18.10.2019

Digital Image Processing (CSE/ECE 478)

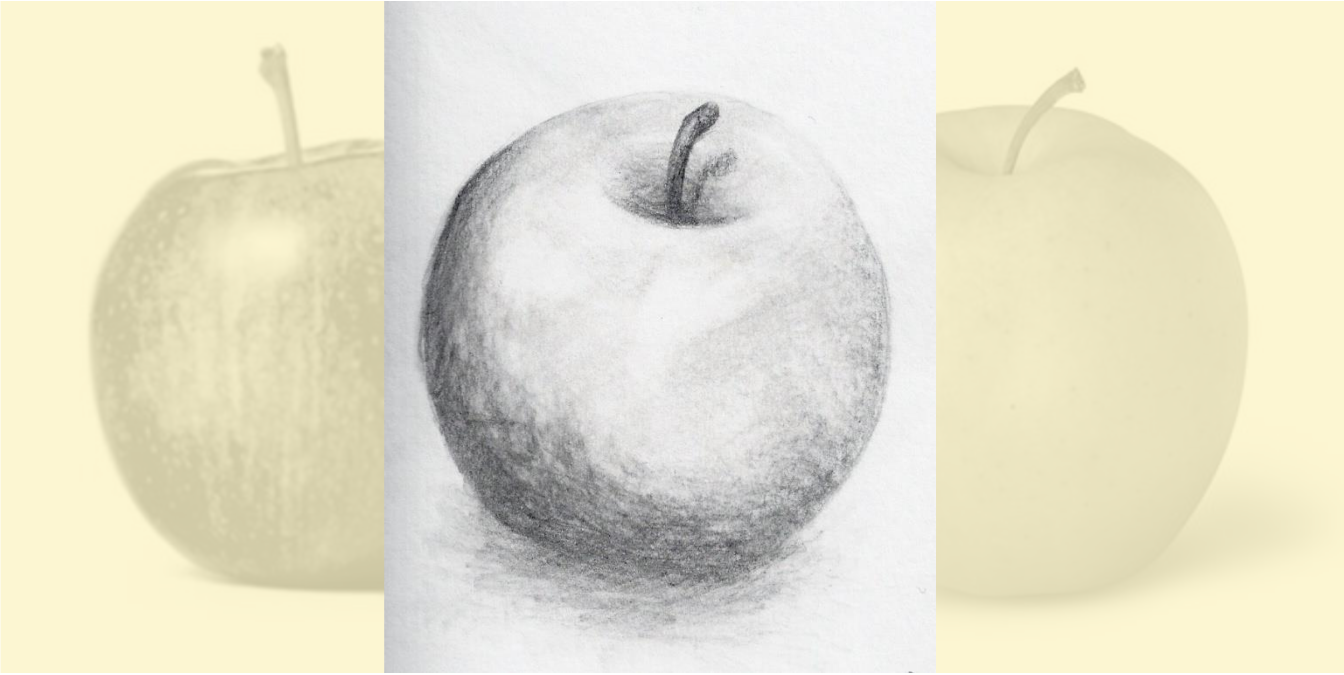
Lecture 17 : Representation and Description

Ravi Kiran

Image Representation & Description

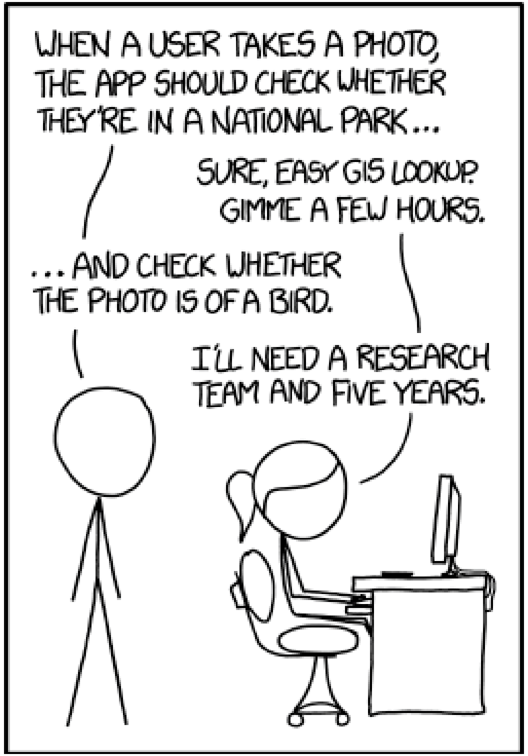
Describe an Apple

Describe an Apple to a Computer

History of Computer Vision

- MIT AI Lab – Summer Project
  - Attach a camera to a computer
  - Transfer the image to computer
  - Describe what it sees
  - Started in 1960s



Courtesy : xkcd

History of Computer Vision

- Computer Vision before 2000

Image Analysis ⇨

Low-level
- Pre-processing
- Extracting Primitives (corners, lines, contours, segments)

- Computer Vision in 2000-2012

Feature Design + ML ⇨

Mid-level
- Image Representation & Description
- Task-specific Feature Design
- Object-level understanding

- Computer Vision in 2013-now

Deep Learning ⇨

High-level
- Image Understanding
- Wild data + Other modalities
- Closer to envisioned AI

Recap : Segmentation

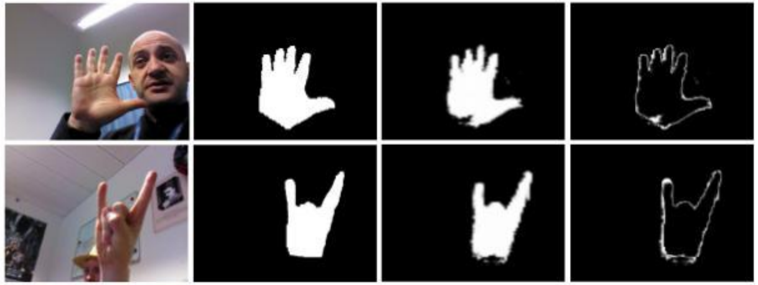
- Enhancement & Morphology
- Segments in terms of
  - Regions
  - Lines
  - Points

Image Analysis

Low-level

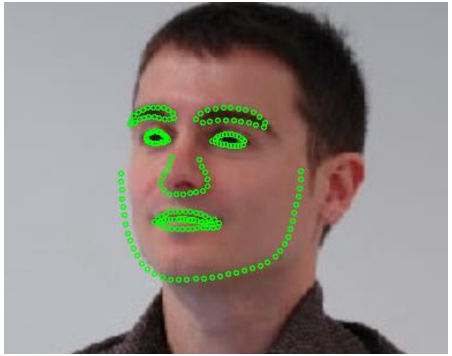
- Pre-processing
- Extracting Primitives

Pre-processing & Segmentation

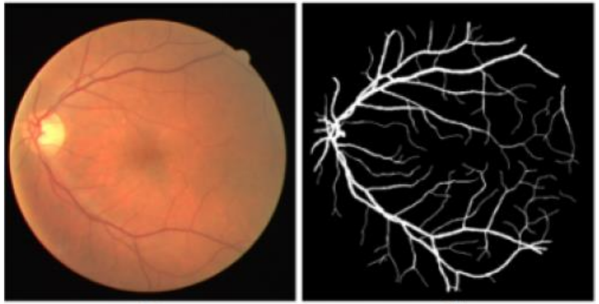


Hand tracking

Scene Text Recognition



Face Applications



Retina Image Analysis

Tracking / Scene Understanding

Driving & Navigation

Image Analysis Paradigm

- Internal vs. External Representation

- External = Shape / Boundary
  - Representation :
  - Description :

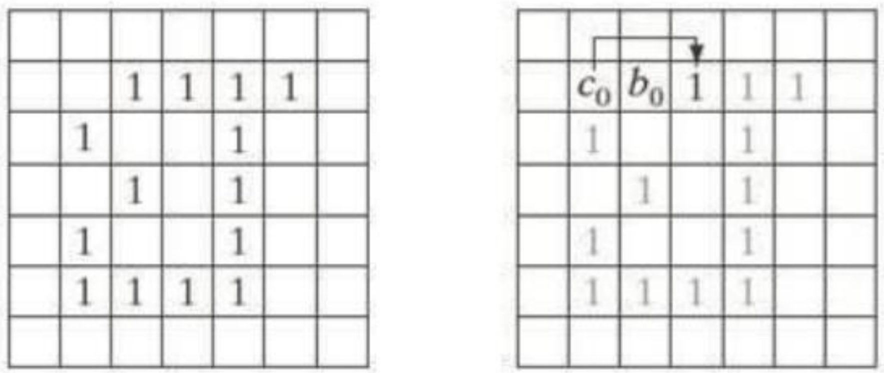
- Internal = Region
  - Representation :
  - Description

Representation and Description

- The Representation of the Object
  - An encoding of the object
  - Truthful but possibly approximate
- A Descriptor of the Object:
  - Only an aspect of the object
  - Suitable for classification
  - Consider invariance to e.g. noise, translation,

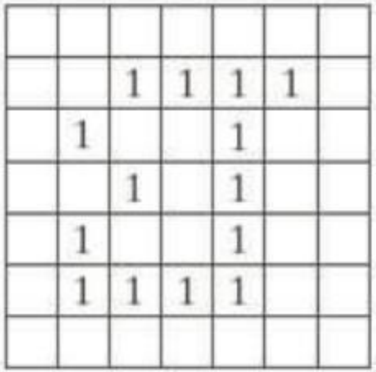
Boundary Following (Moore, 1968)

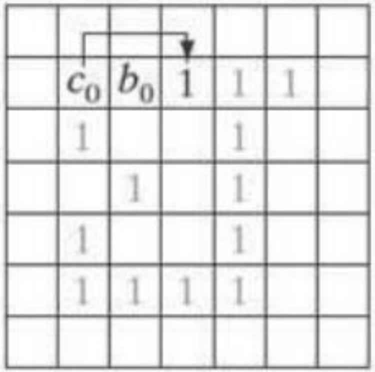
- Given a binary region R or its boundary, the algorithm for following the border of R:
  1. Let the starting point b_0, be the uppermost, leftmost point in the image labeled 1.
  2. Denote by c_0 the west neighbor of b_0. c_0 is always a background point.



Boundary Following

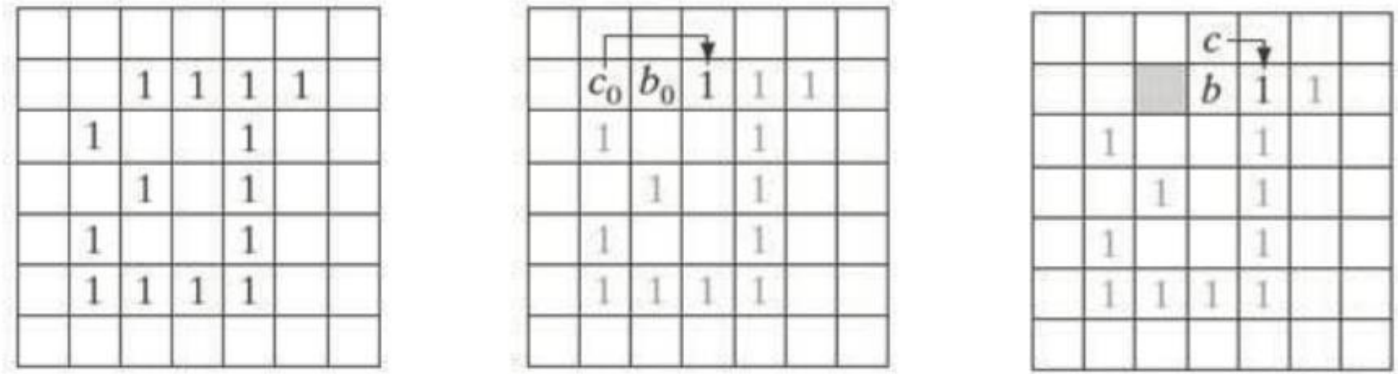
3. Examine the 8-neighbors of b_0, starting at c_0 and proceeding in a clockwise direction.
4. Let b_1 denote the first neighbor encountered whose value is 1.
5. Let c_1 denote the background point immediately preceding b_1 in the sequence.





Boundary Following

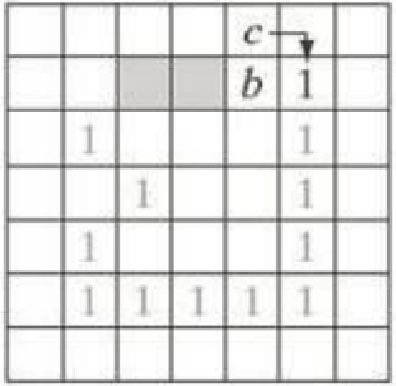
6. Store the locations of b_0 and b_1 for use in Step 10.
7. Let $b=b_1$ and $c=c_1$.
8. Let the 8-neighbors of b, starting at c and proceeding in a clockwise direction, be denoted by $n_1, n_2, \ldots, n_8$. Find the first n_k labeled 1.

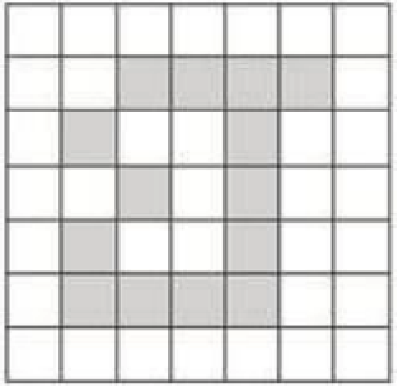


Boundary Following

9. Let $b=n_k$ and $c=n_{k\text{-}1}$.
10. Repeat steps 8 and 9 until $b=b_0$, that is, we have reached the first point and the next boundary point found is b_1.

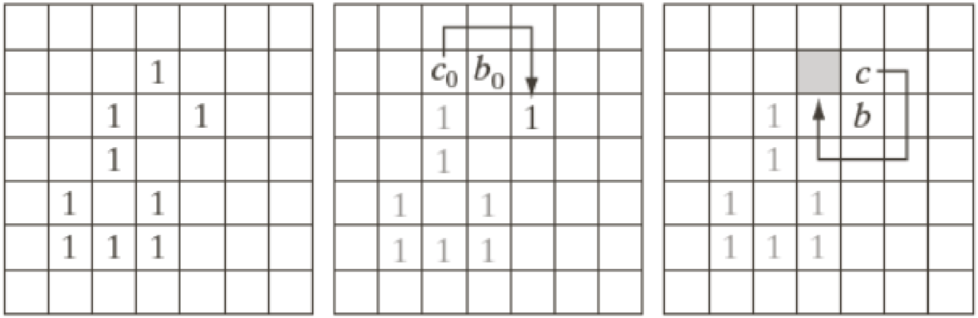
- The algorithm is due to G. Moore [1968]





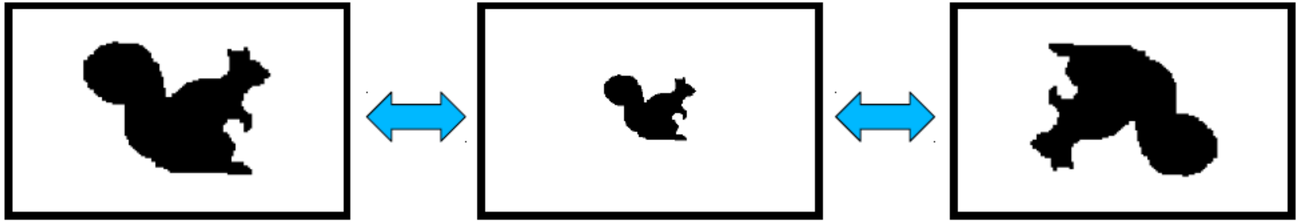
Boundary Following

- The need for the stopping rule "… and the next boundary point found is b_1" is shown below.
- We would only include the spur at the right if we stop when we reach the initial point without checking the next point.



Descriptor Invariance

- What is invariance?
- What invariances are desirable for object recognition?



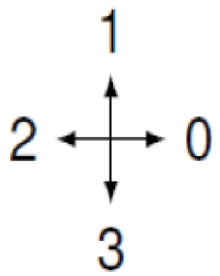
- Not always ideal! (OCR : b vs. P)

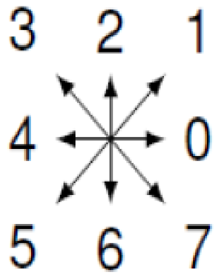
Boundary Description

- Chain Code (Freeman Chain Code)

- Boundary = A connected sequence of straight line segments of specified length and direction.

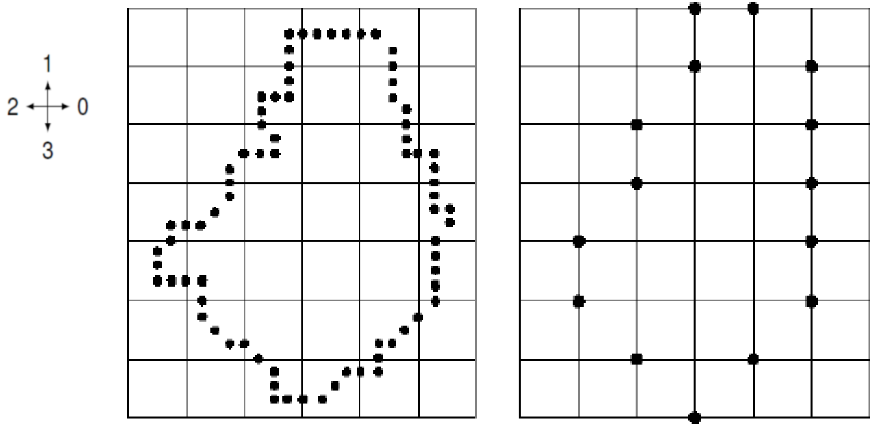
- The direction of each segment is coded by using a numbering scheme such as the ones shown

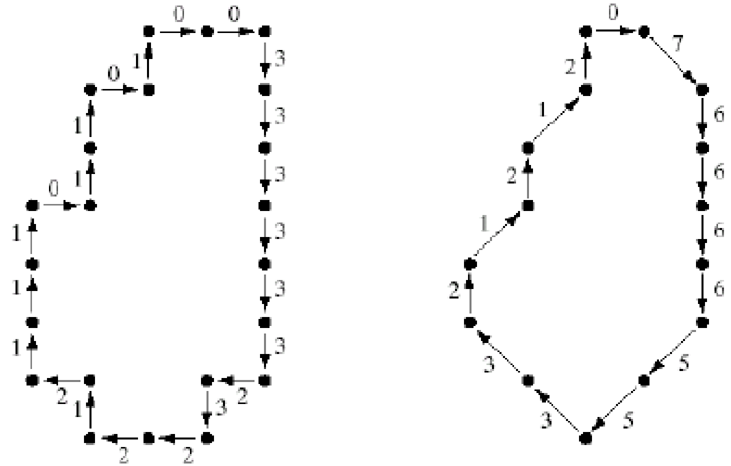




(Freeman) Chain Code

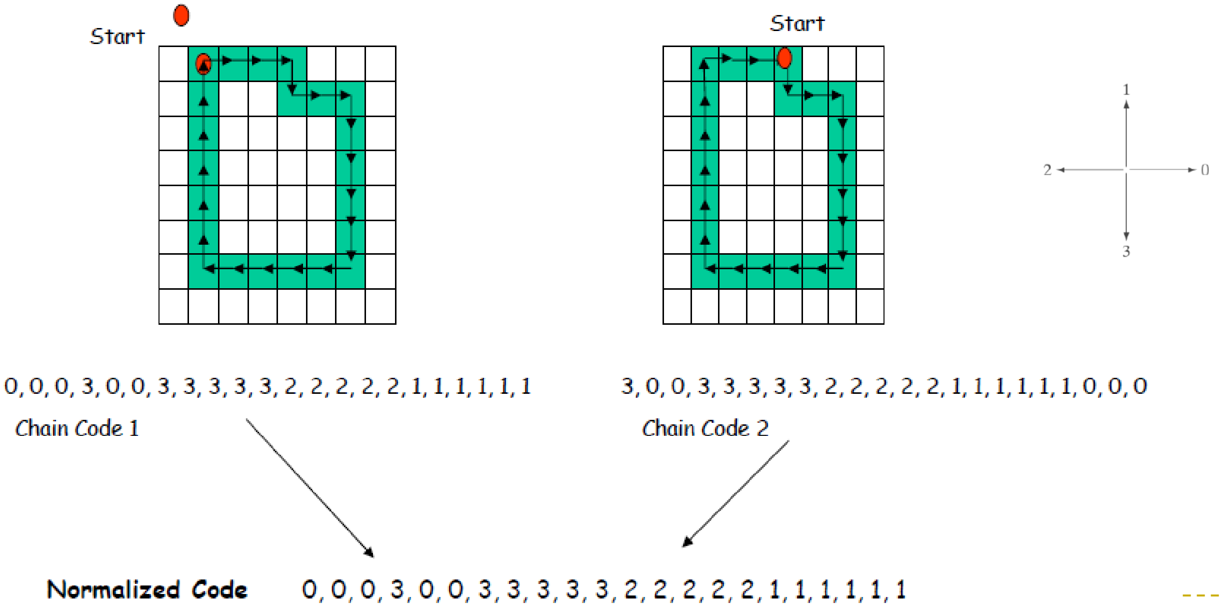
- Long and error-prone to noise
- Solution – downsample





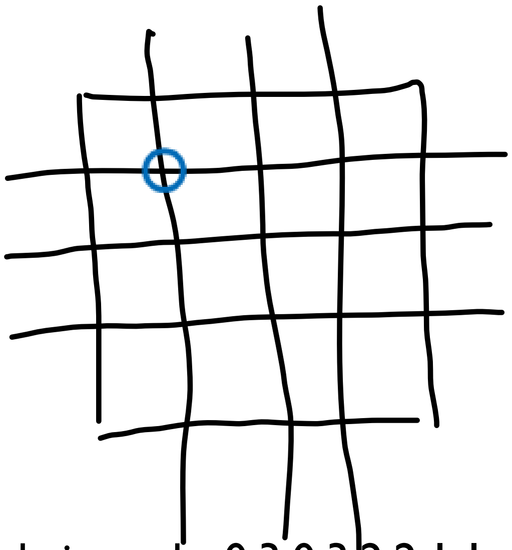
Invariance

- Initialization
  - Treat code as circular, Start with Minimum Integer

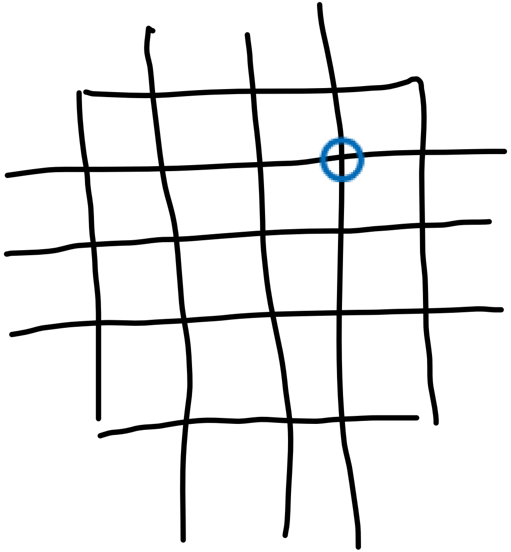


Invariance

- Rotation
  - Instead of directly using code, use first difference of code



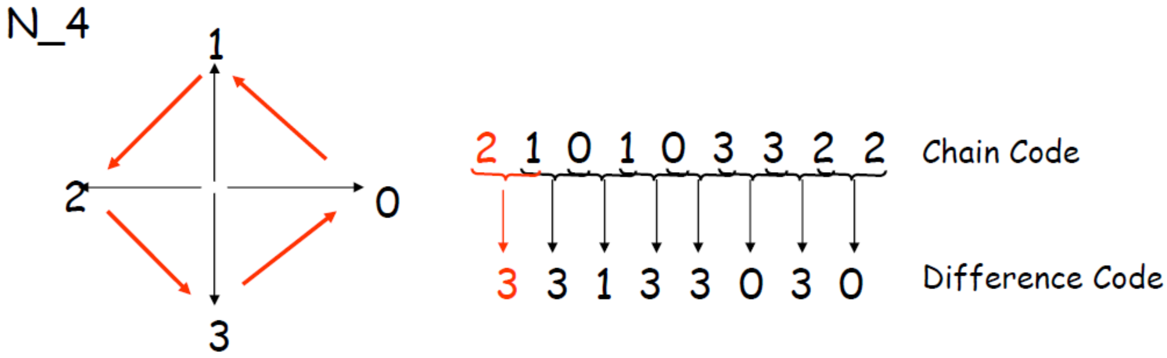
chain code: 0,3,0,3,2,2,1,1



chain code: 3,2,3,2,1,1,0,0

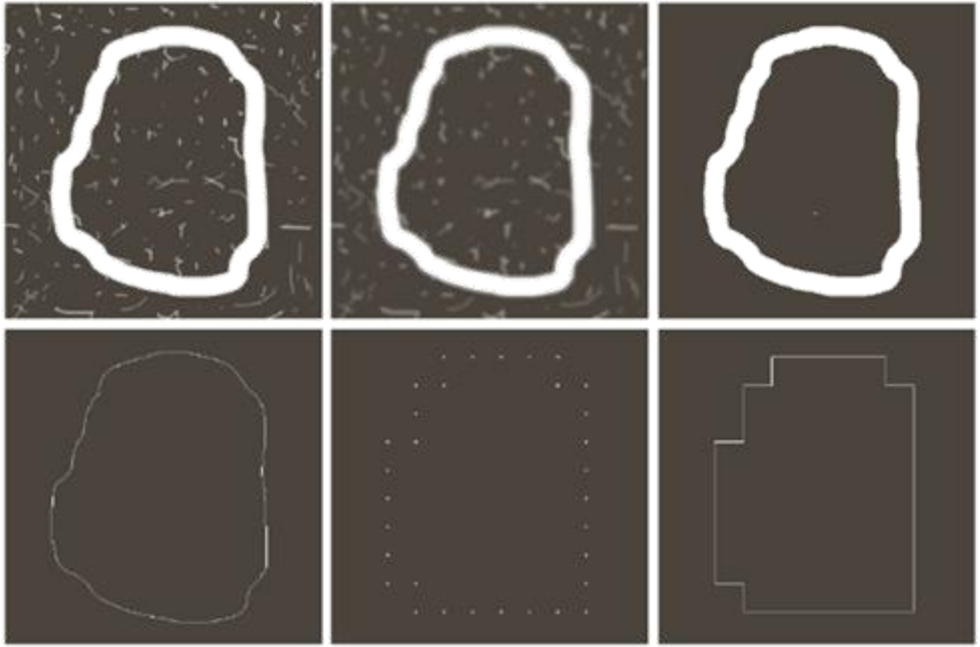
Invariance

- Rotation
  - Instead of directly using code, use first difference of code



Difference: Count the number of separating directions in an anti-clockwise fashion

Example



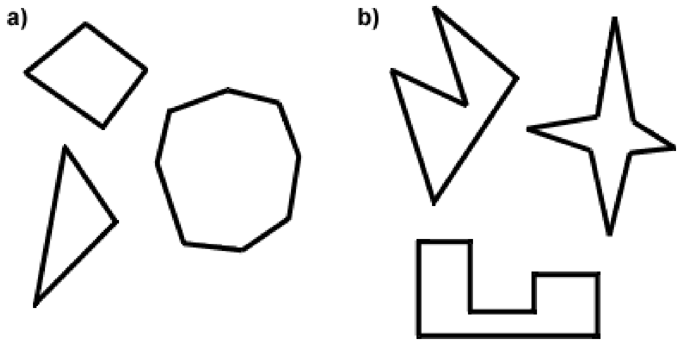
a b c
d e f

FIGURE 11.5 (a) Noisy image. (b) Image smoothed with a 9×9 averaging mask. (c) Smoothed image, thresholded using Otsu's method. (d) Longest outer boundary of (c). (e) Subsampled boundary (the points are shown enlarged for clarity). (f) Connected points from (e).

Polygon Approximations

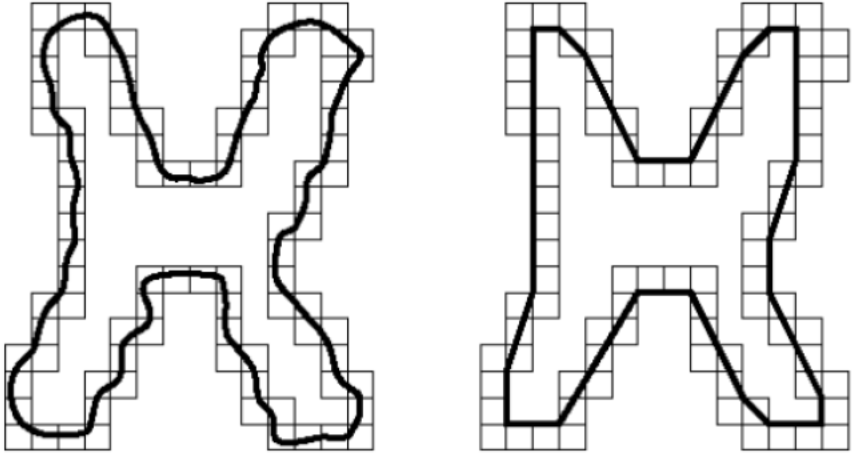
- Digital Boundary as Polygon
  - Simplified Representation
  - Exact when polygon segments = no. of points

- Convex vs. Concave Polygon
  - Interior angle < 180
  - Inside/Outside Test

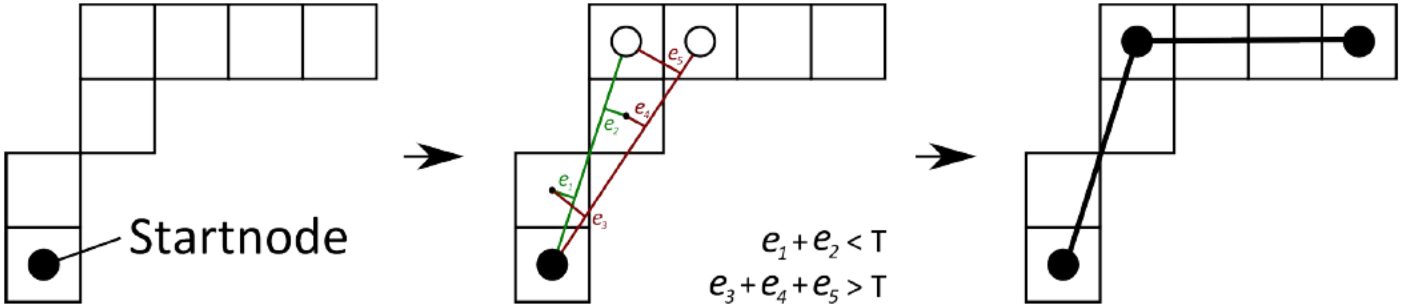


Polygon Approximation

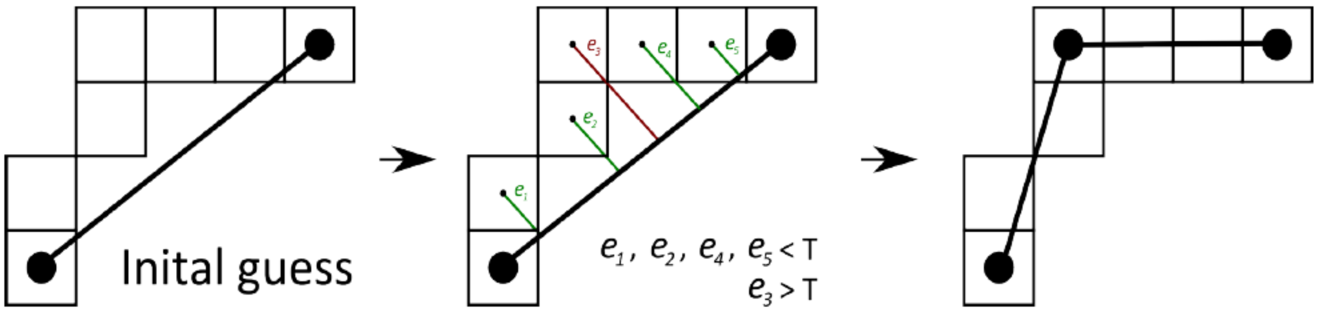
- Minimum Perimeter Polygons
  - Cover the boundary with cells of a chosen size and force a rubber band like structure to fit inside the cells



- **Merging techniques**
  1. Walk around the boundary and fit a least-square-error line to the points until an error threshold is exceeded
  2. Start a new line, go to 1
  3. When the start point is reached the intersections of adjacent lines are the vertices of the polygon



- **Splitting techniques**
  1. Start with an initial guess
  2. Calculate the orthogonal distance from lines to all points
  3. If maximum distance > threshold, create new vertex there
  4. Repeat until no points exceed criterion



Convex hull, deficiency and concavity tree

Convex Hull = minimal enclosing convex region

Convex region = all points can be connected through a straight line inside the region

Convex deficiency = Convex hull – object

The number and distribution of convex deficiency regions may also be useful

=› Concavity tree, generate convex hulls and deficiencies recursively to create at concavity tree

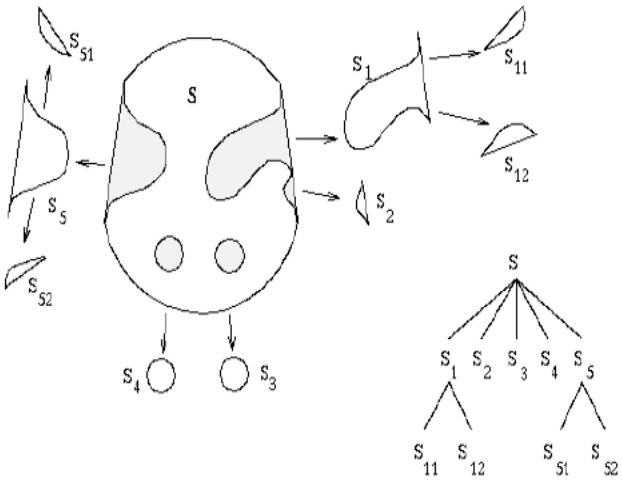
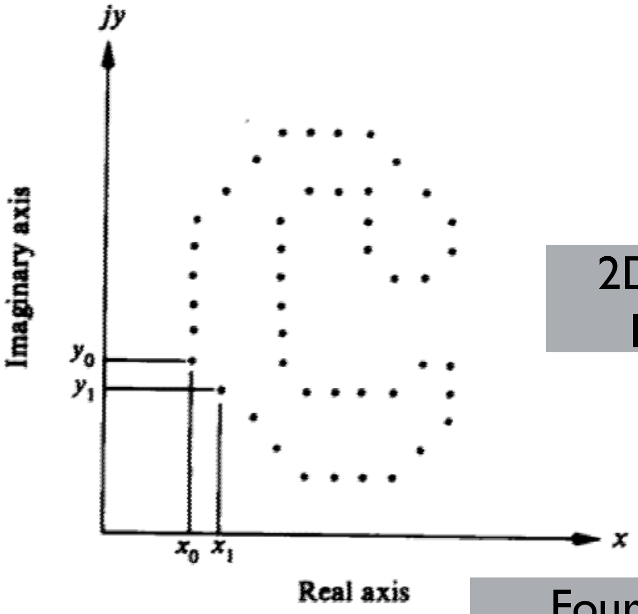


Figure 6.30 *Concavity tree construction: (a) Convex hull and concave residua, (b) concavity tree.*

Boundary description: Fourier Descriptors

- Boundary as a set of points



K point boundary (starting at x_0, y_0):
$(x_0, y_0), (x_1, y_1), (x_2, y_2), \ldots, (x_{K-1}, y_{K-1})$

Can be expressed as $\mathrm{x(k)} = x_k$ and $\mathrm{y(k)} = y_k$

or $\mathrm{s(k)} = [\mathrm{x(k)}, \mathrm{y(k)}],\ k = 0,1,2, \ldots, K-1$

2D as 1D

Treat as a complex number:

$$\mathrm{s(k)} = \mathrm{x(k)} + j\ \mathrm{y(k)}$$

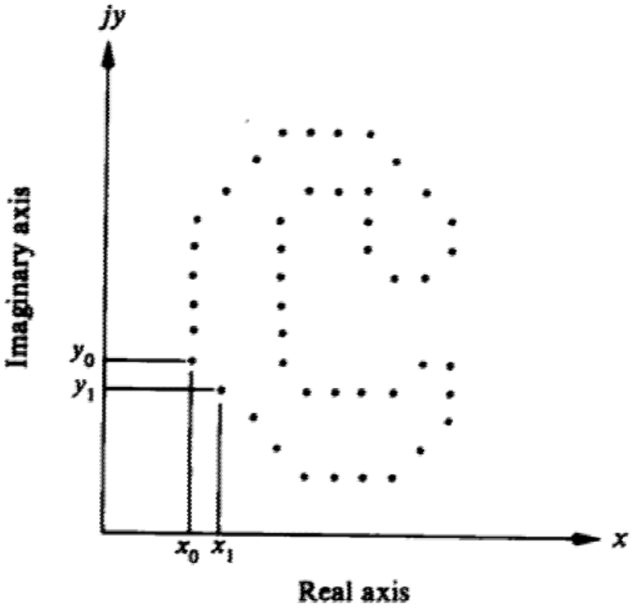
DFT of $\mathrm{s(k)}$:

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

Fourier Descriptor

Fourier Descriptors

- Boundary as a set of points



DFT of s(k):

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

Inverse DFT to restore s(k):

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

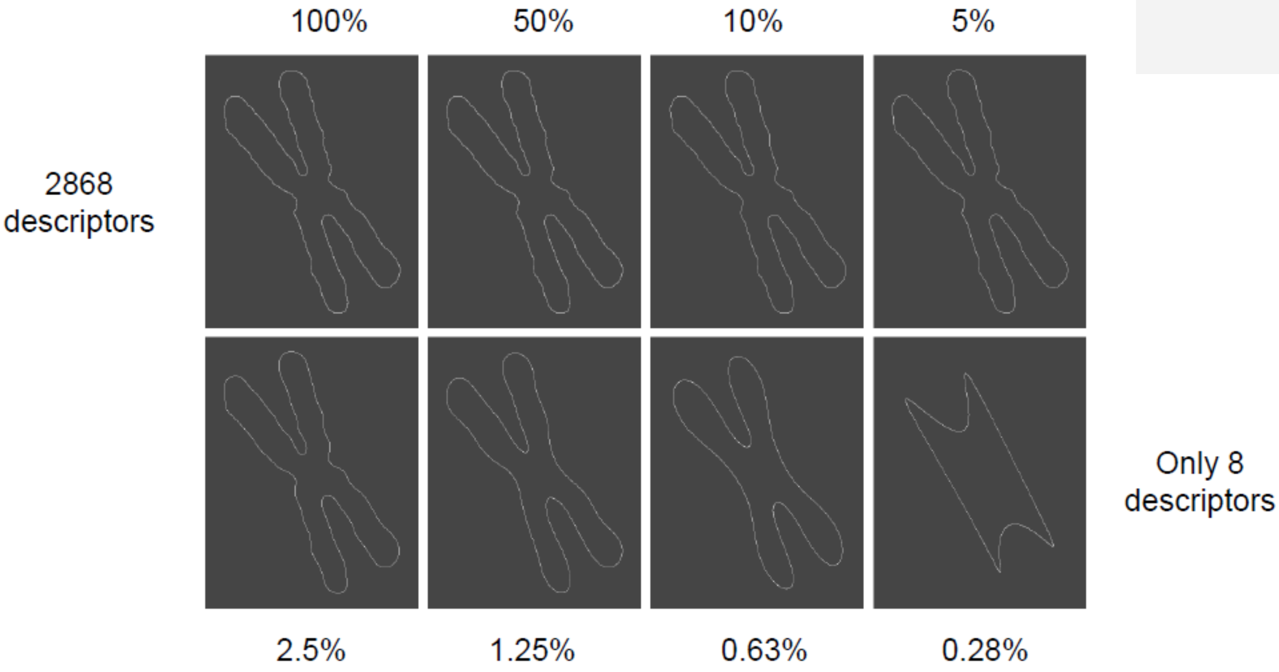
Use only first P coefficients in inverse DFT

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

Fourier Descriptors

- Use only P coefficients for inverse DFT

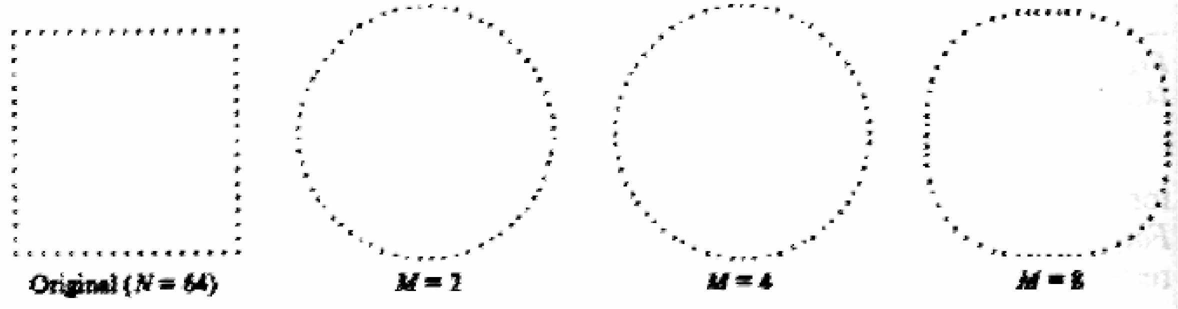
$$\hat{s}(\mathrm{k}) = \frac{1}{P}\sum_{u=0}^{P-1} a(u)e^{j2\pi uk/P}$$



Fourier Descriptors (take away)

1. We only need a few descriptors to capture the gross shape

2. Low order coefficients can be compared to measure the similarity of shapes



Other Boundary Descriptors

- Boundary as 1D signature
- Statistical Moments
- Make your own …

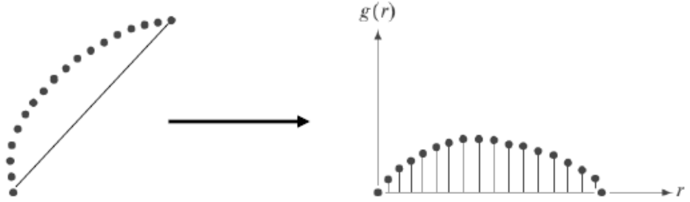
Statistical Moments

Boundary Description using Statistical Moments

$$\mu_n(v) = \sum_{i=0}^{A-1} (v_i - m)^n p(v_i)$$

n-th moment of v

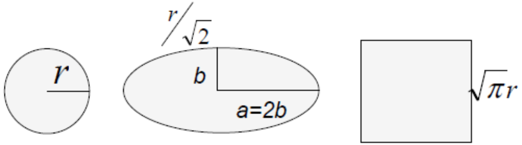
$$m = \sum_{i=1}^{A-1} v_i p(v_i)$$



Internal Descriptors

Region Descriptors - Simple

- Area
- Perimeter
- Compactness → (perimeter)^2/Area
- Circularity Ratio
- Mean/Median intensity
- Max/Min intensity
- Normalized area



$C:$	4π	5π	16
$R_c:$	1	$4/5 \approx 0.8$	$\pi/4 \approx 0.78$

$$R_c = \frac{A}{P^2/4\pi}$$

Area of circle with same perimeter as the shape

Region Descriptors

- Statistical Descriptors
- Topological Descriptors
- Dimensionality Reduction
- Graph-based

Modern Descriptors

- Point Descriptors : SIFT, SURF, DAISY, LBP
- Region Descriptors : HOG, MSER
- Global Descriptors : Bag of Words, GIST
- Introduction to Learned Representation

References

- G&W (11.1 and 11.2)