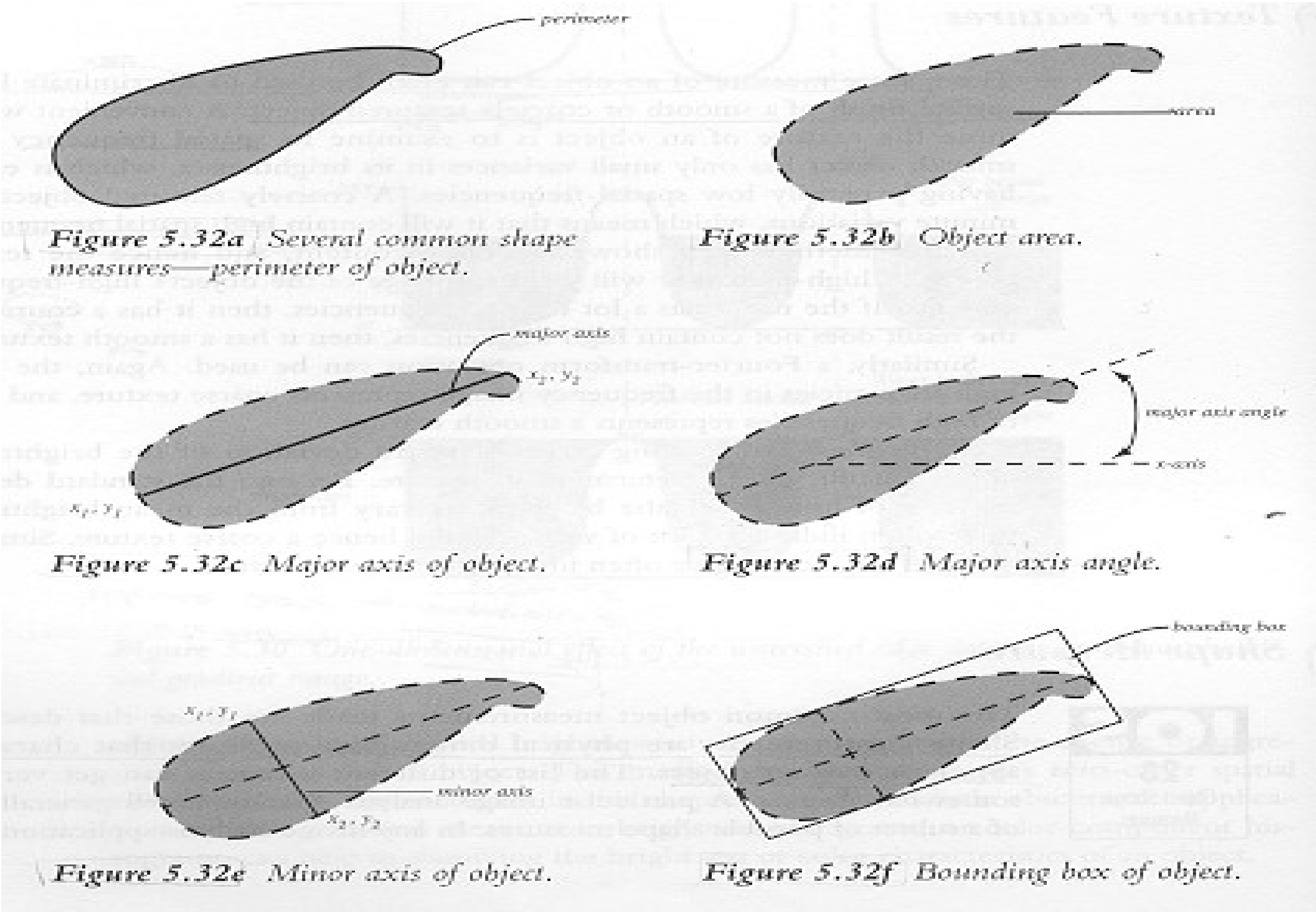
    represents temperature, which facilitates the segmentation of clouds from terrain.
  - Radar Image

    represents the radar cross section, which determine the size of the object being imaged.

# 1.3 Binary Object Features

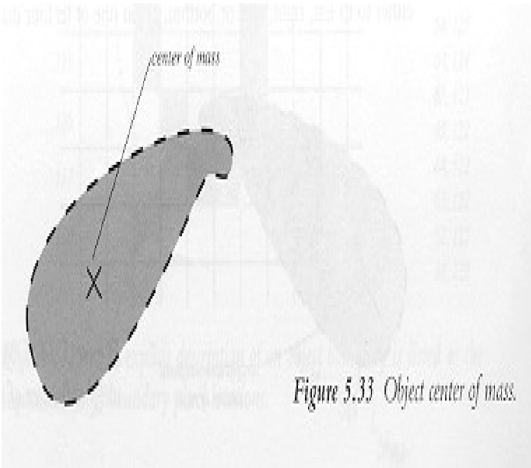
- Area
  - indicates the relative size of the object
- Center of area
  - row and column coordinates of the center of area for the object
- Axis of least second moment
  - provides information about the object’s orientation
- Perimeter
  - help us to locate it in space
  - provide us with information about the shape of the object
- Thinness
  - be used as a measure of roundness.
- Euler number
  - tells us how many closed curves the object contains.

# Feature Example (1)

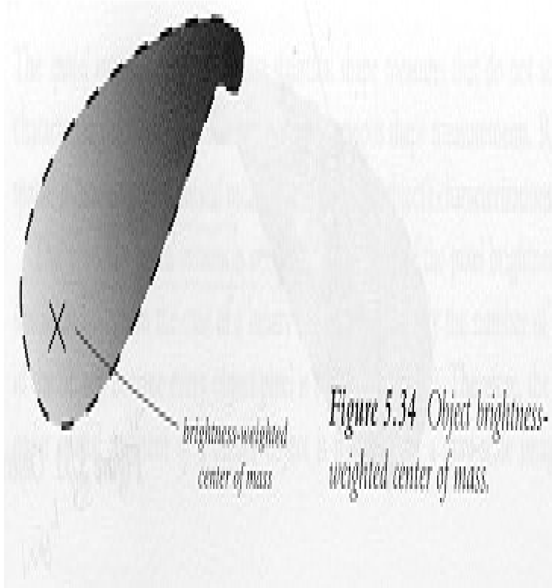


# Feature Example (2)

- Center of mass
- Center of Massx = Sum of object's x-pixel coordinates/ Number of pixels in object




- Gray Center(brightness center of mass)



# 1.4 Histogram Features[1]

$$p_u(x) \equiv \text{Prob}[u = x]$$

$$\cong \frac{\text{No. of pixels with gray level } x}{\text{Total no. of pixels in the region}}$$

• A plot of the gray-level values versus the number of pixels at that value

• The shape of the histogram provides us with information about the nature of the image

$$\text{Moments}: m_i = E[u^i] = \sum_{x=0}^{L-1} x^i p_u(x)$$

$$\text{Absolute moments}: \hat{m}_i = E[|u|^i] = \sum_{x=0}^{L-1} |x|^i p_u(x)$$

$$\text{Central moments}: \mu_i = E\{[\mu - E(u)]^i\} = \sum_{x=0}^{L-1} (x - m_1)^i p_u(x)$$

$$\text{Absolute central moments}: \hat{\mu}_i = E\{|\mu - E(u)|^i\} = \sum_{x=0}^{L-1} |x - m_1|^i p_u(x)$$

$$\text{Entropy}: H = E[-\log_2 p_u] = -\sum_{x=0}^{L-1} p_u(x) \log_2 p_u(x) \text{ bits}$$

# Histogram Features[2]

- Mean
  - average value, general brightness
- Variance
  - tells us something about the contrast
- Skew
  - measures the asymmetry about the mean in the gray-level distribution
- Energy
  - tells us something about how the gray levels are distributed
  - maximum value : 1
  - the lager this value is, the easier it is to compress the image data

# Histogram Features[3]

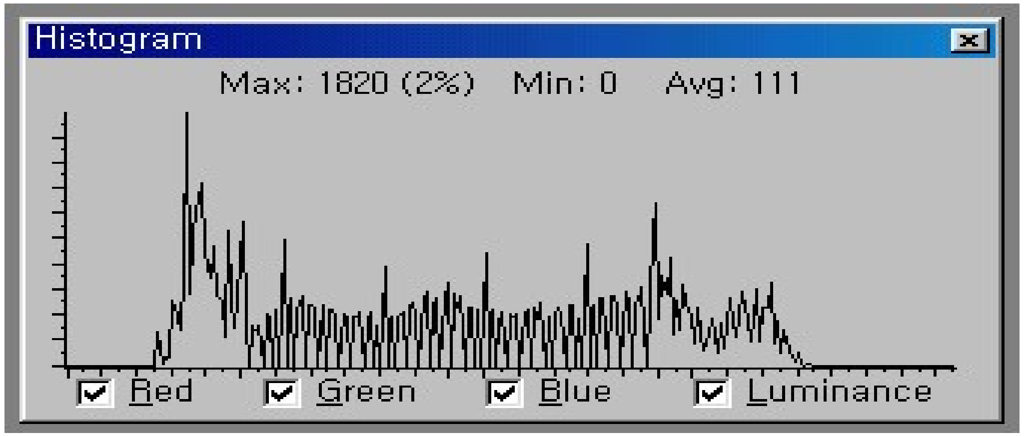
- Entropy
  - a measure that tells us how many bits we need to code the image data

- Second-order histogram features
  - based on a joint probability distribution model
  - provides statistics based on pairs of pixels and their corresponding gray levels
  - be used in identifying texture within an image

# Histogram Features[4]

$$p_u(x_1, x_2) \equiv p_{u_1, u_2}(x_1, x_2) \equiv \text{Prob}[\ u_1 = x_1, u_2 = x_2\ ]$$

$$\cong \frac{\text{No. of pairs of pixels } u_1 = x_1, u_2 = x_2}{\text{Total no. of such pairs of pixels in the region}}$$



# Transform features[1]

- **Fourier Transform**

$$F(u,v) = \frac{1}{N}\sum_{r=0}^{N-1}\sum_{c=0}^{N-1} I(r,c)\, e^{-j2\pi\frac{(ur+vc)}{N}}$$

$$MAGNITDE = |F(u,v)|$$

$$= \sqrt{[R(u,v)]^2 + [I(u,v)]^2}$$

$$PHASE = \Theta(u,v) = \tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$$

  – magnitude : simply peak value
  – phase : information about where objects are in an image, that is object's origin

# Transform features[2]

- **Lowpass Filtering**
  - Pass low frequencies and attenuate or eliminates the high-frequency information.
  - Used for image compression or for hiding effects caused by noise.
  - Blur the image, although this blur is sometimes considered an enhancement.

- **Higpass Filtering**
  - **Pass only high-frequency information.**
  - **Can be used for edge enhancement.**

# Transform features[3]

- Bandpass & Bandreject Filters
  - Are specified by two cutoff frequencies $\rightarrow$ low and high cutoff
  - typically used in image restoration, enhancement and compression

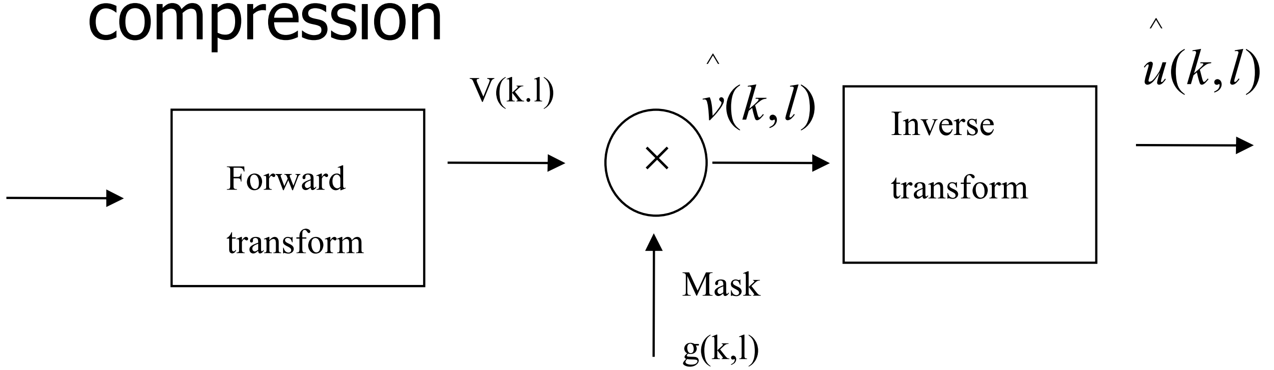
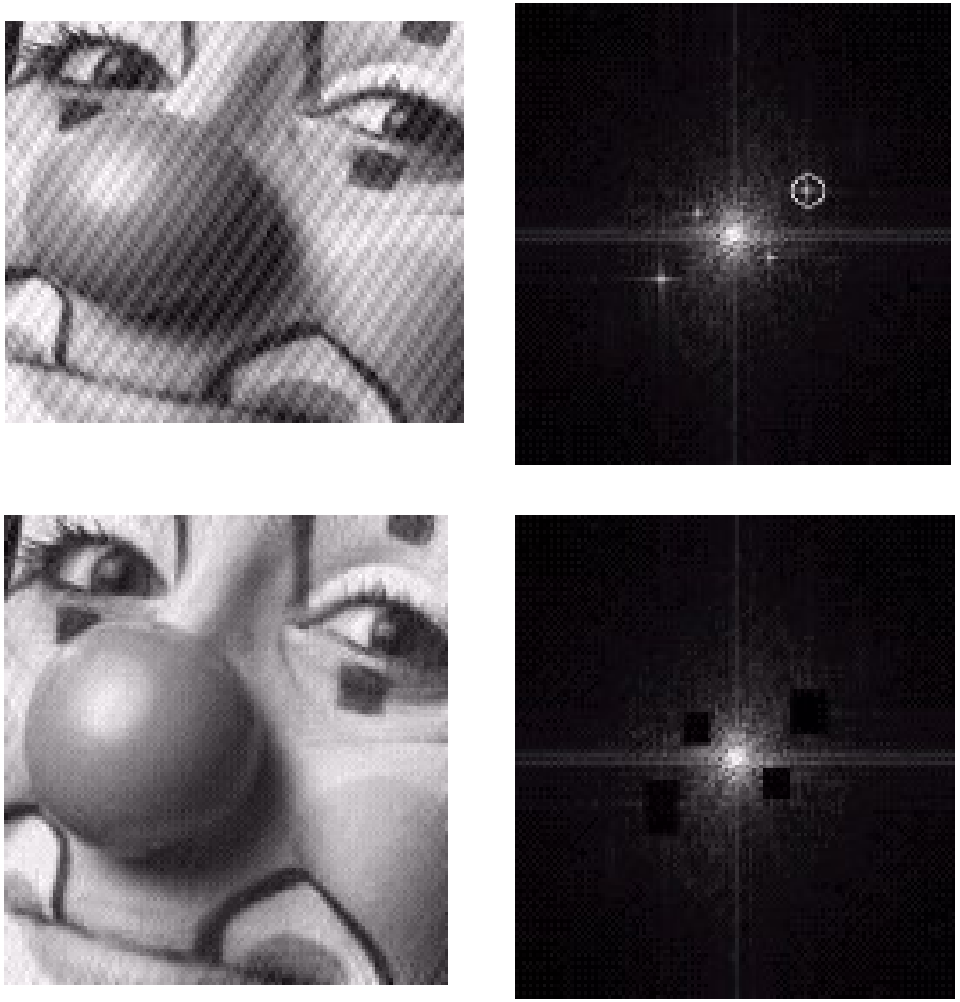



Fig.6 Transform feature extraction

# Fourier Transform Example[4]



(a) Before processing the image is marred by "noise".

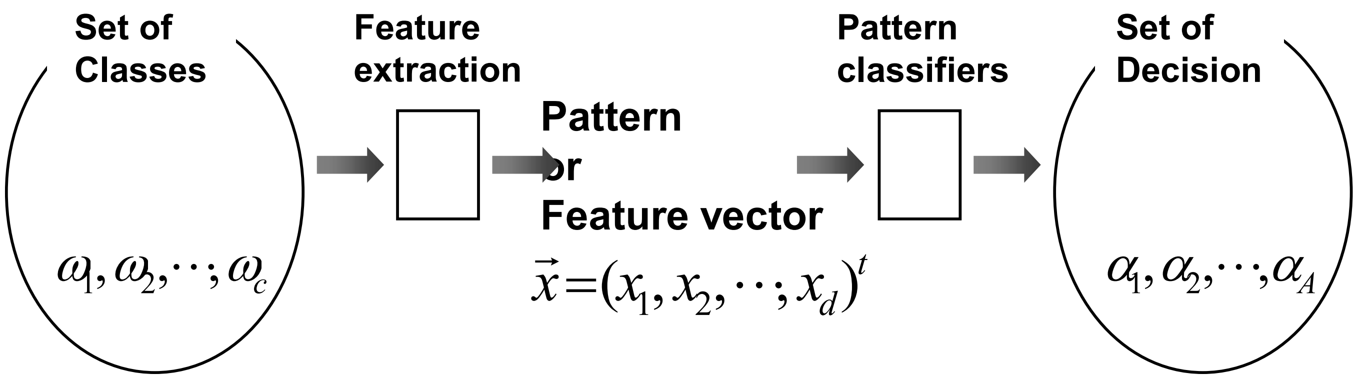
(b) An DFT of the image is displayed. Due to its regularity, the noise pattern stands out as four spikes. One of the spikes has been marked inside an AOI (Area of Interest) preparatory to its deletion.

(c) The image is cleared.

(d) The four noise spikes have been deleted from the DFT so that when the corrected DFT is applied.

# 2. Introduction to Classification

- Feature extraction → classification
- Classification & segmentation
  - Closely related
  - Classification can lead to segmentation, and vice-versa




**C : # of Class**
**A : # of Decisions**
**d : Dimension of feature vector**

- Hole in the Object?

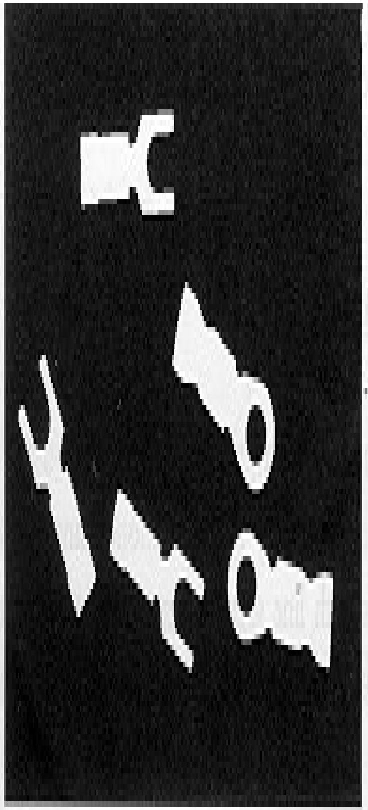
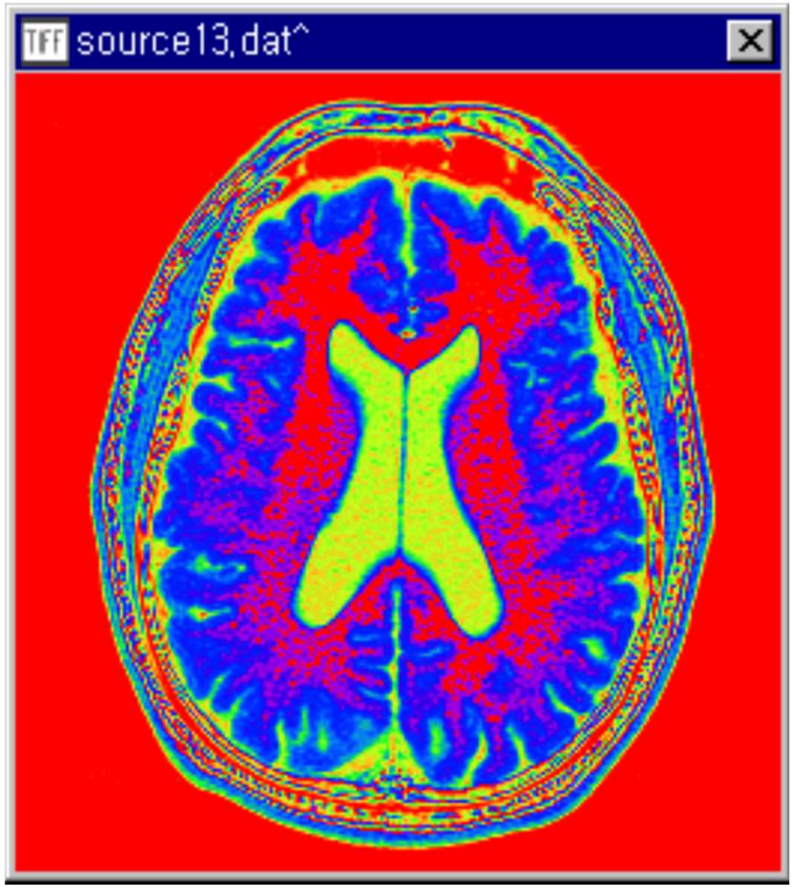


*Figure 5.43 Binary image of ring and spade terminals.*

MRI Classification

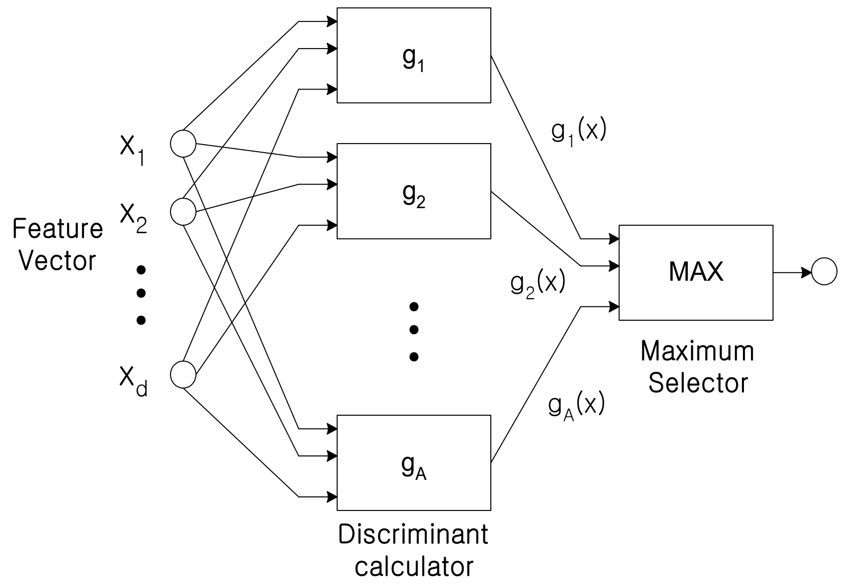


# Supervised vs. Unsupervised Classification

- Supervised
  - Patterns from each class are available
  - The system is "taught" to recognize patterns by means of various adaptive schemes
  - The essentials of this approach
    - A set of training patterns of known classification
    - The implementation of an appropriate learning procedure
- Unsupervised
  - Only a set of training patterns of unknown classification are available
  - Learning without a teacher

# Distribution-free classification

- Do not require a priori probability distribution
  - Decision tree classification
  - Minimum distance classifier
  - K-nearest neighbor classifier
  - Linear discriminant function
- Decision rule
  - Discriminant function




$$\boxed{g_k(\mathbf{x}) > g_i(\mathbf{x}) \qquad k \neq i \Leftrightarrow \mathbf{x} \in S_k}$$

$$S_k = \text{Objects(pattern classes)}$$

# Statistical classification

- Bayes decision rule for minimum risk classifier
- Minimum error rate classifier
- Estimation of probability density function

# Bayes' Decision Rule

- $P(\omega_i)$ : a priori probability
- $P(\omega_i|\vec{x})$ : a posteriori probability
- $P(\vec{x}|\omega_i)$ : class conditional pdf

$$P(\omega_i|\vec{x}) = \frac{P(\omega_i, \vec{x})}{P(\vec{x})}$$

$$= \frac{P(\vec{x}|\omega_i)\, P(\omega_i)}{P(\vec{x})}$$

$$P(\vec{x}) = \sum_{i=1}^{c} P(\vec{x}|\omega_i) P(\omega_i)$$

# Bayes' decision rule(Two Class Problem)

- 2 class problem ($i=1,2$)
  - a pixel belong to class 1 if $p(x|1)p(1) > p(x|2)p(2)$
  - a pixel belong to class 2 if $p(x|1)\ p(1) < p(x|2)p(2)$

# Unsupervised learning or clustering

- Cluster
  - A set of points in the feature space for which their local density is large compared to the density in the surrounding region
- Clustering
  - Useful for image segmentation and for classification of raw data to establish classes and prototypes
  - Useful for vector quantization technique

# General Clustering Algorithm

- Assume that we want to classify N samples $X_1, X_2 \cdots X_N$
- Each sample is to be placed into one of L classes, $\omega_1, \omega_2, \cdots \omega_L$
- The class to which the ith sample is assigned to $\omega_{ki} (i = 1, \cdots N)$
- A classification is $\Omega$ a vector made up of the $\omega_{ki}$ and

$$\Omega = [\omega_{ki} \cdots \varpi_{kN}]^T$$

- Clustering Criterion *J* is a function of $\Omega$ and X* and can be written

$$J = J(\omega_{k1} \cdots \omega_{kN}; X_1 \cdots X_N) = J(\Omega; X^*)$$

- The best classification $\Omega$ satisfies max(J).

# Similarity measure approach

- A general clustering algorithm is based on split and merge idea

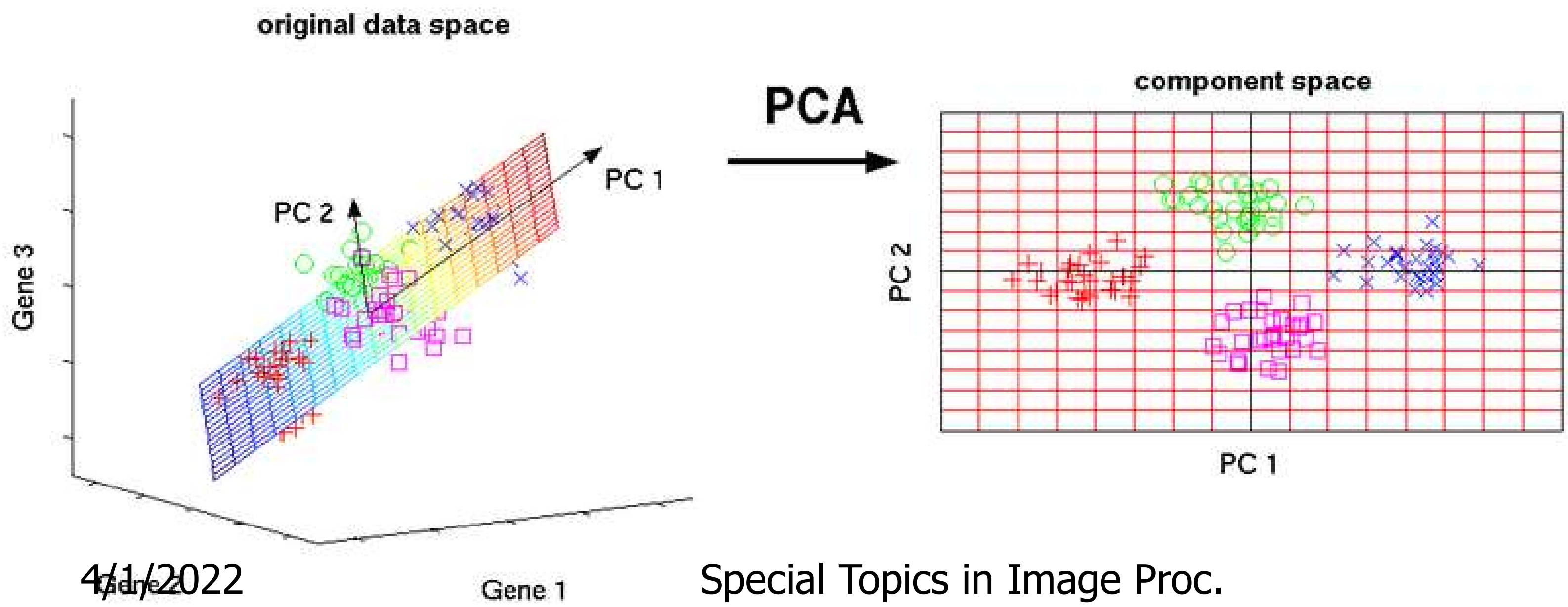
- Basic
  - The input vector are partitioned into subsets by using similarity measure
  - Each partition is tested to check
  - Subsets that are not sufficiently distinct are merged

# Iterative(Isodata) method

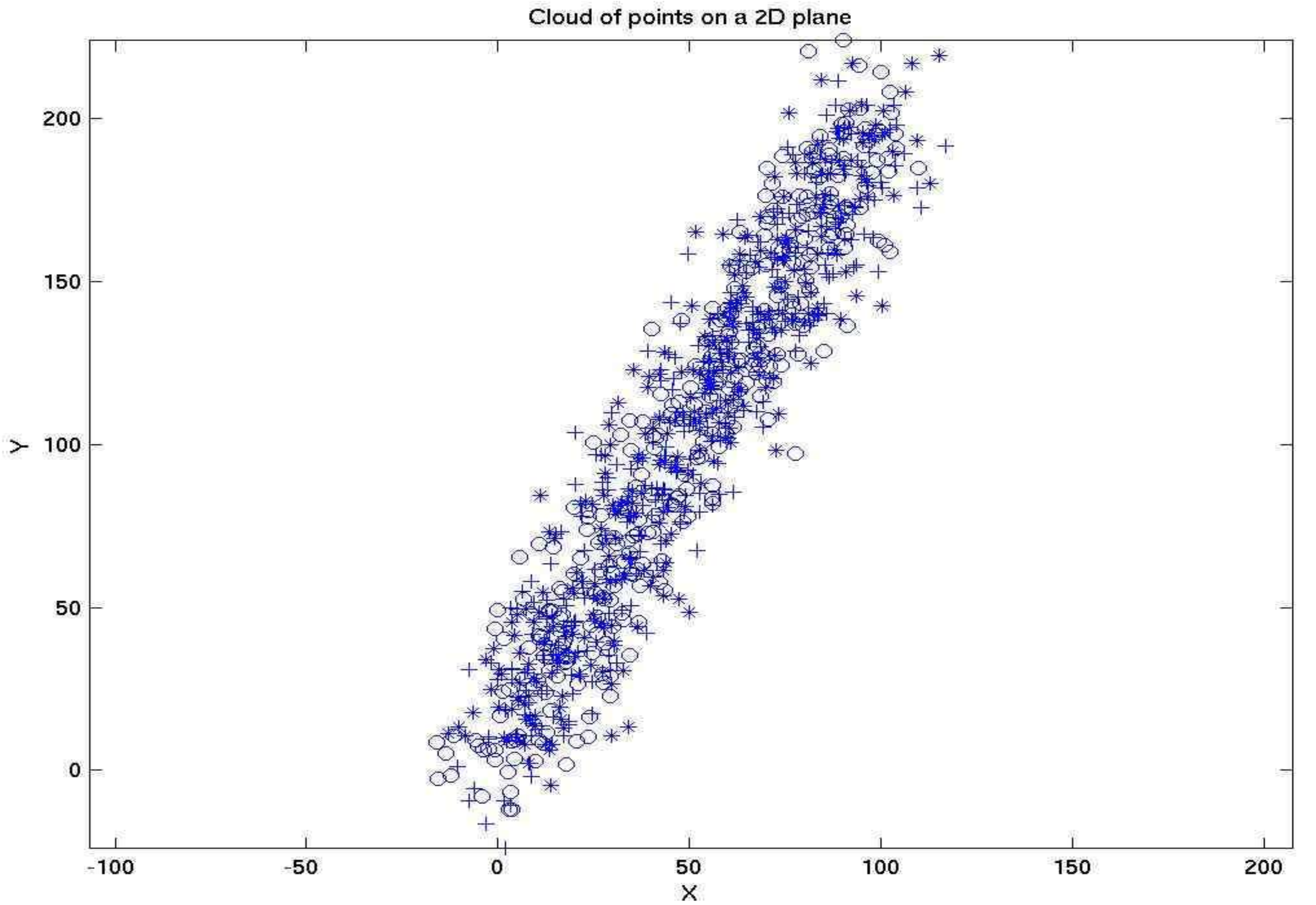
- Basic ISODATA algorithm
  - Choose some initial values for the means $\hat{\mu}_1, \hat{\mu}_2 \cdots \hat{\mu}_c$
  - Classify the ***n*** samples by assigning them to the class of the closest mean
  - Recompute the means as the average of the samples in their class.
  - If any mean changed value, back to second procedure(reclassifying). Otherwise stop.

# 3. Principle Components Analysis (Karhunen-Loeve Transform)

• Principal component analysis (PCA) is a procedure for reducing the dimensionality of the variable space by representing it with a few orthogonal (uncorrelated) variables that capture most of its variability.



# Suppose we are shown a set of points:



# Principle Components Analysis(PCA)

- We want to answer any of the following questions:

  1. What is the trend (direction) of the cloud of points
  2. Which is the direction of maximum variance (dispersion)
  3. Which is the direction of minimum variance (dispersion)

- Supposing I am only allowed to use a 1D description for this set of 2D points, i.e. <u>using 1 coordinate instead of 2 coordinates (x,y)</u>. How should I represent the points in such a way that the overall error is minimized ?
  --- this is data compression problem

  All these questions can be answered using
  Principle Components Analysis (PCA)

# Principal Component Analysis (PCA)

Suppose we have N feature vectors $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_N\}$

We take the mean of the feature vectors $\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$

Form the covariance matrix $C = \sum_{i=1}^{N} \mathbf{y}_i \mathbf{y}_i^T$

where $\mathbf{y}_i = \mathbf{x}_i - \bar{\mathbf{x}}$

Find the eigenvalues and eigenvectors of covariance matrix

$$C\,\mathbf{e}_i = \lambda_i\,\mathbf{e}_i$$

where the $\mathbf{e}_i$ is a principal component with eigenvalue $\lambda_i$

# Principal Component Analysis (PCA)

- One problem with PCA when applied directly to the covariance matrix of the images is that the dimension of covariance matrix is huge!

- For example, for a 256x256 image, X will be of dimension 65536x1. So the covariance matrix of X will be of size 65536x65536 !!!

- To compute the eigenvector of such a huge matrix is clearly inconceivable.

- To solve this problem, Murakami and Kumar proposed a technique in the following paper:

  Murakami, H. and Vijaya Kumar, B.V.K., "Efficient Calculation of Primary Images from a Set of Images", IEEE Trans. Pattern Analysis and Machine Intelligence, vol. PAMI-4, No. 5, Sep 1982.

## To begin, we note that our goal is to find the eigenvectors and eigenvalues of the covariance matrix

$$C = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \overline{\mathbf{x}})(\mathbf{x}_i - \overline{\mathbf{x}})^T = \frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\,\mathbf{y}_i^T$$

We will see later that instead of working directly on $C$, which is a huge matrix, we can find the eigenvectors and eigenvalues of an NxN matrix and then use it to deduce the eigenvectors and eigenvalues of $C$ .

**To see how this can be done, we first note that**

$$\mathbf{y}_i \underbrace{\mathbf{y}_i^T \mathbf{e}}_{\text{scalar}} = (\mathbf{y}_i^T \mathbf{e})\,\mathbf{y}_i$$

## Therefore, if e is an eigenvector of C, then

$$\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\mathbf{y}_i^T\mathbf{e} = \lambda\,\mathbf{e}$$

$$\Rightarrow \quad \frac{1}{N}\sum_{i=1}^{N}(\mathbf{y}_i^T\mathbf{e})\mathbf{y}_i = \lambda\,\mathbf{e}$$

$$\Rightarrow \quad \frac{1}{N}\sum_{i=1}^{N}a_i\mathbf{y}_i = \lambda\,\mathbf{e}$$

Let $\quad b_i = \dfrac{a_i}{\lambda\, N}$

then $\quad \mathbf{e} = \sum_{i=1}^{N} b_i\mathbf{y}_i$

So if $\mathbf{e}$ is an eigenvector of $C$, $\mathbf{e}$ can be expressed as a linear combination of the vectors $\{\mathbf{y}_i\}$.

**Therefore,** $$\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\mathbf{y}_i^T\mathbf{e} = \lambda\sum_{i=1}^{N}b_i\mathbf{y}_i$$

$$\Rightarrow \quad \frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\mathbf{y}_i^T\left(\sum_{j=1}^{N}b_j\mathbf{y}_j\right) = \lambda\sum_{i=1}^{N}b_i\mathbf{y}_i$$

$$\Rightarrow \quad \frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\sum_{j=1}^{N}b_j\,\mathbf{y}_i^T\mathbf{y}_j = \lambda\sum_{i=1}^{N}b_i\mathbf{y}_i$$

$$\Rightarrow \quad \frac{1}{N}\sum_{j=1}^{N}\mathbf{y}_i^T\mathbf{y}_j\,b_j = \lambda\,b_i$$

Therefore, the vector $\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}$ is the eigenvector of the matrix

with (i,j) elements given by $\left\{\frac{1}{N}\mathrm{y}_i^T\mathrm{y}_j\right\}$

So the eigenvector of the matrix $\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\mathbf{y}_i^T$ , i.e. $e$, is given by

$$\mathbf{e} = \sum_{i=1}^{N} b_i \mathbf{y}_i$$

$$= \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_N\end{bmatrix}\mathbf{b}$$

Note that the eigenvalues of the matrix with elements $\left\{\frac{1}{N}\mathbf{y}_i^T\mathbf{y}_j\right\}$ is the same as the eigenvalues of the matrix $\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\mathbf{y}_i^T$

In conclusion, the task of finding the eigenvectors of a huge matrix of size $M$ x $M$ (where $M$ is the number of pixels in an image), is reduced to the task of finding the eigenvectors of a NxN matrix.

# Summary(1)

Suppose we have N feature vectors $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_N\}$

We want to find the principle components. Since a direct computation of covariance matrix is not feasible due to huge dimension of covariance matrix, we do the following steps instead:

**Step 1:**

We take the mean of the feature vectors $\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$

**Step 2:**

Subtract the mean $\bar{\mathbf{x}}$ from each of the feature vectors $\mathbf{x}_i$

$$\mathbf{y}_i = \mathbf{x}_i - \bar{\mathbf{x}}$$

# Summary(2)

**Step 3:**

Form the inner product matrix

$$M = \begin{bmatrix} \mathbf{y}_1^T \mathbf{y}_1 & \mathbf{y}_1^T \mathbf{y}_2 & \cdots & \mathbf{y}_1^T \mathbf{y}_N \\ \mathbf{y}_2^T \mathbf{y}_1 & \mathbf{y}_2^T \mathbf{y}_2 & \cdots & \mathbf{y}_2^T \mathbf{y}_N \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{y}_N^T \mathbf{y}_1 & \mathbf{y}_N^T \mathbf{y}_2 & \cdots & \mathbf{y}_N^T \mathbf{y}_N \end{bmatrix}$$

**Step 4:**

Find the eigenvectors and eigenvalues of M

$$\mathbf{M}\,\mathbf{b} = \alpha\;\mathbf{b}$$

# Summary(3)

**Step 5:**

If $[\mathbf{e}_1\ \mathbf{e}_2\ \mathbf{e}_3 \cdots \mathbf{e}_N]$ and $[\lambda_1\ \lambda_2\ \lambda_3 \cdots \lambda_N]$

are the principle components and the corresponding

eigenvalues respectively of $[\mathbf{x}_1\ \mathbf{x}_2\ \mathbf{x}_3 \cdots \mathbf{x}_N]$

then

$$\mathbf{e}_i = [\mathbf{y}_1\ \mathbf{y}_2\ \mathbf{y}_3 \cdots \mathbf{y}_N]\mathbf{b}_i$$

$$\lambda_i = \alpha_i$$

# Pattern Recognition (2)

# Pattern Recognition

Pattern recognition is:

1. A research area in which patterns in data are found, recognized, discovered, …whatever.

2. A catchall phrase that includes

   - classification
   - clustering
   - data mining
   - ….

# Two Schools of Thought

1. Statistical Pattern Recognition

   The data is reduced to vectors of numbers and statistical techniques are used for the tasks to be performed.

2. Structural Pattern Recognition

   The data is converted to a discrete structure (such as a grammar or a graph) and the techniques are related to computer science subjects (such as parsing and graph matching).

# In this class

1. How should objects to be classified be represented?

2. What algorithms can be used for recognition (or matching)?

3. How should learning (training) be done?

# Classification in Statistical PR

- A class is a set of objects having some important properties in common

- A feature extractor is a program that inputs the data (image) and extracts features that can be used in classification.

- A classifier is a program that inputs the feature vector and assigns it to one of a set of designated classes or to the “reject” class.

# Feature Vector Representation

- X=[x1, x2, … , xn], each xj a real number
- xj may be an object measurement
- xj may be count of object parts
- Example: object rep. [#holes, #strokes, moments, …]

```
00000000010000000000
00000000110000000000
00000000101000000000
00000001000010000000
00000010000010000000
00000100000001000000
00001000000000100000
00001100111111110000
00001111110000010000
00011000000000011000
00010000000000001100
00110000000000000100
00110000000000000110
00100000000000000010
00100000000000000010
01100000000000000010
01000000000000000000
00000000000000000000
```

```
00000000011110000000
00000001100001100000
00000011000000110000
00001100000000011000
00001000000000001000
00001100000000010000
00000111000000100000
00000011100111100000
00000000111100000000
00000011000111000000
00001100000000110000
00011000000000011000
00110000000000001000
00100000000000001100
00010000000000011000
00011000000000010000
00001000000000110000
00000011111110000000
```

# Possible features for char recog.

| (class) character | area | height | width | number #holes | number #strokes | (cx,cy) center | best axis | least inertia |
|---|---|---|---|---|---|---|---|---|
| 'A' | medium | high | 3/4 | 1 | 3 | 1/2,2/3 | 90 | medium |
| 'B' | medium | high | 3/4 | 2 | 1 | 1/3,1/2 | 90 | large |
| '8' | medium | high | 2/3 | 2 | 0 | 1/2,1/2 | 90 | medium |
| '0' | medium | high | 2/3 | 1 | 0 | 1/2,1/2 | 90 | large |
| '1' | low | high | 1/4 | 0 | 1 | 1/2,1/2 | 90 | low |
| 'W' | high | high | 1 | 0 | 4 | 1/2,2/3 | 90 | large |
| 'X' | high | high | 3/4 | 0 | 2 | 1/2,1/2 | ? | large |
| '*' | medium | low | 1/2 | 0 | 0 | 1/2,1/2 | ? | large |
| '-' | low | low | 2/3 | 0 | 1 | 1/2,1/2 | 0 | low |
| '/' | low | high | 2/3 | 0 | 1 | 1/2,1/2 | 60 | low |

# Some Terminology

- Classes: set of m known categories of objects

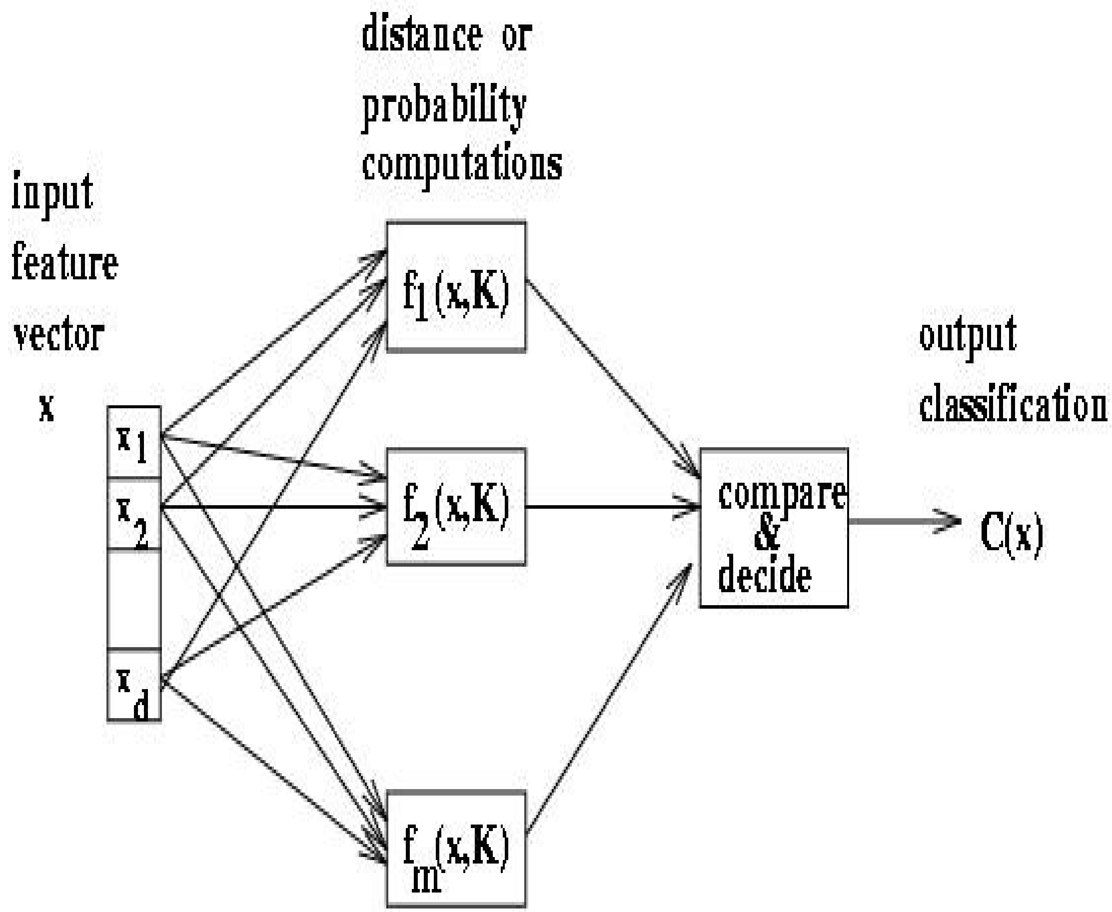
  (a) might have a known description for each

  (b) might have a set of samples for each
- Reject Class:

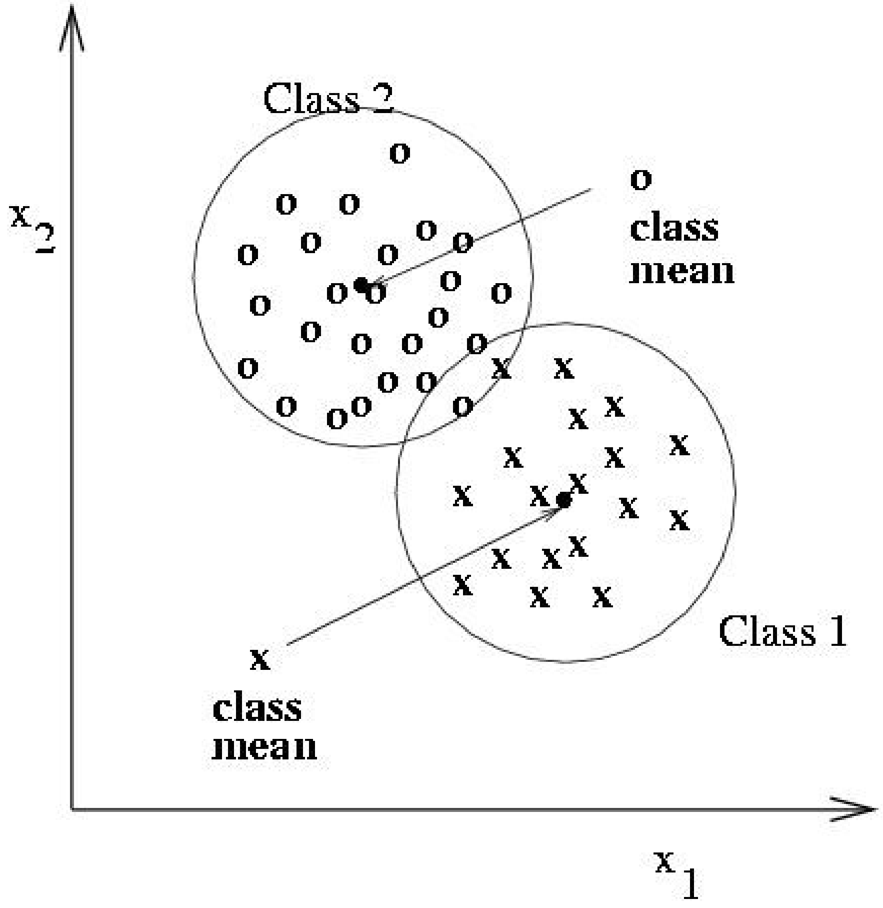
  a generic class for objects not in any of
  the designated known classes
- Classifier: Assigns object to a class based on features

# Discriminant functions

- Functions f(x, K) perform some computation on feature vector x
- Knowledge K from training or programming is usec
- Final stage determines class

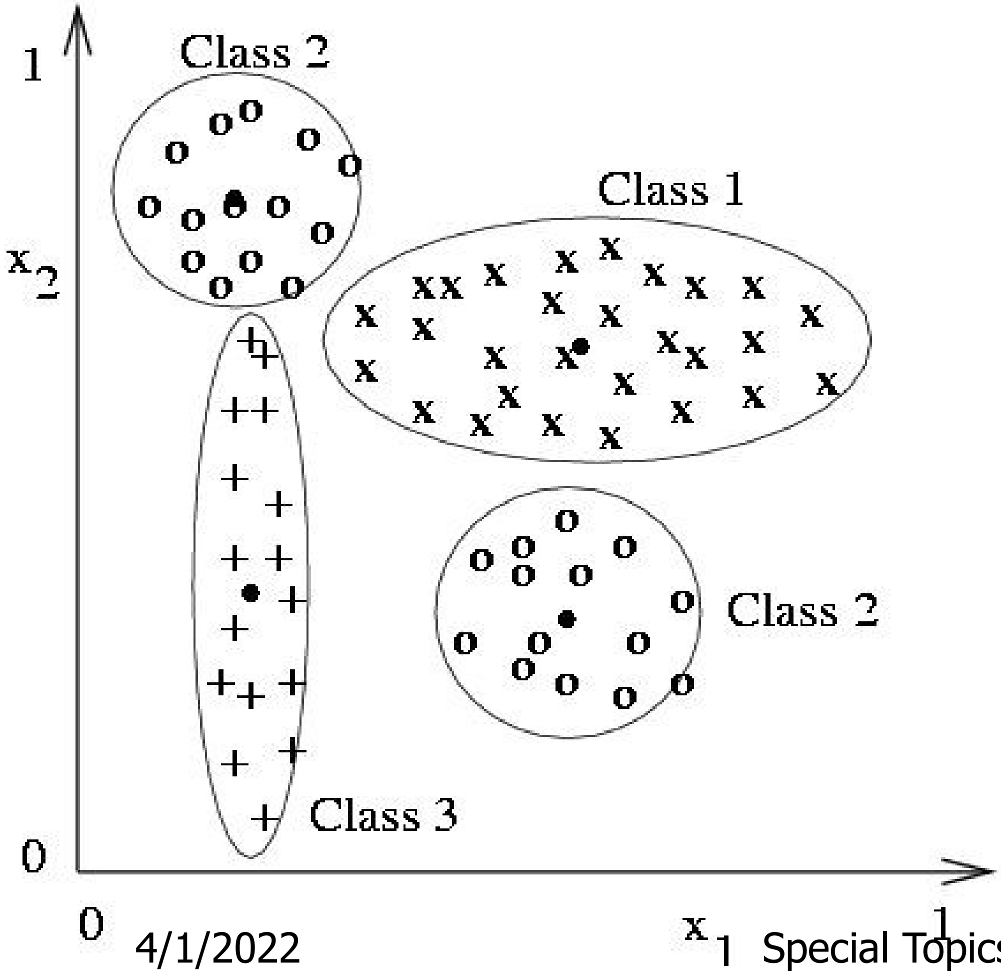


# Classification using nearest class mean




- Compute the Euclidean distance between feature vector X and the mean of each class.
- Choose closest class, if close enough (reject otherwise)

# Nearest mean might yield poor results with complex structure




Class 2 has two modes; where is its mean?

But if modes are detected, two subclass mean vectors can be used

# Scaling coordinates by std dev

We can compute a modified distance from feature vector $\mathbf{x}$ to class mean vector $\mathbf{x_c}$ by scaling by the spread, or *standard deviation*, $\sigma_i$ of class $c$ along each dimension $i$.

**scaled Euclidean distance** from $\mathbf{x}$ to class mean $\mathbf{x_c}$ :

$$\| \mathbf{x} - \mathbf{x_c} \| = \sqrt{\Sigma_{i=1,d}((\mathbf{x}[i] - \mathbf{x_c}[i])/\sigma_i)^2}$$

In the previous 3 class problem, an observed $X$ near the top of the Class 3 distribution will scale to be closer to the mean of Class 3 than to the mean of Class 2. Without scaling, $X$ would be closer to the mean of Class 2.

# Nearest Neighbor Classification

- Keep all the training samples in some efficient look-up structure.
- Find the nearest neighbor of the feature vector to be classified and assign the class of the neighbor.
- Can be extended to K nearest neighbors.

# Bayesian decision-making

- classify into class $\omega_i$ that is most likely based on observations $\mathbf{X}$
- In order to compute the likelihoods given the measurement $\mathbf{X}$, the following distributions are needed.

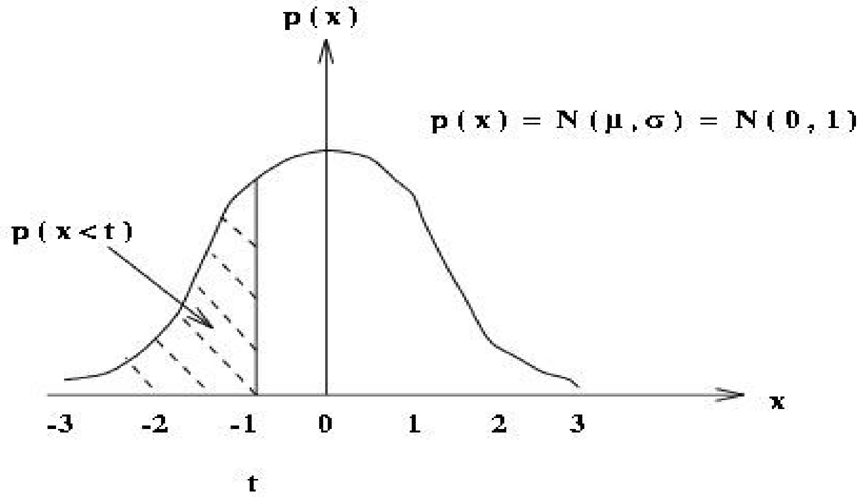
$$\begin{aligned}
&\textbf{class conditional distribution}: \quad p(x|\omega_i) \ \mathit{for\ each\ class} \quad \mathit{omega}_i &(1)\\
&\textbf{a priori probability}: \quad P(\omega_i) \ \mathit{for\ each\ class}\ \omega_i &(2)\\
&\textbf{unconditional distribution}: \quad p(x) &(3)
\end{aligned}$$

- use Bayes rule if all of the classes $\omega_i$ are disjoint

$$P(\omega_i|x) = \frac{p(x|\omega_i)P(\omega_i)}{p(x)} = \frac{p(x|\omega_i)P(\omega_i)}{\Sigma_{i=1,m}\, p(x|\omega_i)P(\omega_i)} \qquad (4)$$

# Normal distribution

- 0 mean and unit std deviation
- Table enables us to fit histograms and represent them simply
- New observation of variable x can then be translated into probability



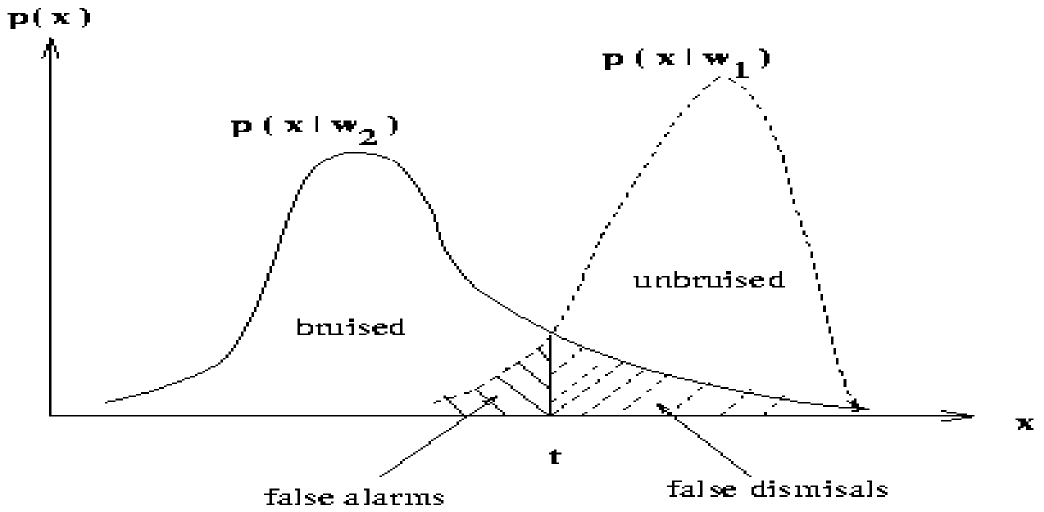

| t | p(x<t) | t | p(x<t) | t | p(x<t) |
|---|---|---|---|---|---|
| -3.0 | 0.0014 | -2.0 | 0.0227 | -1.0 | 0.1587 |
| -2.9 | 0.0019 | -1.9 | 0.0287 | -0.9 | 0.1841 |
| -2.8 | 0.0026 | -1.8 | 0.0359 | -0.8 | 0.2119 |
| -2.7 | 0.0035 | -1.7 | 0.0446 | -0.7 | 0.2420 |
| -2.6 | 0.0047 | -1.6 | 0.0548 | -0.6 | 0.2743 |
| -2.5 | 0.0062 | -1.5 | 0.0668 | -0.5 | 0.3085 |
| -2.4 | 0.0082 | -1.4 | 0.0808 | -0.4 | 0.3446 |
| -2.3 | 0.0107 | -1.3 | 0.0968 | -0.3 | 0.3821 |
| -2.2 | 0.0139 | -1.2 | 0.1151 | -0.2 | 0.4207 |
| -2.1 | 0.0179 | -1.1 | 0.1357 | -0.1 | 0.4602 |
| | | | | 0.0 | 0.5000 |

# Parametric Models can be used

**Parametric Models for Distributions**

A normal distribution characterized by mean $\mu$ and standard deviation $\sigma$ is defined as follows.

$$p(x) = N(\mu, \sigma)(x) = \frac{1}{\sqrt{2\pi}\sigma} exp[-\frac{1}{2}(\frac{x-\mu}{\sigma})^2]$$

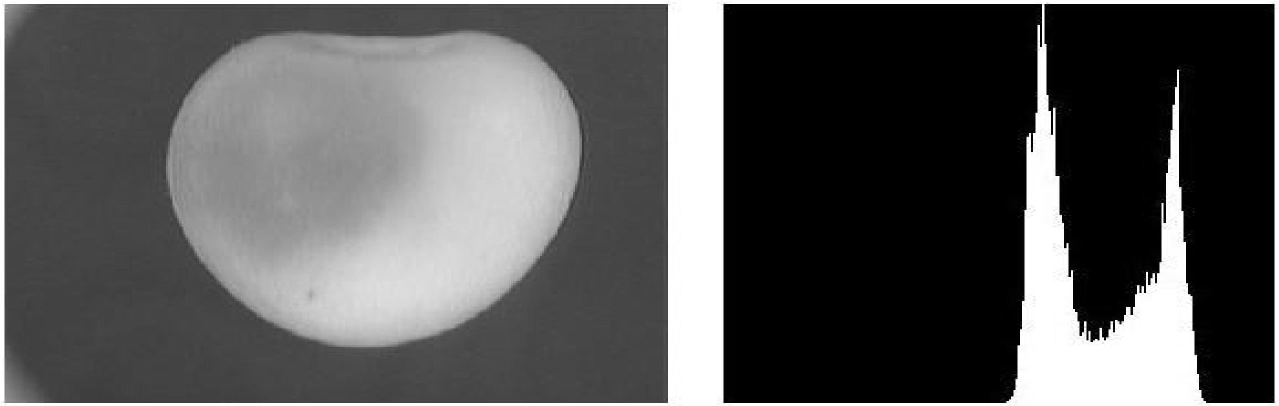



Distributions for intensity measurement $x$ conditioned on whether $x$ is taken from an unbruised or bruised cherry.

# Cherry with bruise

- Intensities at about 750 nanometers wavelength
- Some overlap caused by cherry surface turning away

Cherry with bruise and histogram

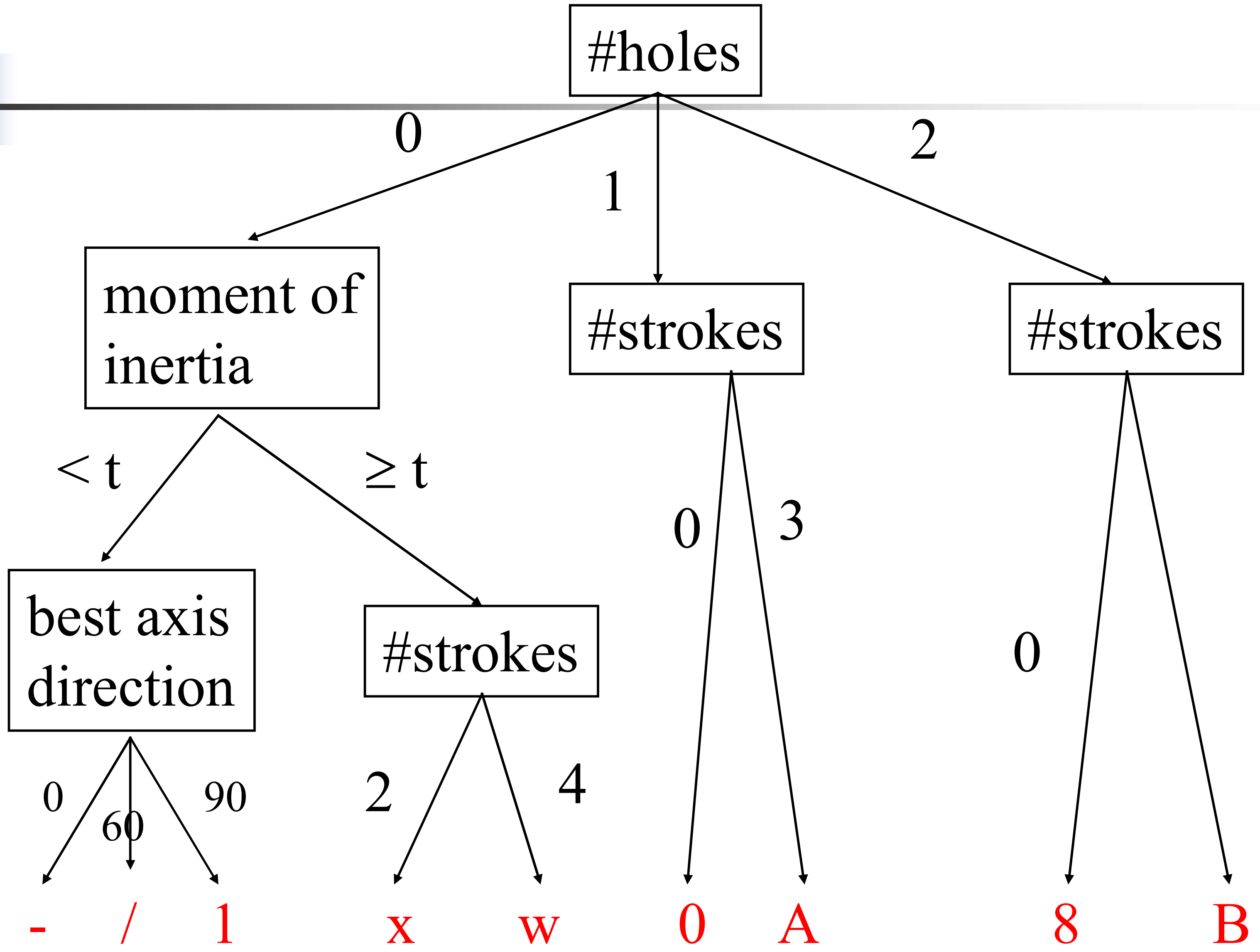


Cherry with bruise and histogram of intensities at about 750 nanometers wavelength. Some of the overlap is explained by unbruised surface that curves away from the viewpoint and thus appears dark like bruised tissue facing the viewpoint.

# Classifiers often used in CV

- Decision Tree Classifiers
- Artificial Neural Net Classifiers
- Bayesian Classifiers and Bayesian Networks (Graphical Models)
- Support Vector Machines

# Decision Trees



# Decision Tree Characteristics

1. Training : How do you construct one from training data?
   Entropy-based Methods

2. Strengths : Easy to Understand

3. Weaknesses : Overtraining

# Entropy

Given a set of training vectors S, if there are c classes,

$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2(p_i)$$

Where pi is the proportion of category i examples in S.

If all examples belong to the same category, the entropy is 0.

If the examples are equally mixed (1/c examples of each class), the entropy is a maximum at 1.0.

e.g. for c=2, $-.5 \log_2 .5 - .5 \log_2 .5 = -.5(-1) - .5(-1) = 1$

# Information Gain

The information gain of an attribute A is the expected reduction in entropy caused by partitioning on this attribute.

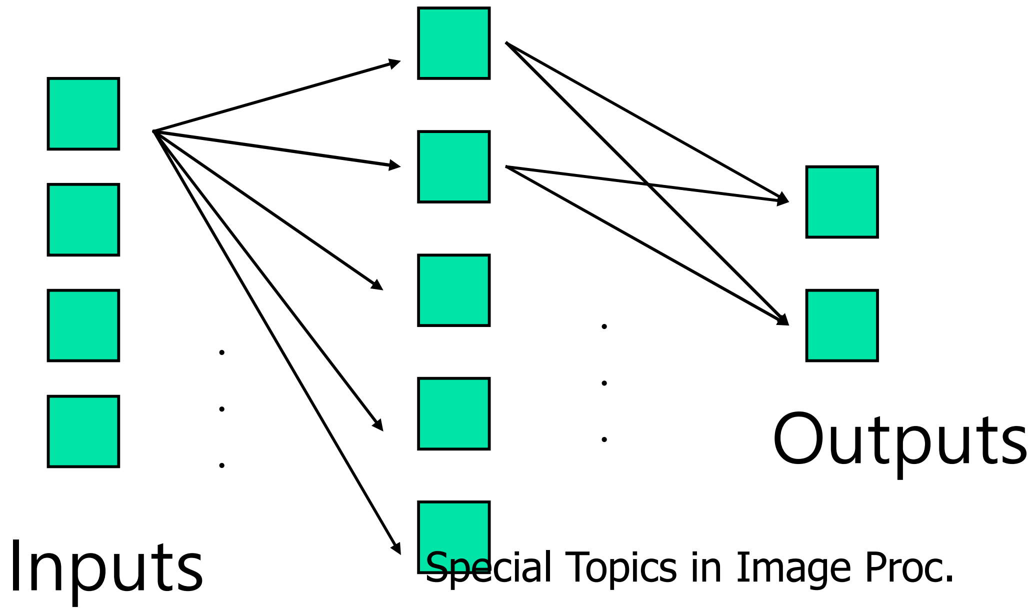
$$\text{Gain}(S,A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|Sv|}{|S|} \text{Entropy}(Sv)$$

where Sv is the subset of S for which attribute A has value v.

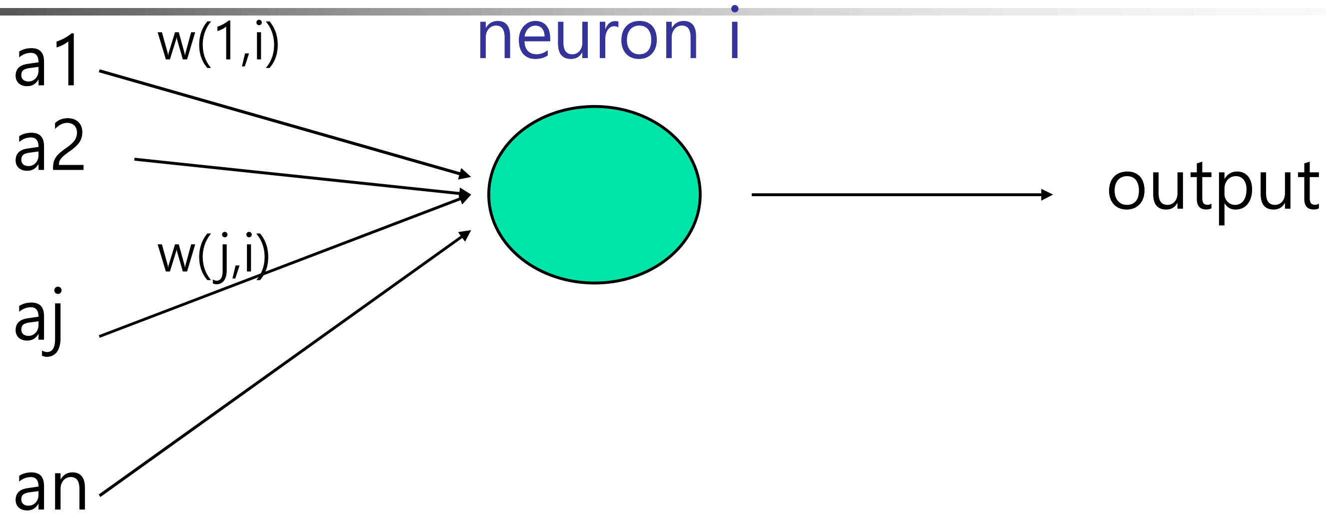
Choose the attribute A that gives the maximum information gain.

# Artificial Neural Nets

- Artificial Neural Nets (ANNs) are networks of artificial neuron nodes, each of which computes a simple function.
- An ANN has an input layer, an output layer, and "hidden" layers of nodes.



# Node Functions




$$\boxed{\text{output} = g\left(\sum aj * w(j,i)\right)}$$

Function g is commonly a step function, sign function, or sigmoid function (see text).

# Neural Net Learning

That's beyond the scope of this text; only simple feed-forward learning is covered.

The most common method is called back propagation.

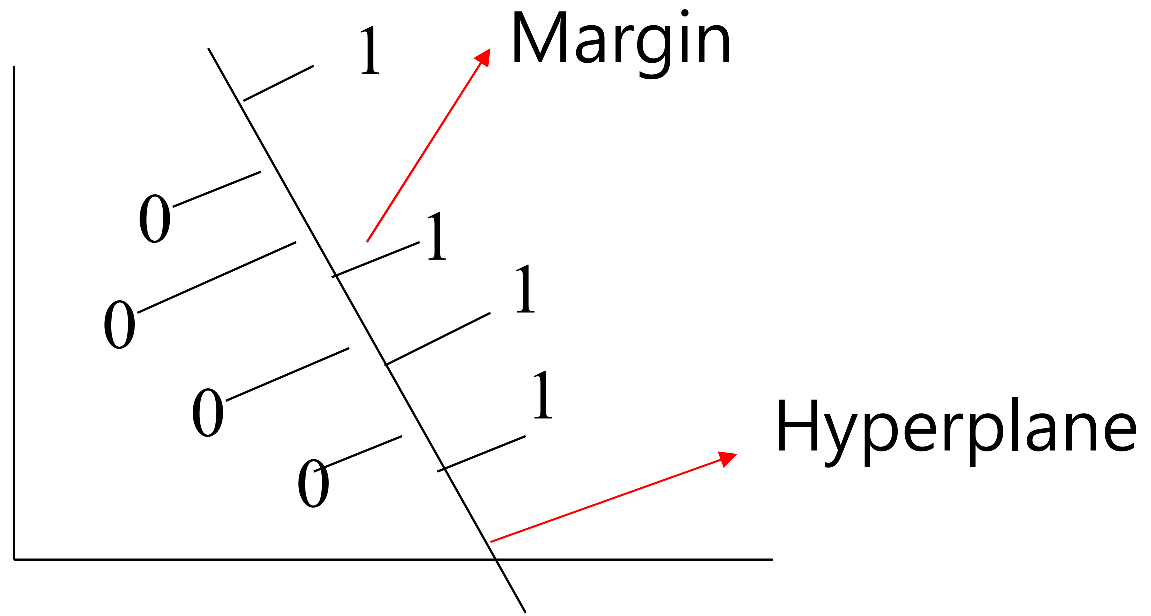
# Support Vector Machines (SVM)

Support vector machines are learning algorithms that try to find a hyperplane that separates the differently classified data the most.

They are based on two key ideas:

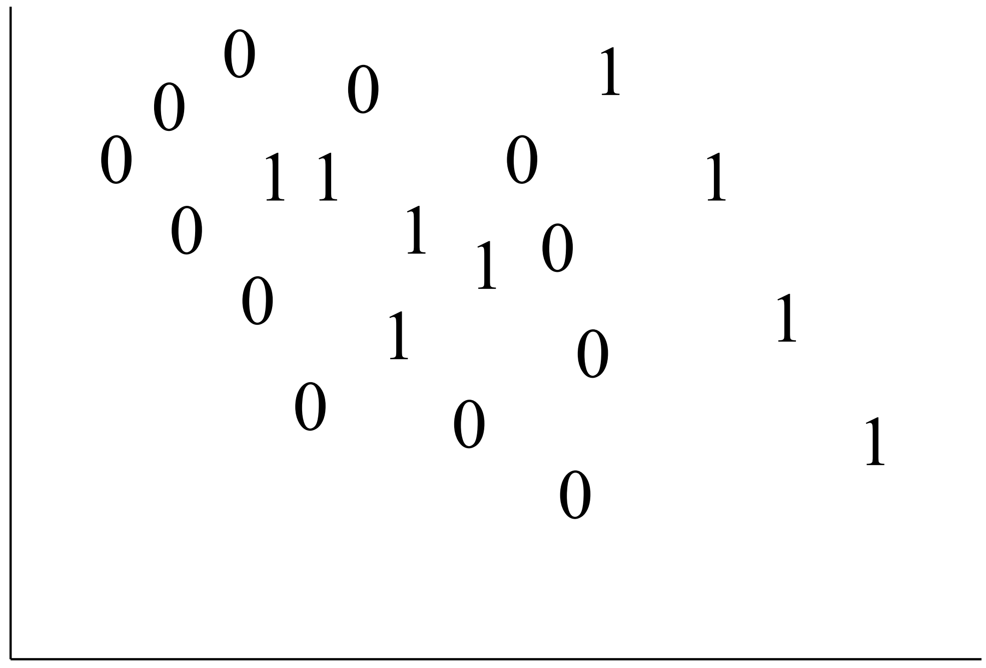
- Maximum margin hyperplanes
- A kernel ‘trick’.

# Maximal Margin




Find the hyperplane with maximal margin for all the points. This originates an optimization problem Which has a unique solution (convex problem).

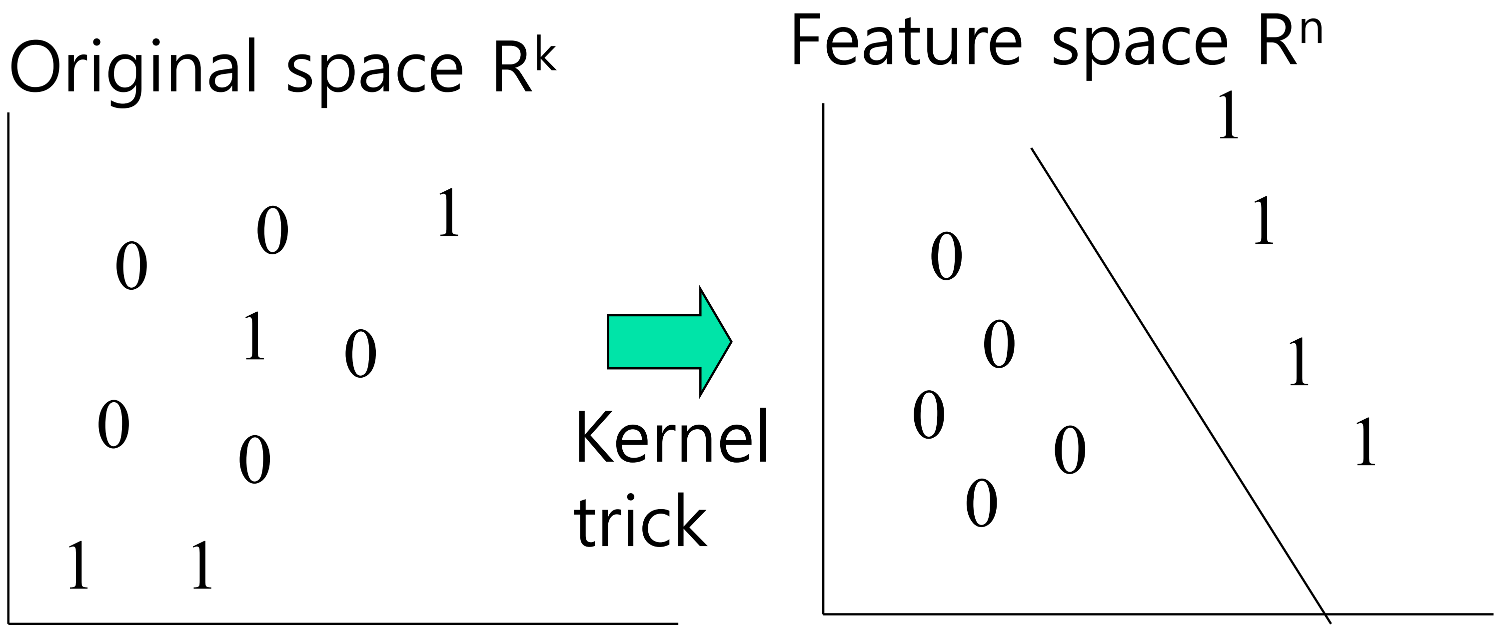
# Non-separable data




What can be done if data cannot be separated with a hyperplane?

# The kernel trick

The SVM algorithm implicitly maps the original data to a feature space of possibly infinite dimension in which data (which is not separable in the original  space) becomes separable in the feature space.

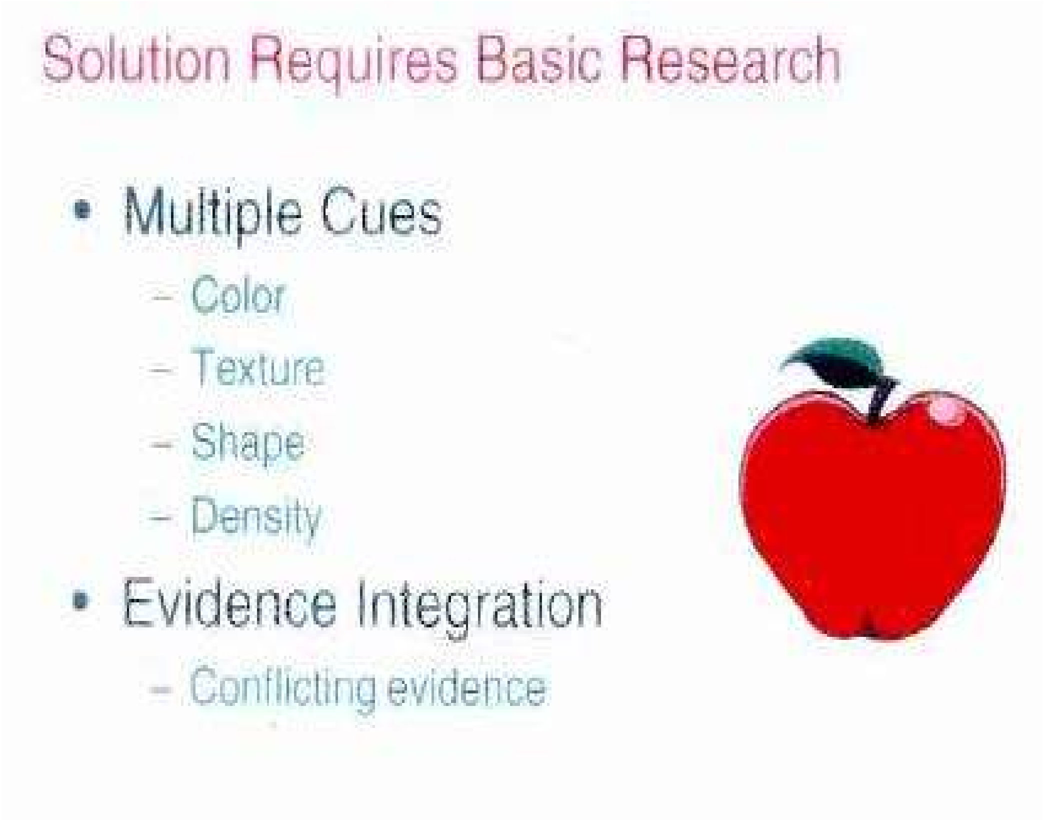


# Veggie Vision by IBM

Ideas about a practical system
to make more efficient the selling and inventory of produce in a grocery store.

# 10 years of R&D now

This information was shared by IBM researchers. Since that time, the system has been tested in small markets and has been modified according to that experience.



# Up to 400 produce types

# Practical problems of application environment

- Shrinkage → sweethearting
- Slow check-out
- Checker training
- Affixing PLUs
- Expensive packaging
- Solid waste

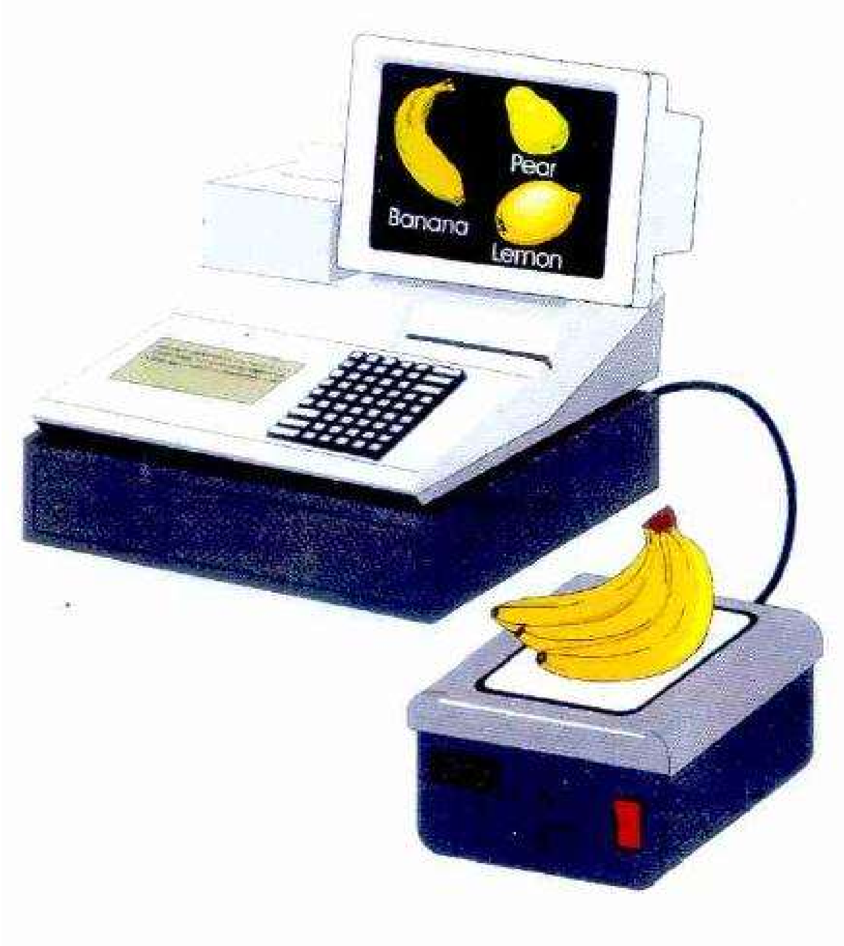
If asked, most grocers and supermarket owners would admit that they would rather not carry produce:

Rather than...

# Engineering the solution

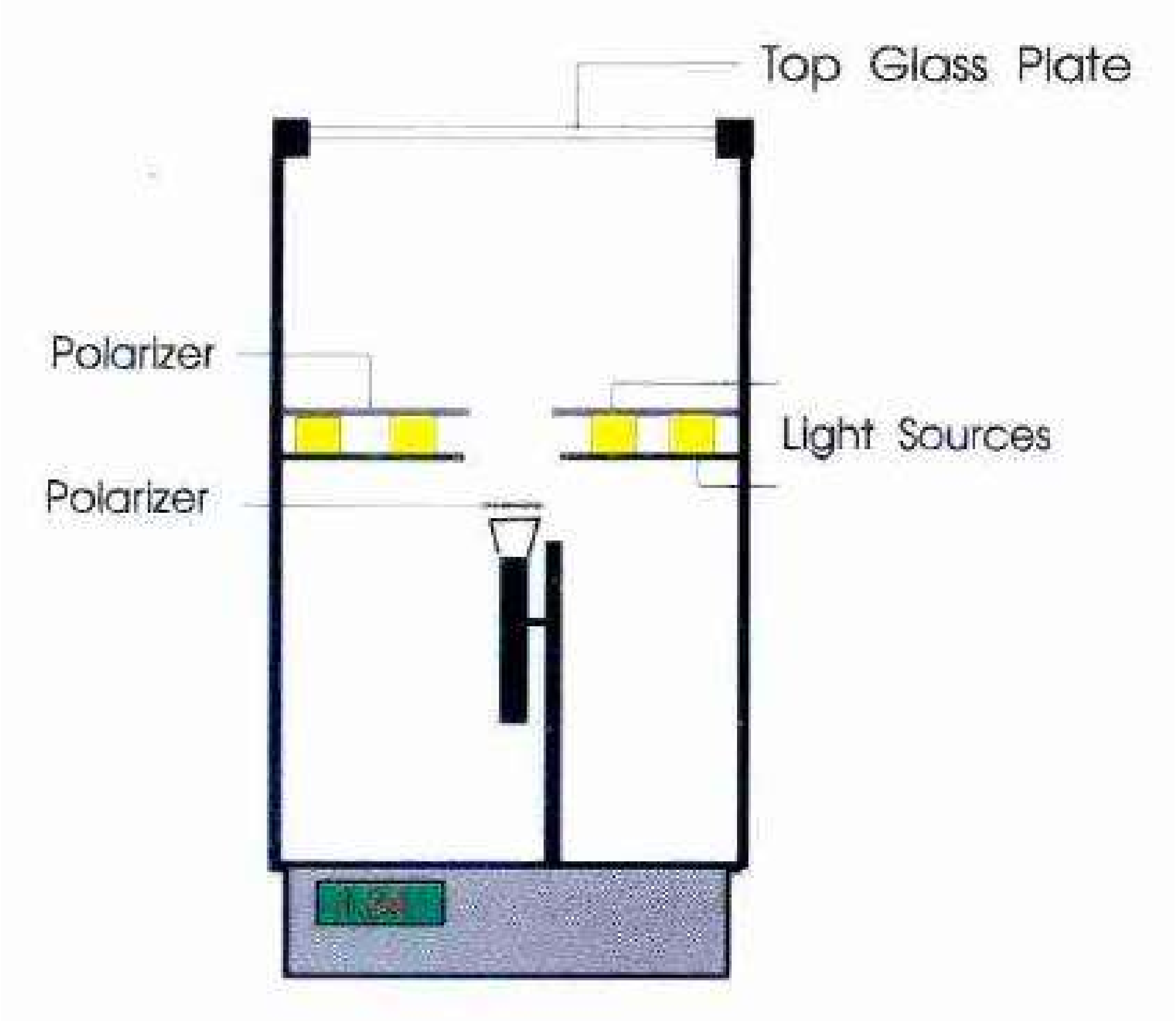
# System to operate inside the usual checkout station

- together with bar code scanner
- together with scale
- together with accounting
- together with inventory
- together with employee
- within typical store environment



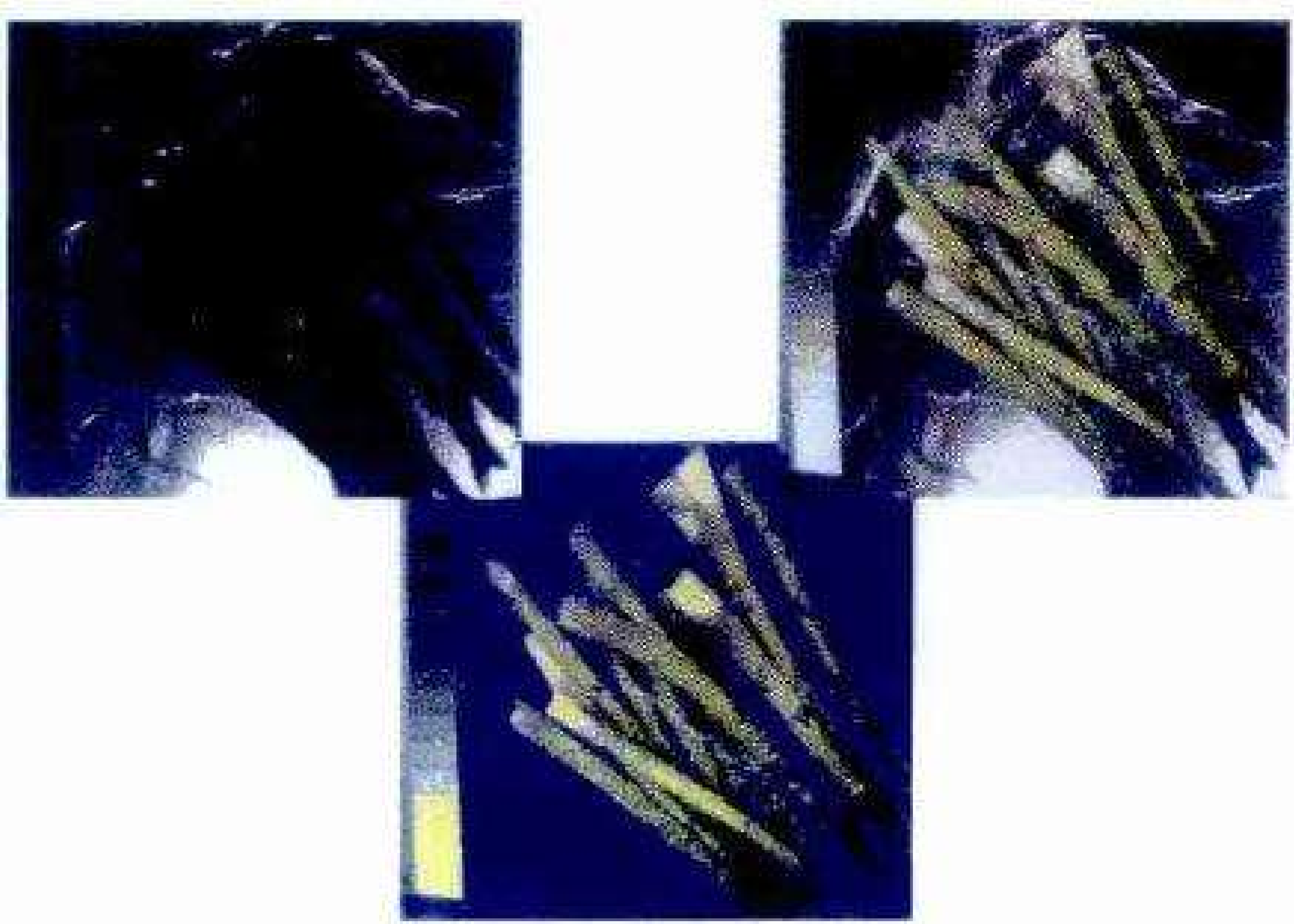

* figure shows system asking for help from the cashier in making final decision on touch screen

# Modifying the scale



# Need careful lighting engineering

# Need to segment product from background



# Quality segmented image obtained

# Design of pattern recognition paradigm (from 1997)

**FEATURES are: color, texture, shape, and size all represented uniformly by histograms**

### Matching

- signature = concatenated histograms
- multiple signatures per produce type
- weighted absolute histogram difference
- **nearest neighbors** retrieved
- "sure" if top 2 are the same type

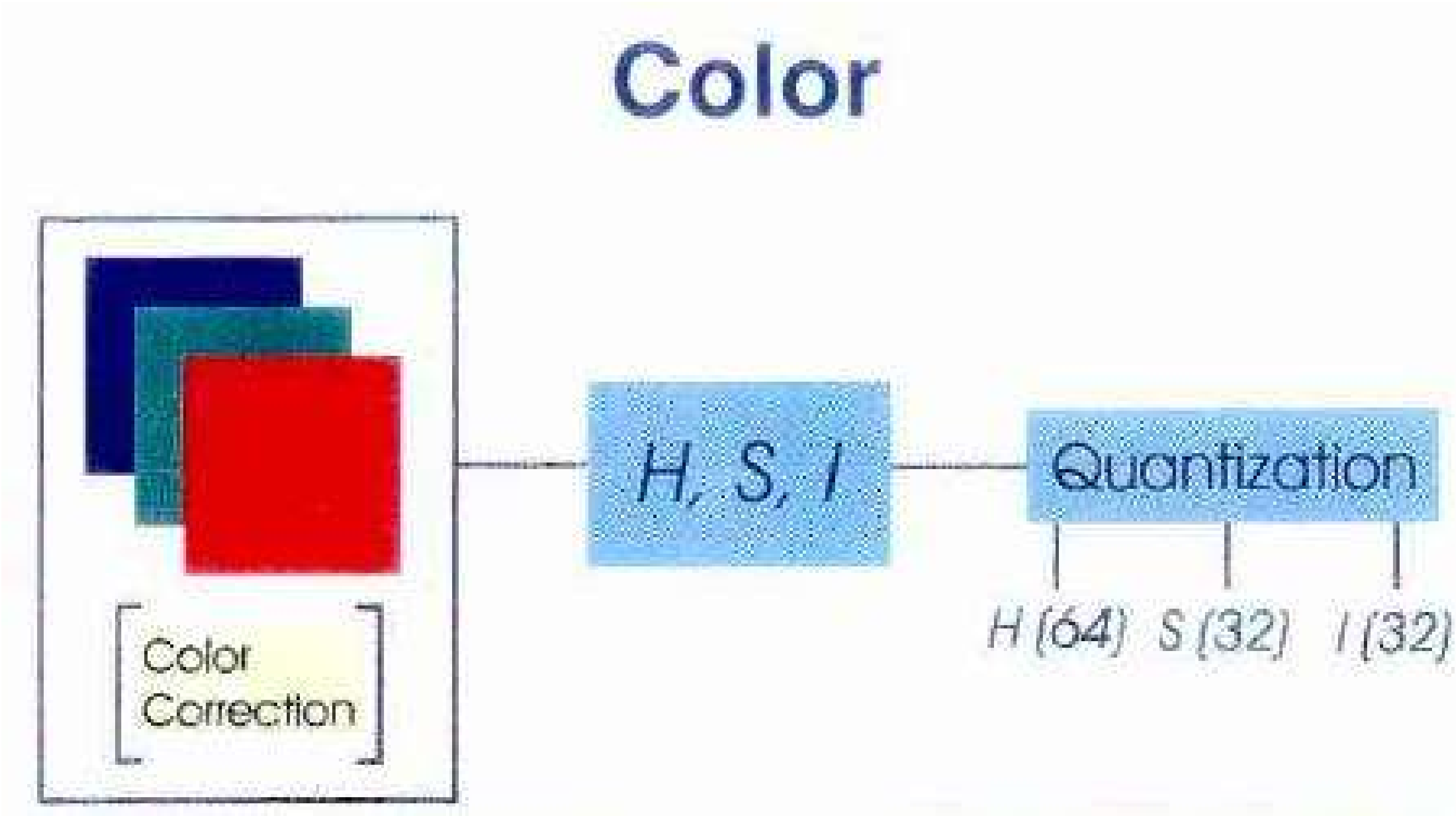
### Training

- batch and incremental modes
- store new signature of labelled example
- **automatic selection** - skip if already "sure"
- signatures per class limited (e.g. 12)
- replace least useful and least recently used

# Matching procedure

- Sample product represented by concatenated histograms: about 400 D
- 350 produce items x 10 samples = 3500 feature vectors of 400D each
- Have about 2 seconds to compare an unknown sample to 3500 stored samples (3500 dot products)
- Analyze the k nearest: if closest 2 are from one class, recognize that class (sure)

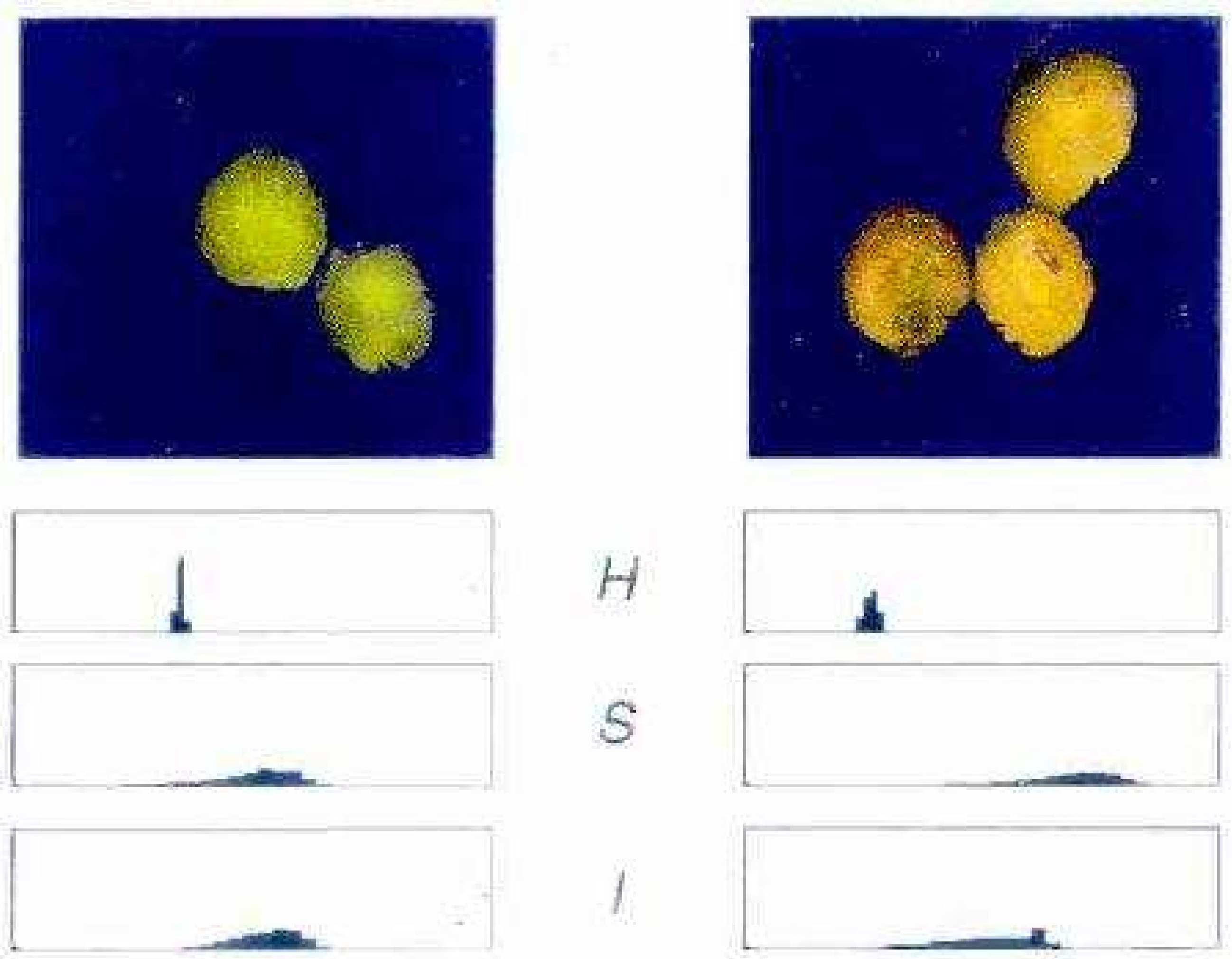
# HSI for pixel color: 6 bits for hue, 5 for saturation and intensity



# Histograms of 2 limes versus 3 lemons

Distribution or population concept adds robustness:

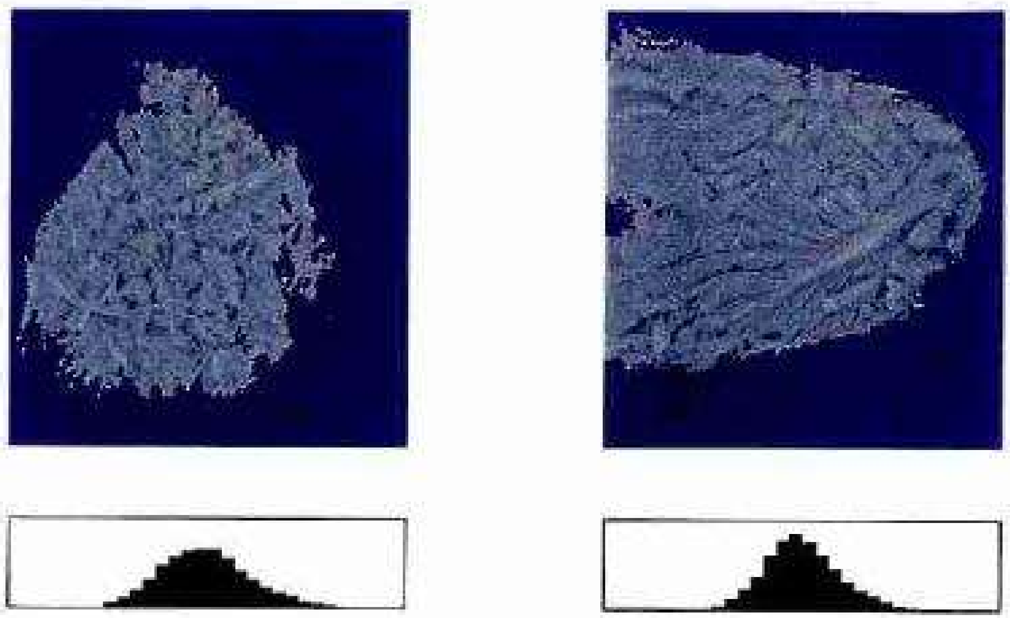
- to size of objects
- to number of objects
- to small variations of color (texture, shape, size)



# Texture: histogram results of LOG filter[s] on produce pixels

- Center surround mask
- Histogram differences

Leafy produce A



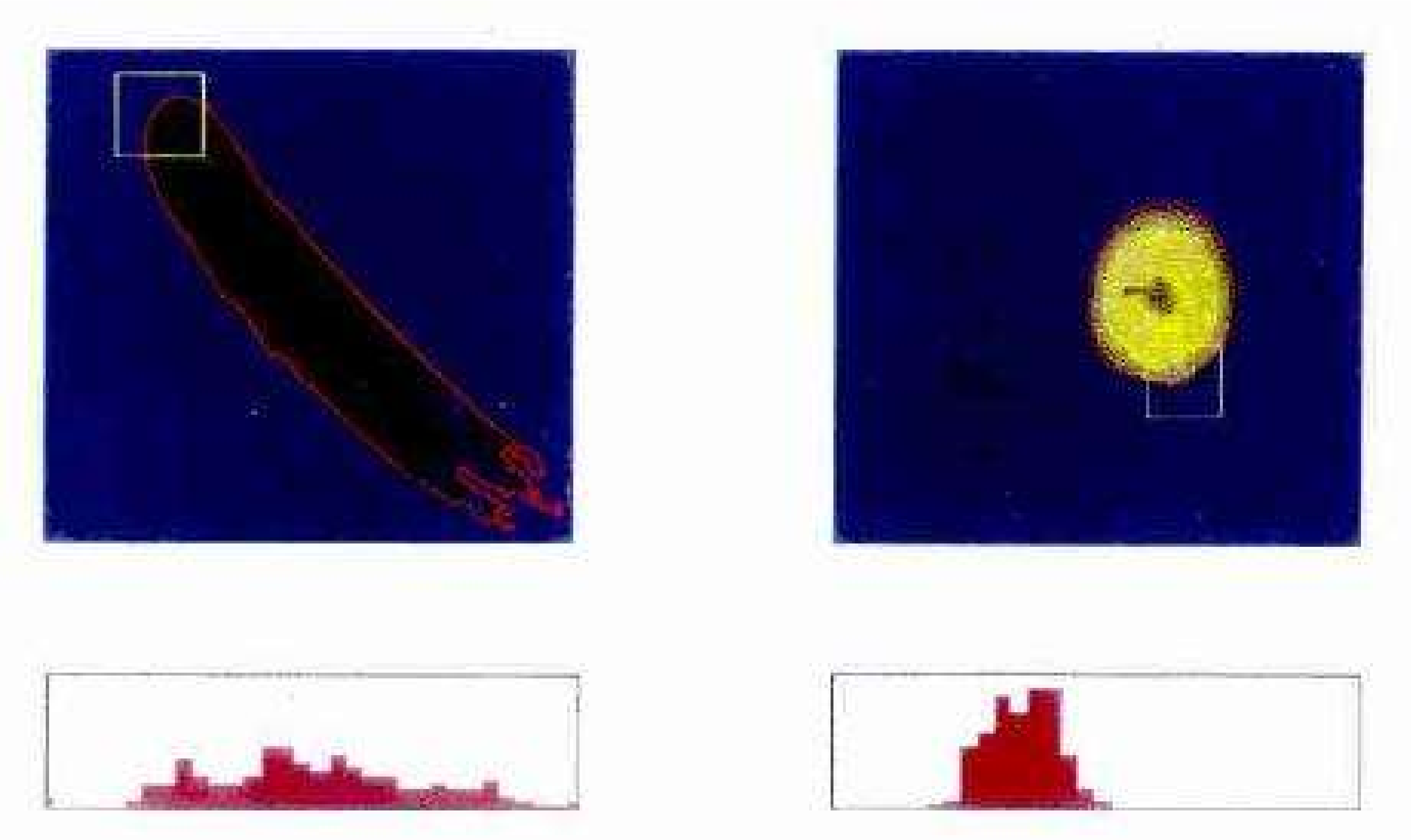
Leafy produce B

# Shape: histogram of curvature of boundary of produce

## Shape

- Chain-Coded Produce Boundary.
- Smooth Boundary.
- Compute Radii of Curvature Around Boundary.
- Histogram of Radii of Curvature.

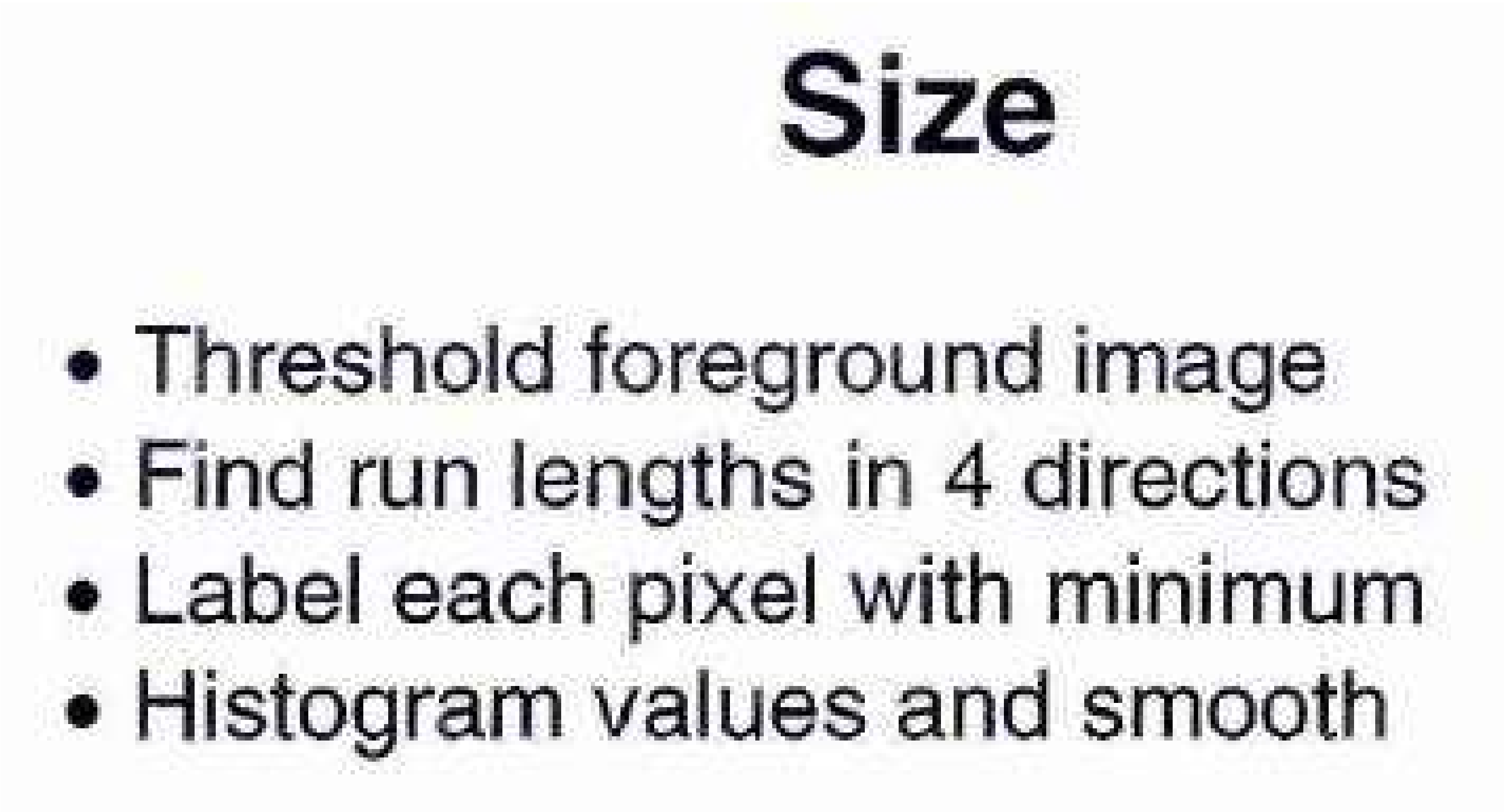
# Banana versus lemon or cucumber versus lime



Large range of curvatures indicates complex object

Small range of curvatures indicates roundish object

# Size is also represented by a histogram

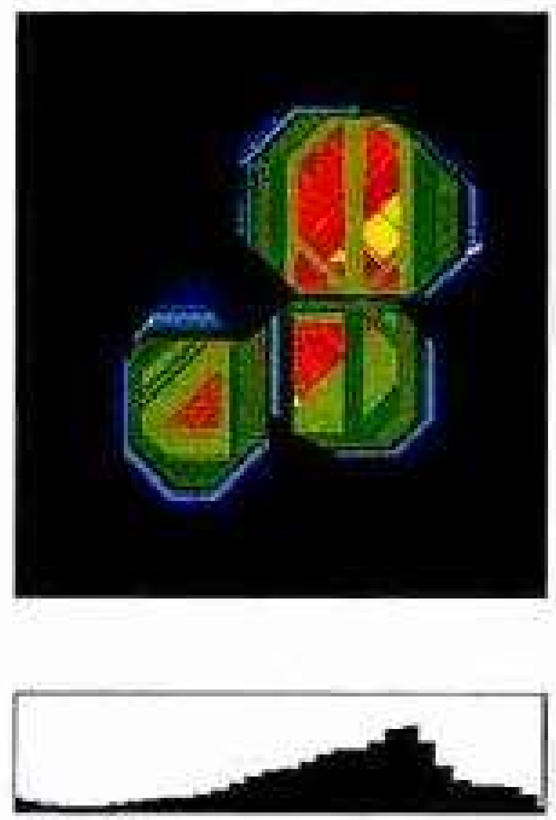


## Size

- Threshold foreground image
- Find run lengths in 4 directions
- Label each pixel with minimum
- Histogram values and smooth

# Each pixel gets a "size" as the minimum distance to boundary

Purple grapes



Chinese eggplants

# Learning and adaptation

- System easy to train: show it produce samples and tell it the labels.
- During service: age out oldest sample; replace last used sample with newly identified one.
- When multiple labeled samples match the unknown, system asks cashier to select from the possible choices.