# Pattern Classification

## EET3053

## Lecture 02: Feature Extraction

**Dr. Kundan Kumar**
Associate Professor
Department of ECE

Faculty of Engineering (ITER)
S'O'A Deemed to be University, Bhubaneswar, India-751030

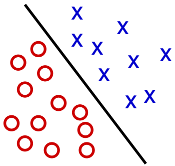
# Feature Extraction

# Topics to be covered

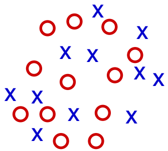
- Boundary Representation
  - Boundary (Border) following
  - Chain codes
  - Polynomial Approximation
  - Signatures
  - Boundary Segments
  - Skeletons
- Boundary Descriptors
  - Some Simple Descriptors
  - Shape Numbers
  - Fourier Descriptors
  - Statistical Moments
- Regional Descriptors
  - Texture: Moment Invariants
  - Texture: GLCM, LBP

# Good features and Bad features

- Extract features which are good for classification.
- Good features:
  - Objects from the same class have similar feature values
  - Objects from different classes have different values.
- Bad Features: features simply do not contain the information needed to separate the classes, doesn't matter how much effort you put into designing the classifier.

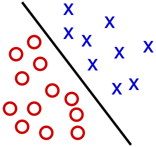


"Good" features

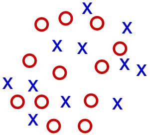


"Bad" features

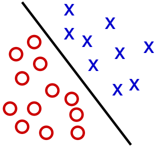
# Feature separability



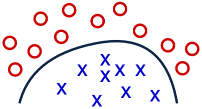
"Good" features



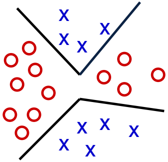
"Bad" features



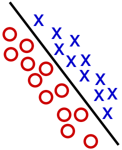
Linear separability



Non-linear separability

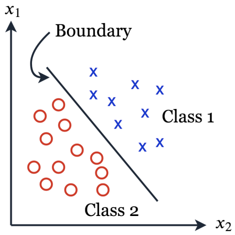


Multi-modal



Highly correlated

# Labeled features



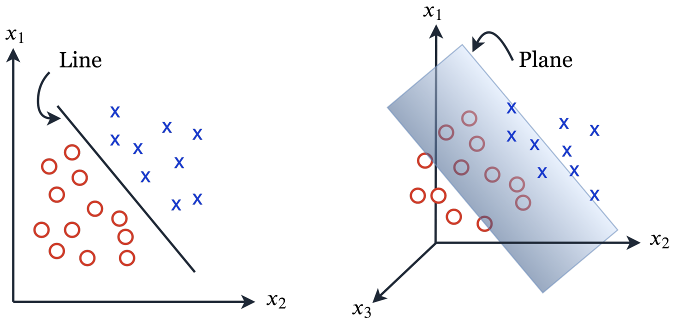

$$\left.\begin{array}{l} f_1 \\ f_2 \\ f_3 \end{array}\right\} \in \omega_1$$

$$\left.\begin{array}{l} f_4 \\ f_5 \\ f_6 \\ f_7 \end{array}\right\} \in \omega_2$$

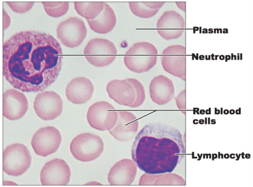
- In general, we use labeled features for supervised learning.
- The mapping from pattern to features that is unique whereas mapping from feature vector to pattern is not immediate.
- So, many patterns may be matched to the same feature of vector.
- In pattern recognition, we never talk about a single pattern. We always talk about feature vector.

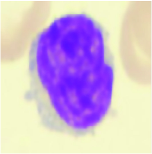
# Nature of separating plane?

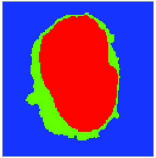
- For 2 dimensional feature space – line
- For 3 dimensional feature space – plane
- For more than 3 dimensional feature space – hyperplane

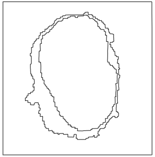


# Boundary Representation







# Binary Images

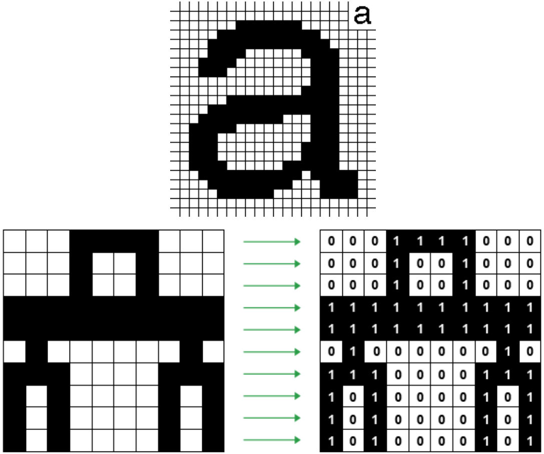
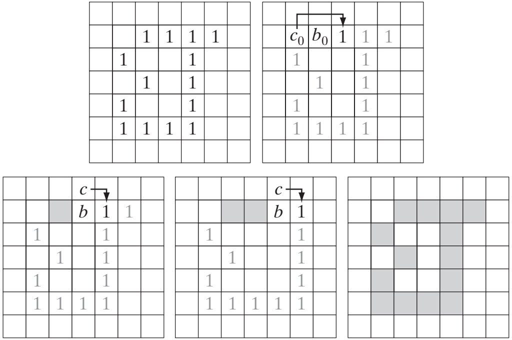



Figure: Binary Images

# Boundary (Border) algorithm

- Assume a binary image in which object and background points are labeled 1 and 0, respectively.
- Assume images are padded with the border of 0's to avoid object merging with the image border



# Boundary algorithm (Moore boundary tracking algorithm)

1. Let the starting point, $b_0$ be the *uppermost, leftmost* point in the image that is labeled 1. Denote by $c_0$ the *west* neighbor of $b_0$. Clearly, $c_0$ always is a background point. Examine the 8-neighbors of $b_0$, starting at $c_0$ and proceeding in a clockwise direction. Let $b_1$ denote the *first* neighbor encountered whose value is 1, and let $c_1$ be the (back-ground) point immediately preceding $b_1$ in the sequence. Store the locations of $b_0$ and $b_1$ for use in Step 5.
2. Let $b = b_1$ and $c = c_1$.
3. Let the 8-neighbors of $b$, starting at $c$ and proceeding in a clockwise direction, be denoted by $n_1, n_2, \ldots, n_8$. Find the first $n_k$ labeled 1.
4. Let $b = n_k$ and $c = n_{k-1}$
5. Repeat Step 3 and 4 until $b = b_0$ and the next boundary point found in $b_1$. The sequence of $b$ points found when the algorithm stops constitutes the set of ordered boundary points.

# Boundary algorithm - stopping criterion

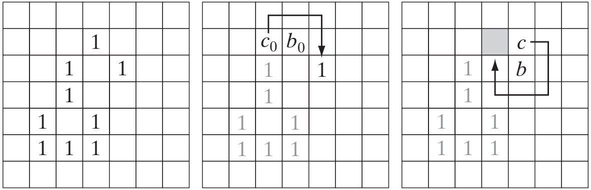



Figure: Illustration of an erroneous result when the stopping rule is such that boundary-following stops when the starting point, $b_0$, is encountered again

# Boundary representation: Chain Codes

- In order to represent a boundary, it is useful to compact the raw data (list of boundary pixels)
- Chain codes: list of segments with defined length and direction
  - 4-directional chain codes
  - 8-directional chain codes

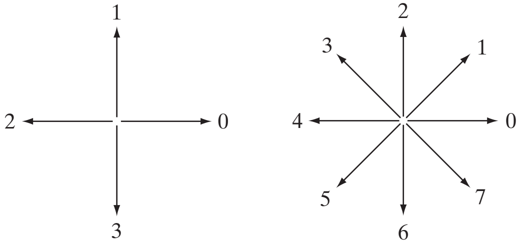



Figure: Direction numbers for (a) 4-directional chain code, and (b) 8-directional chain code

# Boundary representation: Chain Codes

- It may be useful to downsample the data before computing the chain code
  - to reduce the code dimension
  - to remove small detail along the boundary

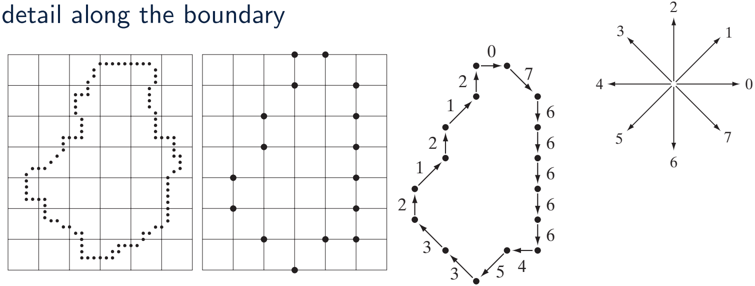
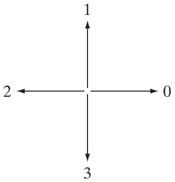


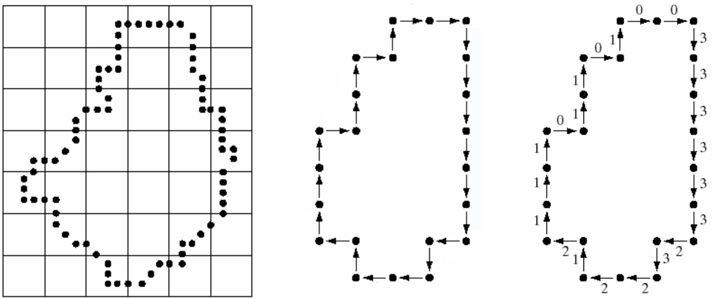

Figure: (a) Digital boundary with resampling grid superimposed, (b) Result of resampling, (c) 8-directional chain-coded boundary.

- Can you draw 4-directional chain-coded boundary?

# Boundary representation: Chain Codes

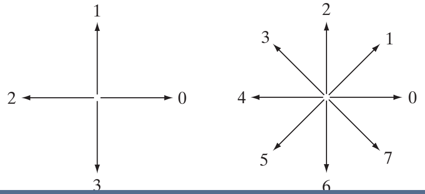



Chain code: 00333333232221211101101

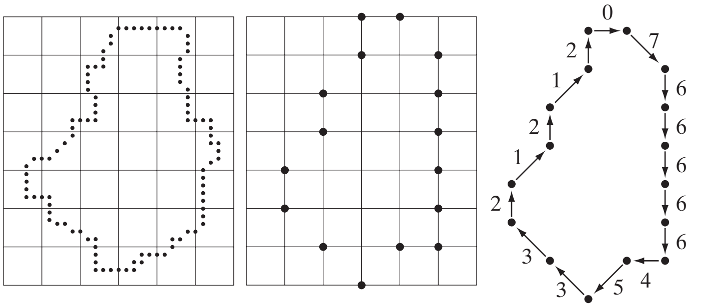


# Boundary representation: Differential Chain Code

- The chain code of a boundary depends on the starting point.
  - normalize with respect to the starting point (circular sequence)
  - the new starting point is the one who gives a sequence of numbers giving the *smallest/largest integer*.
- Normalize with respect to rotation:
  - First difference can be used
  - E.g., $10103322 \Rightarrow 3133030$ (counting CCW) and adding the last transition (circular sequence: $2 \Rightarrow 1$)
    $\Rightarrow$ 31330303 (*Differential Chain Code*)
    $\Rightarrow$ 03033133 (*Independent of starting point*, *i.e., rotation invariant*)



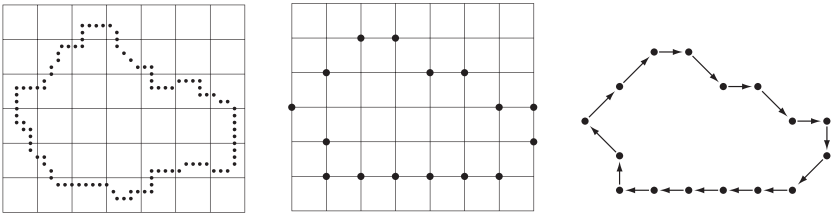
# Differential Chain Code




Can you write the Differential Chain Code?

- Chain code: 0766666453321212
- Differential chain code: 7700006160771716
- Differential chain code (rotation invariant): 0000616077171677

# Differential Chain Code: Validation



Can you write the Differential Chain Code?

- Chain code: 07070654444442311
- Differential chain code: 7171677000061607
- Differential chain code: 000061607717167 7 (validated)
- Is the differential chain code is invariant to rotation at any angle? (HW)

# Polygonal Approximation

- A digital boundary can be approximated with arbitrary accuracy by a polygon.
- In practice, the goal of polygonal approximation is to capture the “essence” of the boundary shape with the fewest possible polygonal segments.
  - Minimum-perimeter polygon
  - Splitting technique

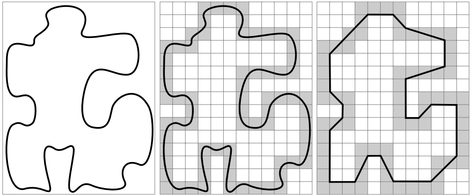


Figure: (a) An object boundary (black curve). (b) Boundary enclosed by cells (in gray). (c) Minimum-perimeter polygon obtained by allowing the boundary to shrink. The vertices of the polygon are created by the corners of the inner and outer walls of the gray region.

# Polygonal Approximation: Minimum-Perimeter Polygon

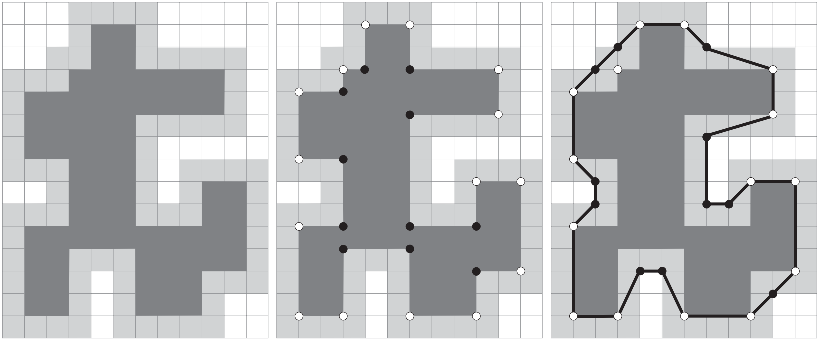


Figure: (a) Region (dark gray) resulting from enclosing the original boundary by cells. (b) Convex (white dots) and concave (black dots) vertices obtained by following the boundary of the dark gray region in the counterclockwise direction. (c) Concave vertices (black dots) displaced to their diagonal mirror locations in the outer wall of the bounding region; the convex vertices are not changed. The MPP (black boundary) is superimposed for reference.

# Polygonal Approximation: Splitting Technique

- One approach to boundary segment splitting is to subdivide a segment successively into two part until a *specified criterion* is satisfied.

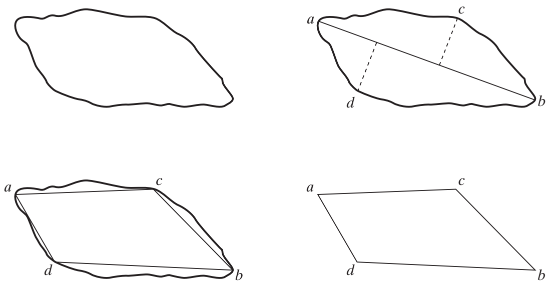



Figure: (a) Original boundary, (b) Boundary divided into segments based on extreme points, (c) Joining of vertices, (d) Resulting polygon

# Polygonal Approximation: Splitting Technique

- For a closed boundary, the best starting points usually are two farthest points in the boundary.
- Farthest point can be obtained by *Karhunen-Loeve transform (KLT)*.
- The maximum perpendicular distance from a boundary segment to the line joining its two end points not exceed a preset threshold.
- Splitting procedure with a threshold equal to 0.25 times the length of line $ab$.

# Signatures

- A signature is a 1-D representation of a boundary (which is a 2-D thing): it should be easier to describe.
  e.g., distance form the centroid vs angle.

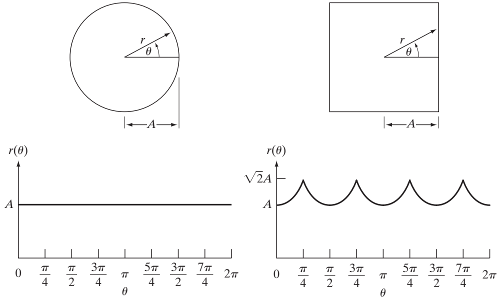



Figure: Distance-versus-angle signatures. (a) $r(\theta)$, is constant, (b) the signature consists of repetitions of the pattern $r(\theta) = Asec(\theta)$ for $0 \leq \theta \leq \pi/4$ and $r(\theta) = Acsc(\theta)$ for $\pi/4 < \theta \leq \pi/2$

# Signatures

- Signatures are invariant to translation, but variant to rotation.
- Invariant to rotation: depends on the starting point
  - the starting point could be the farthest point from the *centroid*.
- Scaling varies the amplitude of the signature
  - invariance can be obtained by normalizing between 0 and 1, or
  - by dividing by the variance of the signature (does not work on circle)

# Boundary Segments

- Decomposing a boundary into segments often is useful.
- Decomposition reduces the boundary's complexity and thus simplifies the description process.
- In this case use of the *convex hull* of the region enclosed by the boundary is a powerful tool for robust decomposition of the boundary.

# Boundary Segments

- *Convex hull* $H$ of an arbitrary set $S$ is the smallest convex set containing $S$.
- The set difference $H - S$ is called the *convex deficiency* $D$ of the set $S$.
- Note that in principle, this scheme is independent of region size and orientation.

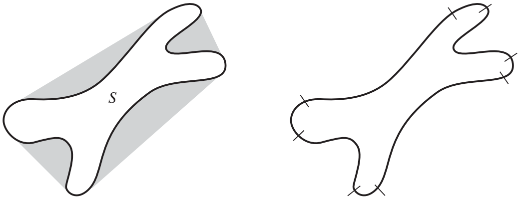
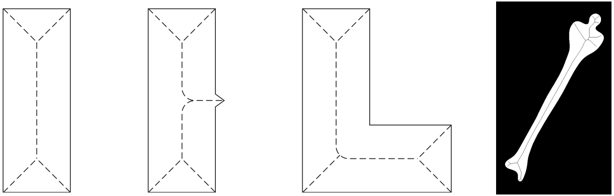



Figure: (a) A region, $S$, and its convex deficiency (shaded). (b) Partitioned boundary.

# Skeletonization

- One way to represent a shape is to reduce it to a graph, by obtaining its *skeleton* via thinning (*skeletonization*)
- *MAT (Medial axis transformation)* is composed by all the points which have more than one closest boundary points (“*prairie fire concept*”)



(a) (b)

Figure: (a) Medial axes (dashed) of three simple regions,(b) Human leg bone and skeleton of the region

# Boundary Features/Descriptors

# Simple descriptors

- *length* of a boundary is one of its simplest descriptors.
  - The number of pixels along a boundary gives a rough approximation of its length.
  - For a chain coded curve with unit spacing:

$$\textit{length} = \text{Horizontal} + \text{Vertical} + \sqrt{2} \times \text{Diagonal}$$

- *diameter* (length of the major axis)

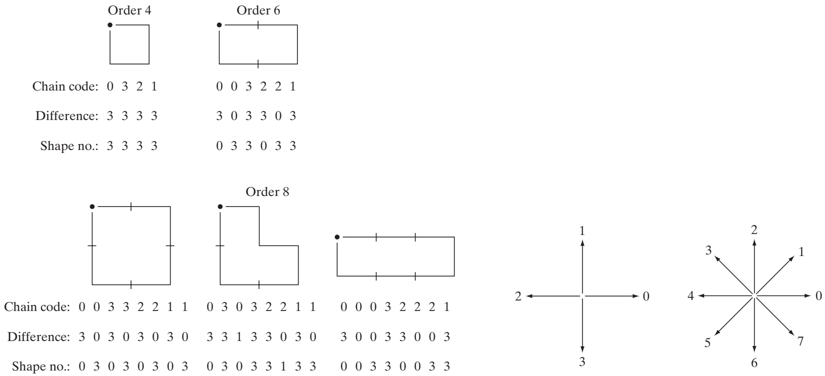
$$\text{Diam}(B) = \max_{i,j} [D(p_i, p_j)]$$

- The *minor axis* of a boundary is defined as the line perpendicular to the *major axis*.
- *Basic rectangle* (formed by the major and the minor axis; encloses the boundary) and its

$$\text{eccentricity} = \frac{\text{major axis}}{\text{minor axis}}$$

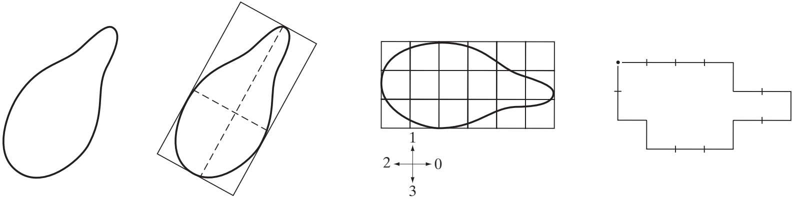
# Shape Number

- Shape number: the first difference as smallest magnitude (treating the chain code as a circular sequence)
- Order of a shape: the number of digits in Shape number.



# Shape Number

- It is advisable to normalize the grid orientation by aligning the chain code grid to the basic rectangle.




Chain code: 0 0 0 0 3 0 0 3 2 2 3 2 2 2 1 2 1 1

Difference: 3 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0

Shape no.: 0 0 0 3 1 0 3 3 0 1 3 0 0 3 1 3 0 3

# Fourier Descriptors

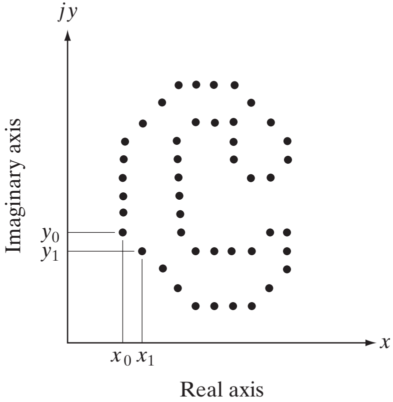



Figure: A digital boundary and its representation as a complex sequence. The point $(x_0, y_0)$ and $(x_1, y_1)$ shown are (arbitrarily) the first two points in the sequence.

# Fourier Descriptors

- Each coordinate pair treat as a complex number

$$s(k) = x(k) + jy(k)$$

for $k = 0, 1, 2, \ldots, N-1$.

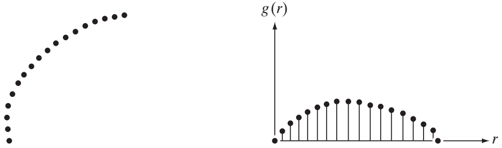
- The discrete Fourier transform (DFT) of $s(k)$ is

$$a(u) = \sum_{k=0}^{N-1} s(k)e^{-j2\pi uk/N}$$

for $u = 0, 1, 2, \ldots, N-1$

- $a(u)$ are Fourier Descriptors.

# Statistical moments

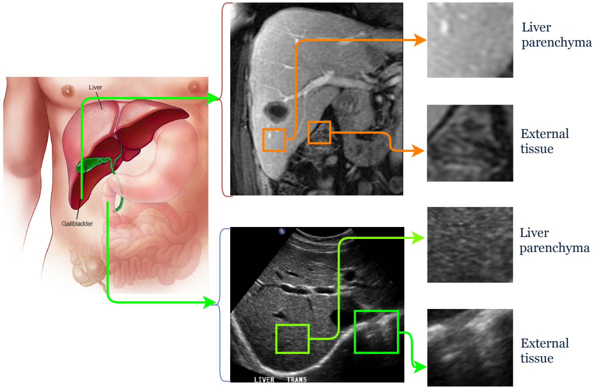



- Once a boundary is described as a 1-D function, statistical moments (mean, variance, and a few higher-order central moments) can be used to describe it.

$$\mu_n(z) = \sum_{i=0}^{N-1} (z_i - m)^n p(z_i)$$

$$m = \sum_{i=0}^{N-1} z_i p(z_i)$$

# Regional Features/Descriptors



# Some simple Descriptors

- The *area of a region* is defined as the number of pixels in the region.
- The *perimeter of a region* is the length of its boundary.
- *Compactness of a region*, defined as $(perimeter)^2/area$, and is minimal for a disk-shape region.
- A slightly different descriptor of compactness is the *circularity ratio*, defined as the ratio of the area of a region to the area of a circle (the most compact shape).
- *Region descriptors*:
  - mean and median of the gray levels,
  - minimum and maximum gray-level values, and
  - number of pixels with above and below the mean.

# Region Features

- There are following region features
  - Colors, e.g., RGB values, HSV value, L*a*b
  - Intensity, e.g. Gray Values
  - Textures
- Further texture is divided into two classes:
  - Spatial Domain Features
    - Structural Features, e.g., LBP, Wavelets
    - Statistical Features, e.g., GLCM, Orientation Histogram
  - Transformed Domain Features
    - Gabor Filters

# Texture

- An important approach to region description is to quantify its texture content.

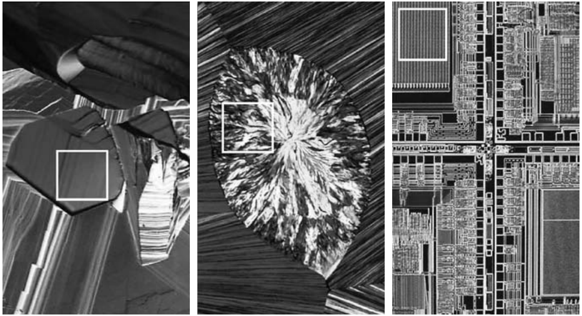


Figure: The white squares mark, from left to right, smooth, coarse, and regular textures. These are optical microscope images of a superconductor, human cholesterol, and a microprocessor. (Courtesy of Dr. Michael W. Davidson, Florida State University.)

# Texture: Statistical approaches

- Compute the *histogram* of the *area of interest*.
- The $n^{th}$ *moment* of $z$ about the mean is

$$\mu_n(z) = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i)$$

$$m = \sum_{i=0}^{L-1} z_i p(z_i)$$

- The *second moment* $(n = 2)$ is of particular importance in texture description. It is a measure of gray-level contrast that can be used to establish descriptors of relative *smoothness*.
- For example, the texture measure, $R$, is 0 for areas of contrast intensity (the variance is 0 here) and approaches 1 for large value of $\sigma^2(z)$

$$R = 1 - \frac{1}{1 + \sigma^2(z)}$$

# Texture: Statistical approaches

- The *third moment*

$$\mu_3(z) = \sum_{i=0}^{L-1} (z_i - m)^3 p(z_i)$$

is a measure of the *skewness* of the histogram while the *fourth moment* is a measure of its relatives flatness.

- Some useful additional texture measures bases on histograms include a measure of "*uniformity*", given by

$$\text{Uniformity} = \sum_{i=0}^{L-1} p^2(z_i)$$

- *Average entropy* measure

$$\text{Entropy} = -\sum_{i=0}^{L-1} p(z_i) \log_2 p(z_i)$$

# Texture: Statistical approaches

Figure: Texture measures for the subimages shown in previous slide

| Texture | Mean | Standard deviation | *R* (normalized) | Third moment | Uniformity | Entropy |
|---|---|---|---|---|---|---|
| Smooth | 82.64 | 11.79 | 0.002 | −0.105 | 0.026 | 5.434 |
| Coarse | 143.56 | 74.63 | 0.079 | −0.151 | 0.005 | 7.783 |
| Regular | 99.72 | 33.73 | 0.017 | 0.750 | 0.013 | 6.674 |

# Image Histograms

- The histogram of a digital image with intensity levels in the range $[0, L-1]$ is a discrete function

$$h(r_k) = n_k \tag{1}$$

where, $r_k$ is the $k$th intensity value and $n_k$ is the number of pixels in the image with intensity $r_k$

- Normalized histogram

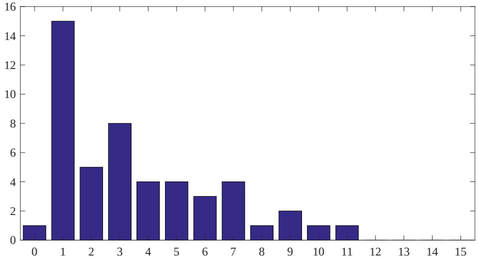
$$p(r_k) = \frac{r_k}{MN} \qquad \text{for } k = 0, 1, 2, \ldots, L-1. \tag{2}$$

- $p(r_k)$ is an estimate of the probability of occurrence of intensity level $r_k$ in an image.

# Compute histogram

Compute the histogram of the given image. First find out the number graylevels in the image (how many bit image?).

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0 | 2 | 4 | 5 | 3 | 1 |
| 9 | 1 | 1 | 4 | 7 | 2 | 1 |
| 10 | 3 | 7 | 3 | 5 | 3 | 3 |
| 11 | 2 | 3 | 3 | 3 | 2 | 1 |
| 7 | 5 | 6 | 6 | 7 | 6 | 1 |
| 1 | 4 | 1 | 1 | 4 | 9 | 1 |
| 2 | 8 | 1 | 1 | 5 | 1 | 1 |



# Texture: Gray level co-occurrence matrix (GLCM)

- Gray Level Co-occurance Matrix: $G_{l,\theta}(i,j)$
  where $i = 0, 1, 2, \ldots, L-1$, $j = 0, 1, 2, \ldots, L-1$, $L$ is maximum intensity level.

| | | | | | |
|---|---|---|---|---|---|
| 1 | 1 | 7 | 5 | 3 | 2 |
| 5 | 1 | 6 | 1 | 2 | 5 |
| 8 | 8 | 6 | 8 | 1 | 2 |
| 4 | 3 | 4 | 5 | 5 | 1 |
| 8 | 7 | 8 | 7 | 6 | 2 |
| 7 | 8 | 6 | 2 | 6 | 2 |

Image $f$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 0 | 0 | 0 | 1 | 1 | 0 |
| 2 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 3 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 5 | 2 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 6 | 1 | 3 | 0 | 0 | 0 | 0 | 0 | 1 |
| 7 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 2 |
| 8 | 1 | 0 | 0 | 0 | 0 | 2 | 2 | 1 |

Co-occurrence matrix **G**

- Above calculation is just for demonstration. For real images, GLCM matrix dimension is $L \times L$, where index varies as $i = 0, 1, 2, \ldots, L-1$, $j = 0, 1, 2, \ldots, L-1$.

# GLCM Features

- *Maximum probability*: Measure of the strongest response of $G$. The range of value is $[0, 1]$.

$$\text{Maximum probability} = \max_{i,j} p_{ij}$$

- *Contrast*: A measure of intensity contrast between a pixel and its neighbor over the entire image. The range of values is 0 (When G is constant) to $(L-1)^2$.

$$\text{Contrast} = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} (i-j)^2 p_{ij}$$

- *Inverse Element Difference Moment*: A measure of intensity contrast between a pixel and its neighbor.

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{p_{ij}}{(i-j)^k} \qquad \text{for } i \neq j$$

# GLCM Features

- *Uniformity/Energy*: A measure of how intensities are uniformly distributed.

$$\text{Uniformity} = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} {p_{ij}}^2$$

- *Homogeneity*: Measures the spatial closeness of the distribution of elements in $G$ to the diagonal. The range of values is [0,1], with the maximum being achieved when $G$ is a diagonal matrix.

$$\text{Homogeneity} = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{p_{ij}}{1 + |i - j|}$$

also defined as

$$\text{Homogeneity} = \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{p_{ij}}{1 + (i - j)^2}$$

# GLCM Features

- *Entropy*: Measures the randomness of the elements of $G$. The entropy is 0 when all $p_{ij}$'s are 0 and is maximum when all $p_{ij}$'s are equal. The maximum value is $2\log_2 L$.

$$\text{Entropy} = -\sum_{i=0}^{L-1}\sum_{j=0}^{L-1} p_{ij}\log_2 p_{ij}$$

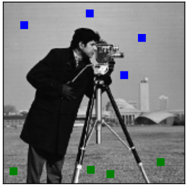
- *Correlation*: A measure of how correlated a pixel is to its neighbor over the entire image. Range of values is 1 to $-1$.

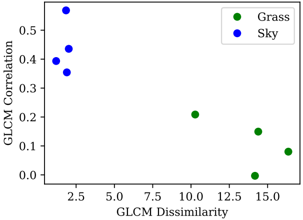
$$\text{Correlation} = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} \frac{(i-m_r)(j-m_c)p_{ij}}{\sigma_r\sigma_c}$$

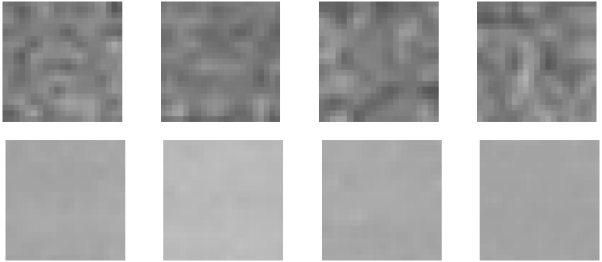
$$m_r = \sum_{i=0}^{L-1} i \sum_{j=0}^{L-1} p_{ij} \qquad m_r = \sum_{j=0}^{L-1} j \sum_{i=0}^{L-1} p_{ij}$$

$$\sigma_r^2 = \sum_{i=0}^{L-1}(i-m_r)^2\sum_{j=0}^{L-1} p_{ij} \qquad \sigma_c^2 = \sum_{j=0}^{L-1}(j-m_c)^2\sum_{i=0}^{L-1} p_{ij}$$

# GLCM feature Visualization


Original Image



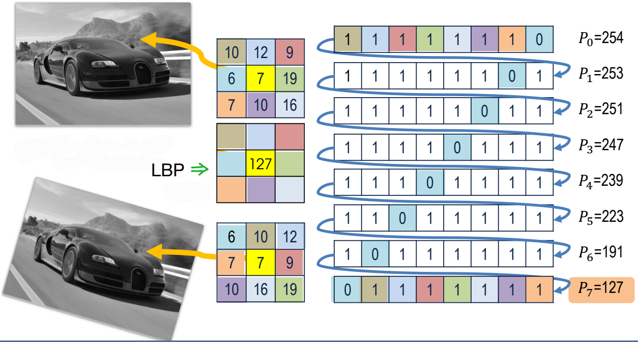



# Local Binary Pattern

- Basic Local Binary Pattern is governed by

$$b_k = \begin{cases} 1 & \text{if } g_k \geqslant g(x) \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad LBP_{ri}(x) = \min\{P_j\}$$

where $P_j$ is decimal equivalent of binary sequence $b_j$.



# Local Binary Pattern: Example

| | | | | |
|---|---|---|---|---|
| 5 | 4 | 2 | 2 | 1 |
| 3 | 5 | 8 | 1 | 3 |
| 2 | 5 | 4 | 1 | 2 |
| 4 | 3 | 7 | 2 | 7 |
| 1 | 4 | 4 | 2 | 6 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | |
| $2^7$ | $2^6$ | | | | $2^2$ | | $2^0$ | $= 197$ |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | |
| $2^7$ | | | | $2^3$ | | $2^1$ | $2^0$ | $= 139$ |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | |
| | | | $2^4$ | | $2^2$ | $2^1$ | $2^0$ | $= 23$ |

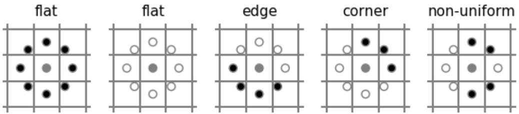
⋮

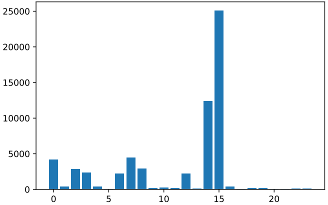
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | |
| $2^7$ | $2^6$ | $2^5$ | | | | $2^1$ | | $= 226$ |

| | | | | |
|---|---|---|---|---|
| | | | | |
| | ? | | | |
| | | 23 | | |
| | | | | |
| | | | | |

Can you compute LBP at the position (?)?

# Local Binary Pattern: Example



# References

[1] Hart, P. E., Stork, D. G., & Duda, R. O. (2000). Pattern classification. Hoboken: Wiley.

[2] Gonzalez, R. C., Woods, R. E., & Masters, B. R. (2009). Digital image processing.

Thank you!