Lecture

Image Segmentation: Edge Detection

Basics

- Edge detection is an important tool in image analysis, and is necessary for applications of computer vision in which objects need to be recognized by their outlines.

- An edge detection algorithm should show the locations of major edges in the image while ignoring false edges caused by noise.

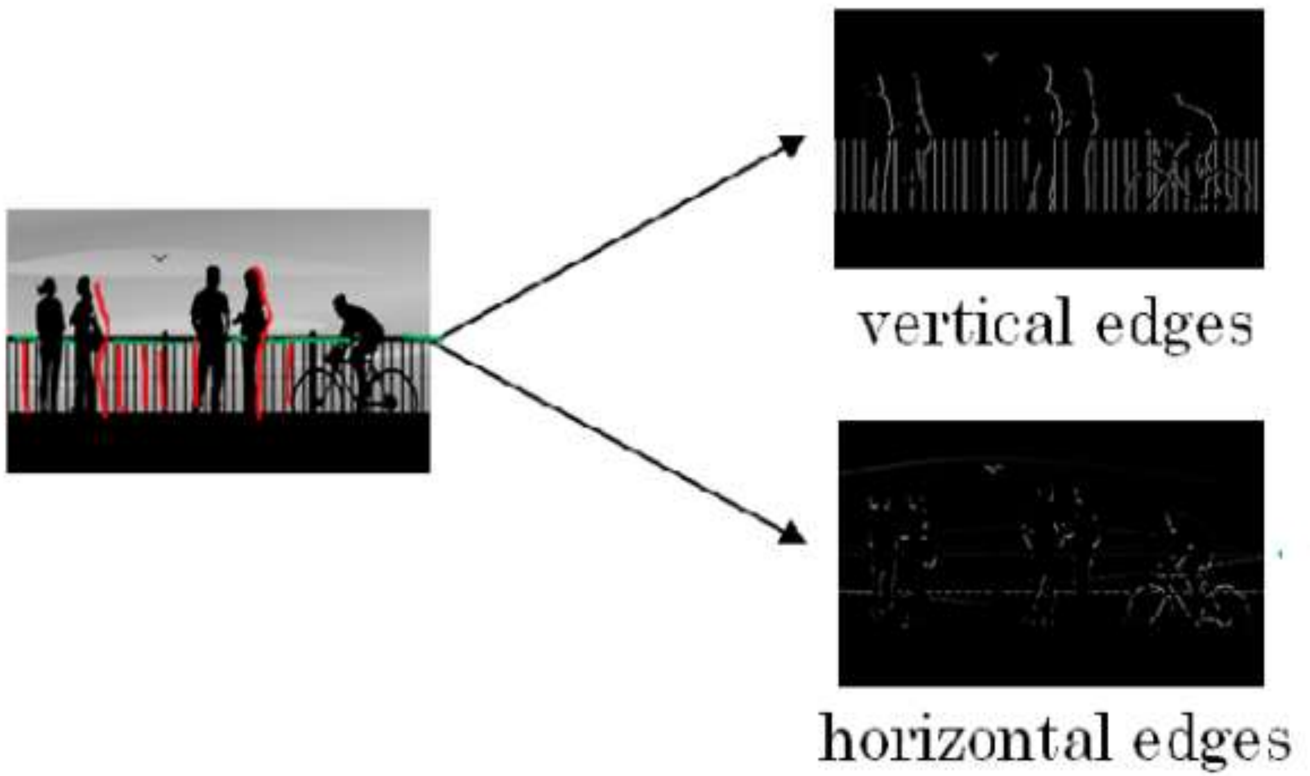
- It involves identifying and locating sharp discontinuities in an image, which typically correspond to significant changes in intensity or color.

Definition: image segmentation

- **Edges:** Edges are abrupt changes in intensity, discontinuity in image brightness or contrast; usually edges occur on the boundary of two regions.

- An edge can be defined as a set of connected pixels that forms a boundary between two disjoint regions.

- There are three types of edges-
  - Horizontal edges
  - Vertical edges
  - Diagonal edges


vertical edges
horizontal edges

Original Image	Applying Horizontal Mask	Applying Horizontal Mask

Horizontal Edge Detection-

- Detects edges where there is a significant change in intensity in the vertical direction, i.e., from top to bottom.
- Example: In an image, horizontal edges might correspond to boundaries like horizons or rows of text.
- Kernel (Filter): A simple filter for detecting horizontal edges is-

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Contd..

- This filter highlights transitions between dark and light regions along the vertical direction.

- It detects where there is a change in intensity between the top and bottom halves of a region.

Vertical Edge Detection-

- Detects edges where there is a significant change in intensity in the horizontal direction, i.e., from left to right.

- **Example:** Vertical edges may correspond to the edges of buildings, trees, or text columns in an image.

- Kernel (Filter): A simple filter for detecting vertical edges is-

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Consider a small 3x3 grayscale image where each number represents the intensity of a pixel-

$$\text{Image} = \begin{bmatrix} 50 & 50 & 50 \\ 100 & 100 & 100 \\ 200 & 200 & 200 \end{bmatrix}$$

Horizontal Edge Detection Kernel

- The kernel for horizontal edge detection is-

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Contd..

- We apply the kernel to each 3x3 region in the image by performing a convolution. For simplicity, let's apply it to the center pixel, assuming the edges are padded.

- Convolution result for the center pixel (2nd row, 2nd column)-

- ???????

- The high value (450) indicates a strong horizontal edge between the middle and bottom rows.

Vertical Edge Detection Kernel

- The kernel for vertical edge detection is:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

- Apply the kernel to the center pixel-

- The result is 0, meaning there is no vertical edge in this region since the image intensity changes only along the horizontal direction.

Diagonal Edge-

- Consider a new matrix that includes a diagonal intensity change-

$$\text{Image} = \begin{bmatrix} 50 & 100 & 150 \\ 100 & 150 & 200 \\ 150 & 200 & 250 \end{bmatrix}$$

Contd..

- The kernel for 45-degree diagonal edge detection is-

$$\text{45-degree Kernel} = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$$

Contd..

- The result is 0, indicating no strong diagonal edge for this pixel.

Contd..

Edge Detection Operators are of two types:

1. Gradient – based operator which computes first-order derivations in a digital image like, Sobel operator, Prewitt operator, Robert operator

2. Gaussian – based operator which computes second-order derivations in a digital image like, Canny edge detector, Laplacian of Gaussian

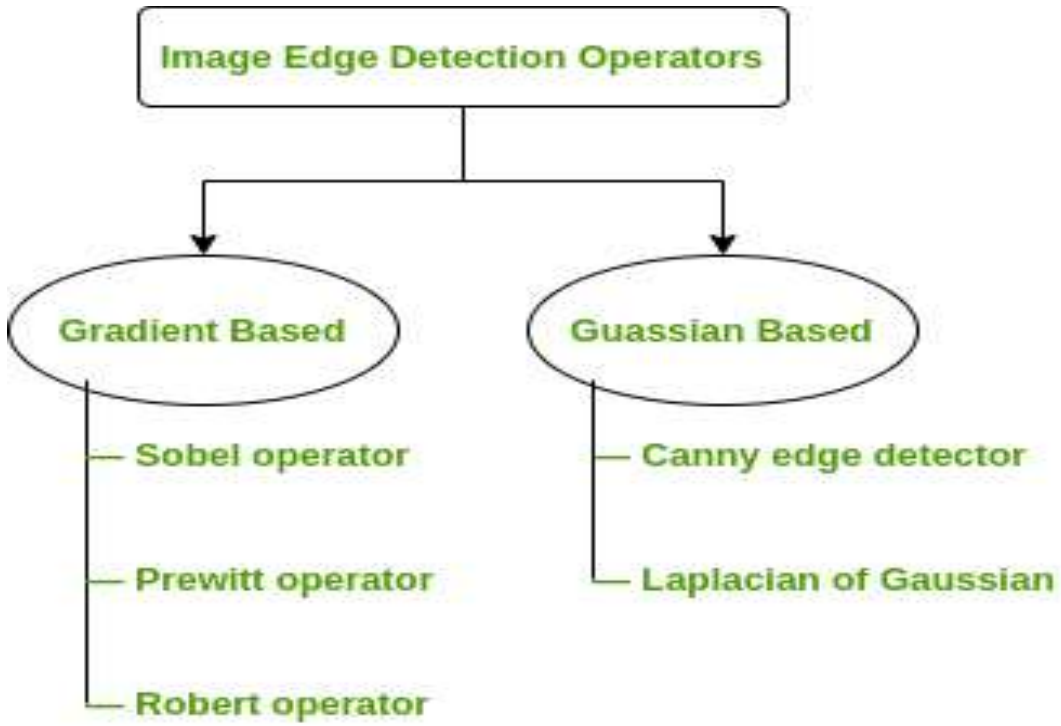


Image Edge Detection Operators
Gradient Based
Guassian Based
Sobel operator
Prewitt operator
Robert operator
Canny edge detector
Laplacian of Gaussian

$$T(x,y) = \text{function}(\text{block}(x,y)) - C$$

Where,

- T(x,y) is the threshold value for the pixel located at coordinates (x,y).
- block(x,y) refers to a local window (or block) of pixels centered around the pixel (x,y). This block is used to determine the local characteristics of the image.
- function(block(x,y)) can be the average (mean) or weighted sum of the pixel intensities within the block.
- C is a constant value that is subtracted from the calculated threshold to fine-tune the result.

- Consider a 5×5 grayscale image with the following pixel intensities-

$$\begin{bmatrix} 120 & 125 & 130 & 135 & 140 \\ 118 & 122 & 128 & 136 & 145 \\ 116 & 119 & 127 & 138 & 150 \\ 112 & 117 & 129 & 137 & 148 \\ 110 & 114 & 126 & 134 & 146 \end{bmatrix}$$

Solution-1. Adaptive Threshold Calculation for Pixel at (2,2)

- Step 1: Identify the 3×3 block centered around the pixel at (2,2)-

$$\begin{bmatrix} 120 & 125 & 130 \\ 118 & 122 & 128 \\ 116 & 119 & 127 \end{bmatrix}$$

- Step 2: Calculate the average intensity of the block-

$$\text{Average} = \frac{120 + 125 + 130 + 118 + 122 + 128 + 116 + 119 + 127}{9} = \frac{1105}{9} \approx 123$$

- Step 3: Calculate the adaptive threshold using the formula = T(x,y) =Average−C

  T(2,2)=123−5=118

Step 4: Apply the threshold to the pixel at (2,2)

- The pixel value (128) is greater than the threshold (118), so it would be set to 1 (white).

2. Adaptive Threshold Calculation for Pixel at (4,4)

Step 1: Identify the 3×3 block centered around the pixel at (4,4)

Step 2: Calculate the average intensity of the block:

Step 3: Calculate the adaptive threshold using the formula = T(x,y) =Average−C

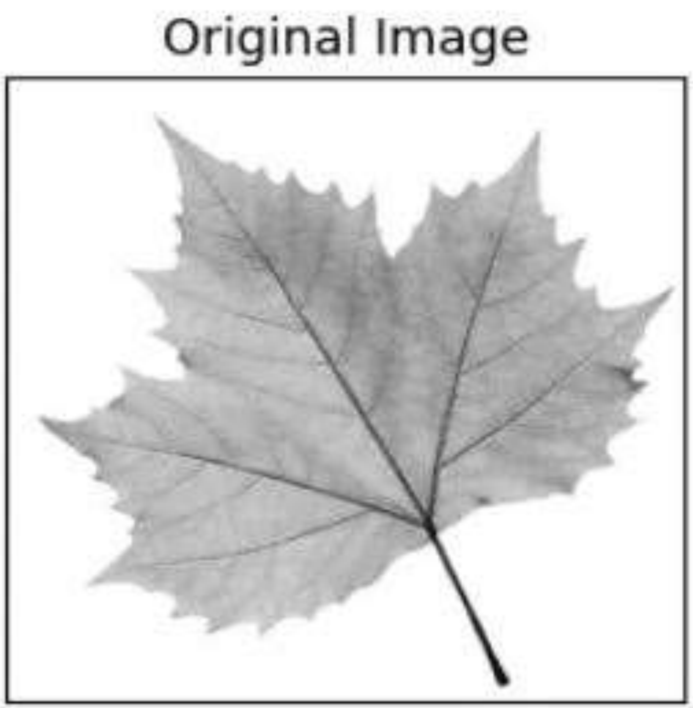
Step 4: Apply the threshold to the pixel at (4,4)

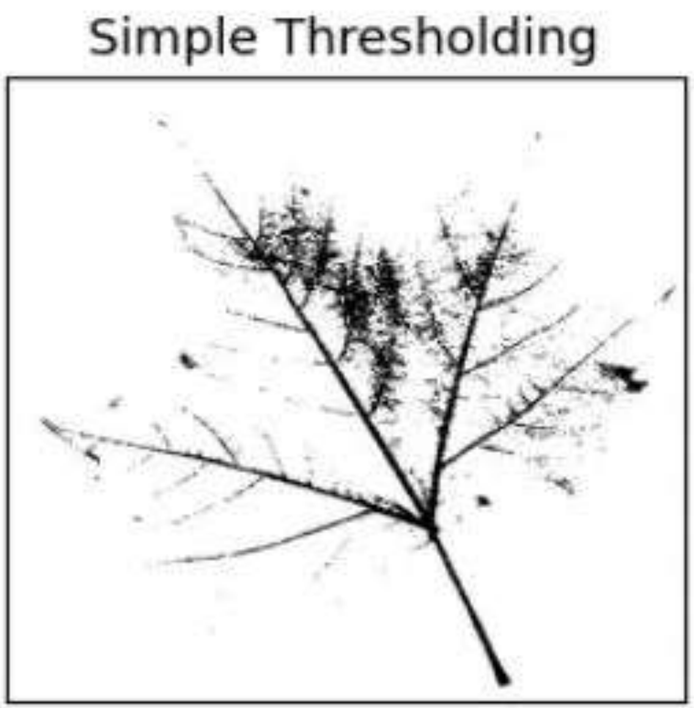
Contd..

Summary of Threshold Calculations

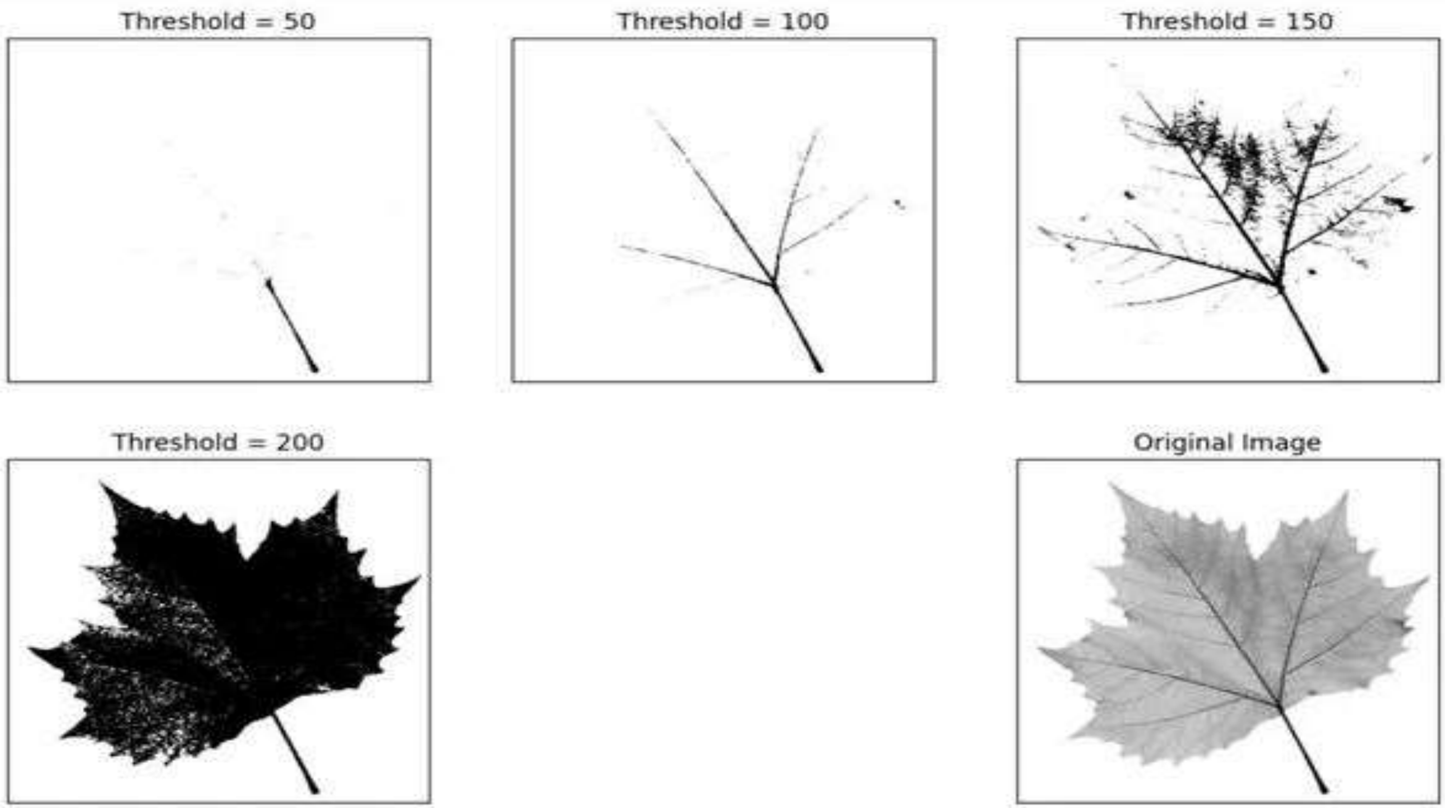
- Pixel at (3,3): Threshold = 112, Pixel value = 127, Result = 1 (white)
- Pixel at (2,2): Threshold = 118, Pixel value = 128, Result = 1 (white)
- Pixel at (4,4): Threshold = 132, Pixel value = 137, Result = 1 (white)

Contd..



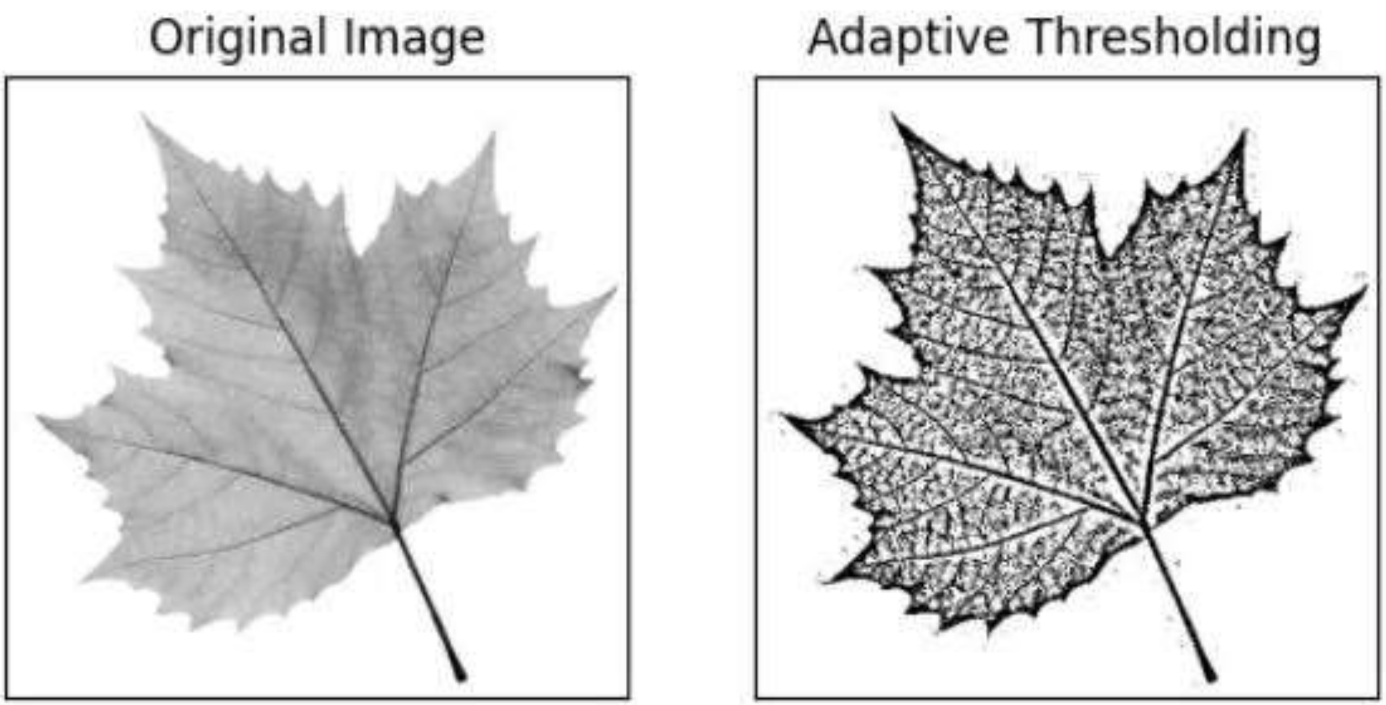


Contd..



Threshold = 50
Threshold = 100
Threshold = 150
Threshold = 200
Original Image

Contd..



Region-based Segmentation

3. Otsu's Method

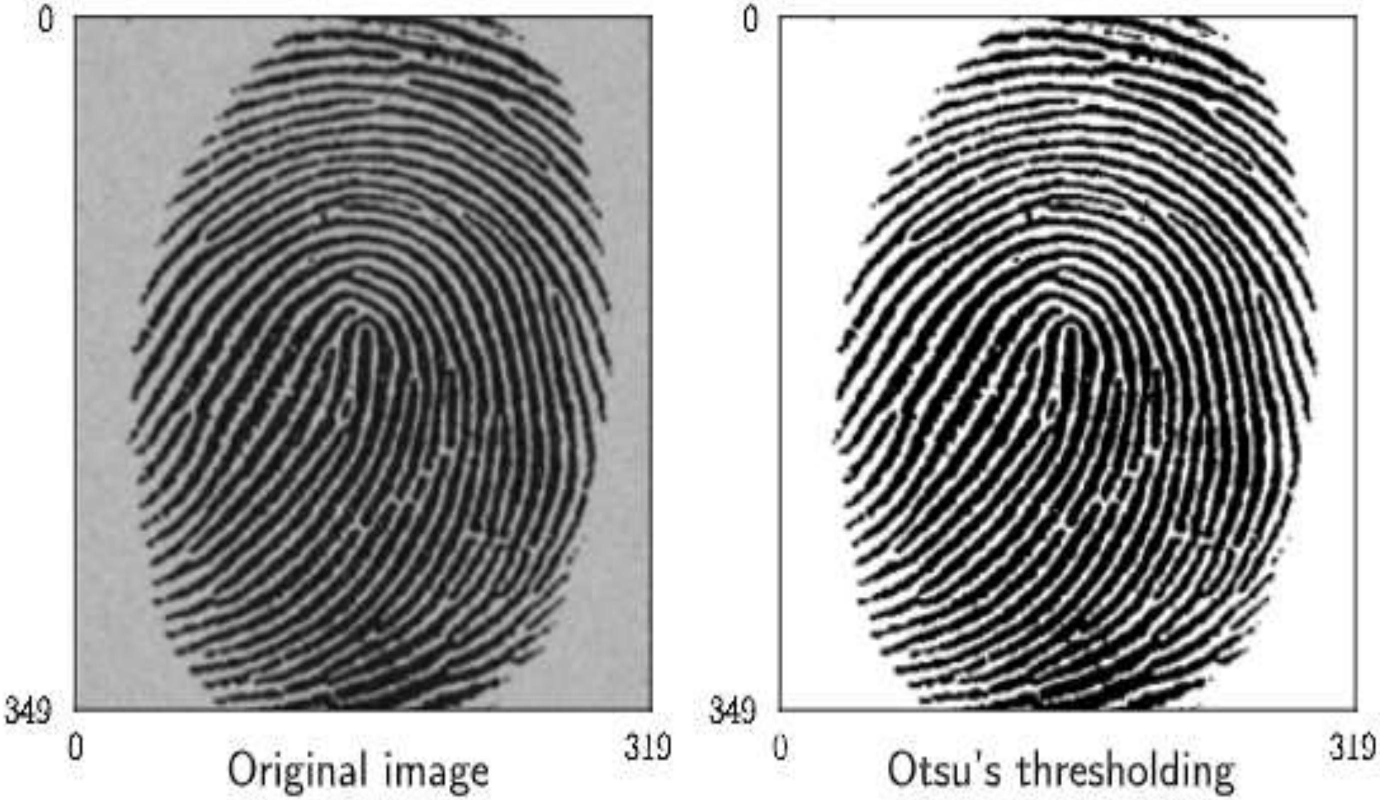
- An automatic thresholding technique that determines the optimal threshold value by minimizing the within-class variance or maximizing the between-class variance.

- Particularly useful when the image has a bimodal histogram (two peaks representing object and background).

- Otsu's approach is optimum in the sense that it minimizes the in-class variance or maximize between class variance, where a class means a set of pixel belonging to the region.

Steps

1. Compute the Grayscale Histogram
2. Compute the Cumulative Distribution Function
3. Compute the Mean Grayscale Intensity Value of the Image
4. Compute the Between-Class Variance for Each Possible Threshold Value
5. Find the Threshold Value That Maximizes the Between-Class Variance


0
349
0
319
Original image
0
349
0
319
Otsu's thresholding

Reference

- Books: "Digital Image Processing" by Rafael C. Gonzalez and Richard E. Woods.