Lec. 7

Image Segmentation

Assist. Prof. Dr. Saad Albawi

Segmentation

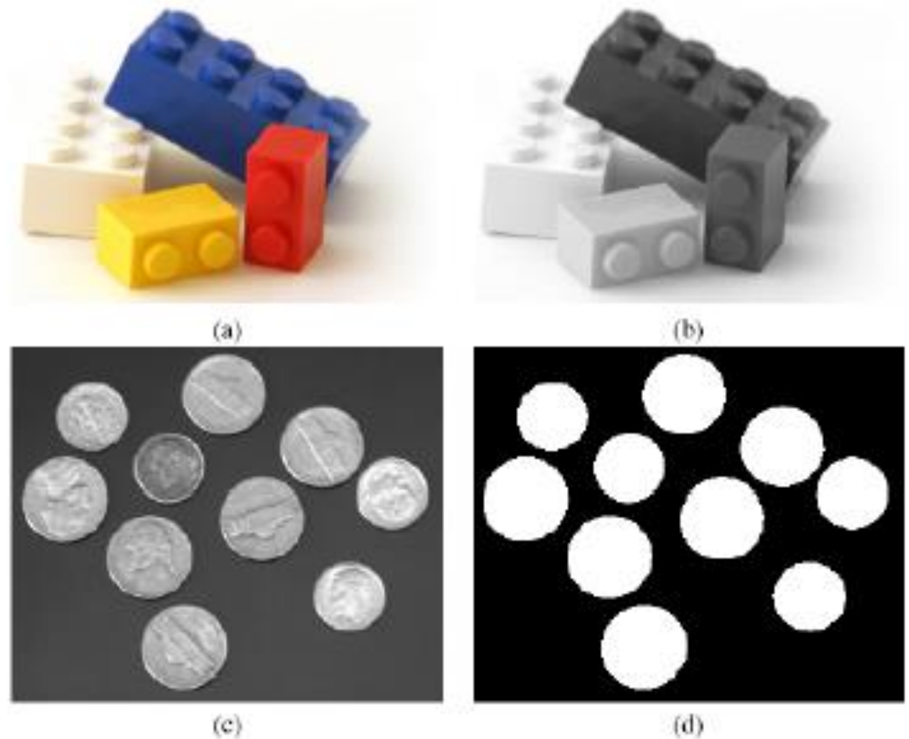
- Partitioning image into homogeneous regions with respect to some characteristic
  - Gray level, texture, color, motion, context (foreground, background)
- Useful mid-level representation of an image
- Facilitates computer vision tasks

Examples

(a) A hard image to segment due to uneven lighting, projected shadows, and occlusion among objects

(b) Grayscale of (a). Segmentation of this image is even harder (even impossible with some methods)

(c) An image easy to segment.



(a) (b) (c) (d)

Applications

- Object recognition
- Image annotation
- Video summarization
- Background subtraction
- Medical Imaging
- Image compression

Regions and Edges

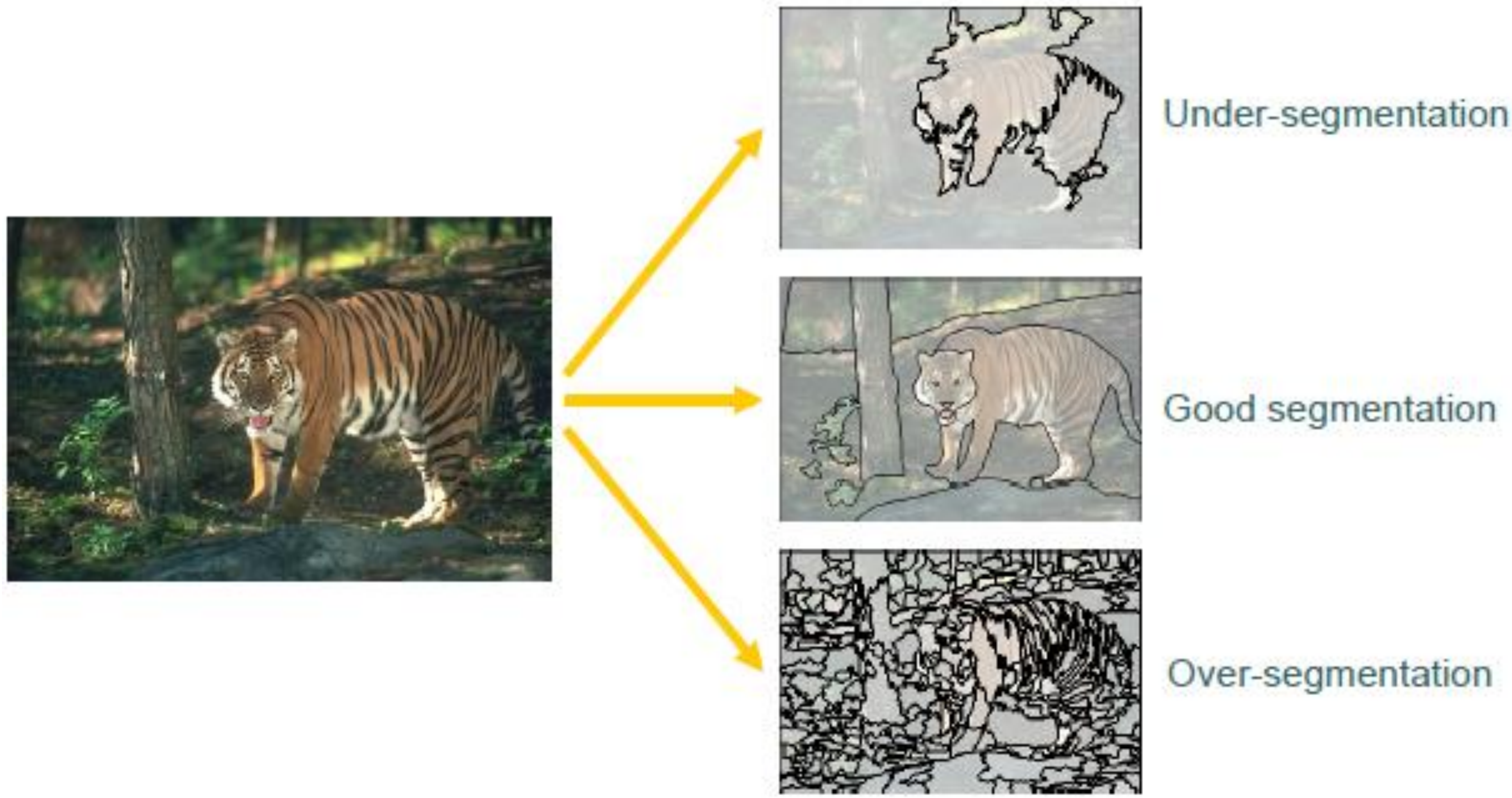
- **Regions** are found based on *SIMILARITIES* between values of adjacent pixels
  - We could “trace” regions to obtain **edges**
- **Edges** are found based on *DIFFERENCES* between values of adjacent pixels
  - We could “fill” closed contours to obtain **regions**

Strategies

- Top-down segmentation
  - Pixels belong together because they come from the same object

- Bottom-up segmentation
  - Pixels belong together because they look similar

Over & Under Segmentation



Segmentation Techniques

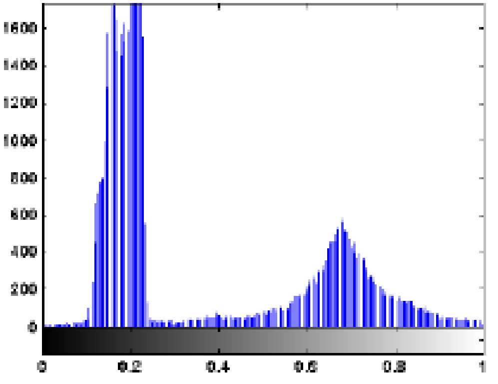
- Intensity-based (non-contextual): based on pixel distributions (i.e., histograms).

- Region-based (contextual): rely on adjacency and connectivity criteria between a pixel and its neighbors.
  - Region growing
  - Region splitting

Intensity-Based Segmentation

Intensity-Based Segmentation

- Rely on pixel statistics (histogram properties) to determine which pixels belong to “foreground” objects and which pixels should be labeled as “background.”



By Oge Marques

Image Thresholding

- This is usually performed by comparing each pixel intensity against a reference value (*threshold*) and replacing the pixel with a value (say 1 or 0) that means "foreground" or "background" depending on the outcome of the comparison.

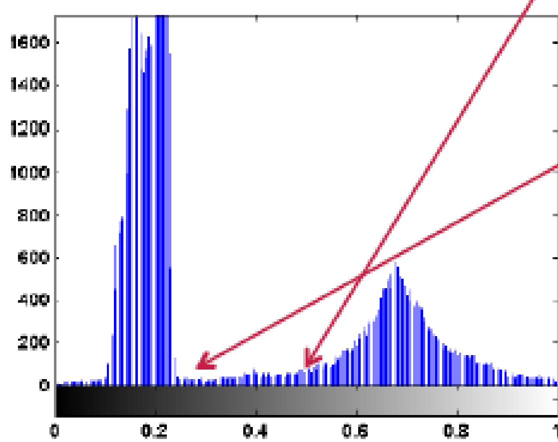
$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{otherwise} \end{cases}$$

- In particular we will look at:
  - What is thresholding?
  - Simple thresholding
    - Single value thresholding can be given mathematically as follows:

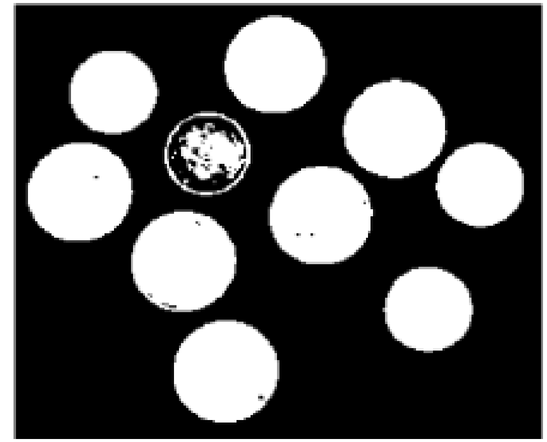
$$g(x, y) = \begin{cases} 1 \ if \ f(x, y) > T \\ 0 \ if \ f(x, y) \leq T \end{cases}$$

    - How to Choose T? Based on the histogram of an image. Partition the image histogram using a single global threshold
    - The success of this technique very strongly depends on how well the histogram can be partitioned

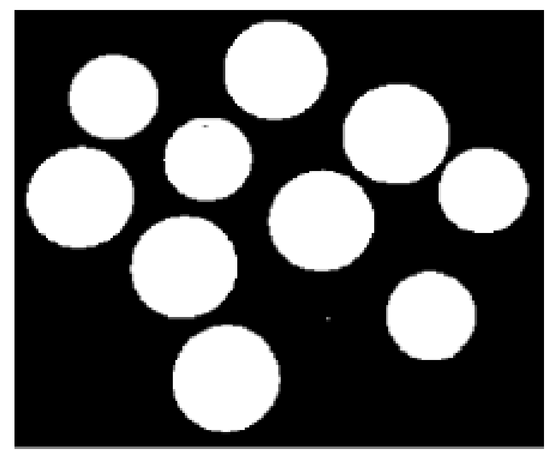
Global Thresholding



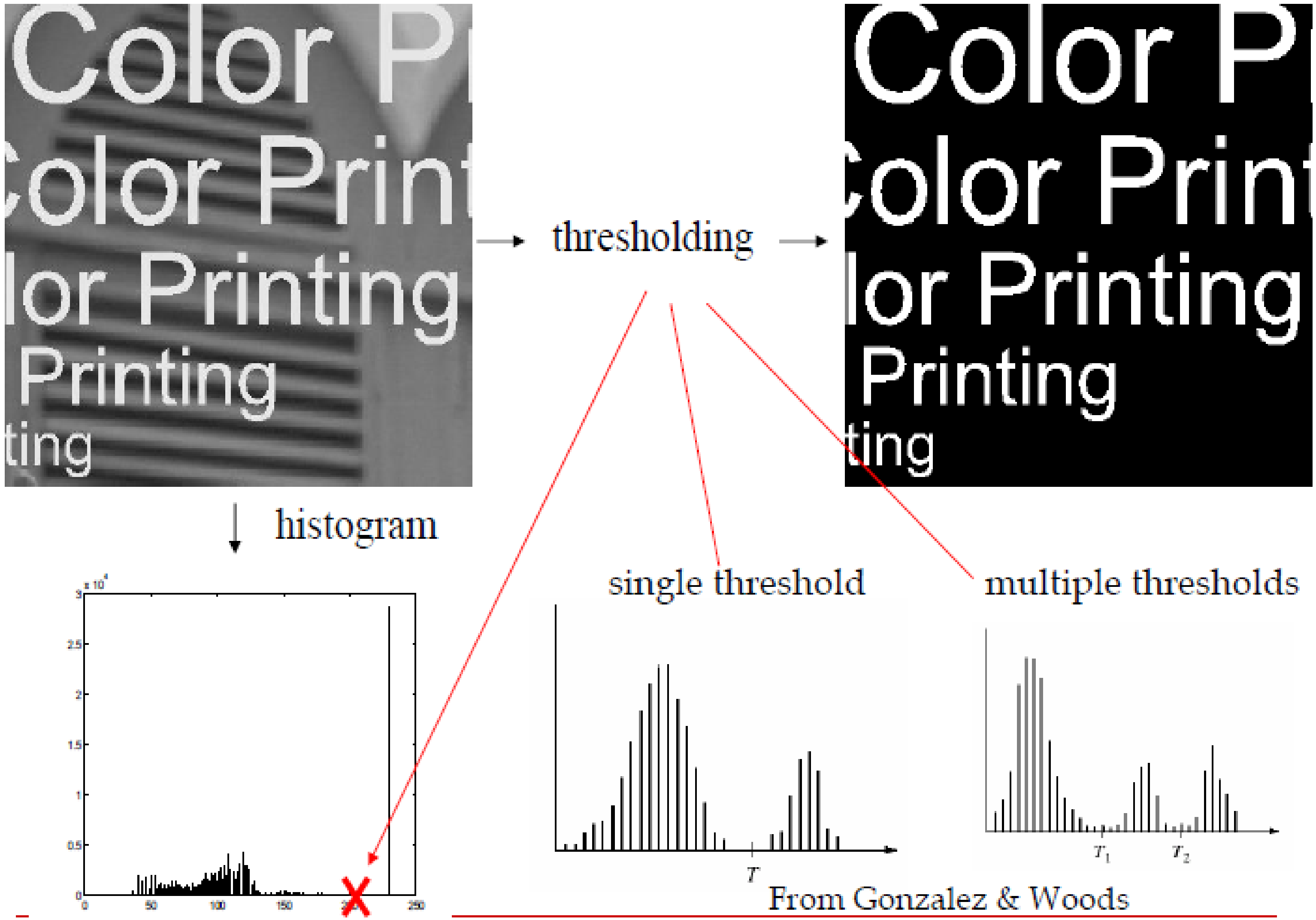
T = 0.4947



T = 0.25



Thresholding Method



- The basic global threshold, T, is calculated as follows:
  1. Select an initial estimate for T (typically the average grey level in the image)
  2. Segment the image using T to produce two groups of pixels: G_1 consisting of pixels with grey levels $>T$ and G_2 consisting pixels with grey levels $\leq T$
  3. Compute the average grey levels of pixels in G_1 to give μ_1 and G_2 to give μ_2

4. Compute a new threshold value:

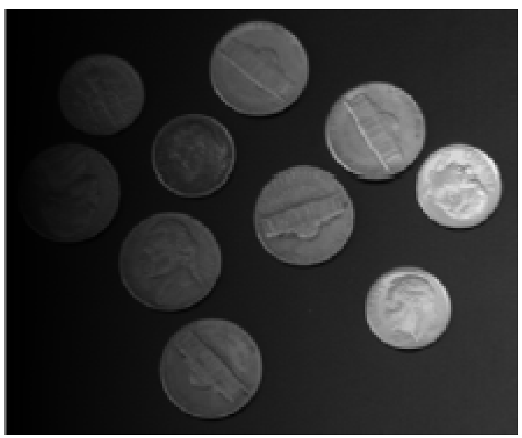
$$T_{new} = \frac{\mu_1 + \mu_2}{2}$$

5. Repeat steps 2 – 4 until the difference in T in successive iterations is less than a predefined limit T_∞

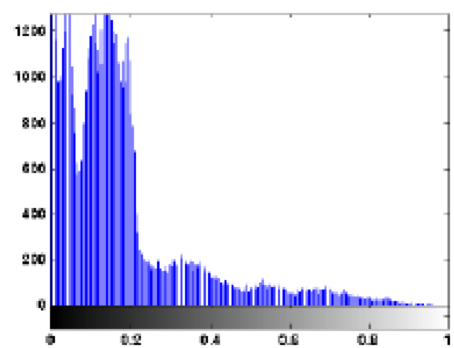
- This algorithm works very well for finding thresholds when the histogram is suitable

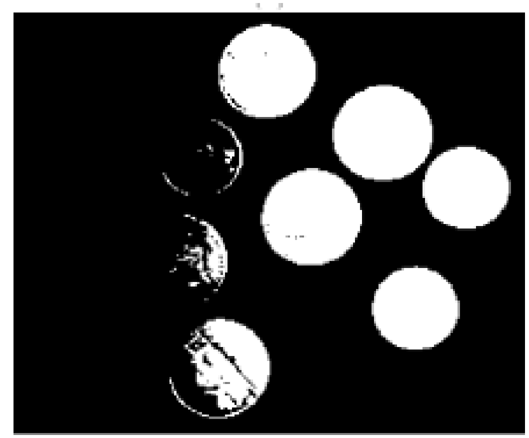
Image Thresholding and Illumination

- Even easy images may become hard to segment



Histogram lost its bimodal shape.

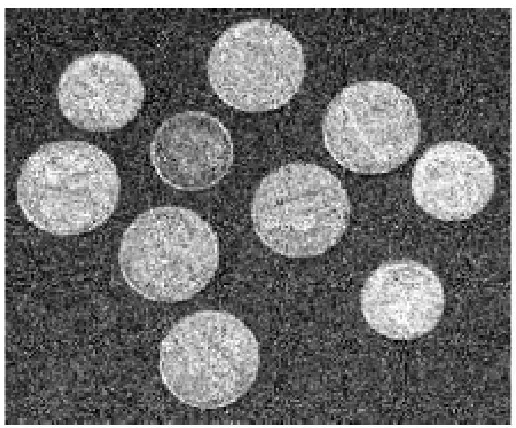




Result of segmentation with T = 0.25.

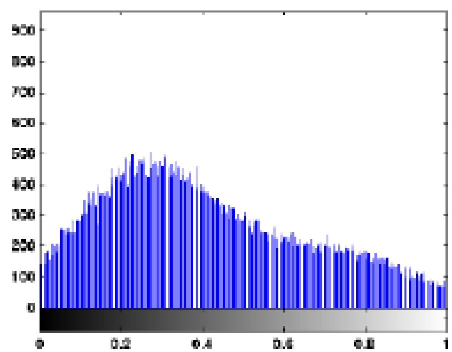
Image Thresholding and Noise

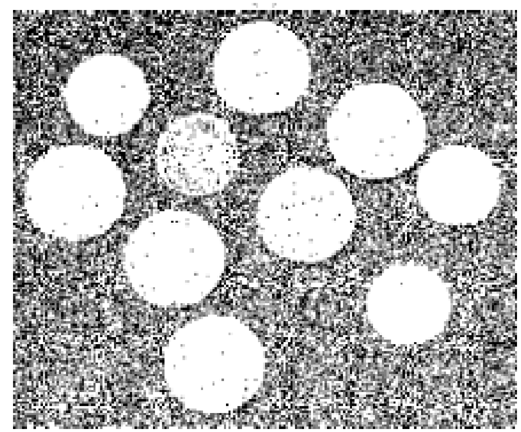
- Even easy images may become hard to segment



Gaussian noise with zero mean and variance 0.03.

Histogram lost its bimodal shape.



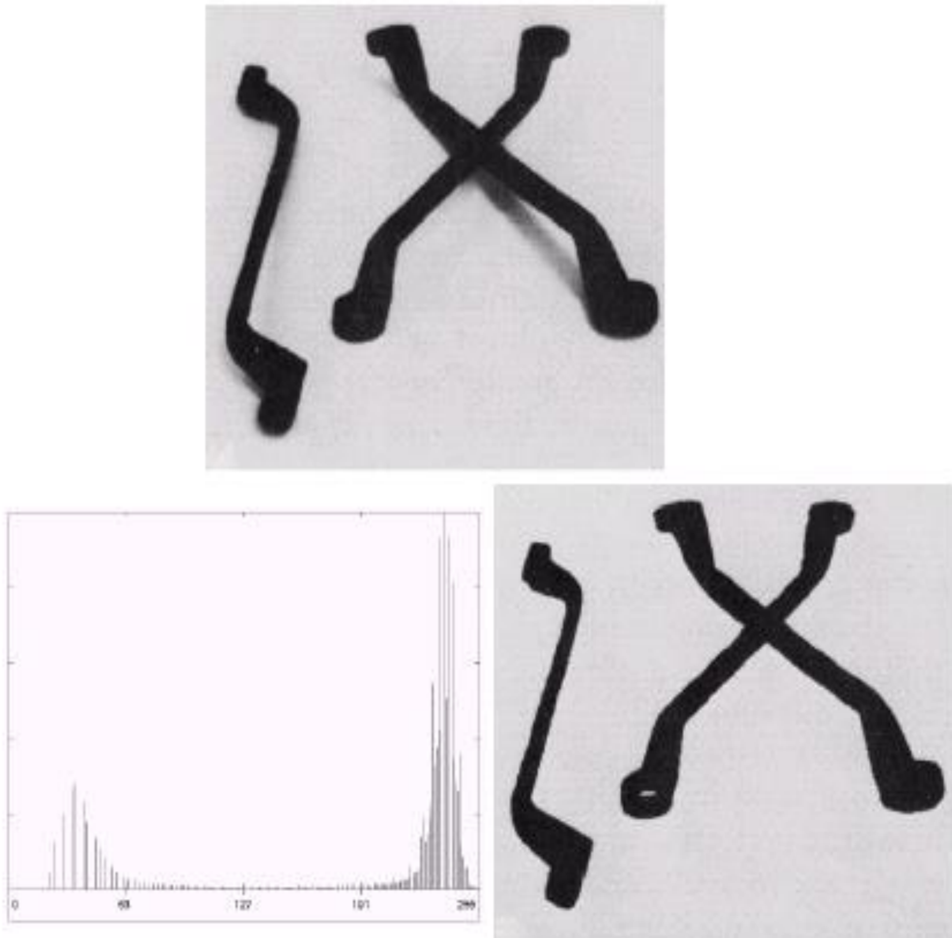


Result of segmentation with T = 0.25.

Local Thresholding

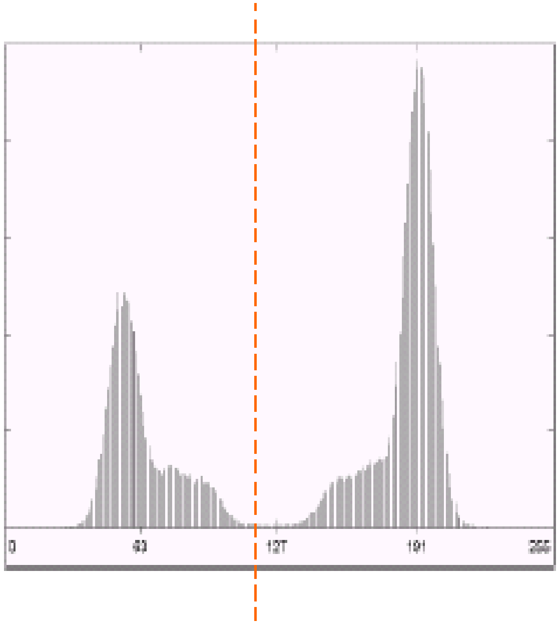
- Threshold blocks of pixels, one block at a time.
  - If the blocks that are too small: large computational cost.
  - If the blocks that are too large: results may not be substantially better than the ones obtained with global thresholding.

- **Global Thresholding: When does it work?**



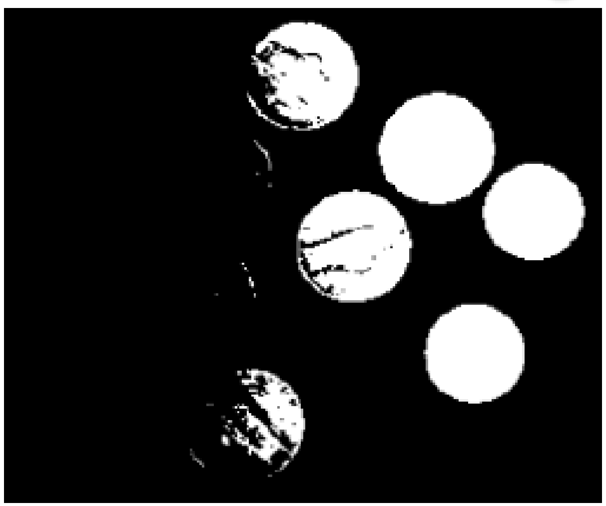
a
b c

FIGURE 10.28 (a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

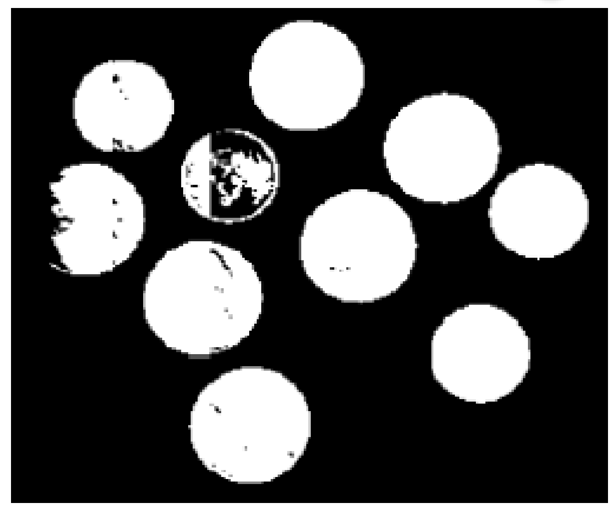

0
191

Local Thresholding Example

Global thresholding

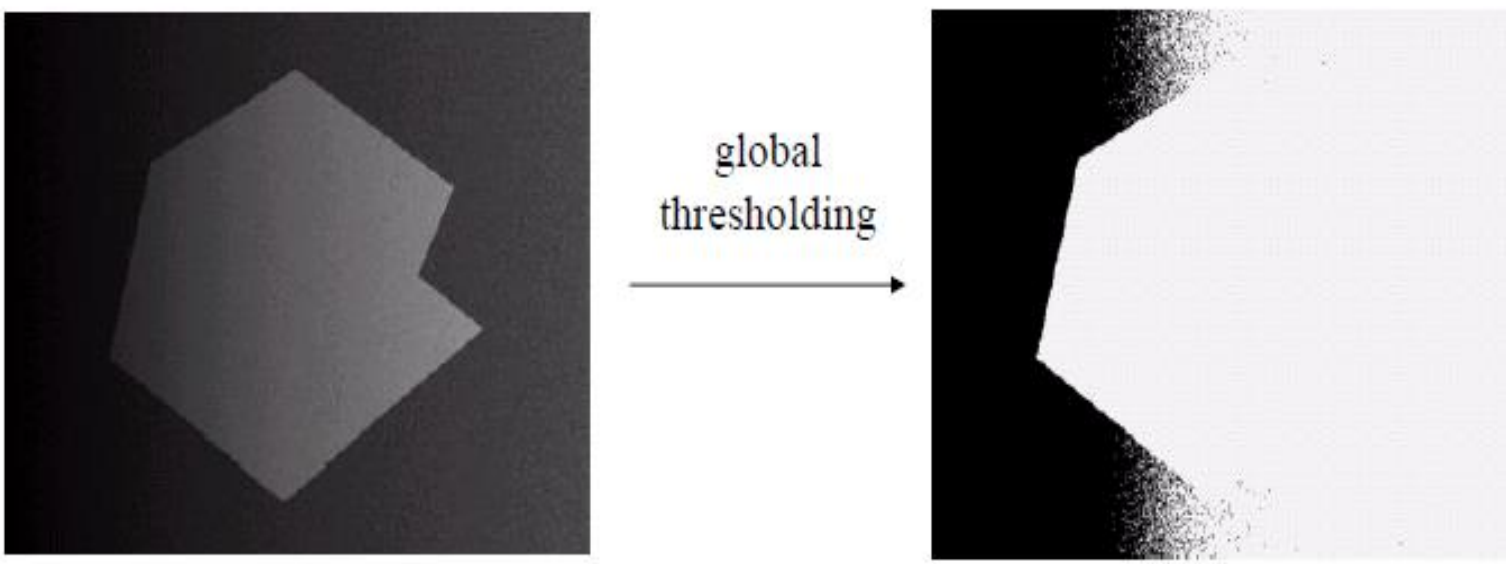


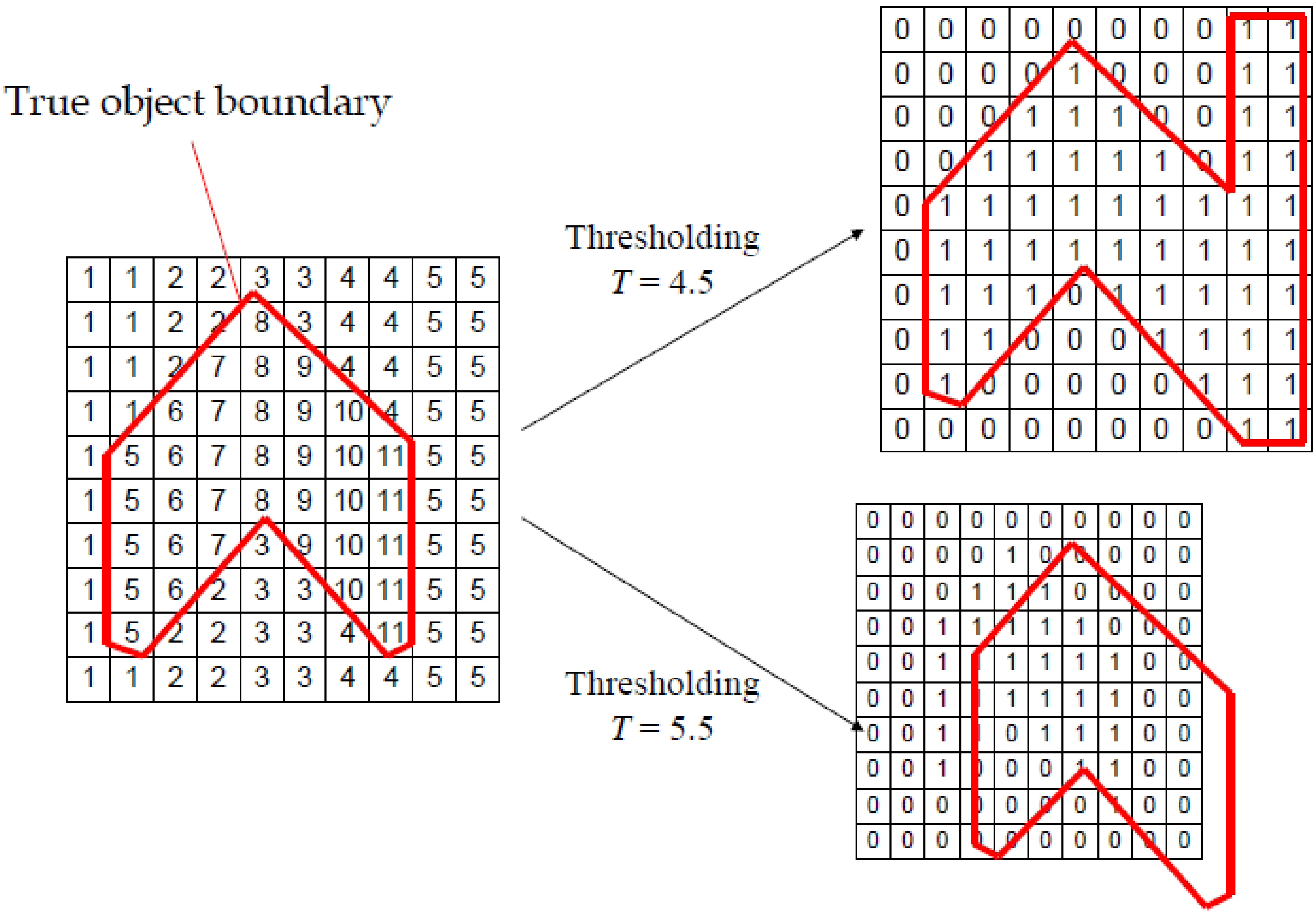
Local thresholding



By Oge Marques

- **Global Thresholding: When does it not work?**
  - A meaningful global threshold may not exist
  - Uneven illumination can really upset a single valued thresholding scheme
  - Image-dependent




True object boundary
1 1 2 2 3 3 4 4 5 5
1 1 2 2 8 3 4 4 5 5
1 1 2 7 8 9 4 4 5 5
1 1 6 7 8 9 10 4 5 5
1 5 6 7 8 9 10 11 5 5
1 5 6 7 8 9 10 11 5 5
1 5 6 7 3 9 10 11 5 5
1 5 6 2 3 3 10 11 5 5
1 5 2 2 3 3 4 11 5 5
1 1 2 2 3 3 4 4 5 5
Thresholding
T = 4.5
0 0 0 0 0 0 0 0 1 1
0 0 0 0 1 0 0 0 1 1
0 0 0 1 1 1 0 0 1 1
0 0 1 1 1 1 1 0 1 1
0 1 1 1 1 1 1 1 1 1
0 1 1 1 1 1 1 1 1 1
0 1 1 1 0 1 1 1 1 1
0 1 1 0 0 0 1 1 1 1
0 1 0 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 1 1
Thresholding
T = 5.5
0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0
0 0 0 1 1 1 0 0 0 0
0 0 1 1 1 1 1 0 0 0
0 0 1 1 1 1 1 1 0 0
0 0 1 1 1 1 1 1 0 0
0 0 1 1 0 1 1 1 0 0
0 0 1 0 0 0 1 1 0 0
0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0

Adaptive Thresholding Method

- An approach to handling situations in which single value thresholding will not work is to divide an image into sub images and threshold these individually
- Since the threshold for each pixel depends on its location within an image this technique is said to *adaptive*

- **Improvement over Global Solution**
  - Spatially adaptive thresholding
  - Localized processing

1	1	2	2	3	3	4	4	5	5
1	1	2	2	8	3	4	4	5	5
1	1	2	7	8	9	4	4	5	5
1	1	6	7	8	9	10	4	5	5
1	5	6	7	8	9	10	11	5	5
1	5	6	7	8	9	10	11	5	5
1	5	6	7	3	9	10	11	5	5
1	5	6	2	3	3	10	11	5	5
1	5	2	2	3	3	4	11	5	5
1	1	2	2	3	3	4	4	5	5

Split →

1	1	2	2	3
1	1	2	2	8
1	1	2	7	8
1	1	6	7	8
1	5	6	7	8

3	4	4	5	5
3	4	4	5	5
9	4	4	5	5
9	10	4	5	5
9	10	11	5	5

1	5	6	7	8
1	5	6	7	3
1	5	6	2	3
1	5	2	2	3
1	1	2	2	3

9	10	11	5	5
9	10	11	5	5
3	10	11	5	5
3	4	11	5	5
3	4	4	5	5

spatially adaptive threshold selection

1	1	2	2	3
1	1	2	2	8
1	1	2	7	8
1	1	6	7	8
1	5	6	7	8

Thresholding
$T = 4$

0	0	0	0	0
0	0	0	0	1
0	0	0	1	1
0	0	1	1	1
0	1	1	1	1

3	4	4	5	5
3	4	4	5	5
9	4	4	5	5
9	10	4	5	5
9	10	11	5	5

Thresholding
$T = 7$

0	0	0	0	0
0	0	0	0	0
1	0	0	0	0
1	1	0	0	0
1	1	1	0	0

1	5	6	7	8
1	5	6	7	3
1	5	6	2	3
1	5	2	2	3
1	1	2	2	3

Thresholding
$T = 4$

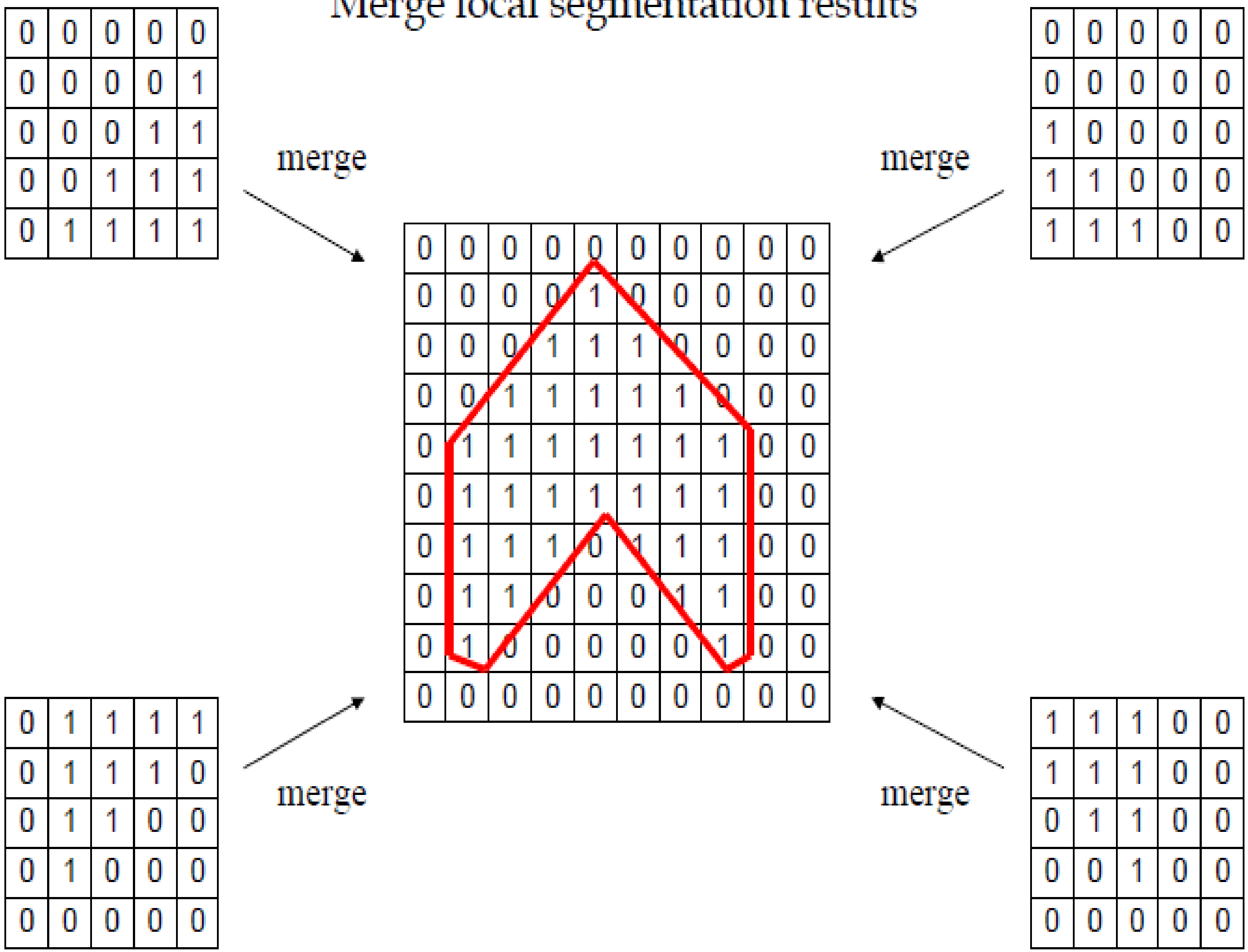
0	1	1	1	1
0	1	1	1	0
0	1	1	0	0
0	1	0	0	0
0	0	0	0	0

9	10	11	5	5
9	10	11	5	5
3	10	11	5	5
3	4	11	5	5
3	4	4	5	5

Thresholding
$T = 7$

1	1	1	0	0
1	1	1	0	0
0	1	1	0	0
0	0	1	0	0
0	0	0	0	0

Merge local segmentation results



Limitations of Thresholding

- Operates on each image pixel independently.
  - Pixels must have similar values.
- Spatial coherency cannot be satisfied.
  - Pixels do not have to be connected.

Region-based Segmentation

Region-based Segmentation

- A pixel cannot be considered a part of an object based solely on its gray value.
- Incorporates measures of connectivity among pixels in order to decide whether those pixels belong to the same region (or object).

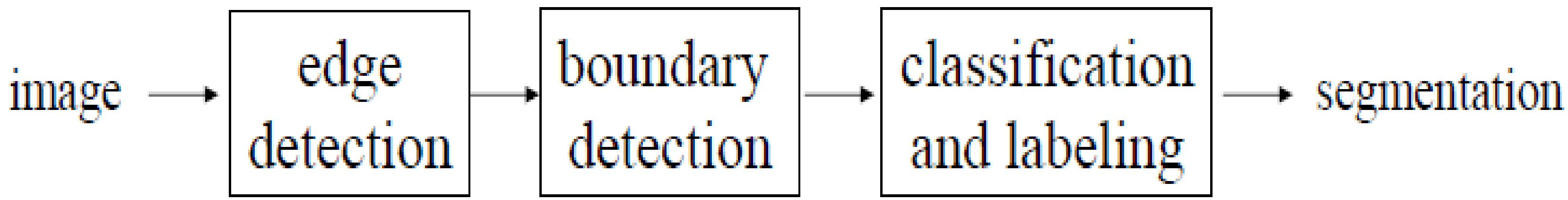
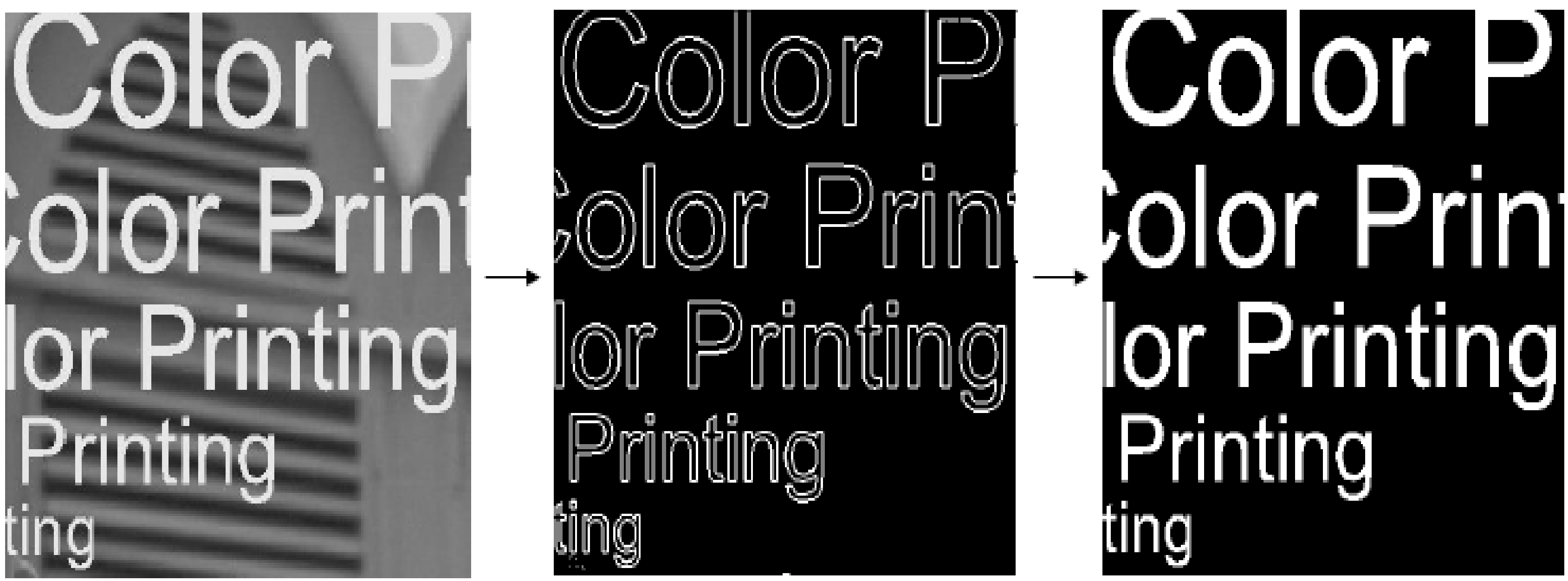


image
edge detection
boundary detection
classification and labeling
segmentation



Region-Based Segmentation

Divide an image I into n regions $R_1, R_2, \ldots, R_n$ such that:

1. $\bigcup_{i=1}^{n} R_i = I$
2. R_i is a connected region, $i = 1, 2, \cdots, n$.
3. $R_i \cap R_j = \emptyset$ for all i and j, $i \neq j$.
4. $P(R_i)$ = TRUE for $i = 1, 2, \cdots, n$.
5. $P(R_i \cup R_j)$ = FALSE for any adjacent regions R_i and R_j.

where $P(R_i)$ is a logical predicate defined over the points in set R_i and $\emptyset$ is the empty set.

Region-Based Segmentation

- Logical predicates (also called *homogeneity criteria*) include:
  - Local mean relative to global mean: the average intensity in a region is significantly different than the average gray level in the whole image.
  - Local standard deviation relative to global mean: The standard deviation of the pixel intensities in a region is less than a small percentage of the average gray level in the whole image.
  - Variance: At least a certain percentage of the pixels in a region are within two standard deviations of the local mean.

Region Growing Algorithm

```
Let f(x,y) be the input image
Define a set of regions R1, R2, ..., Rn, each consisting of a
          single seed pixel
repeat
   for i = 1 to n do
      for each pixel p at the border of Ri do
         for all neighbors of p do
            Let (x,y) be the neighbor's coordinates
            Let Mi be the mean gray level of pixels in Ri
            if the neighbor is unassigned and
                      |f(x,y) - Mi| <= Delta then
               Add neighbor to Ri
               Update Mi
            end if
         end for
      end for
   end for
until no more pixels can be assigned to regions
```


- **Region Growing**
  - Start from a seed, and let it grow (include similar neighborhood)

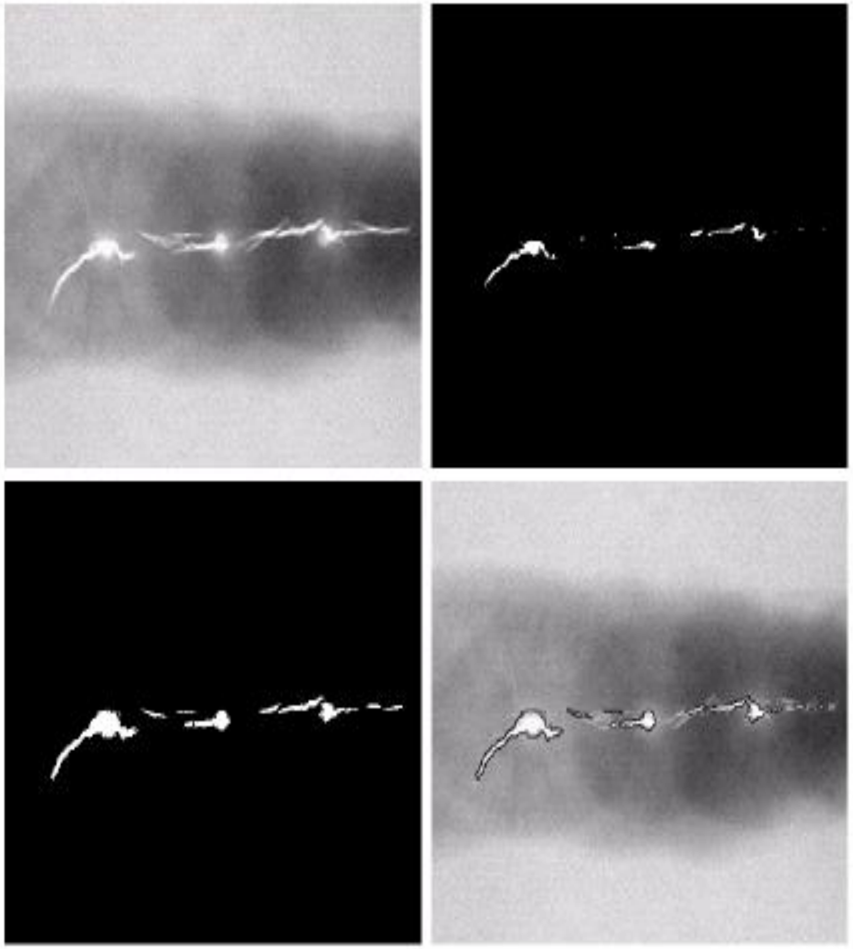
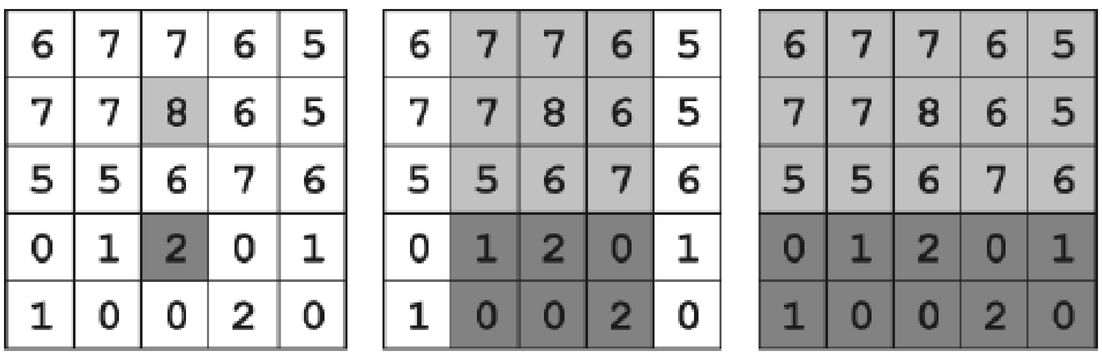


FIGURE 10.40 (a) Image showing defective welds. (b) Seed points. (c) Result of region growing. (d) Boundaries of segmented defective welds (in black). (Original image courtesy of X-TEK Systems, Ltd.).

Key: similarity measure

Region-Growing Example

$$P(R_i) = \begin{cases} \text{TRUE} & \text{if } |f(x,y) - \mu_i| \le \Delta \\ \text{FALSE} & \text{otherwise} \end{cases}$$



Seed pixels

Results after first iteration, delta = 3

Results after second iteration

Region Growing Limitations

- Significantly different results may be obtained when switching between 4-connectivity and 8-connectivity criteria.
- Segmentation results are sensitive to the choice of logical uniformity predicate.
- The number of seeds provided by the user may not be sufficient to assign every pixel to a region, or some may belong to the same region.

Region Splitting and Merging

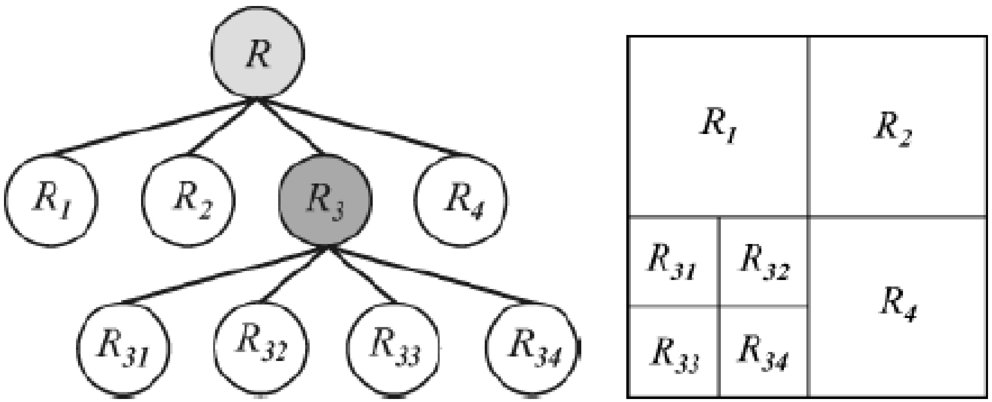
- Start from the entire image and partition (split) it into smaller sub-images until each resulting region is considered homogeneous by some criterion.

- Merge two or more adjacent regions into one region if they satisfy the homogeneity criterion.

Region Splitting and Merging

1. Define a logical uniformity predicate $P(R_i)$.
2. Compute $P(R_i)$ for each region.
3. Split into four disjoint quadrants any region R_i for which $P(R_i)$ = FALSE.
4. Repeat steps 2 and 3 until all resulting regions satisfy the uniformity criterion, i.e., $P(R_i)$ = TRUE.
5. Merge any adjacent regions R_j and R_k for which $P(R_j \cup R_k)$ = TRUE.
6. Repeat step 5 until no further merging is possible.

Region Splitting and Merging

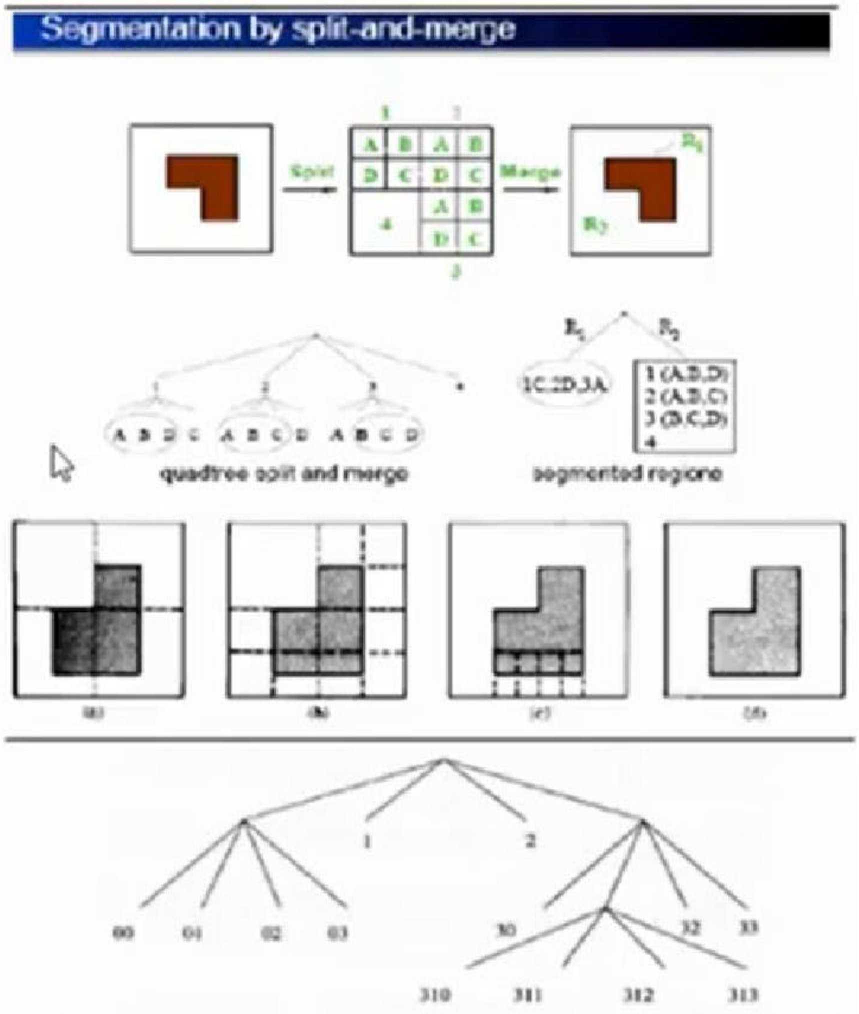


Region-Based Segmentation

Region Splitting and Merging Segmentation

Algorithm:

- If a region R is inhomogeneous (P(R)=FALSE), then R is split into four sub-regions.
- If two adjacent regions R_i,R_j are homogeneous ($P(R_i \cup R_j)$=TRUE), they are then merged.
- The algorithm stops when no further splitting or merging is possible.



Region-Based Segmentation

Region Splitting and Merging Segmentation

Example:

Apply the split and merge technique to segment the image shown in fig. below.

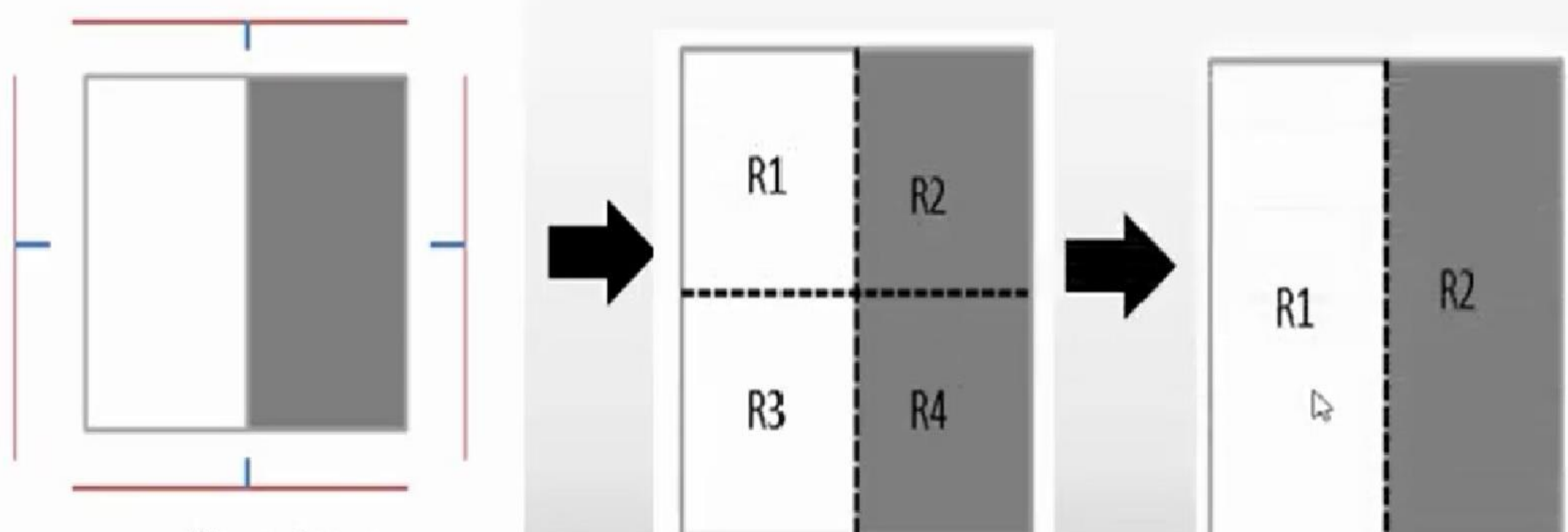


Figure: Image

Region-Based Segmentation

Region Splitting and Merging Segmentation

Example:

1	1	1	1	1	1	1	2
1	1	1	1	1	1	1	0
3	1	4	9	9	8	1	0
1	1	8	8	8	4	1	0
1	1	6	6	6	3	1	0
1	1	5	6	6	3	1	0
1	1	5	6	6	2	1	0
1	1	1	1	1	1	0	0

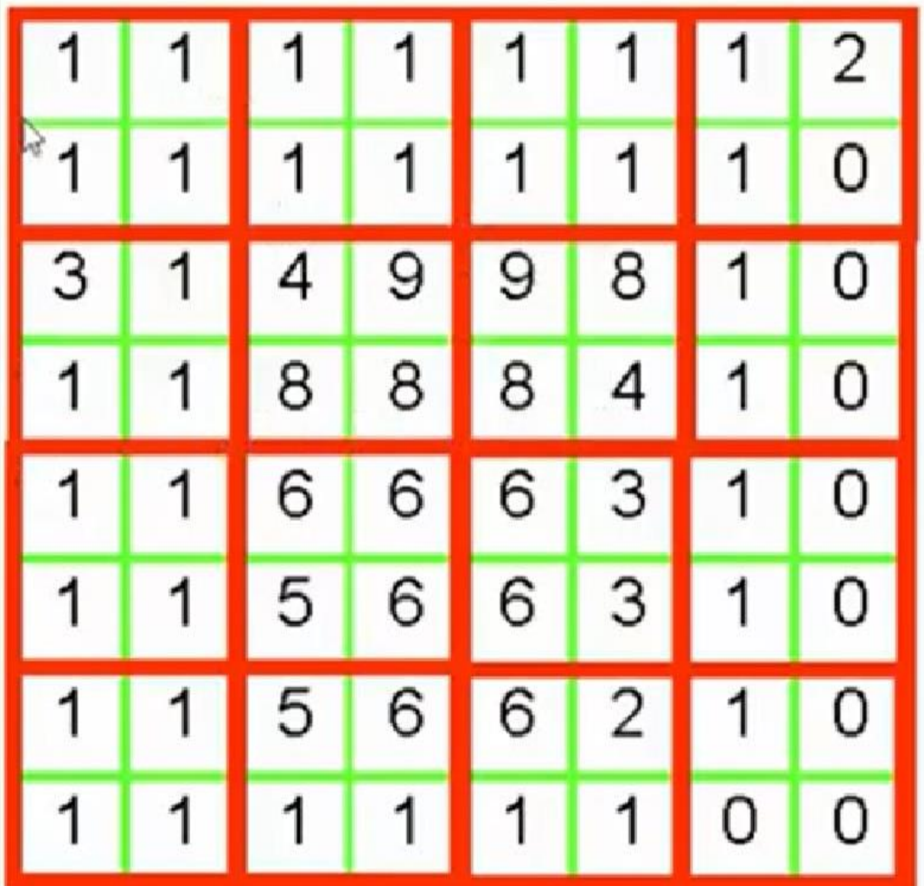
Sample image

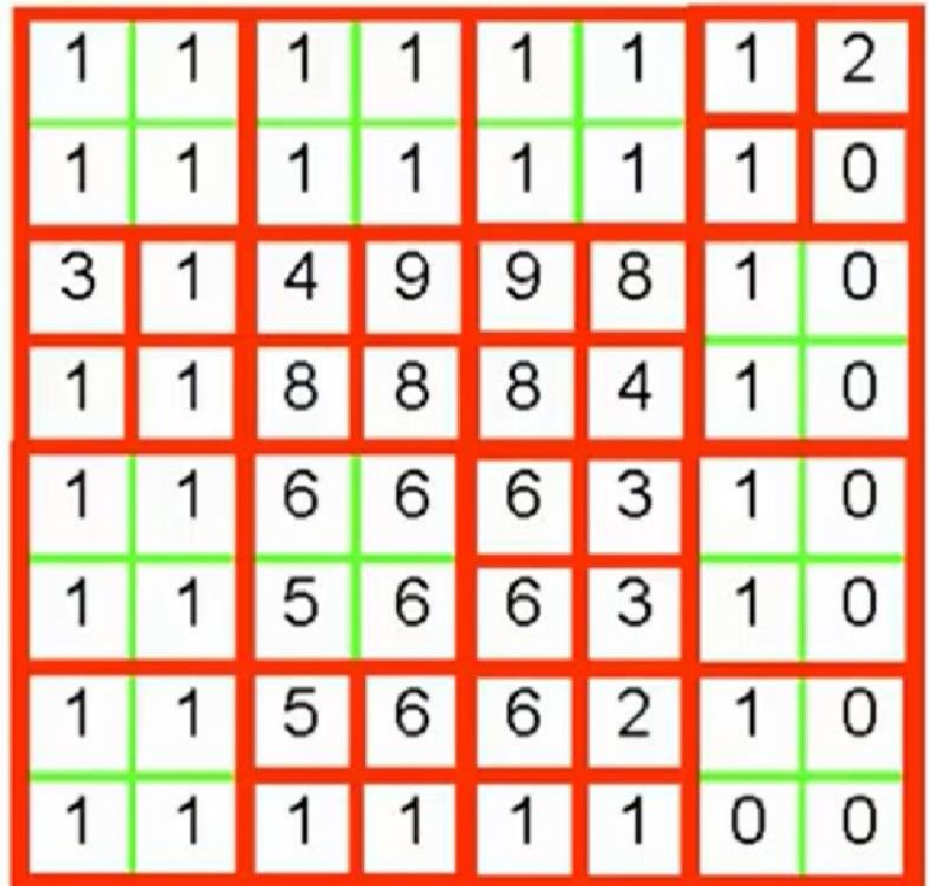
1	1	1	1	1	1	1	2
1	1	1	1	1	1	1	0
3	1	4	9	9	8	1	0
1	1	8	8	8	4	1	0
1	1	6	6	6	3	1	0
1	1	5	6	6	3	1	0
1	1	5	6	6	2	1	0
1	1	1	1	1	1	0	0

First split

Region-Based Segmentation

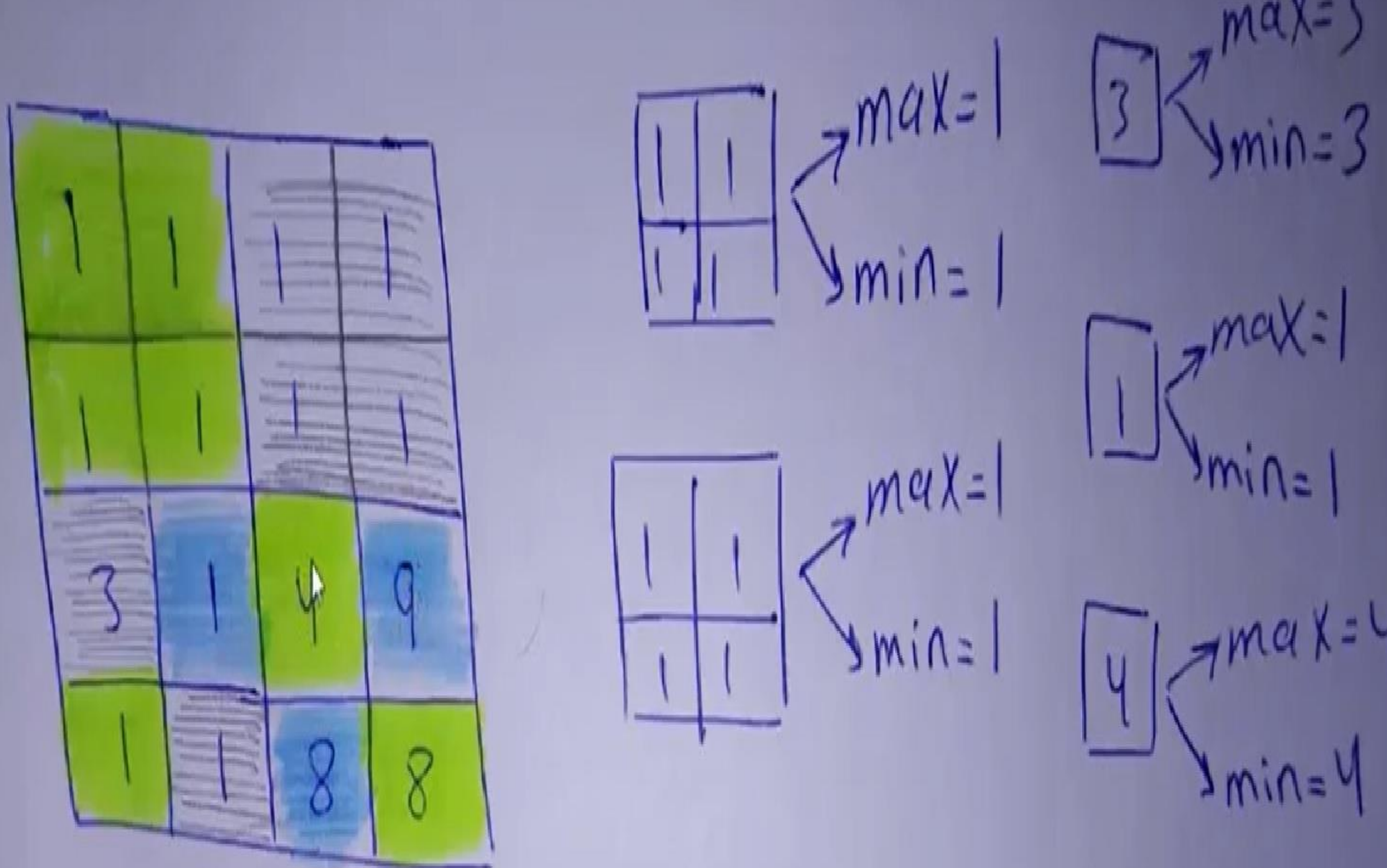
Region Splitting and Merging Segmentation

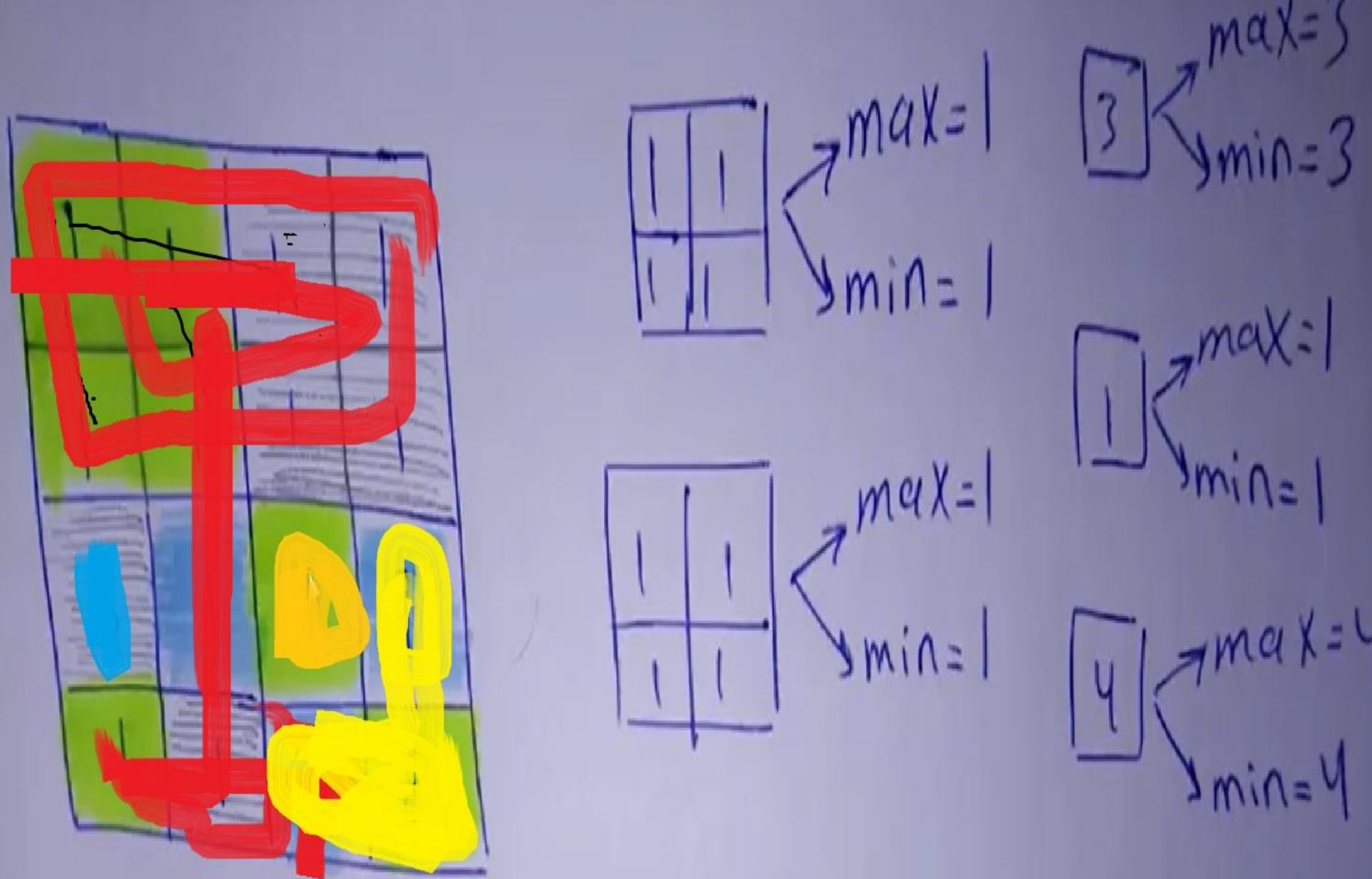




Second split

Third split


max=1
min=1
max=3
min=3
max=1
min=1
max=1
min=1
max=4
min=4



max=1
min=1
3
max=3
min=3
1
max=1
min=1
max=1
min=1
4
max=4
min=4

Region-Based Segmentation

Region Splitting and Merging Segmentation

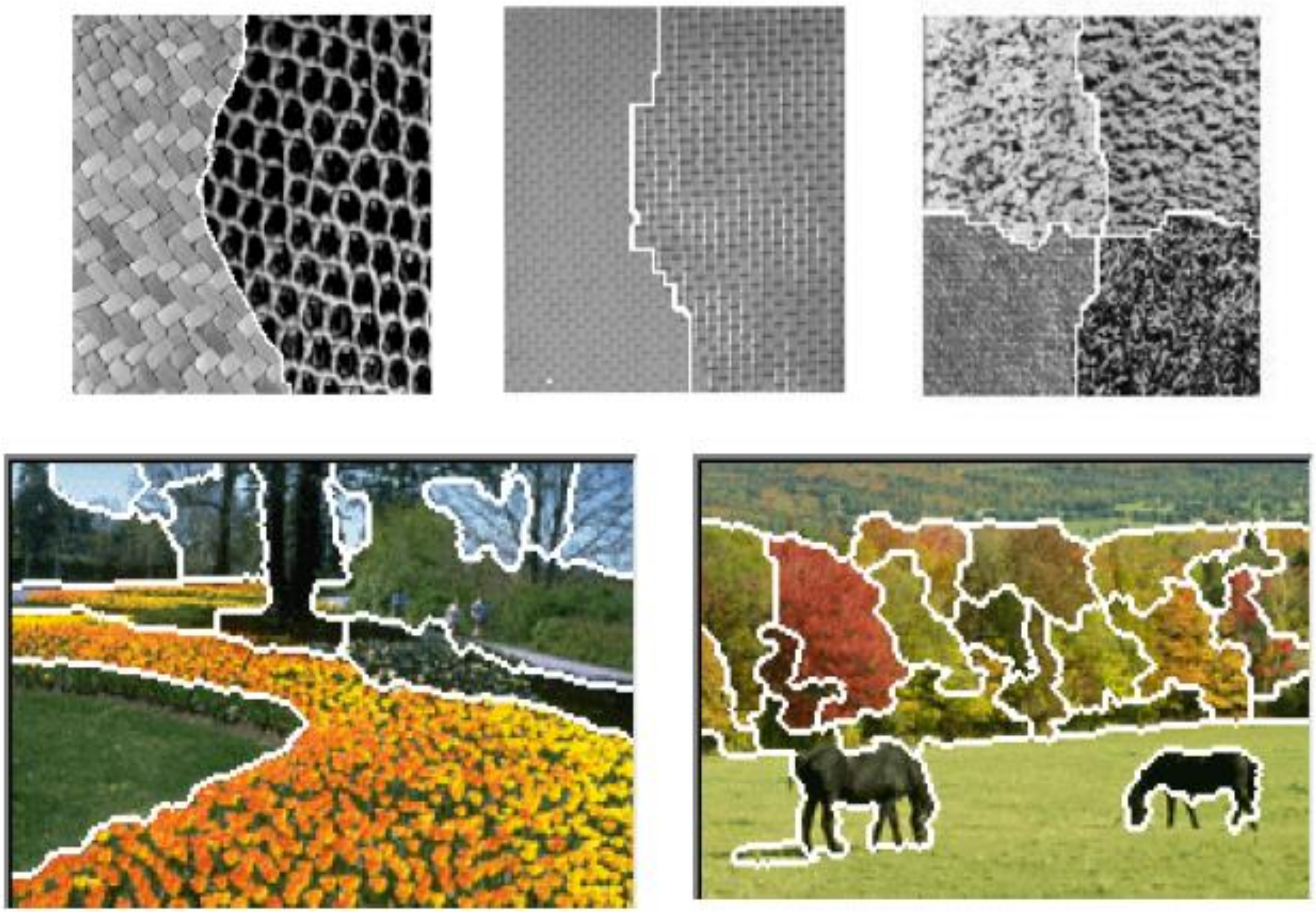
1	1	1	1	1	1	1	2
1	1	1	1	1	1	1	0
3	1	4	9	9	8	1	0
1	1	8	8	8	4	1	0
1	1	6	6	6	3	1	0
1	1	5	6	6	3	1	0
1	1	5	6	6	2	1	0
1	1	1	1	1	1	0	0

Merge

1	1	1	1	1	1	1	2
1	1	1	1	1	1	1	0
3	1	4	9	9	8	1	0
1	1	8	8	8	4	1	0
1	1	6	6	6	3	1	0
1	1	5	6	6	3	1	0
1	1	5	6	6	2	1	0
1	1	1	1	1	1	0	0

Final result

Hard Problem: Textures



Similarity measure makes the difference