Informatique et systèmes de communication ISC

Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

Image Segmentation

Jaime Barranco
jaime.barranco@hes-so.ch

03.03.2025

Structure

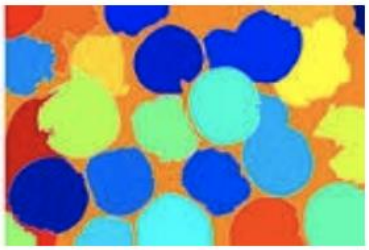
1. Introduction

2. Threshold-based segmentation
   - Histogram extrema
   - Otsu's method
   - K-means algorithm
   - Optimal thresholding
   - Histogram estimation
   - Enhancing threshold segmentation
     - Morphology
     - Local thresholding

3. Edge-based segmentation
   - Edge linking
   - Watershed segmentation
   - Active contours

1. Introduction

- **Part 1 - Image segmentation:**
  - It is the process of partitioning an image into multiple segments.
  - Its goal is to simplify and/or change the content of the image into something easier to analyze.
  - Is an essential step in many computer vision applications, such as:
    - Image analysis.
    - Object representation.
    - Visualization.
    - Etc.



1. Introduction

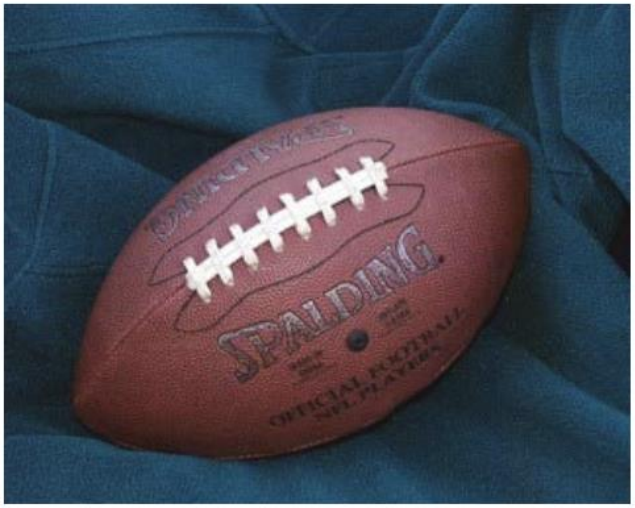
Definition: Image segmentation is the process of partitioning a digital image into multiple segments.

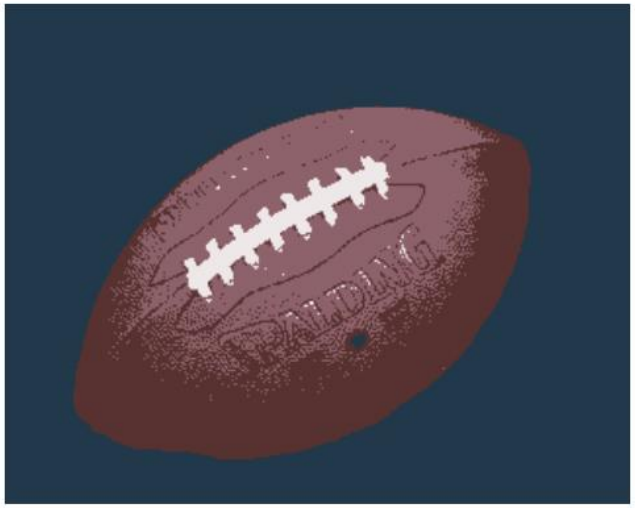
Goal: To simplify and/or change the representation of an image into something more meaningful and easier to analyze.

Result: A label is assigned to each pixel in the image. The pixels with the same label share certain characteristics.

Example:

Color-based segmentation





1. Introduction

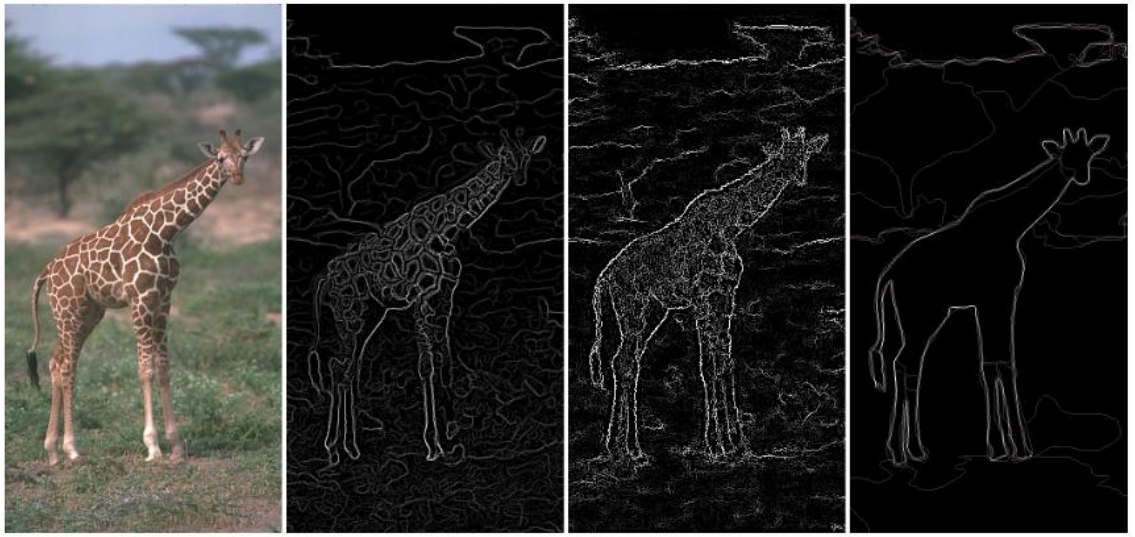
Definition: Image segmentation is the process of partitioning a digital image into multiple segments.

Goal: To simplify and/or change the representation of an image into something more meaningful and easier to analyze.

Result: A label is assigned to each pixel in the image. The pixels with the same label share certain characteristics.

Example:

Edge-based segmentation



1. Introduction

Definition: Image segmentation is the process of partitioning a digital image into multiple segments.

Goal: To simplify and/or change the representation of an image into something more meaningful and easier to analyze.

Result: A label is assigned to each pixel in the image. The pixels with the same label share certain characteristics.

Example:

Texture-based segmentation

1. Introduction

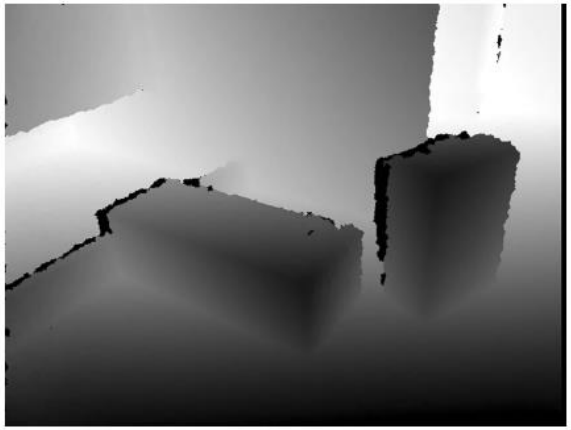
Definition: Image segmentation is the process of partitioning a digital image into multiple segments.

Goal: To simplify and/or change the representation of an image into something more meaningful and easier to analyze.

Result: A label is assigned to each pixel in the image. The pixels with the same label share certain characteristics.

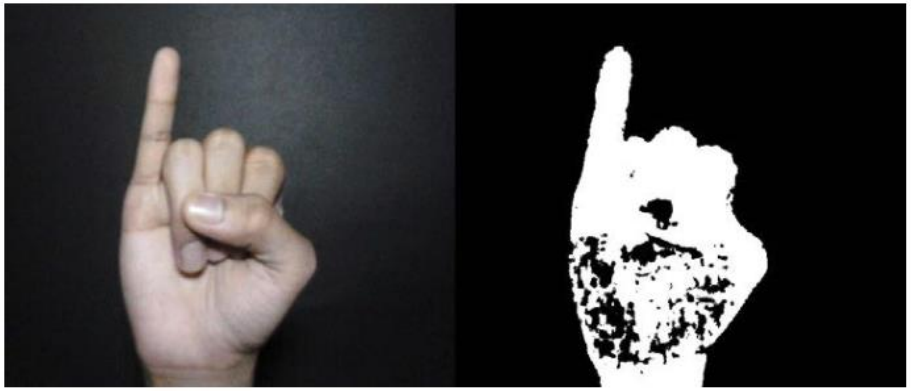
Example:

Depth-based segmentation



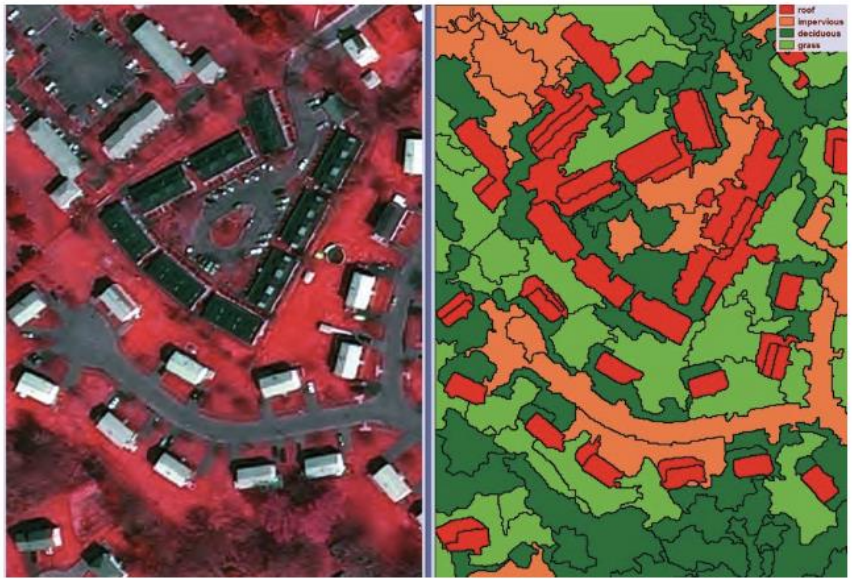
1. Introduction

- **Objective 1:** Decompose the image into parts for further analysis.
  - In simple cases, the characteristics of the objects in the image are well known and it is easy to distinguish the parts that need to be analyzed further.
    - Example: Human face and/or hands segmentation.



1. Introduction

- **Objective 2:** Performing a change of representation.
  - The pixels of the image must be organized into higher-level units that are: more meaningful, more efficient for further analysis or both.

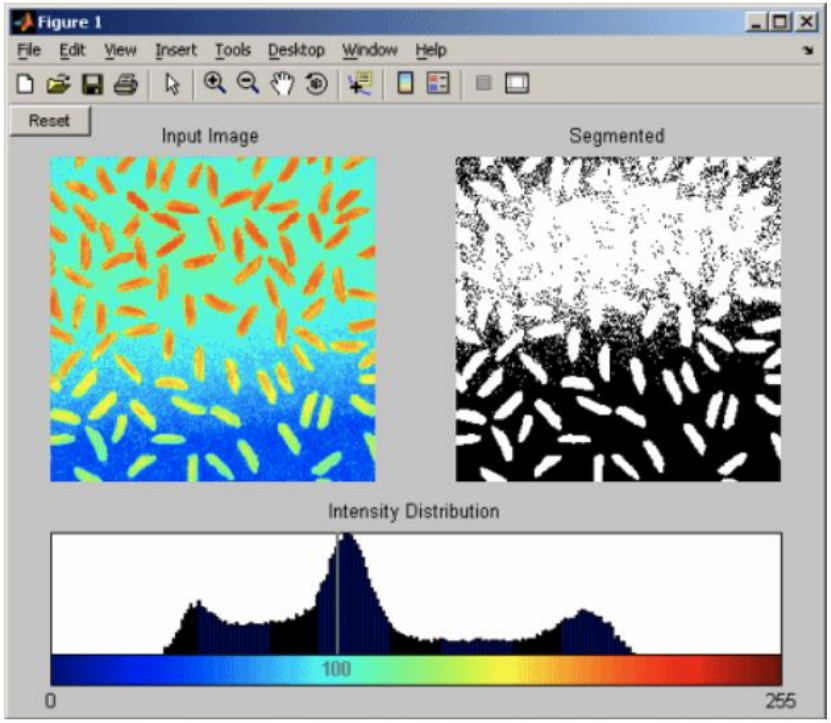


1. Introduction

- **Segmentation categories:**
  - There is a huge amount and variety of existing segmentation algorithms.
  - Typically, the segmentation algorithms can be classified into the following types:
    - Threshold-based segmentation.
    - Edge-based segmentation.
    - Region-based segmentation.
    - Clustering-based segmentation.
    - Matching-based segmentation.

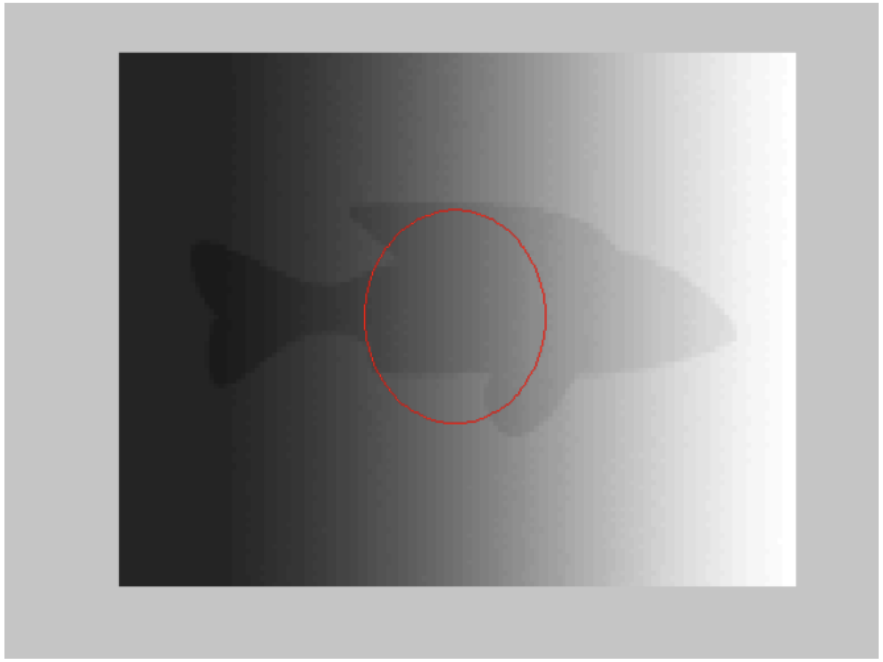
1. Introduction

- **Threshold-based segmentation:**
  - Color histogram thresholding techniques are used to segment the images.
  - They can be applied directly to an image, but they can also be combined with pre- and post-processing techniques.



1. Introduction

- **Edge-based segmentation:**
  - The edges detected in an image are assumed to represent object boundaries.
  - These edges are used to identify the objects of interest.
  - Once the edges have been located, the objects are segmented by filling them in.

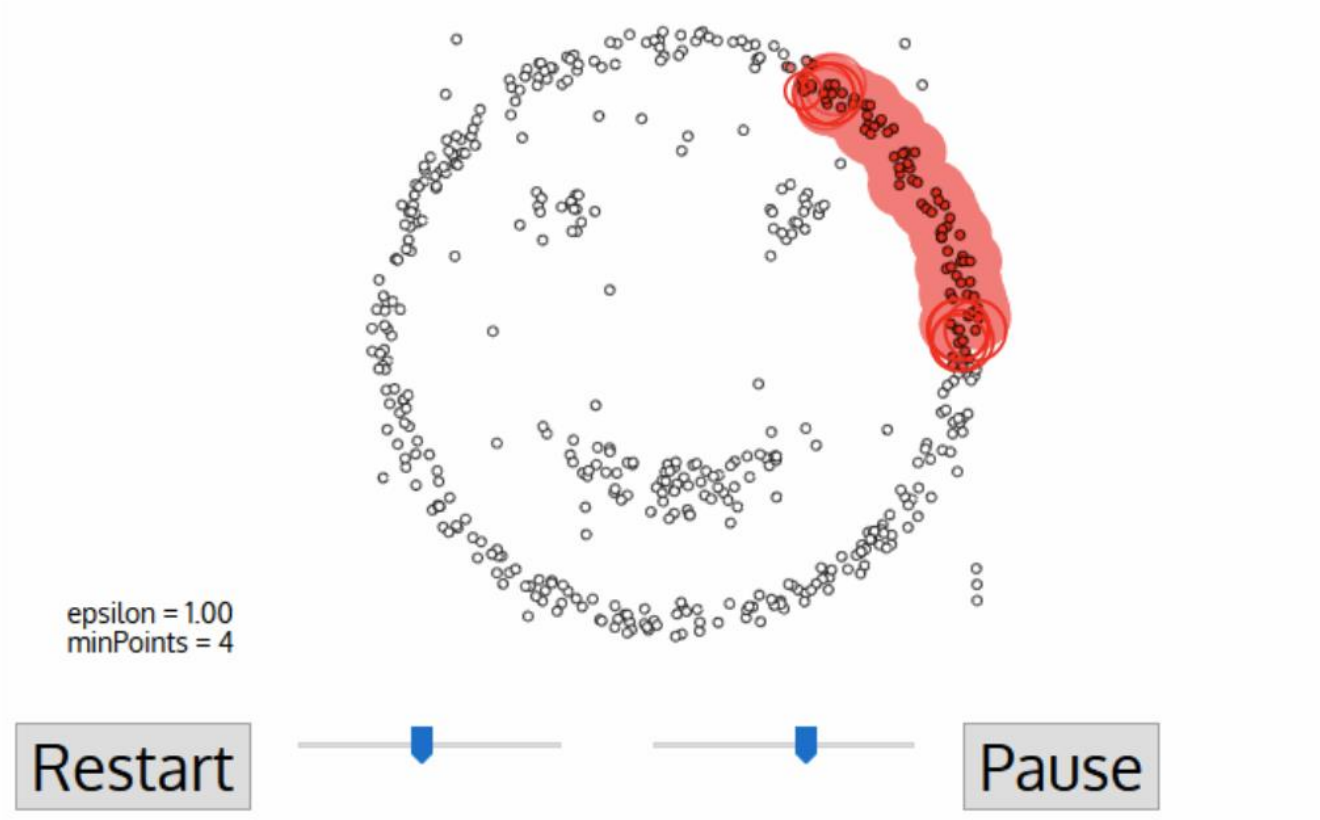


1. Introduction

- **Region-based segmentation:**
  - As opposite to edge-based segmentation, these techniques start in a small region inside an object and then grow until it meets the object boundaries.

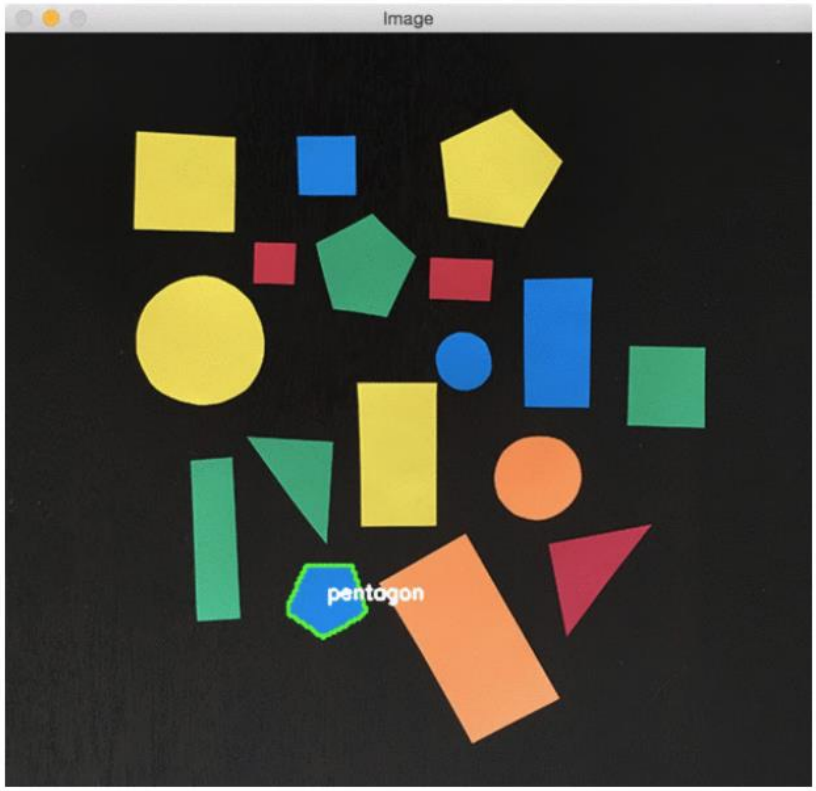
1. Introduction

- **Clustering-based segmentation:**
  - These methods attempt to group patterns that are similar in some sense.



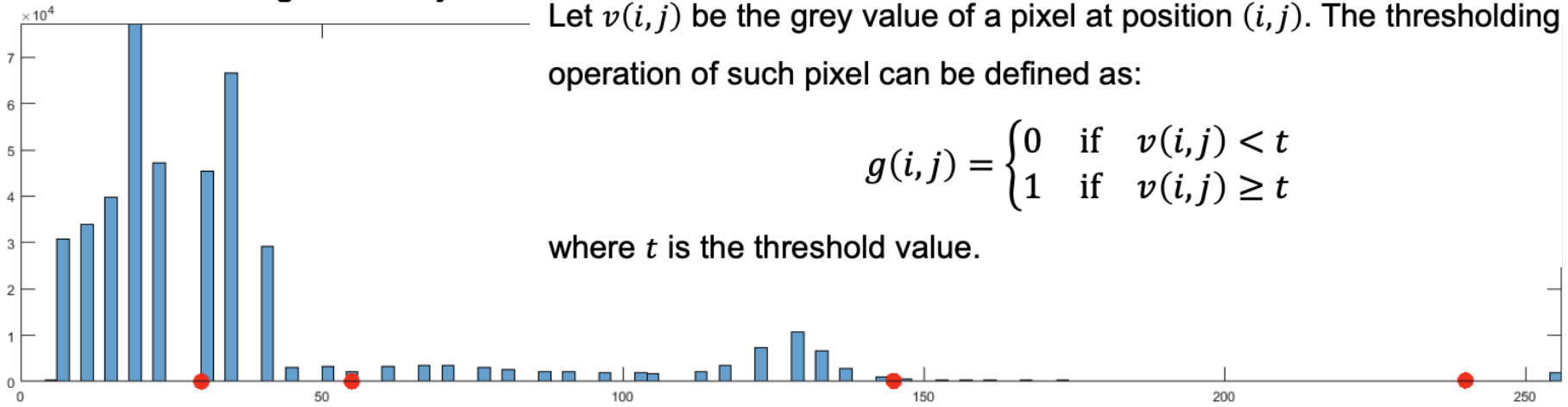
1. Introduction

- **Matching-based segmentation:**
  - Once we know what an object we want to locate looks like, we use this knowledge to locate it in the image.



2. Threshold-based segmentation

- Histogram analysis:



Let $v(i,j)$ be the grey value of a pixel at position (i,j). The thresholding operation of such pixel can be defined as:

$$g(i,j) = \begin{cases} 0 & \text{if} \quad v(i,j) < t \\ 1 & \text{if} \quad v(i,j) \geq t \end{cases}$$

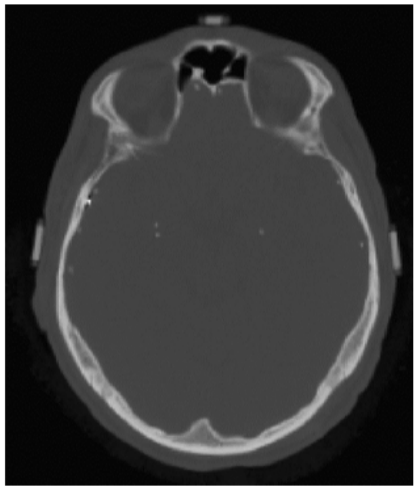
where t is the threshold value.

$t = 30$ $t = 55$ $t = 145$ $t = 240$

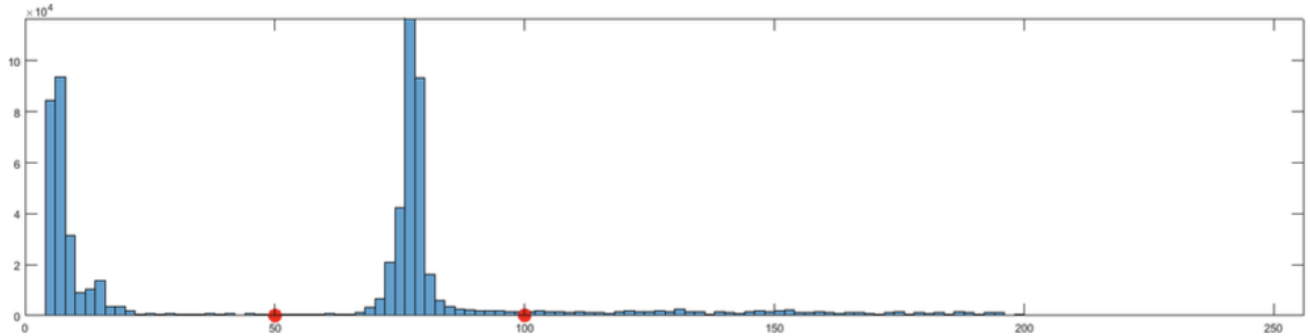
2. Threshold-based segmentation

- When several desired segments can be distinguished by their grey values, several thresholds can be used as follows:

$$g(i,j) = \begin{cases} 0 & \text{if} \quad v(i,j) < t_1 \\ 1 & \text{if} \quad t_1 \le v(i,j) < t_2 \\ 2 & \text{if} \quad t_2 \le v(i,j) < t_3 \\ \vdots & \vdots \quad \vdots \\ n & \text{if} \quad t_n \le v(i,j) \end{cases}$$

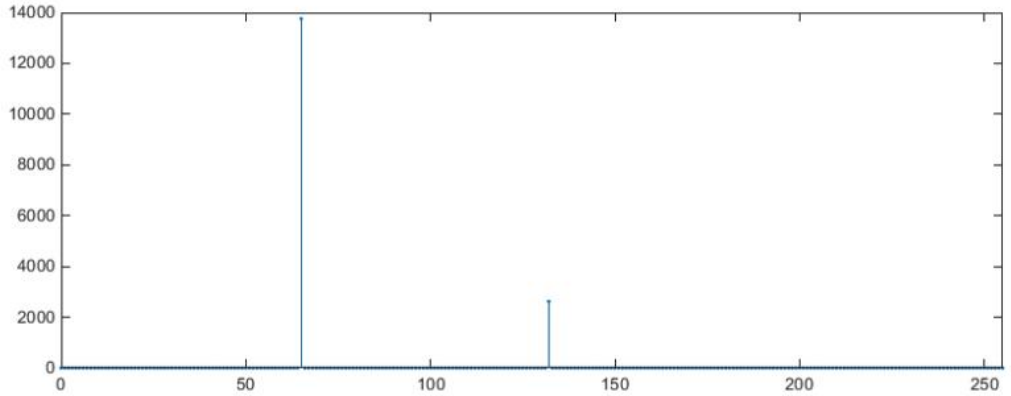


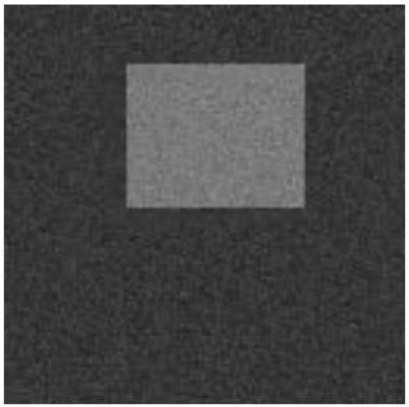
  - The image is segmented into $n + 1$ segments identified by the grey values 0 to n.

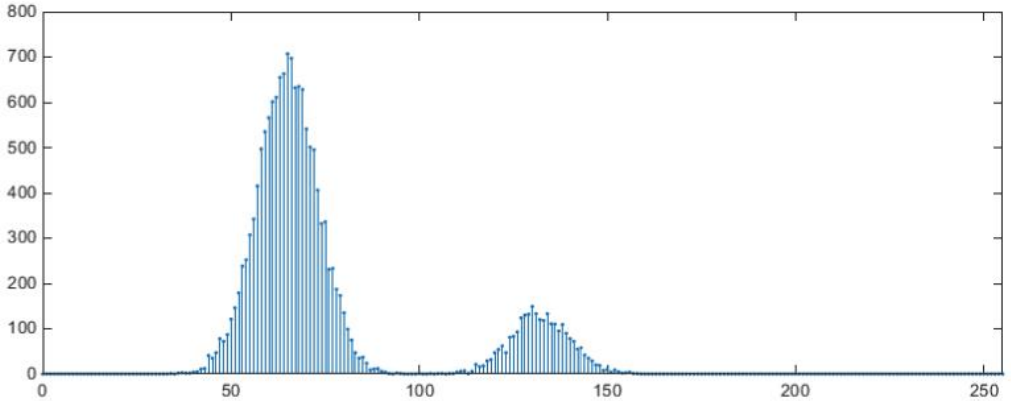


2.1. Histogram extrema

- Typically, the images to segment will have a multimodal (multiple peaks) histogram.





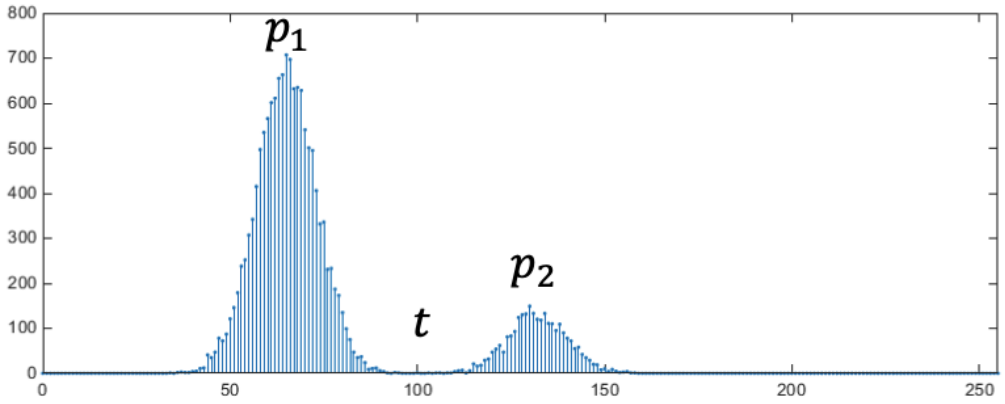


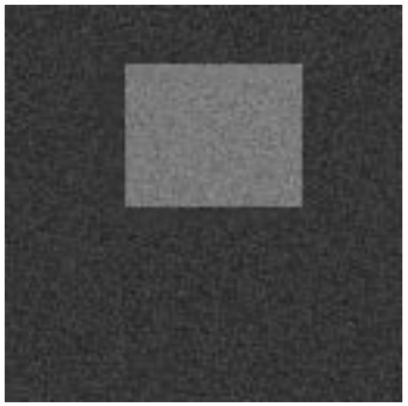
2.1. Histogram extrema

- Option 1: Use the maxima (peaks) to establish the segmentation threshold.
  - It can be set as the value between the two peaks.

$$t = \frac{p_1 + p_2}{2}$$

  - In this example: $t = \frac{65+135}{2} = 100$

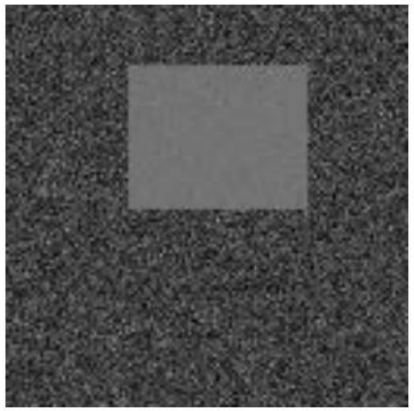




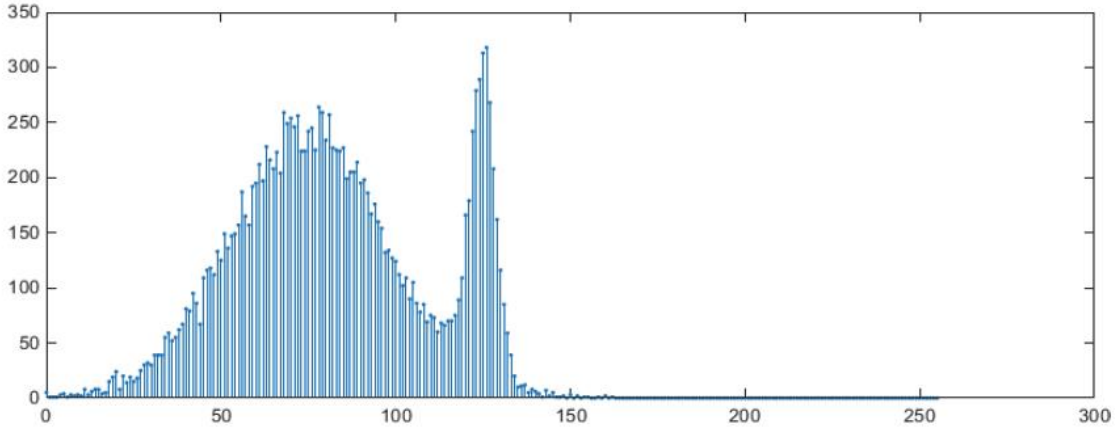
2.1. Histogram extrema

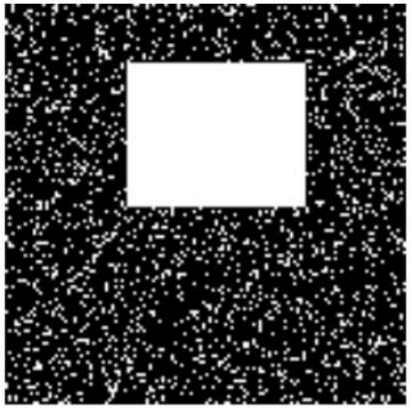
- Option 1: Use the maxima (peaks) to establish the segmentation threshold.
  - It can be set as the value between the two peaks.

$$t = \frac{p_1 + p_2}{2}$$



  - In this another example: $t = \frac{75+125}{2} = 100$





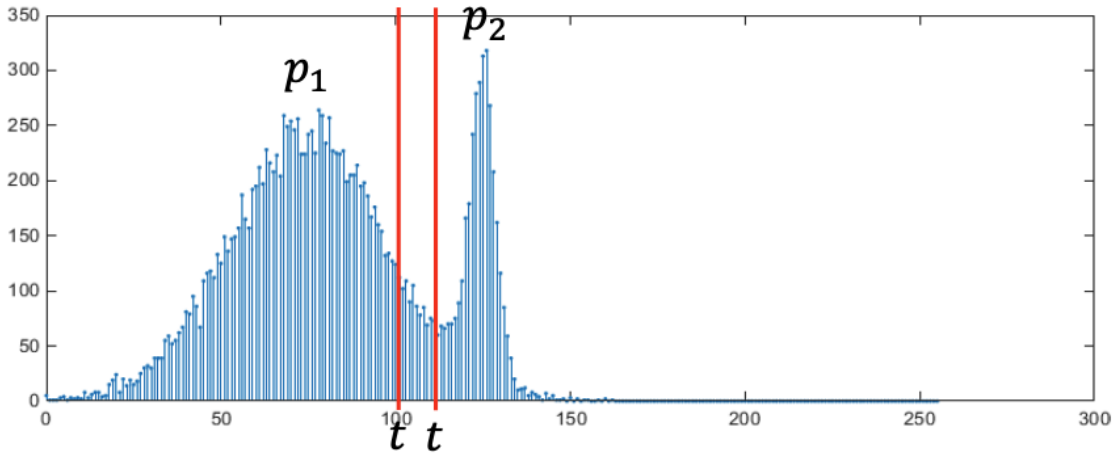
2.1. Histogram extrema

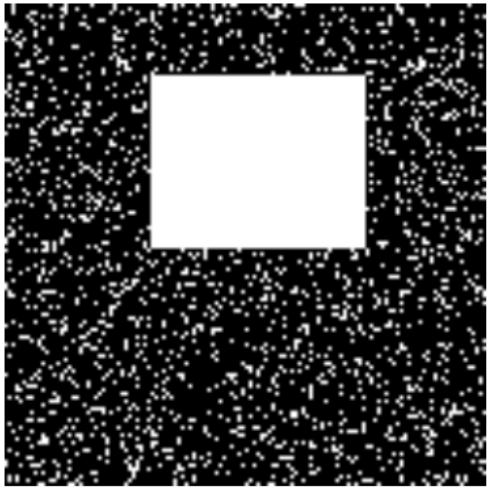
- Option 2: Use the minimum between the two peaks:

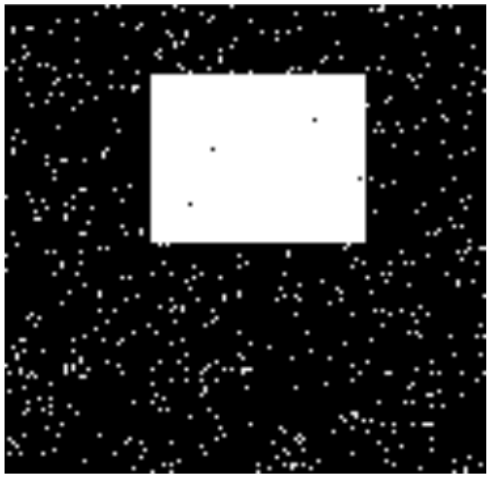
$$t = \arg \min_{v \in [p_1, p_2]} H(v)$$

  - Where $H(v)$ gives the histogram value at $v(i,j)$.
  - In this example:

$$t = \frac{75+125}{2} = 100 \qquad t = \arg \min_{v \in [p_1, p_2]} H(v) = 114$$







2.2. Otsu's method

- It assumes that the image contains two classes of pixels following a bi-modal histogram. Then, it calculates the optimum threshold separating the two classes according to any of the following two criteria (both criteria are equivalent):
  - **Intra-class variance minimization:**
    - Segments should have relatively homogeneous grey values → The threshold must minimizes the variance of the grey values within.
  - **Inter-class variance maximization:**
    - Alternatively, a threshold that maximizes the variance between objects and background can be selected.

2.2. Otsu's method

- Otsu's method chooses the optimal threshold t by maximizing the between-class variance, which is equivalent to minimizing the within-class variance:

  - The total variance (the sum of the within-class variance and the between-class variance) is constant for different partitions.

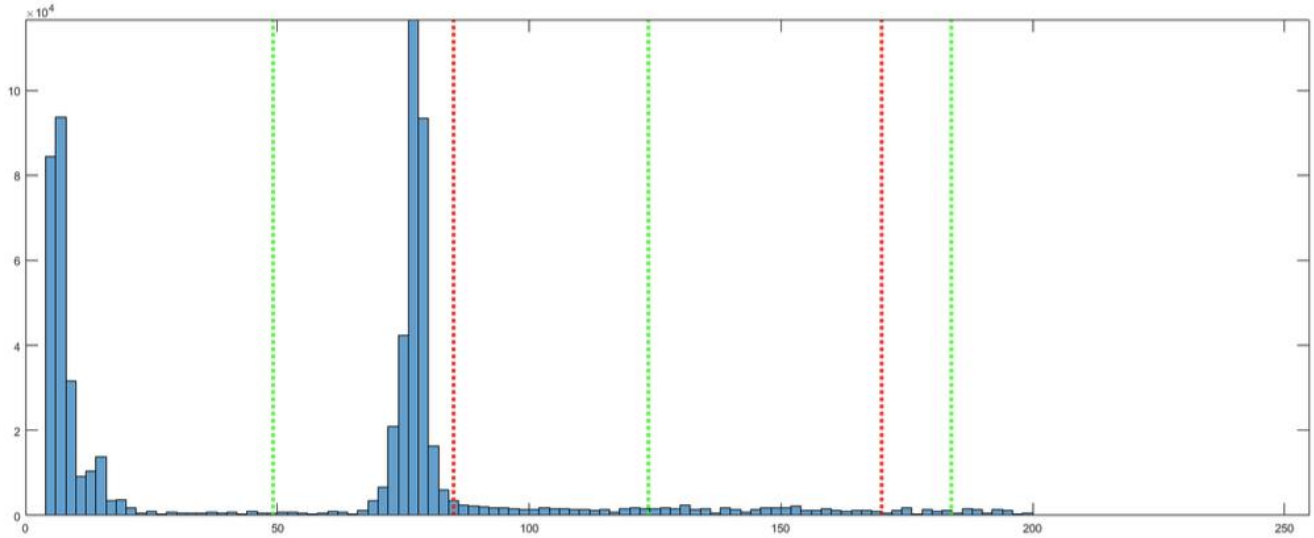
$$t = \arg\left\{\max_t\{\sigma_b^2(t)\}\right\} = \arg\left\{\min_t\{\sigma_w^2(t)\}\right\}$$

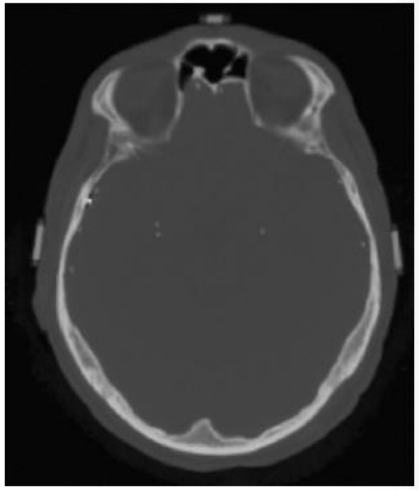
- If more than two segments are required, the Otsu's method can be extended to use multiple thresholds.

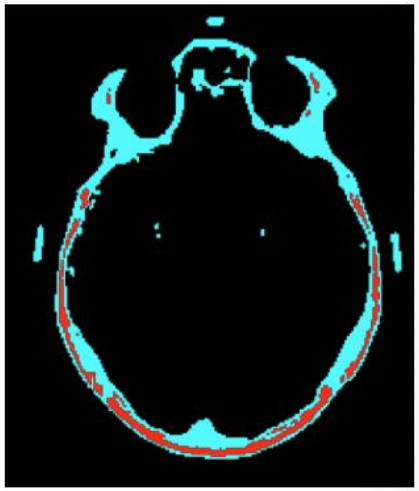
  - Multidimensional optimization is required → Complex and computationally expensive.
  - More practical solution → K-means algorithm.

2.3. K-means algorithm

1. Initialization:
   a) Distribute the $K-1$ thresholds over the histogram (red lines) → $K=3$ in the example.
   b) Segment the image according such thresholds.
   c) For each segment, compute the mean pixel value (green lines).

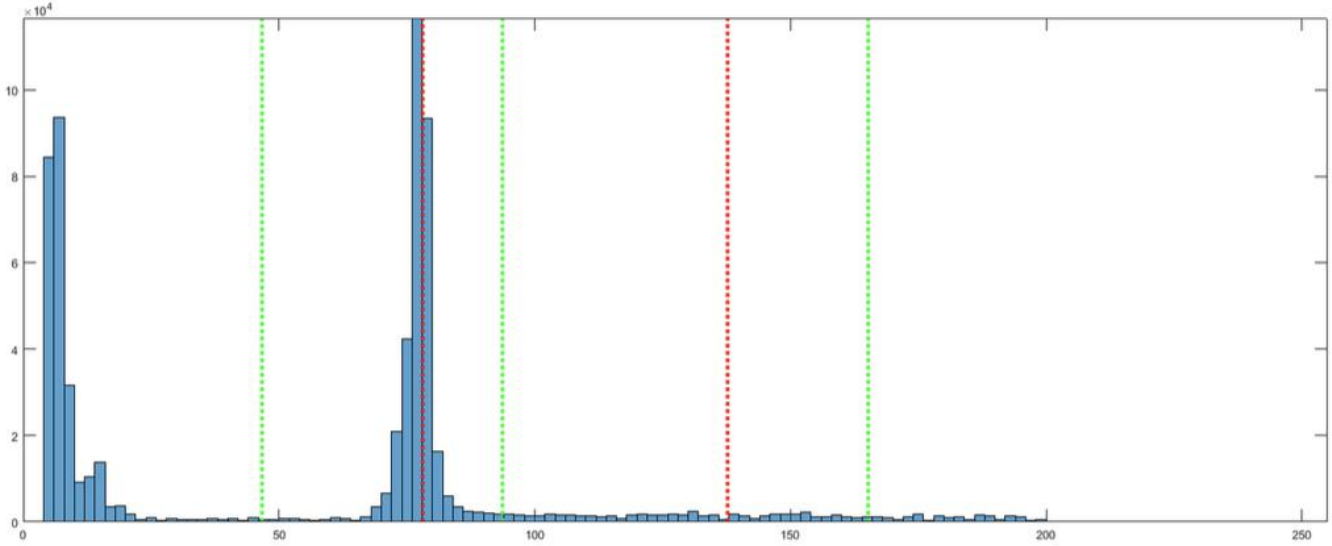


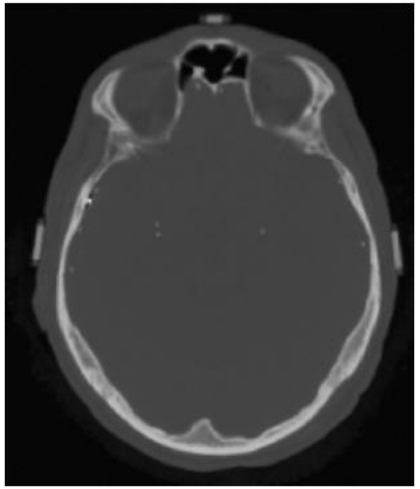


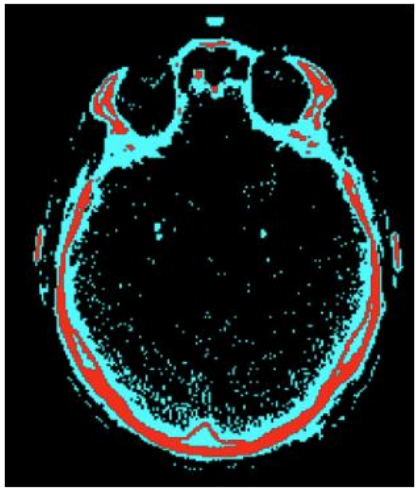


2.3. K-means algorithm

2. Reset the clusters centers to the computed mean values.
3. Reset the thresholds to be midway the cluster centers (red lines).
4. Segment the image and compute new means (green lines).
5. Go to stem 2. Iterate until cluster centers do not move.

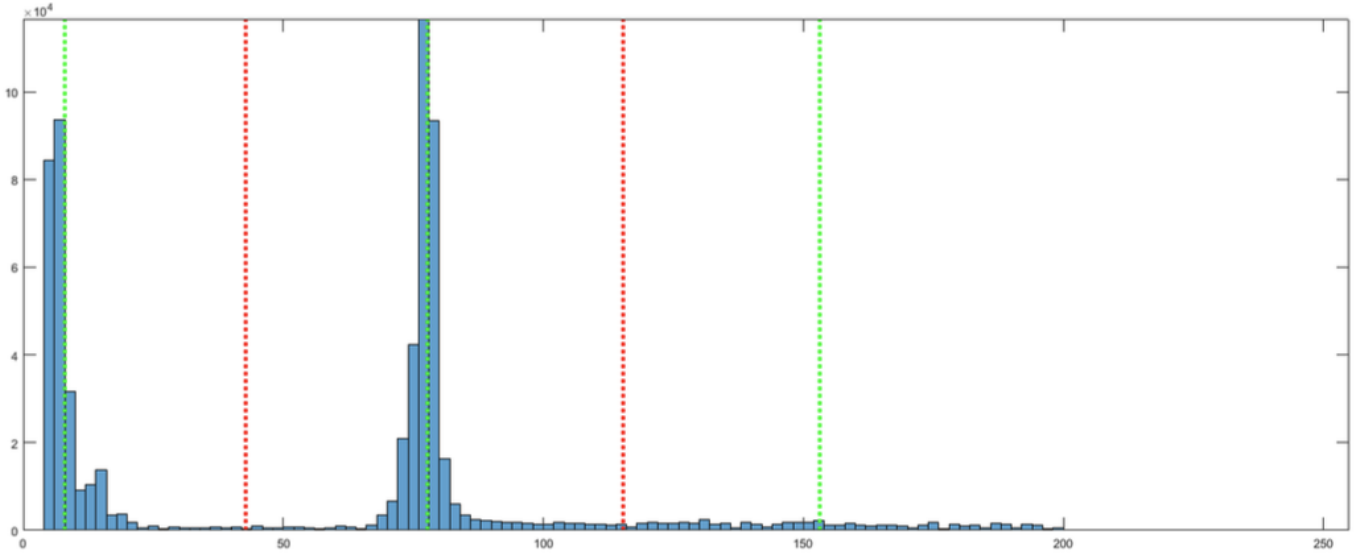


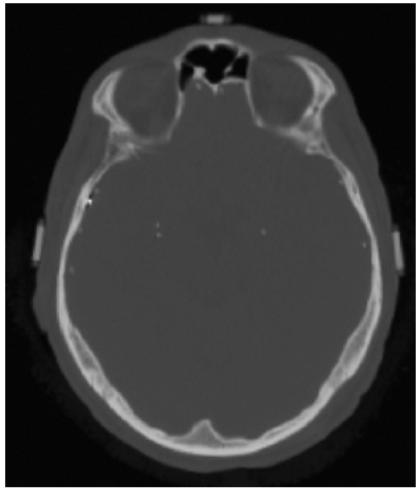


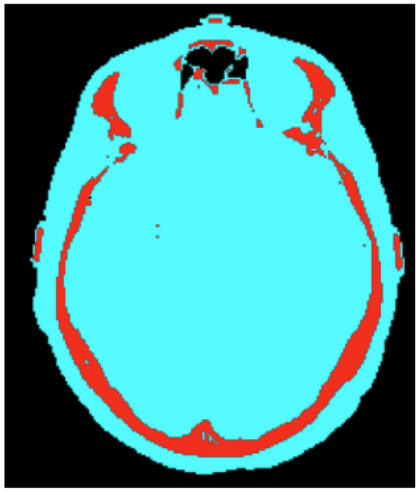


2.3. K-means algorithm

- In the illustrated example, the final segmentation has been reached in 7 iterations.
- It can be observed that the result is successfully.





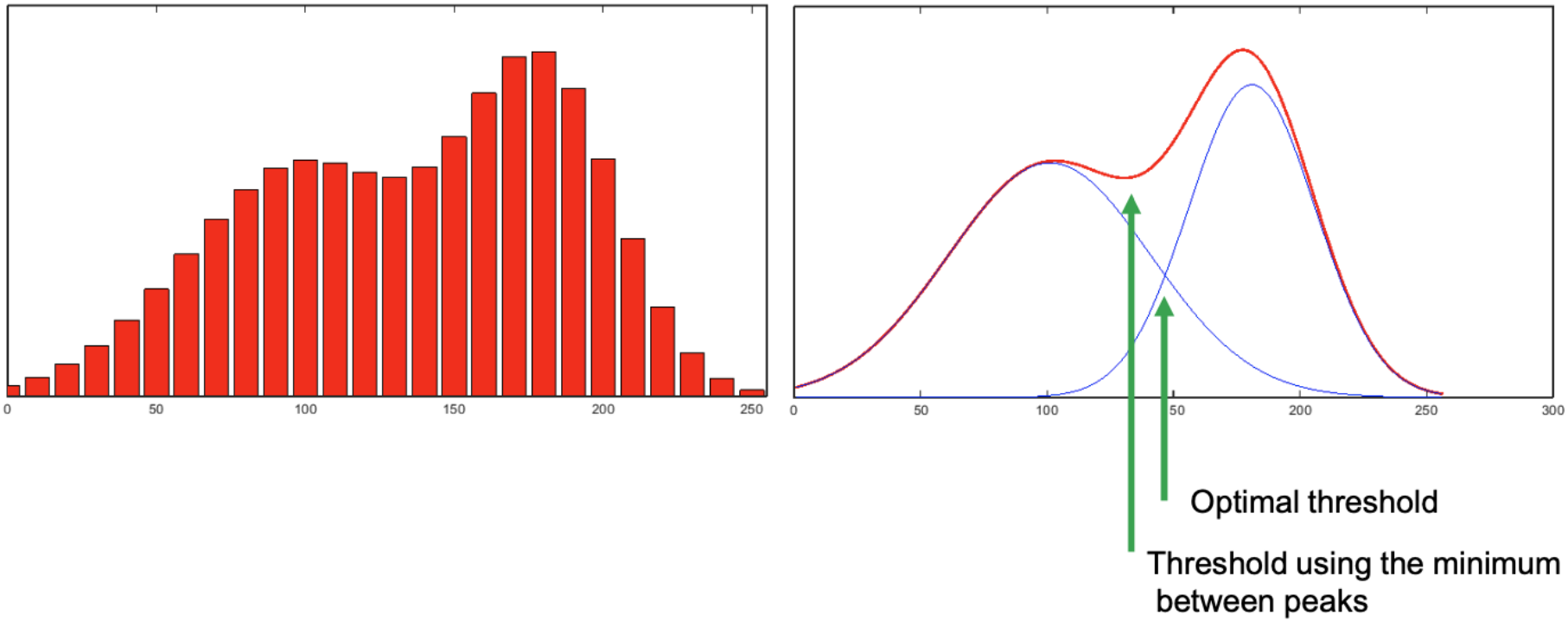


2.3. K-means algorithm

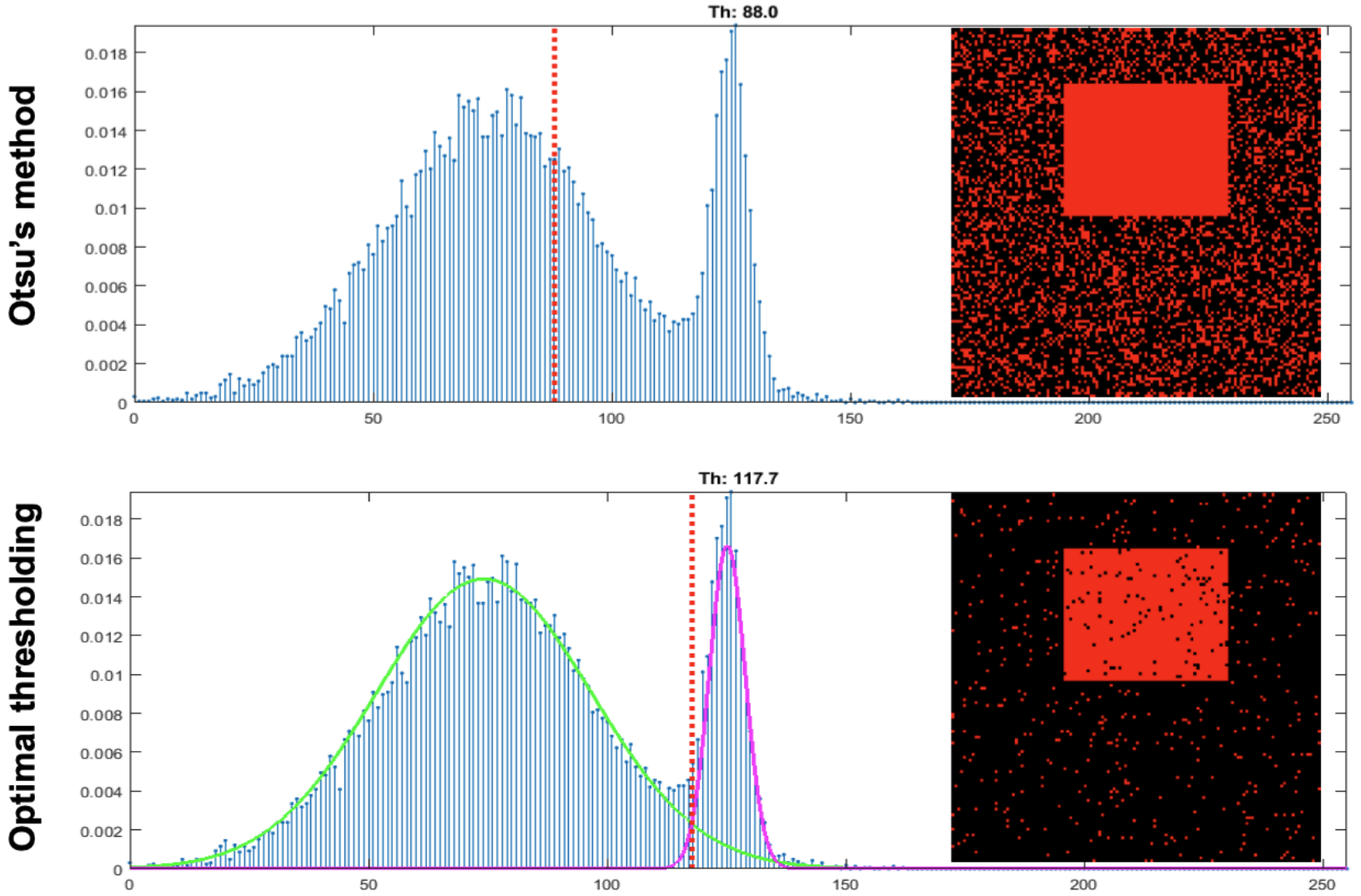
- **Drawbacks:**
  - The value of K must be set before running the algorithm.
    - What is the adequate one?
  - The result depends on the initialization.
    - What are the adequate ones?

2.4. Optimal thresholding

- Example of optimal thresholding:
  - Left: Image histogram.
  - Right: Approximation of the histogram using two Gaussians.



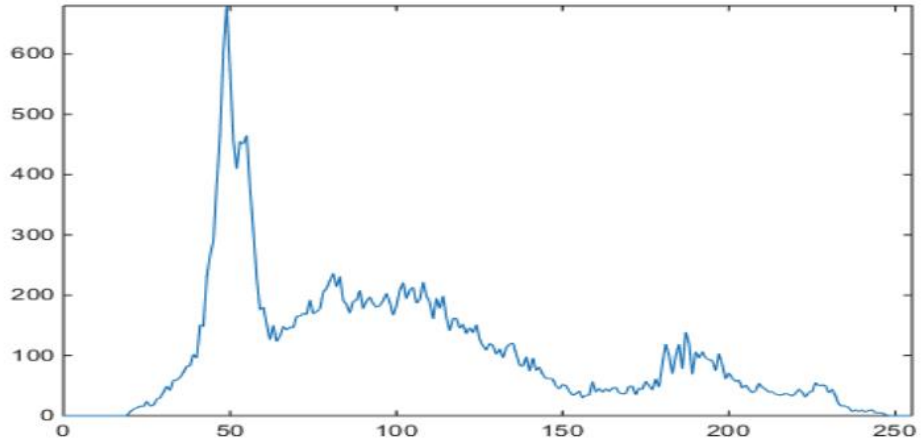
2.4. Optimal thresholding



2.5. Histogram estimation

- Estimation of the histogram → A smooth curve that best fits the real image histogram.
- Optimal thresholding method → It uses an estimation of the histogram.
- Other methods (e.g. those requiring finding extrema) → They also benefit from using a smooth estimate.
  - The chance of getting stuck in local extrema is decreased.
  - The process is less susceptible to noise.

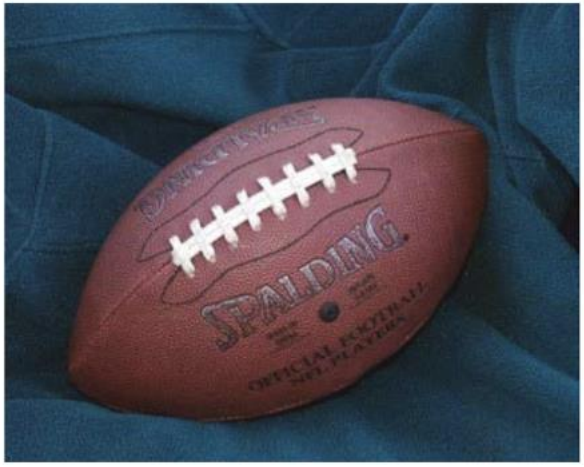
Conclusion: The histogram estimation is very useful in many cases.

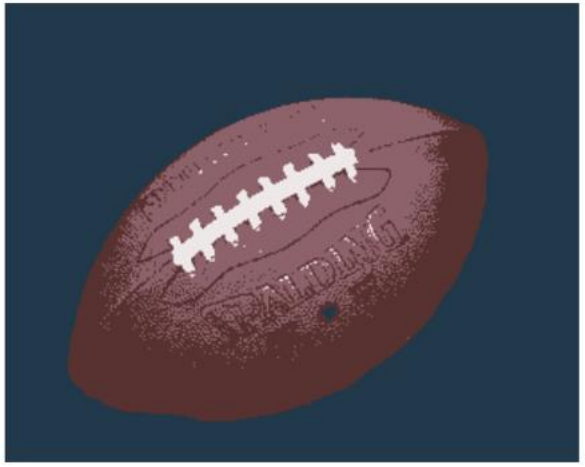


2.5. Histogram estimation

- Estimation of the histogram → A smooth curve that best fits the real image histogram.
- Optimal thresholding method → It uses an estimation of the histogram.
- Other methods (e.g. those requiring finding extrema) → They also benefit from using a smooth estimate.
  - The chance of getting stuck in local extrema is decreased.
  - The process is less susceptible to noise.

Conclusion: The histogram estimation is very useful in many cases.





2.5. Histogram estimation

- **Kernel Density Estimation (KDE):**
  - It is the most frequently used nonparametric modeling method.
  - It estimates the histogram by a sum of local functions (kernels, K) centered at each sample.
  - Typical kernels: Gaussian, triangular and rectangular.
  - Each distribution is identical to all others.
  - The only parameter that must be stablished is their variance.

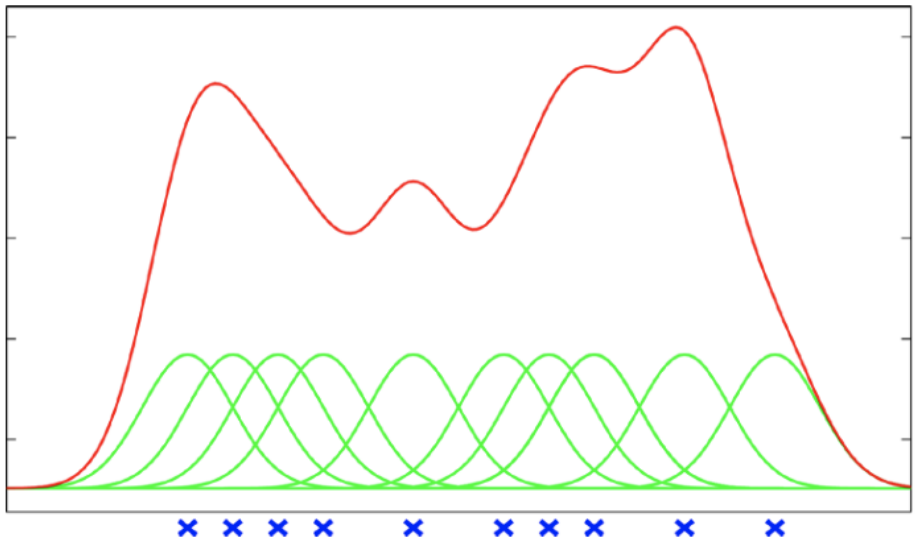
$$f(v) = \frac{1}{N}\sum_{i=1}^{N} K(v - v_i)$$

$$f(v) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(v - v_i)^2}{2\sigma^2}\right)$$

N → Number of samples.

σ^2 → Variance of the kernels.

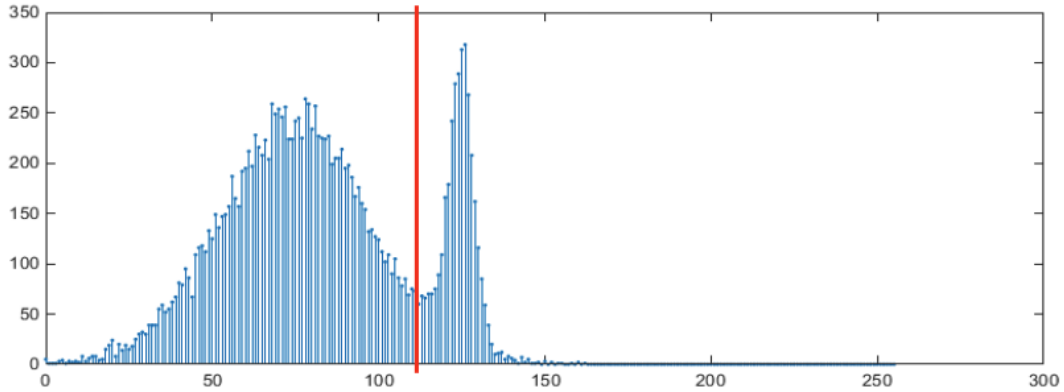
v_i → Grey value of each sample.



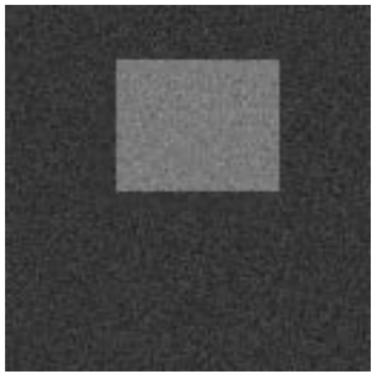
2.6. Enhancing threshold segmentation

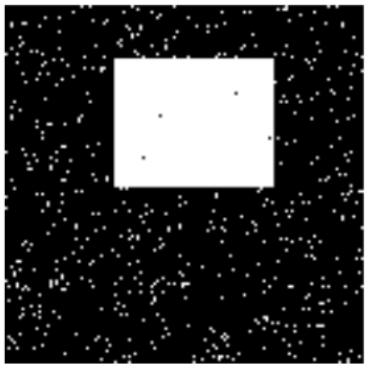
- **Post processing using a median filter:**

  - Each output pixel contains the median value in a $n \times n$ neighborhood around the corresponding pixel in the input image.



$t = 114$



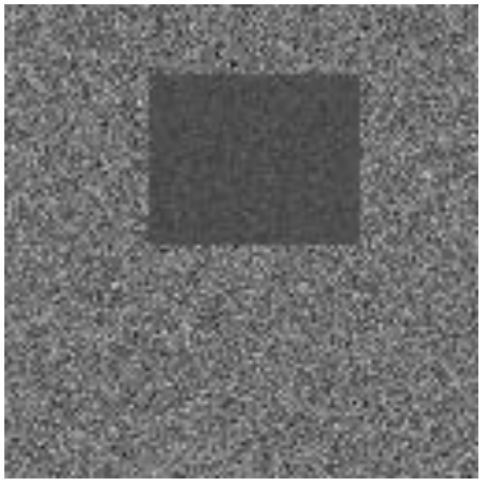


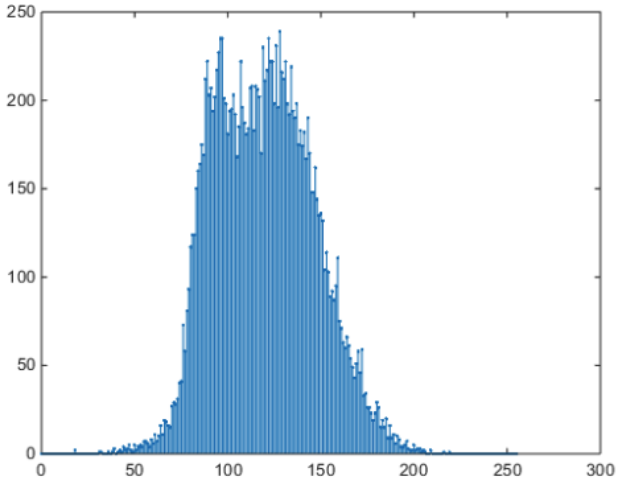
$n = 3$

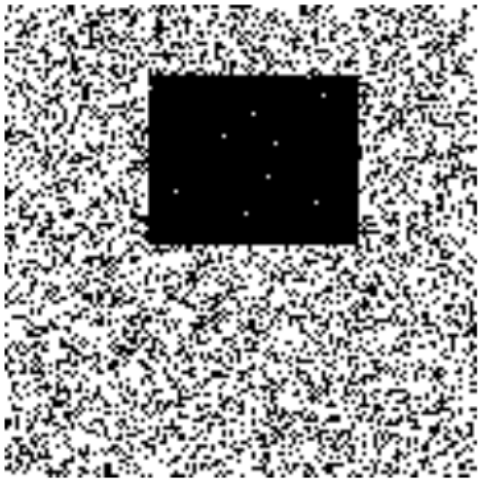
$n = 5$

2.6. Enhancing threshold segmentation

- **Pre and post processing using morphology:**



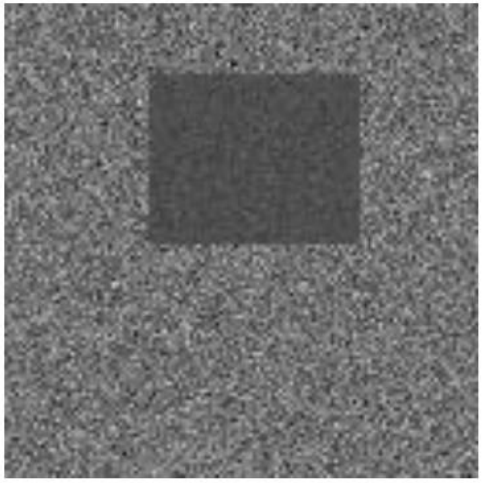


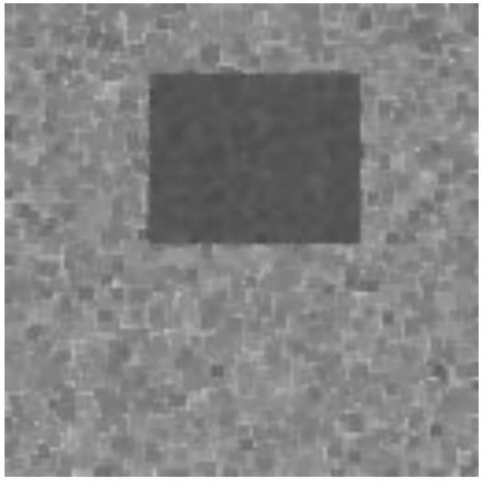


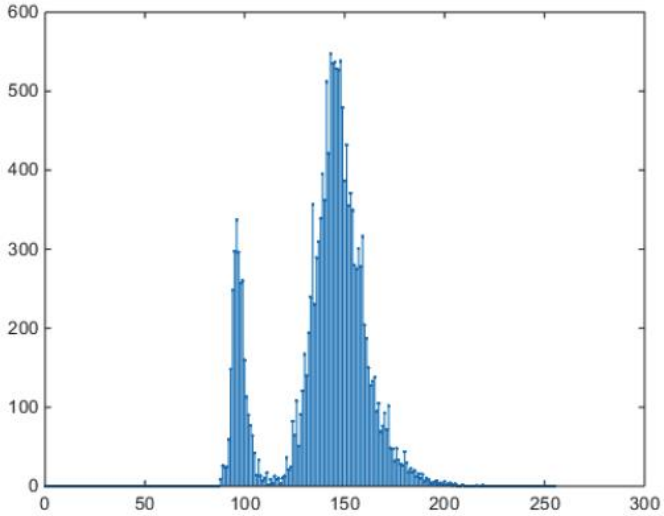
  - This is the result if we apply a 3 × 3 median filter in a post processing stage.

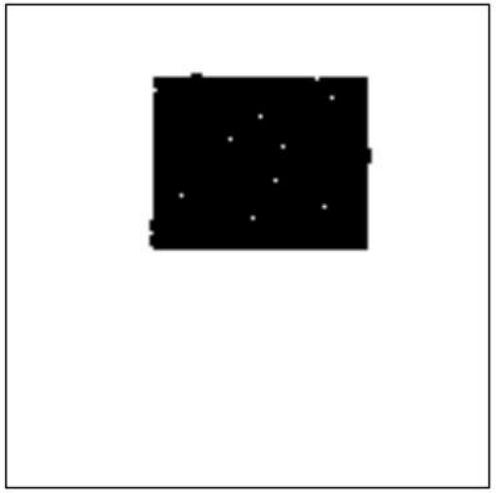
2.6. Enhancing threshold segmentation

- **Pre and post processing using morphology:**





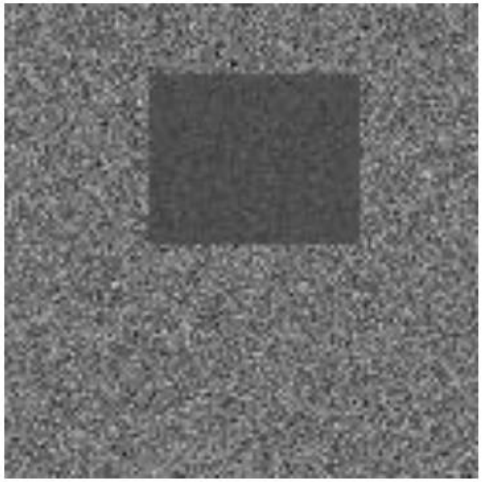


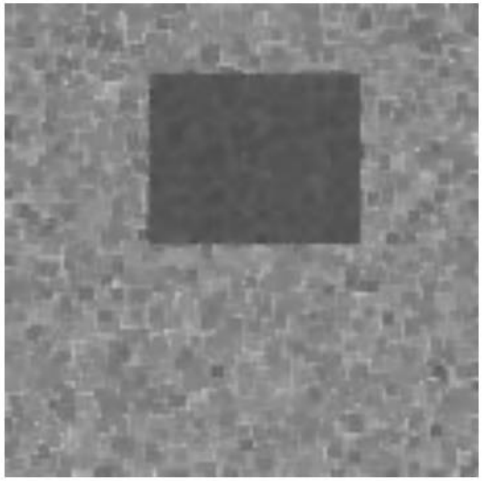


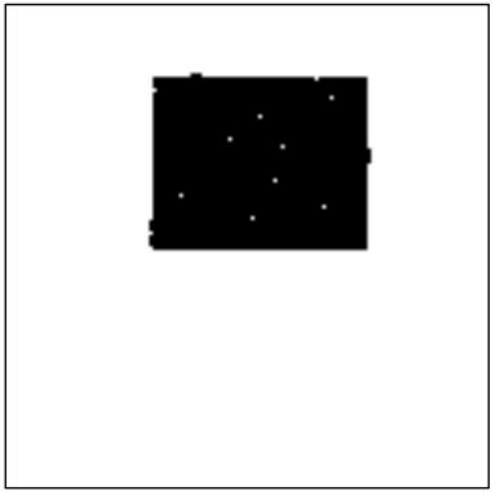
- Pre processing: First, we apply a 3×3 closing operation (dilation followed by an erosion) on the original image.
- The result is a new image with an histogram easier to analyze.
- The noisy segments have been removed.

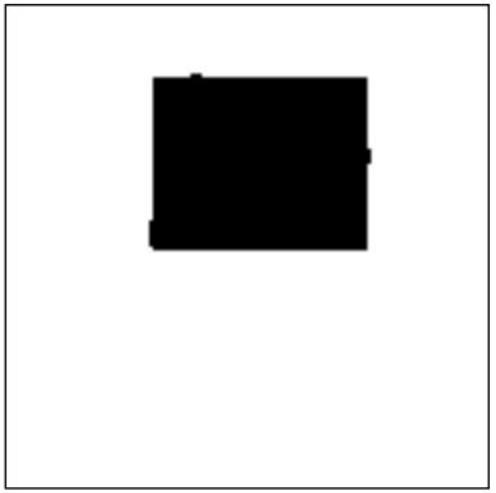
2.6. Enhancing threshold segmentation

- **Pre and post processing using morphology:**





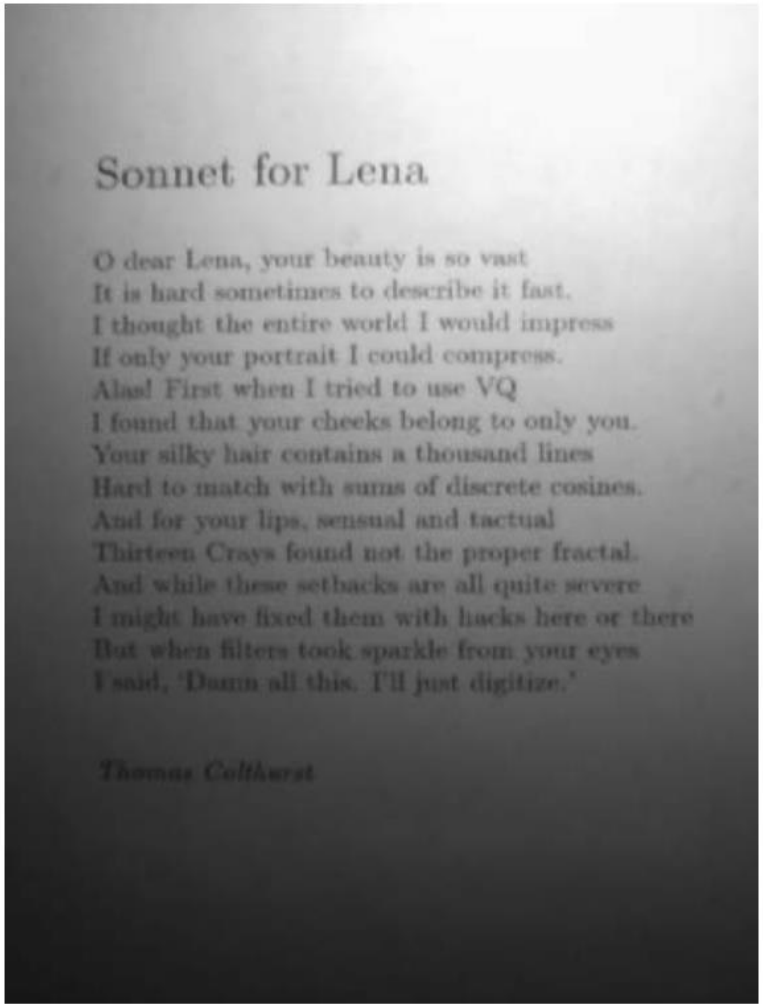


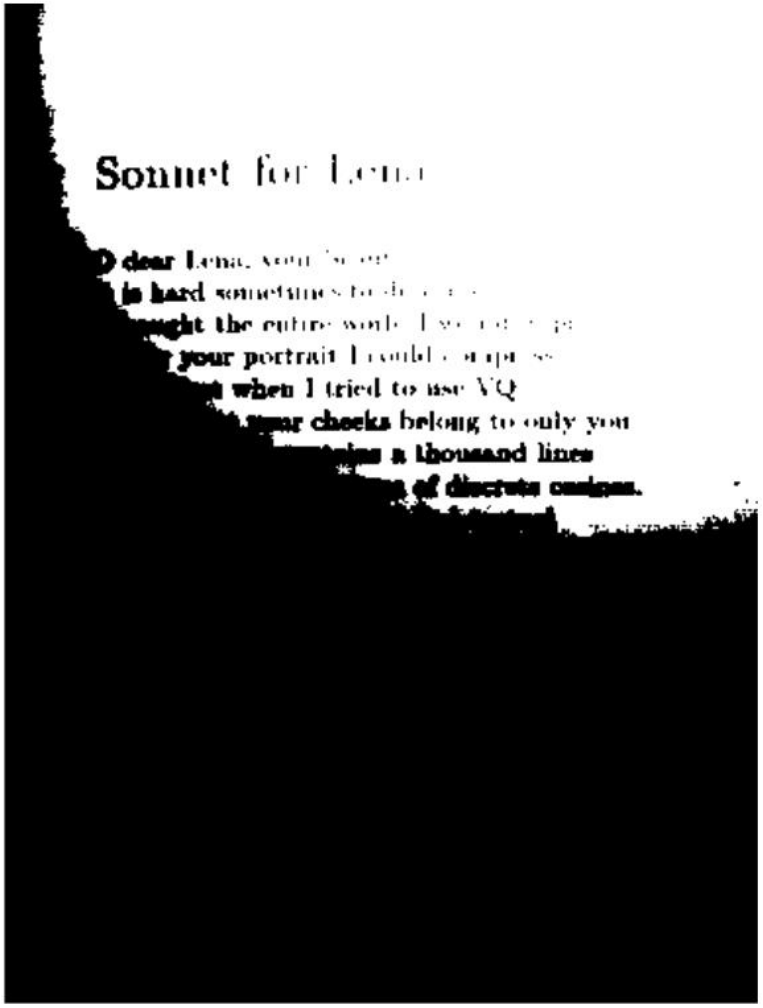


- Post processing: Finally, we apply a 3×3 opening operation (erosion followed by a dilation) **on the segmented image**.
- The result is a segmented image in which the holes have been removed.

2.6. Enhancing threshold segmentation

- **Local thresholding:**



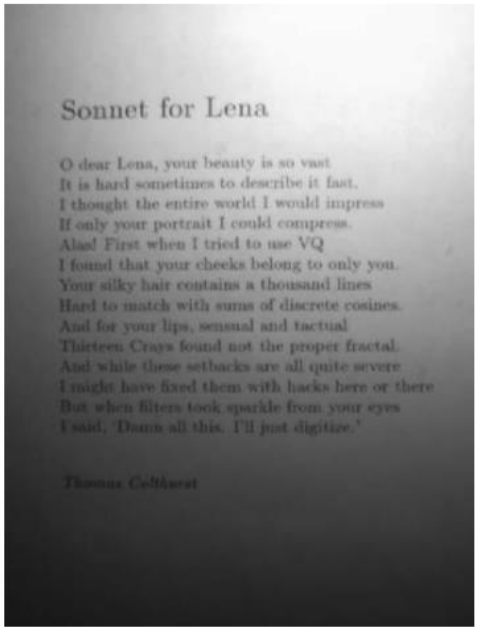


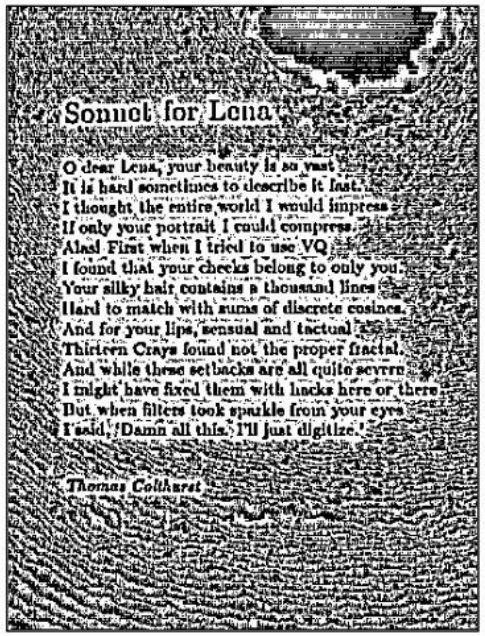
2.6. Enhancing threshold segmentation

- **Local thresholding:**
  - This problem can be solved by tiling the images into subimages.
    - The grey value of the object to segment must be relatively constant in each subimage.
    - An appropriate threshold could be found for each subimage.
    - The final segmentation will be the result of merging all the segmentation results.
  - Problems:
    - How many subimages?
    - How many thresholds per subimage?
      - It could be necessary to apply different techniques in the different subimages.

2.6. Enhancing threshold segmentation

- **Local thresholding:**
  - Example - Image with 2 gray levels and small segments:

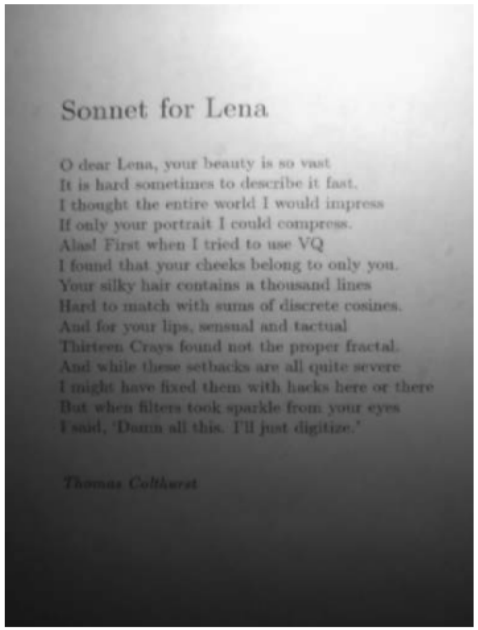


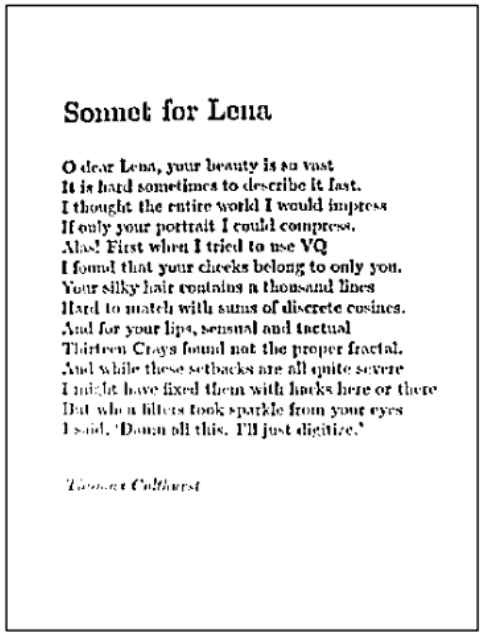


- The threshold applied to each pixel is the mean of a 7×7 neighborhood.
- Good results in the area surrounding the text → There are enough foreground and background pixels in the local neighborhood of each pixel.
- Bad results in too uniform areas → The range of intensity values is very small and their mean is close to the value of the center pixel

2.6. Enhancing threshold segmentation

- **Local thresholding:**
  - Example - Image with 2 gray levels and small segments:





POSSIBLE SOLUTION:

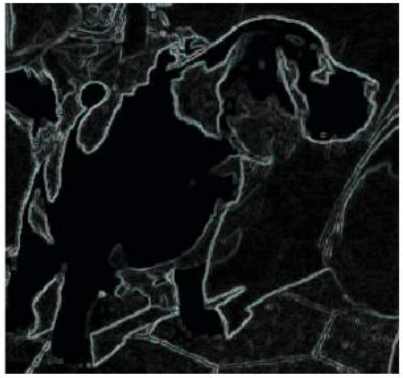
- These errors can be improved if the threshold employed is not the mean, but **the mean minus a constant**, C.
- In this way, all the pixels in uniform areas will be set as background.

- This result has been obtained by thresholding each pixel with the mean of its 7×7 neighborhood minus $C = 7$.

3. Edge-based segmentation

- Since objects are fully represented by their edges, the images can be segmented into objects by finding the edges of such objects.

- The main stages in edge-based segmentation are:



1. Compute the edge image (Sobel, Prewitt, Canny, etc.).

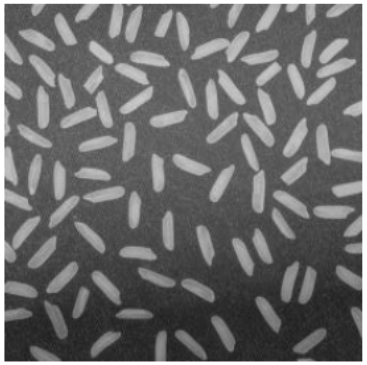
2. Process the edge image to maintain only closed object boundary.

3. Fill the object boundaries to obtain the segmented object.

3. Edge-based segmentation

- **Algorithm: edge-based segmentation**
  - This is a simple example to show how to segment an object when its boundary is perfectly closed (rare case).

1. Obtain the edge image (that is, the gradients of the image, G).
   - In this example we have used the module of the edge vectors obtained with the Sobel operator.



I

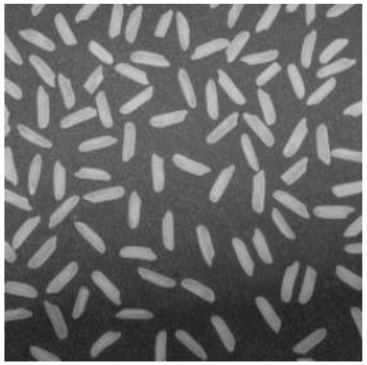
G

$$G_h = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I \quad G_v = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I$$

$$G = \sqrt{G_h^2 + G_v^2}$$

3. Edge-based segmentation

- **Algorithm: edge-based segmentation**
  2. Threshold G to have a binary image, G_{th}, showing the most prominent edges in the image.
     - The threshold used is typically very low → It allows removing noisy gradients but preserves most edges.



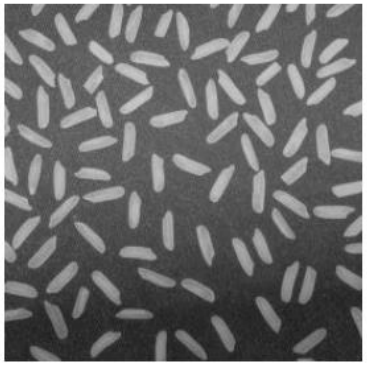
I

G

G_{th}

3. Edge-based segmentation

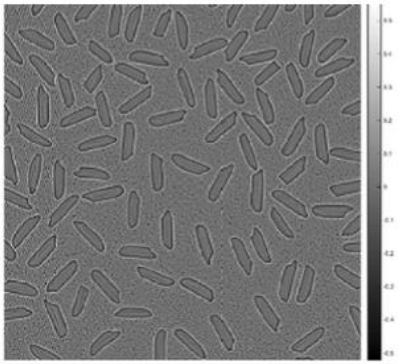
- **Algorithm: edge-based segmentation**
  3. Compute a Laplacian image, ΔI, from I.
     - Any discrete or continuous Laplacian operator can be used.
     - The Laplacian is a differential operator given by the divergence of the gradient of a function:
       - Positive values when the gradient is growing.
       - Negative values when the gradient is decreasing.
       - Nulls if the gradient is 0.



I

G

G_{th}



ΔI

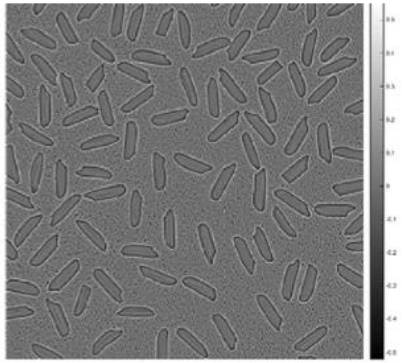
3. Edge-based segmentation

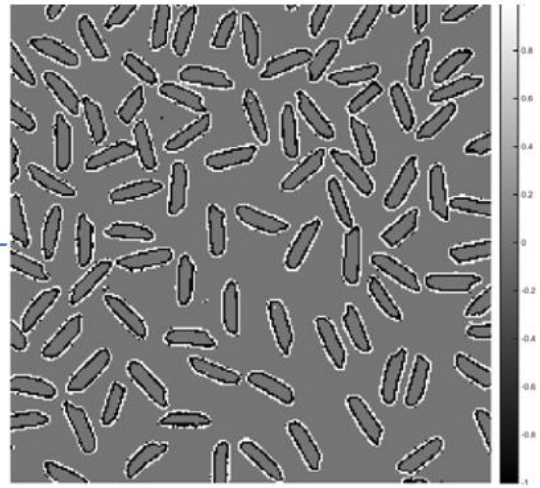
- **Algorithm: edge-based segmentation**

  4. Compute the image $g = G_{th} \cdot \mathrm{sgn}(\Delta I)$.

G_{th}

ΔI

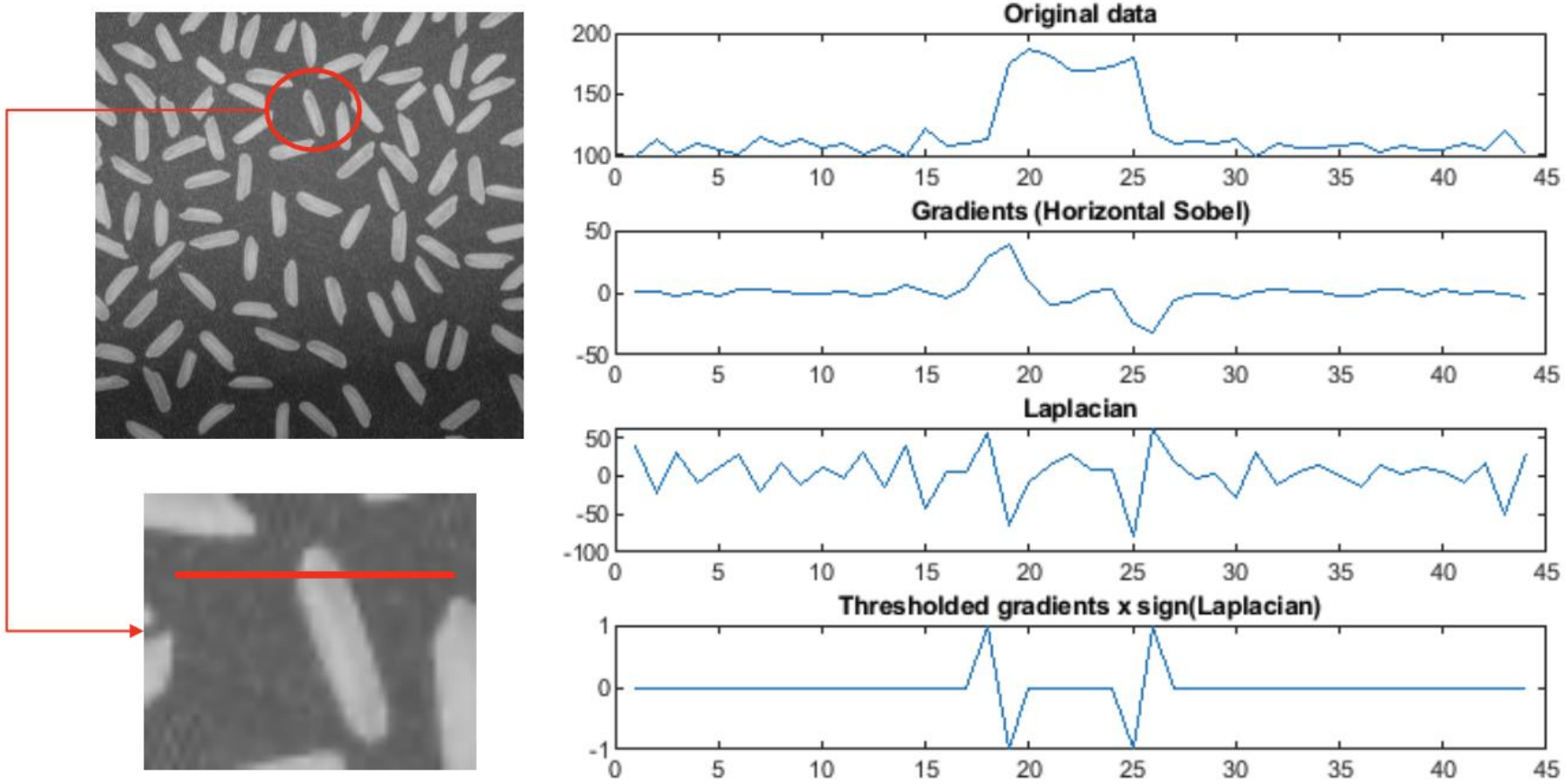




- This image has only 3 possible values:
  - 0 at non-edge pixels.
  - 1 at edge-pixels on the bight side of the edges.
  - -1 at the edge-pixels on the dark side of the edges.

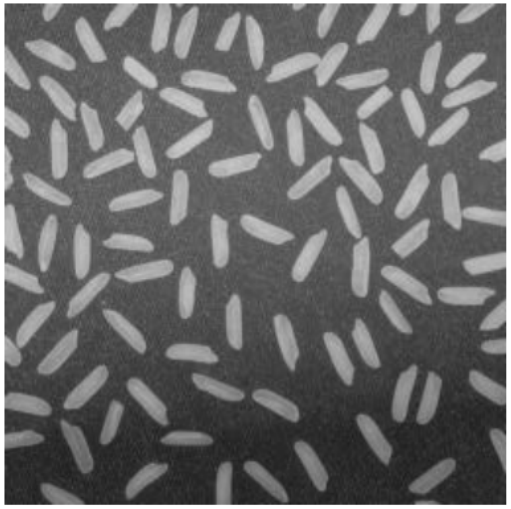
3. Edge-based segmentation

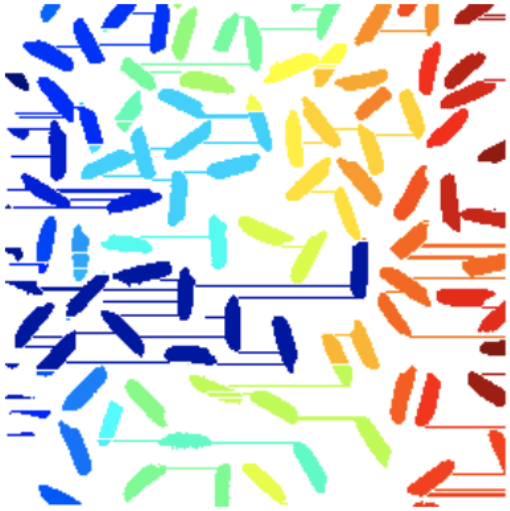
- **Algorithm: edge-based segmentation**



3. Edge-based segmentation

- **Algorithm: edge-based segmentation**
  5. The image g is traversed from left to right:
     - Adjacent pixels with values 1 and -1 → Entering an object.
     - Adjacent pixels with values -1 and 1 → Leaving an object.





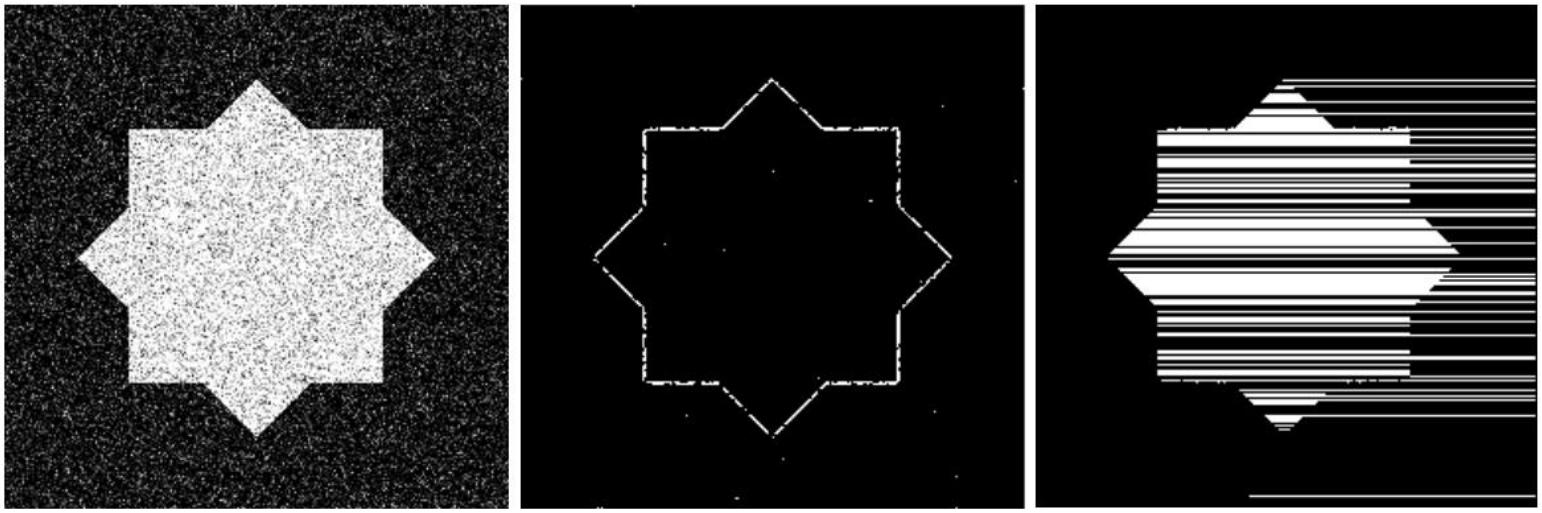
- The errors are due to the noise in the original → It results in erroneous sign transitions.

3. Edge-based segmentation

- **Algorithm: edge-based segmentation**
  - Result in an image with better defined edges.

3.1. Edge linking

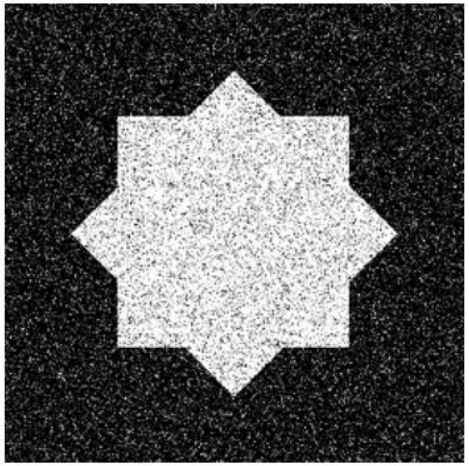
- Edge detection rarely provides perfect and closed boundaries.
  - There are spurious detected edges where they should not be.
    - It can be solved with morphology operations.
  - There are gaps where there should be edges.
    - Edge linking strategies are required to obtain continuous boundaries.



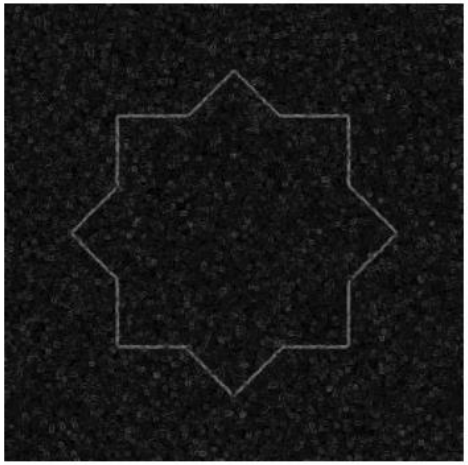
3.1. Edge linking

- **The Hough transform (example):**
  - We want to segment the following image:
    - We know that the object to segment is the intersection of a square and a diamond → Its boundaries are limited by 8 straight lines.

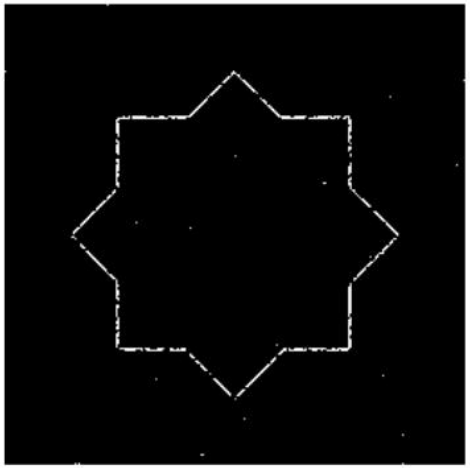
1. Obtain the edge image G.
2. Threshold G to have a binary image, G_{th}.



I



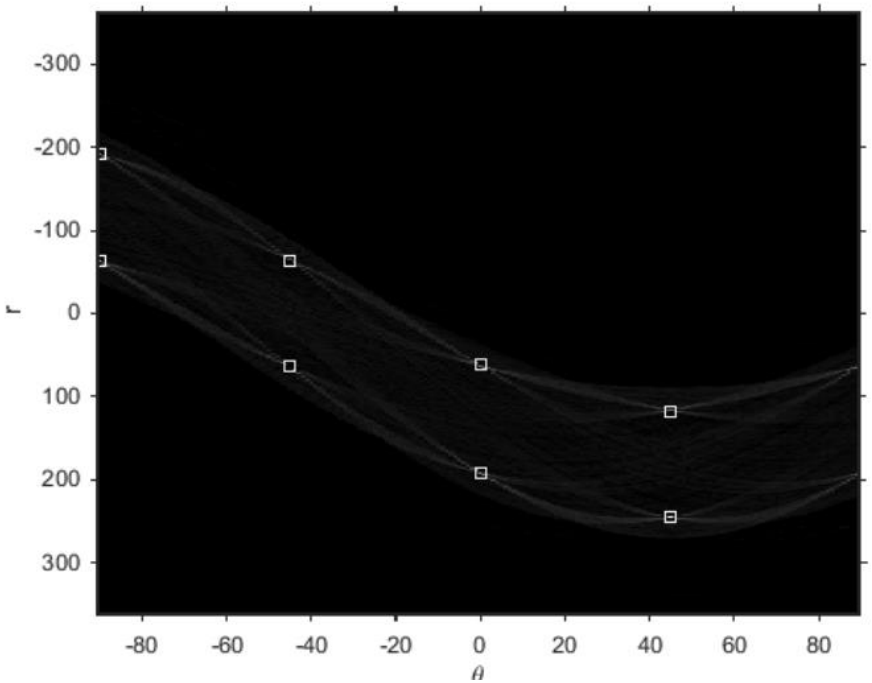
G

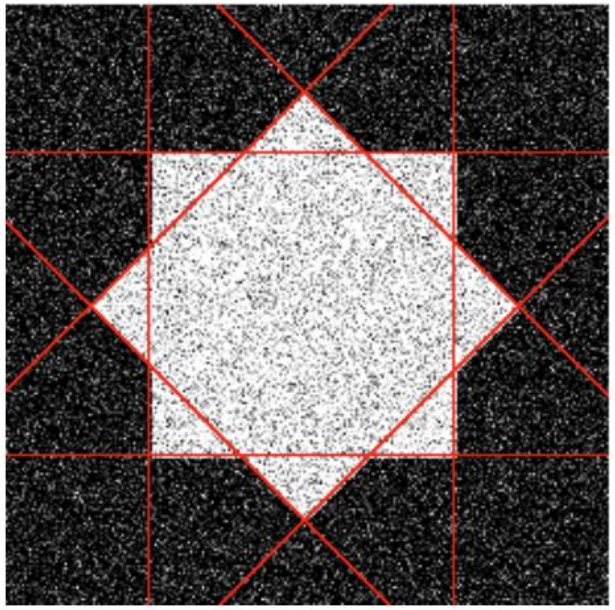


G_{th}

3.1. Edge linking

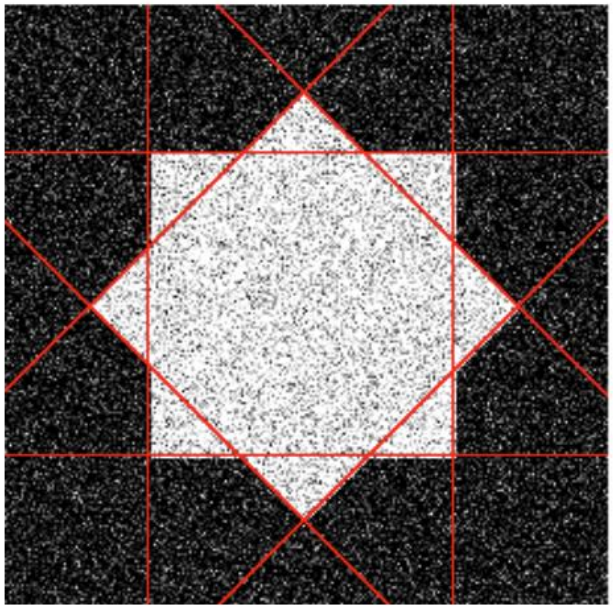
- **The Hough transform (example):**
  3. Compute the Hough transform, H (one Gaussian per edge point in G_{th}).
  4. Select the maximum 8 values.
     - These maxima will correspond to the 8 straight lines determining the boundaries of the object to segment.

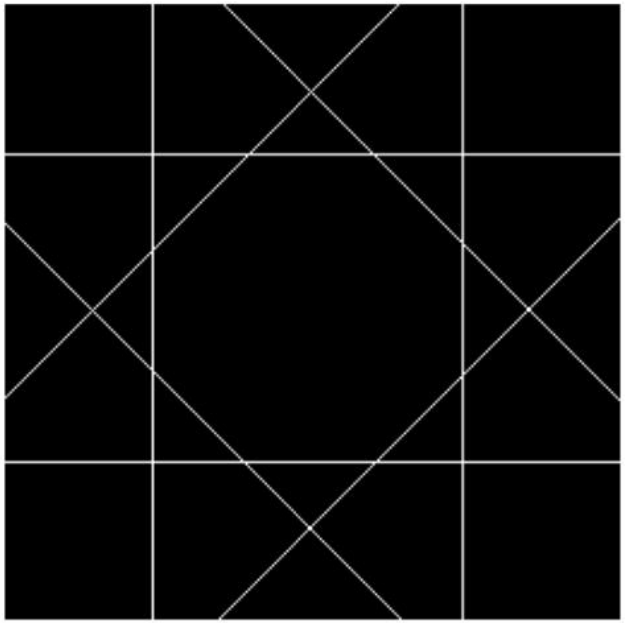




3.1. Edge linking

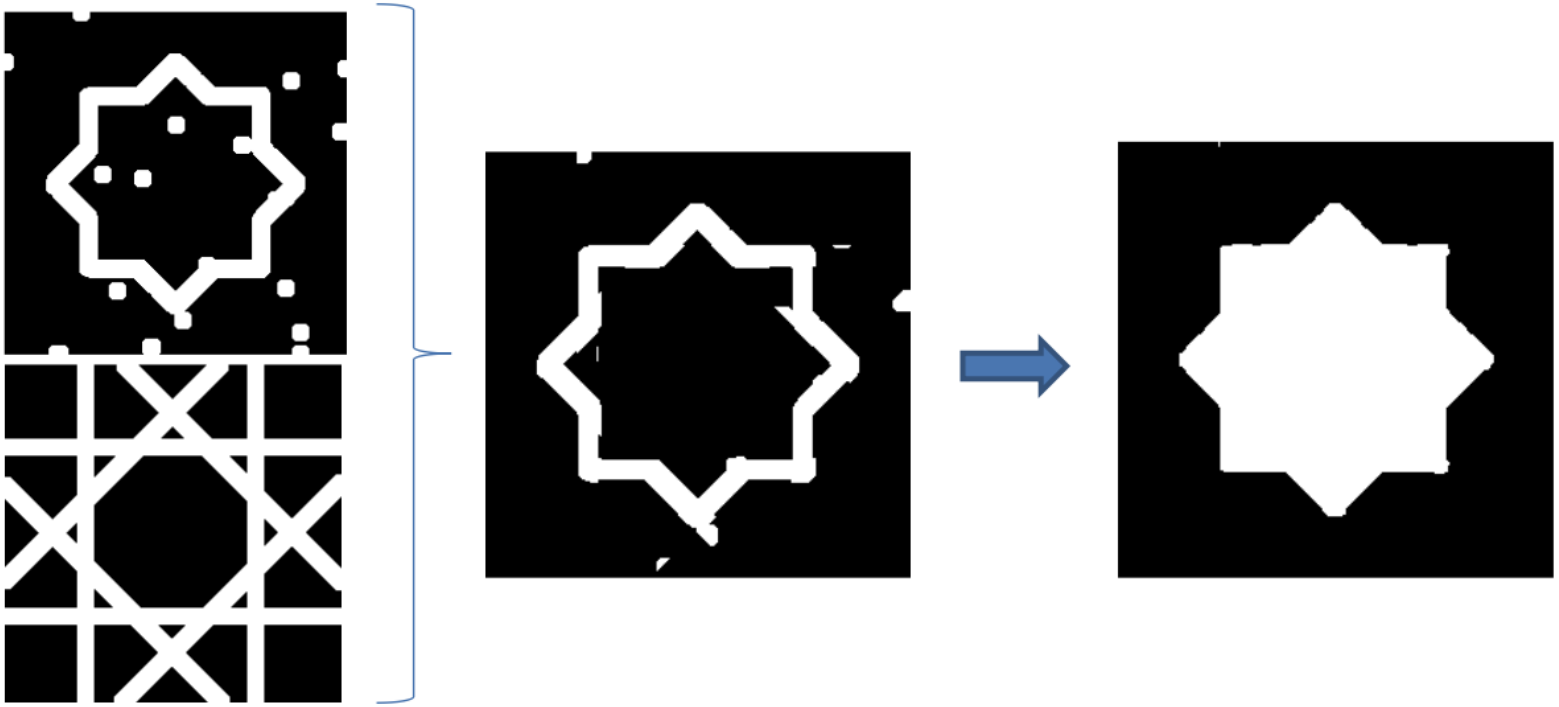
- **The Hough transform (example):**
  5. Create a binary image from the detected lines:
     a. For each line, compute its cut points with the image boundaries.
     b. Determine what pixels belong to the straight lines between the computed cut points (e.g. using Bresenham algorithm).



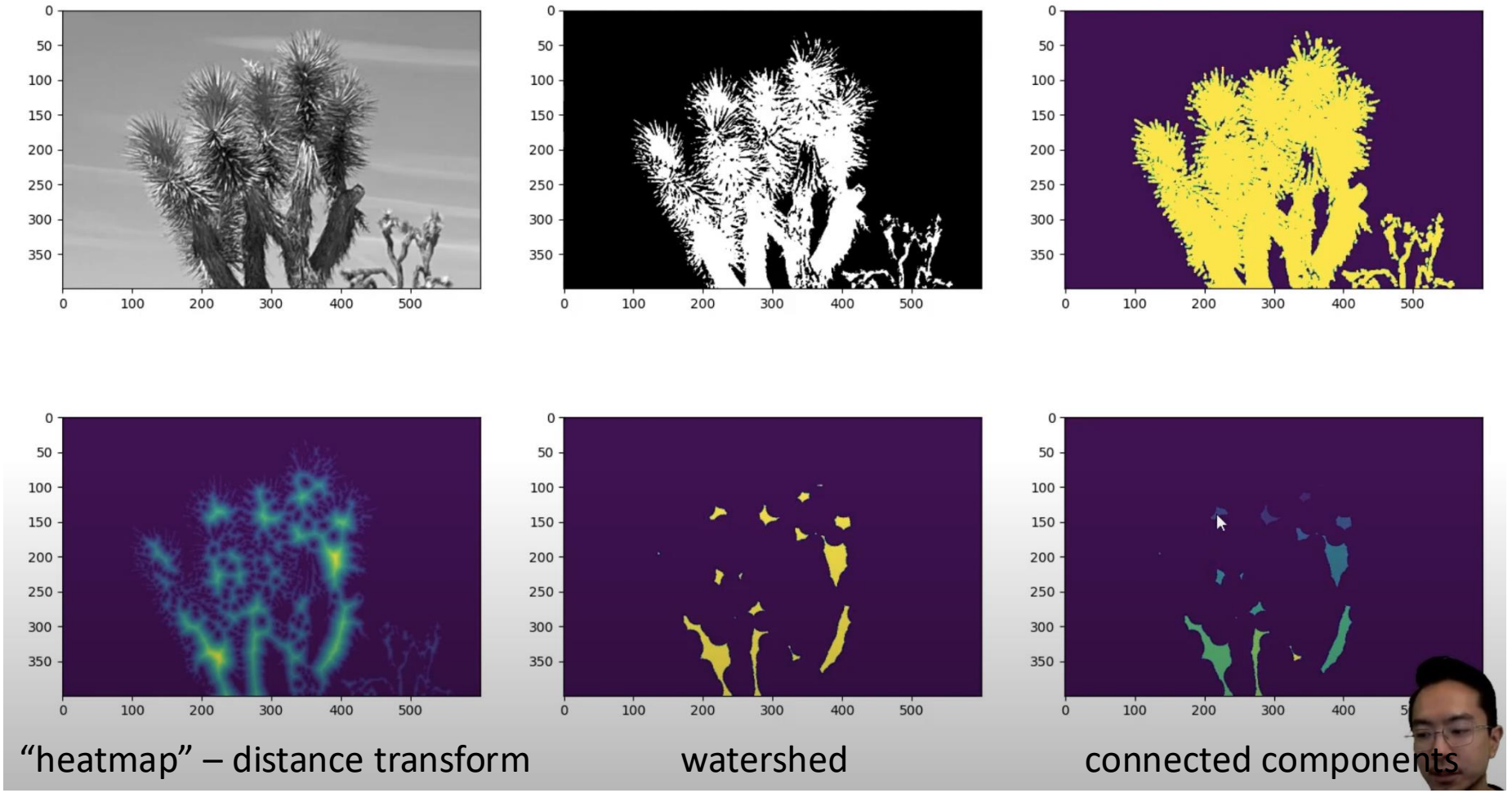


3.1. Edge linking

- **The Hough transform (example):**
  6. Post processing:
     a. Dilate both G_{th} and the binary image obtained in point 5.
     b. Compute the intersection between such images.
     c. Erode and fill the result.

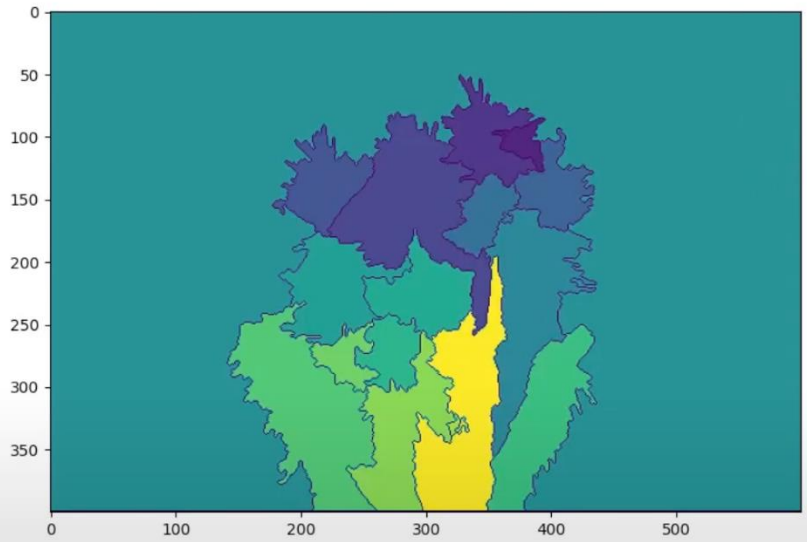


3.2. Watershed segmentation



https://www.youtube.com/watch?v=3MUxPn3uKSk

3.2. Watershed segmentation



https://www.youtube.com/watch?v=3MUxPn3uKSk

3.3. Active contours