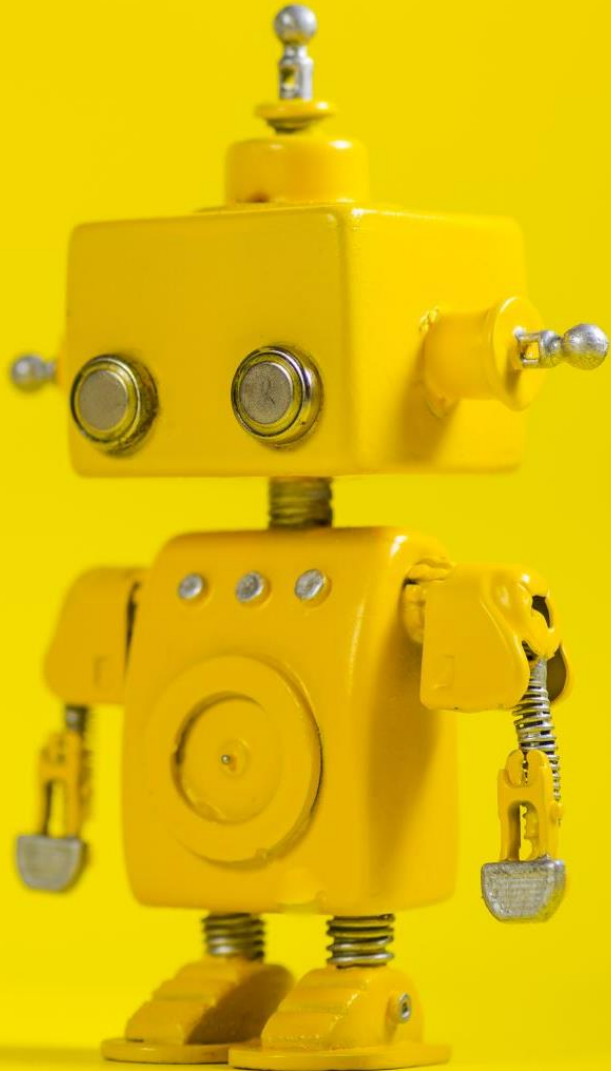


CAP 4453

Robot Vision

Dr. Gonzalo Vaca-Castaño
gonzalo.vacacastano@ucf.edu





Administrative details

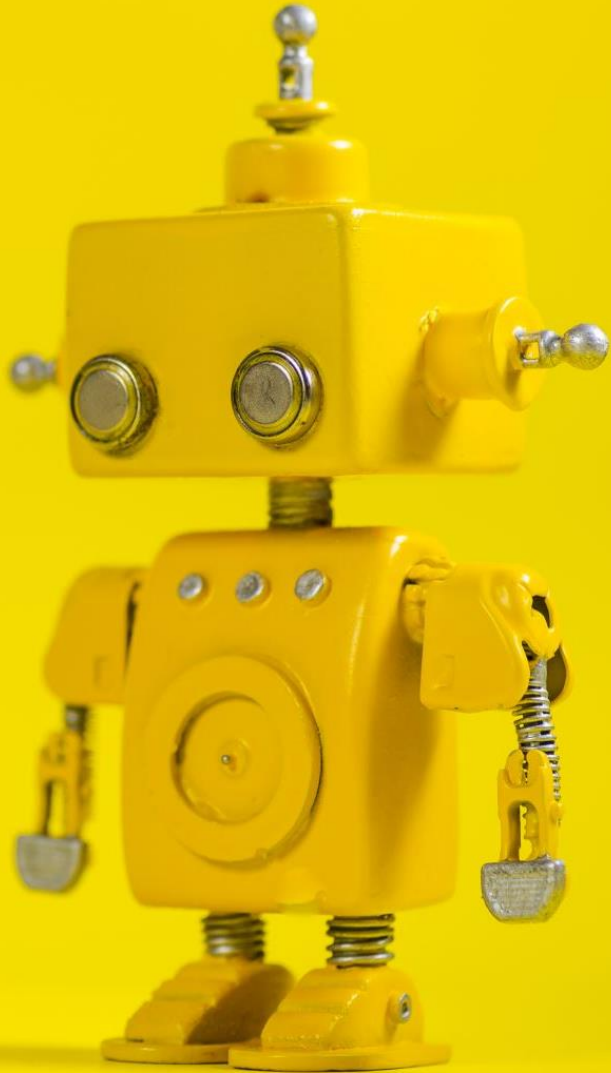
- Allow grader to review your homework:
 - moazam4453@gmail.com
- Homework 1 review
- Any issues with hw3 ?



Credits

- Some slides comes directly from:
 - Yogesh S Rawat (UCF)
 - Noah Snavelly (Cornell)
 - Ioannis (Yannis) Gkioulekas (CMU)
 - Mubarak Shah (UCF)
 - S. Seitz
 - James Tompkin
 - Ulas Bagci
 - L. Lazebnik

Short Review from last class

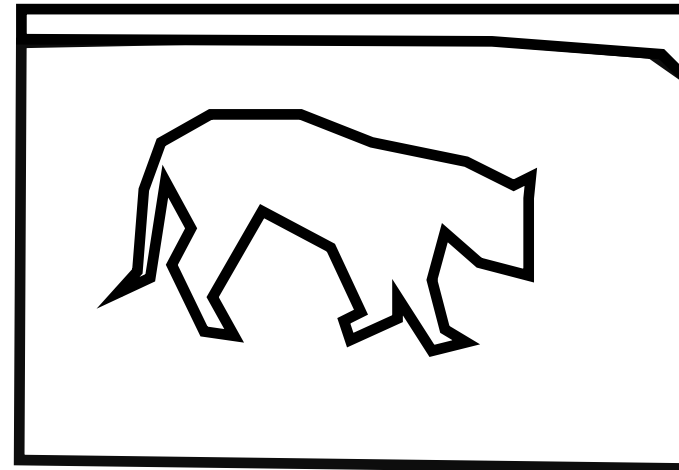
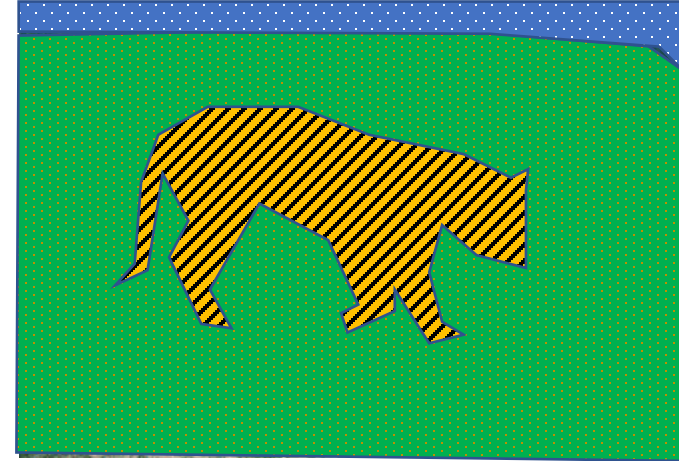


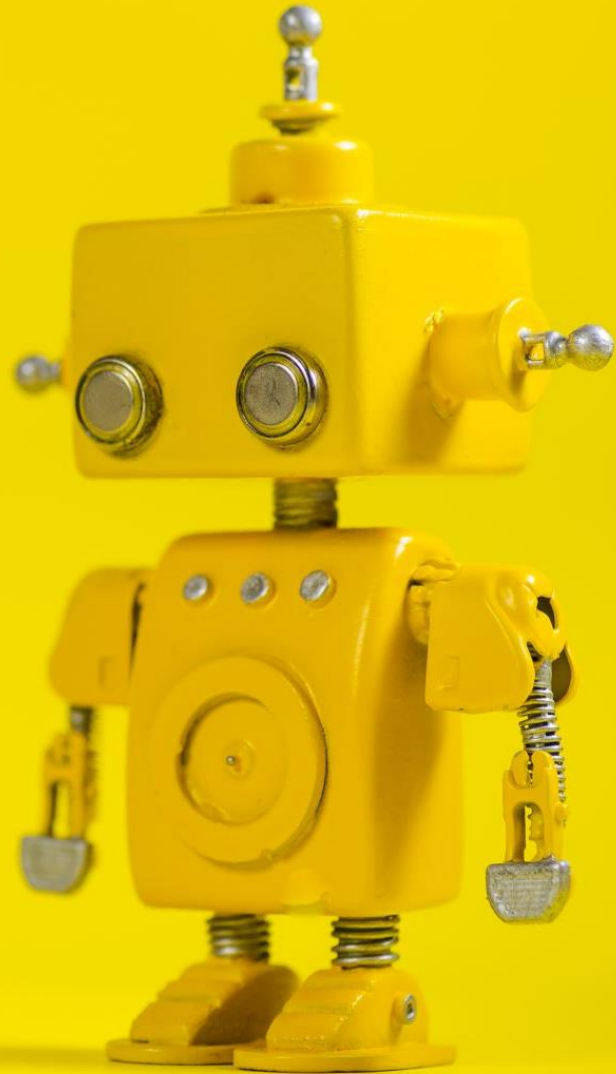


Last 2 classes

- Gradient operators
 - Prewit
 - Sobel
- Marr-Hildreth (Laplacian of Gaussian)
- Canny (Gradient of Gaussian)

Regions \leftrightarrow Boundaries





Robot Vision

7. Segmentation I



Outline

- Image segmentation basics
- Thresholding based
 - Binarization
 - Otsu
- Region based
 - Merging
 - Splitting
- Clustering based
 - K-means (SLIC)



Outline

- **Image segmentation basics**
- Thresholding based
 - Binarization
 - Otsu
- Region based
 - Merging
 - Splitting
- Clustering based
 - K-means (SLIC)

Image segmentation

- Partition an image into a collection of set of pixels
 - Meaningful regions (coherent objects)
 - Linear structures (line, curve, ...)
 - Shapes (circles, ellipses, ...)





Image segmentation

- Content base image retrieval
- Machine vision
- Medical imaging applications
- Object detection (face detection, ..)
- 3D reconstruction
- Object/motion tracking
- ...

Image segmentation

- In computer vision, image segmentation is one of the oldest and most widely studied problems
 - Early techniques -> region splitting or merging
 - Recent techniques -> Energy minimization, hybrid methods, and deep learning

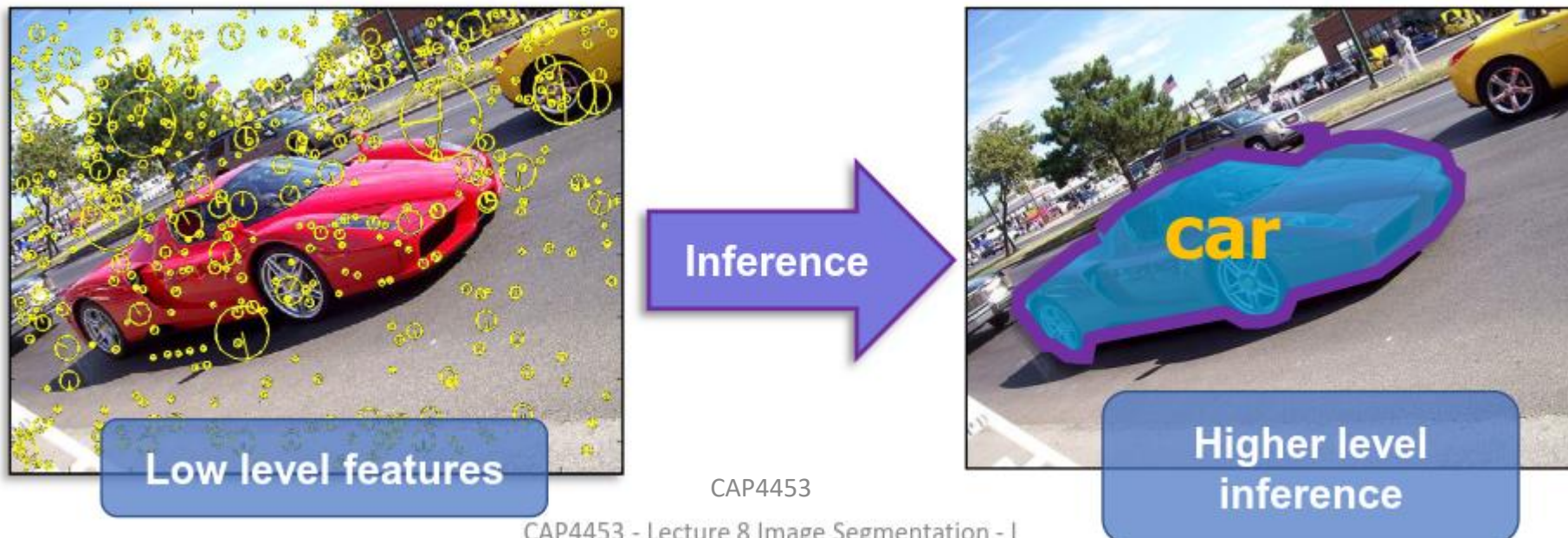


Image segmentation methods

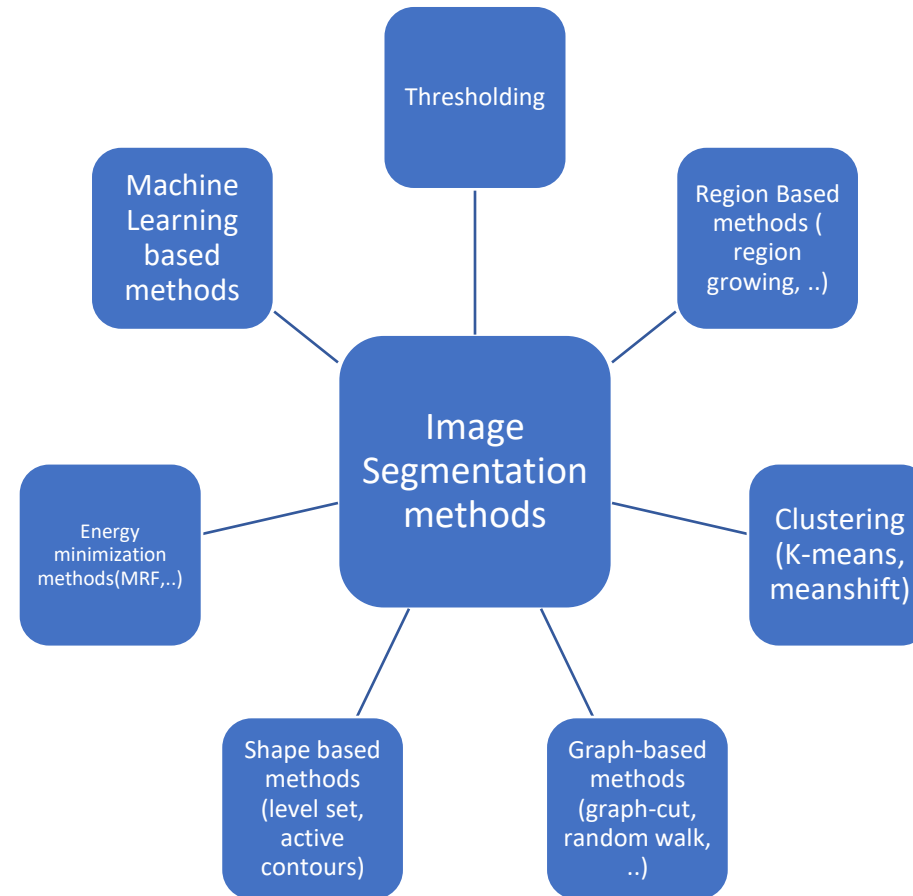


Image segmentation

- Image segmentation partitions an image into regions called segments.

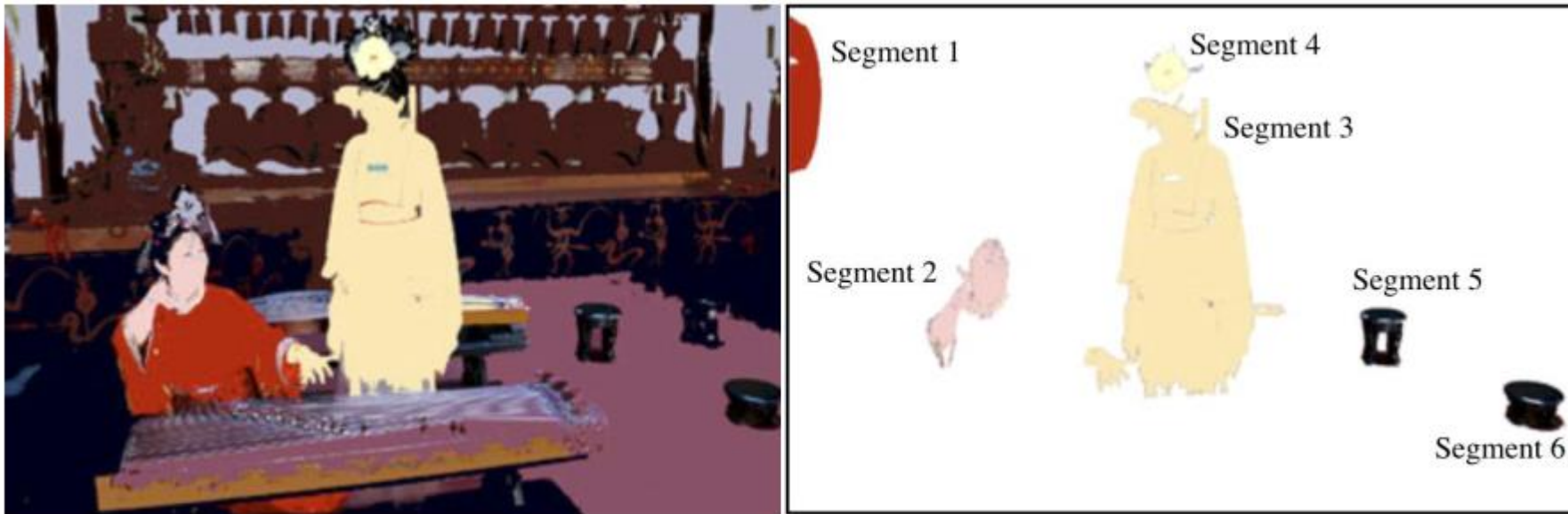
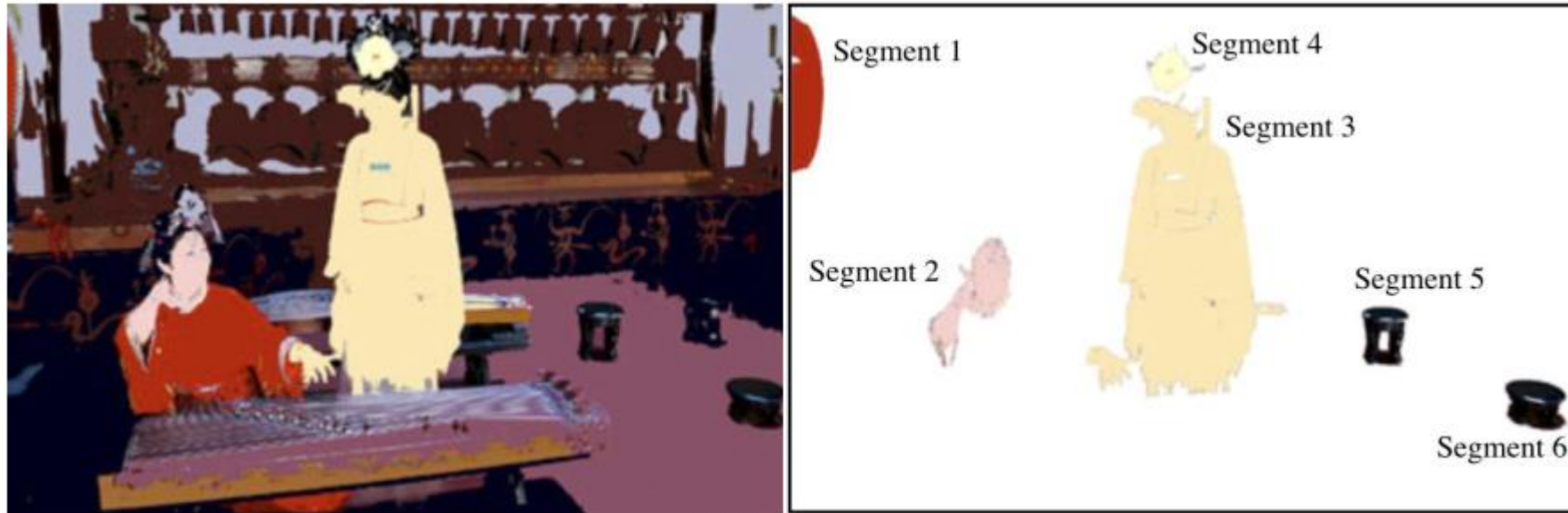


Image segmentation

- Image segmentation partitions an image into regions called segments.



- Image segmentation creates segments of connected pixels by analyzing some similarity criteria:
 - intensity, color, texture, histogram, features



Outline

- Image segmentation basics
- **Thresholding based**
 - **Binarization**
 - Otsu
- Region based
 - Merging
 - Splitting
- Clustering based
 - K-means (SLIC)

Image binarization

- Image binarization applies often just one global threshold T for mapping a scalar image I into a binary image



Image binarization

- Image binarization applies often just one global threshold T for mapping a scalar image I into a binary image

$$J(x, y) = \begin{cases} 0 & \text{if } I(x, y) < T \\ 1 & \text{otherwise.} \end{cases}$$

Image binarization

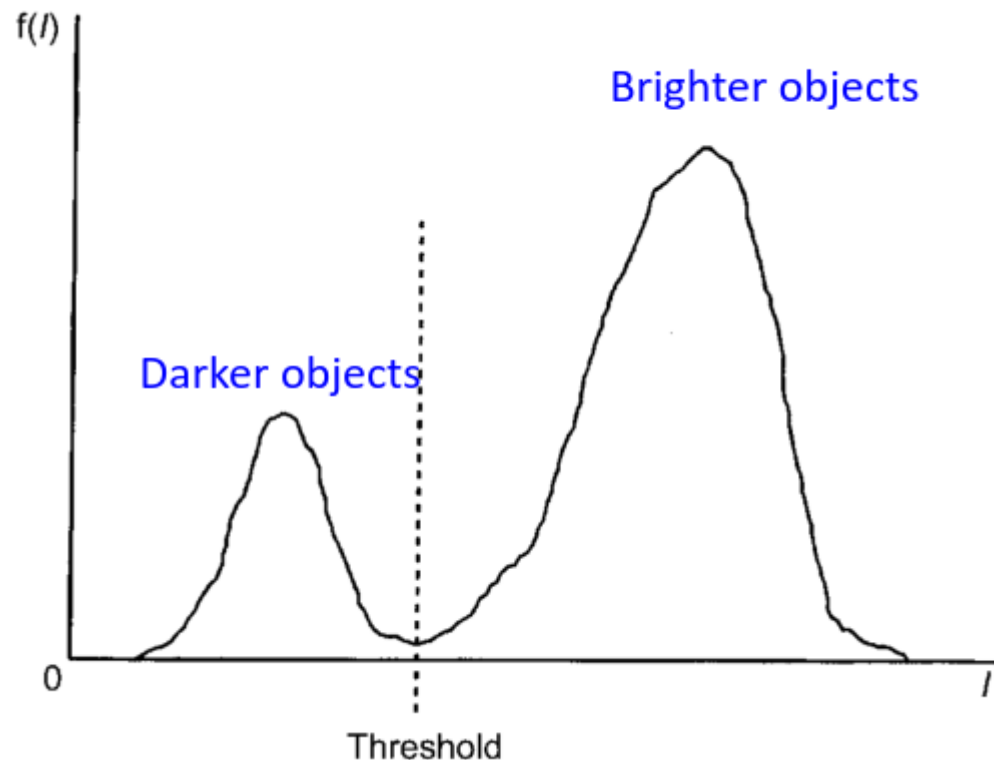
- Image binarization applies often just one global threshold T for mapping a scalar image I into a binary image

$$J(x, y) = \begin{cases} 0 & \text{if } I(x, y) < T \\ 1 & \text{otherwise.} \end{cases}$$

- The global threshold can be identified by an optimization strategy aiming at creating “large” connected regions and at reducing the number of small-sized regions, called artifacts.

Image binarization

- Thresholding: Most frequently employed method for determining threshold is based on histogram analysis of intensity levels



Peak on the left of the histogram corresponds to dark objects

Peak on the right of the histogram corresponds to brighter objects

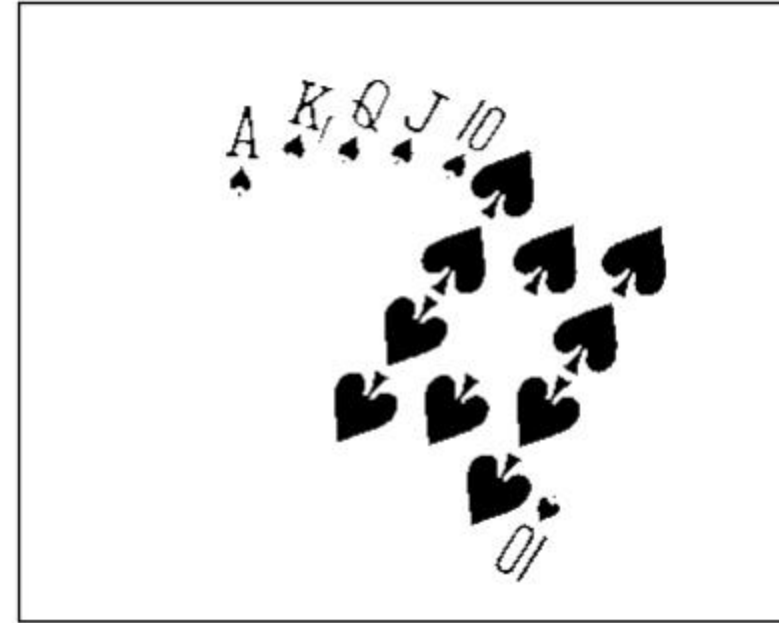
Difficulties

1. The valley may be so broad that it is difficult to locate a significant minimum
2. Number of minima due to type of details in the image
3. Noise
4. No visible valley
5. Histogram may be multi-modal

Thresholding examples

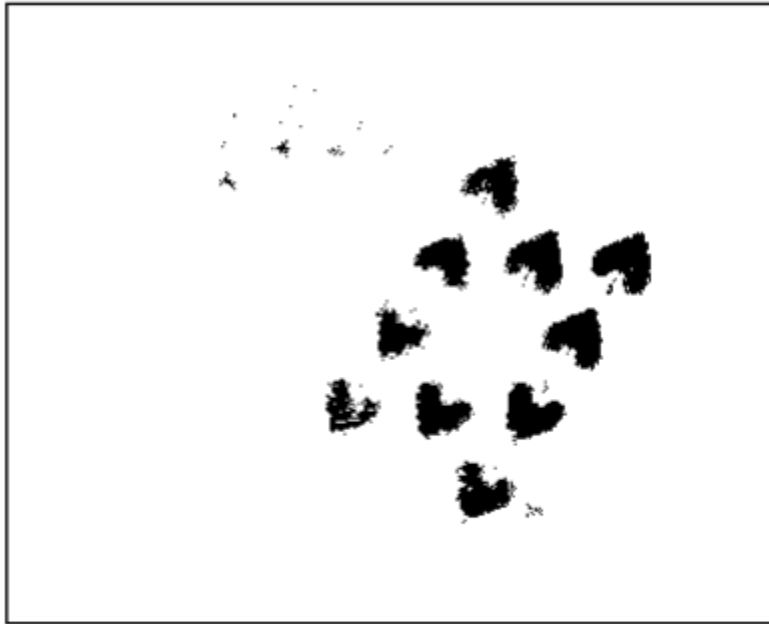


Original Image

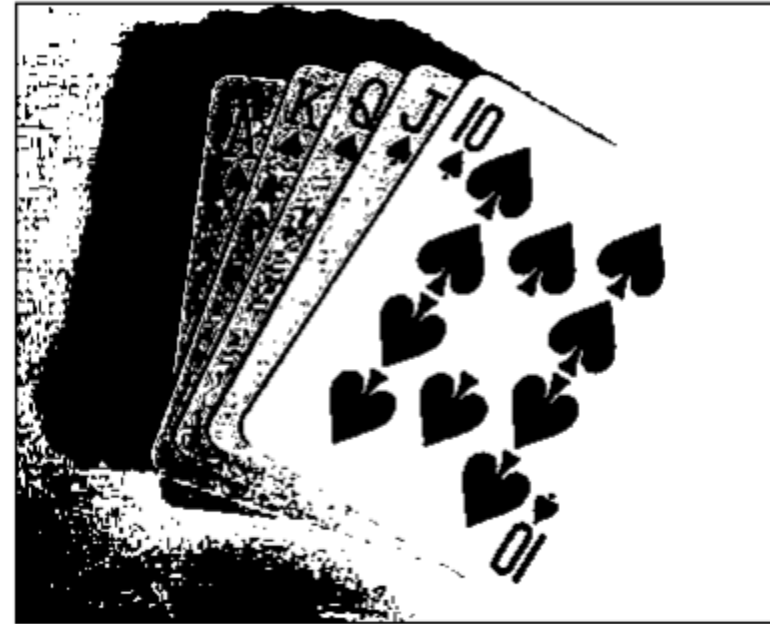


Thresholded Image

Thresholding examples

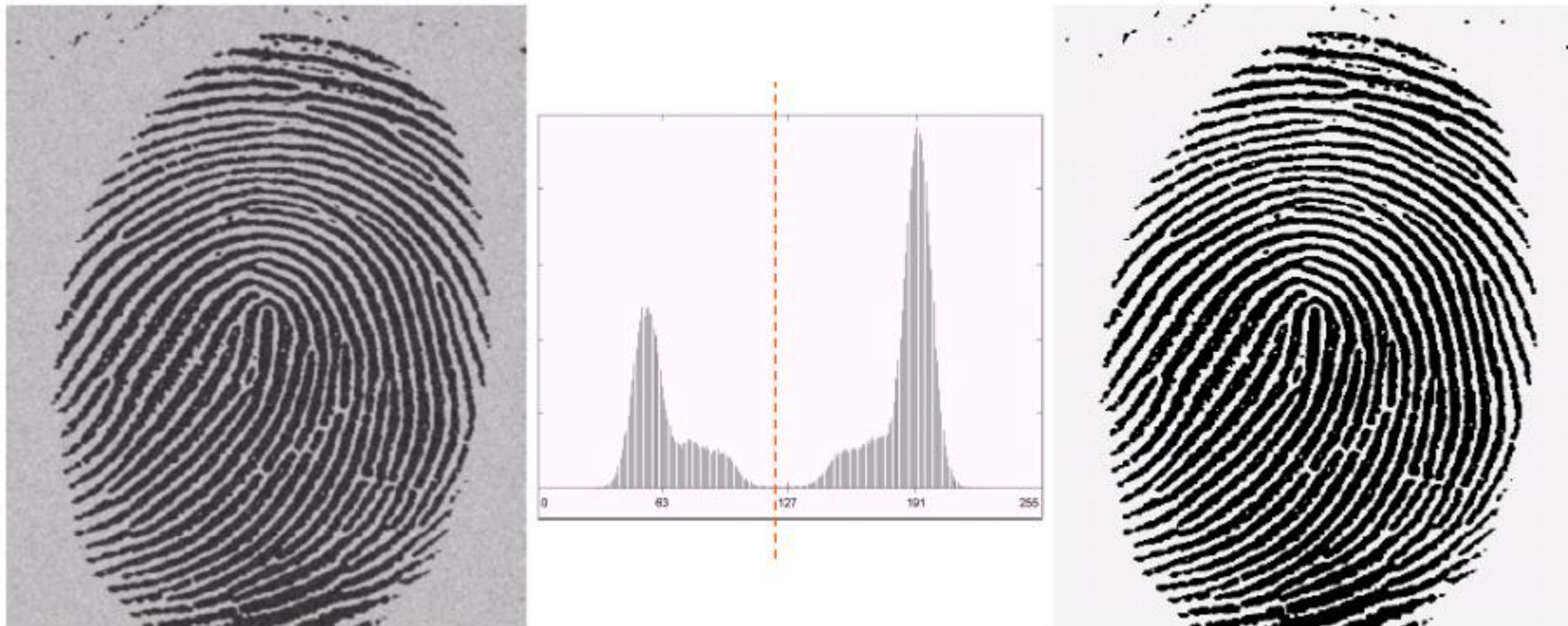


Threshold Too Low

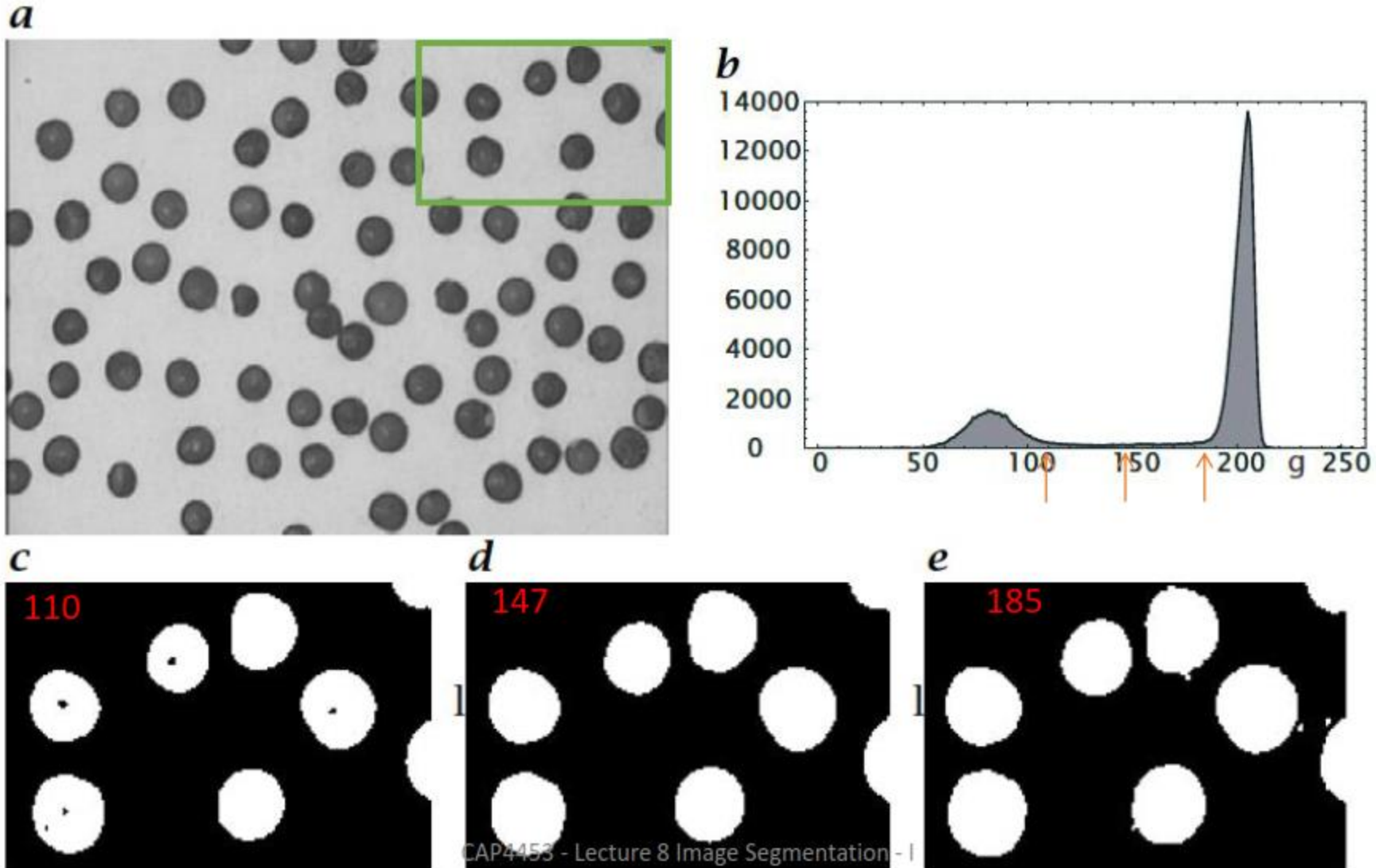


Threshold Too High

Thresholding examples



Thresholding examples





Outline

- Image segmentation basics
- Thresholding based
 - Binarization
 - **Otsu**
- Region based
 - Merging
 - Splitting
- Clustering based
 - K-means (SLIC)

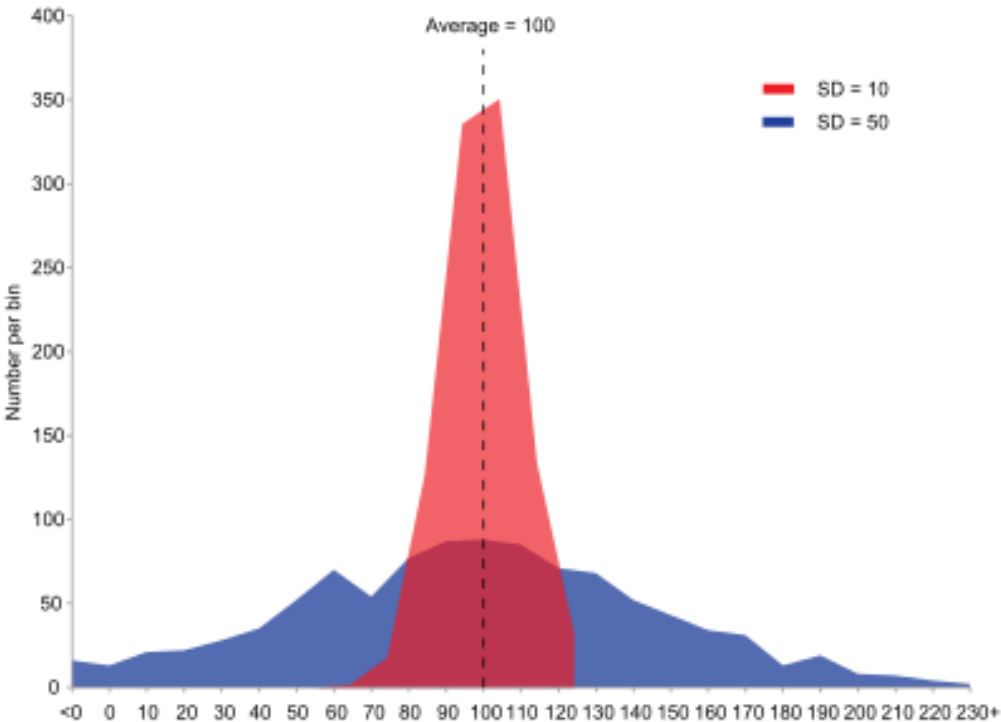
Variance

$$\text{Var}(X) = \mathbf{E}[(X - \mu)^2] = \sigma_X^2$$

where μ is the expected value. That is,

$$\mu = \sum_{i=1}^n p_i x_i.$$

$$p_i = \frac{\text{\#values in } i}{\text{sum all values (area under curve)}}$$



[Variance - Wikipedia](#)

Variance

$$\text{Var}(X) = \mathbb{E}[(X - \mu)^2] = \sigma_X^2$$

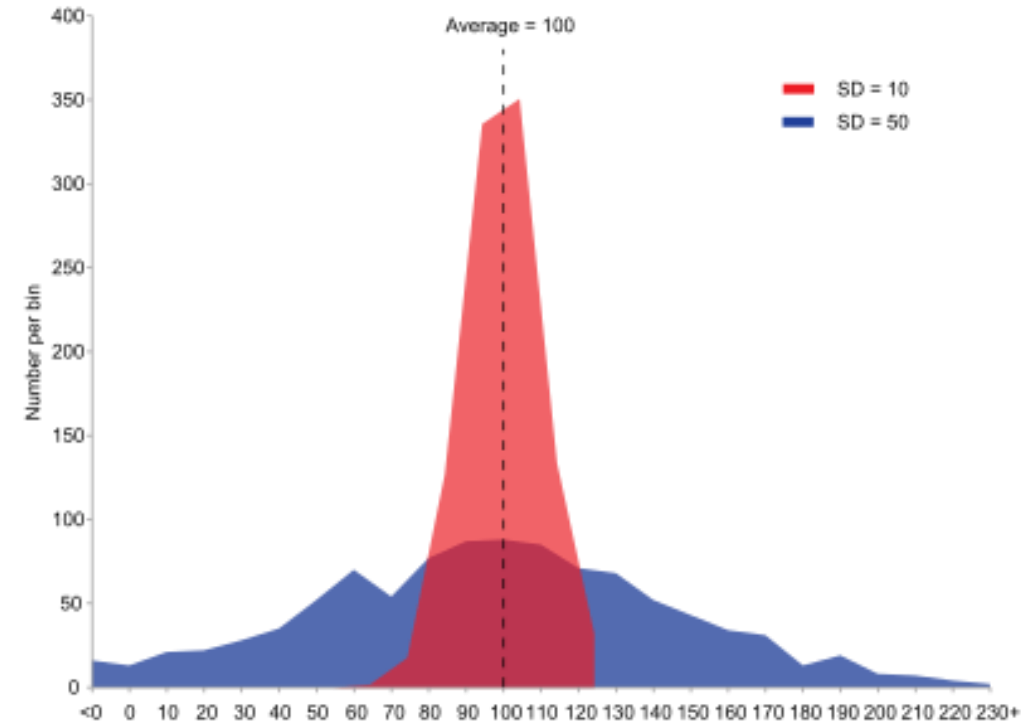
Discrete random variable [\[edit \]](#)

If the generator of random variable X is discrete with probability mass function $x_1 \mapsto p_1, x_2 \mapsto p_2, \dots, x_n \mapsto p_n$, then

$$\text{Var}(X) = \sum_{i=1}^n p_i \cdot (x_i - \mu)^2,$$

where μ is the expected value. That is,

$$\mu = \sum_{i=1}^n p_i x_i.$$

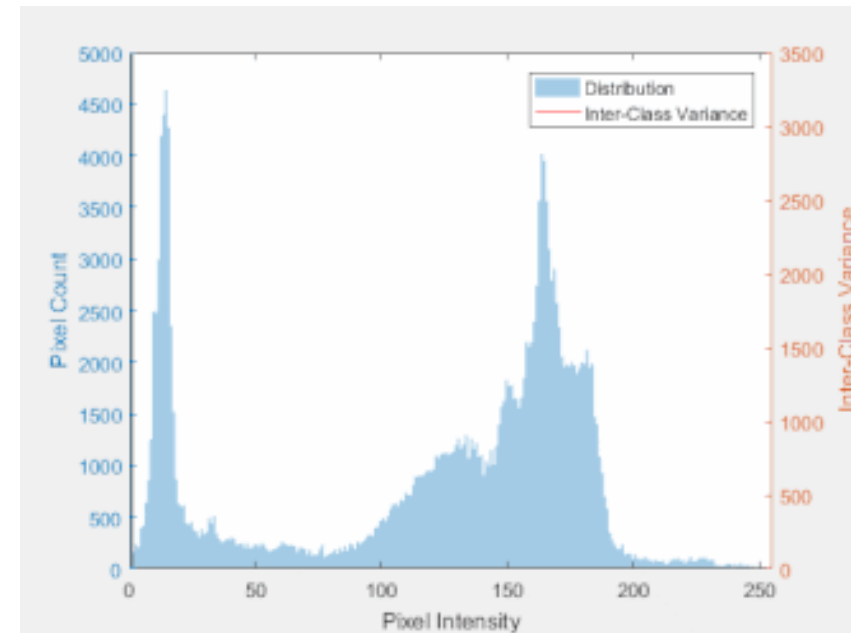


[Variance - Wikipedia](#)

Otsu thresholding

- Definition: The method uses grey-value histogram of the given image I as input and aims at providing the best threshold (foreground/background)
- Otsu's algorithm selects a threshold that maximizes the between-class variance σ_b^2 or minimize within-class variance σ_w^2
- For each threshold t in $[0, 255]$, pixels can be separated into two classes, $C1$ and $C2$; those pixels whose $P_i < t$ are put into $C1$, otherwise into $C2$
- The possibilities of $C1$ and $C2$ separated by t , denoted as $W1$ and $W2$, respectively. For example,

$$W1 = (\text{\#pixels in } C1) / (\text{total pixels count}).$$
- Given H , $W1$, and $W2$, for each t , compute the between-class variance σ_b^2 or within-class variance σ_w^2 ($\sigma_b^2 \rightarrow$ red curve)
- optimal cut t^* corresponds to t whose σ_b^2 is maximum or σ_w^2 is minimum.



Otsu thresholding

- Definition: The method uses grey-value histogram of the given image I as input and aims at providing the best threshold (foreground/background)
- Otsu's algorithm selects a threshold that maximizes the between-class variance σ_b^2 .

Option 1: maximum of:

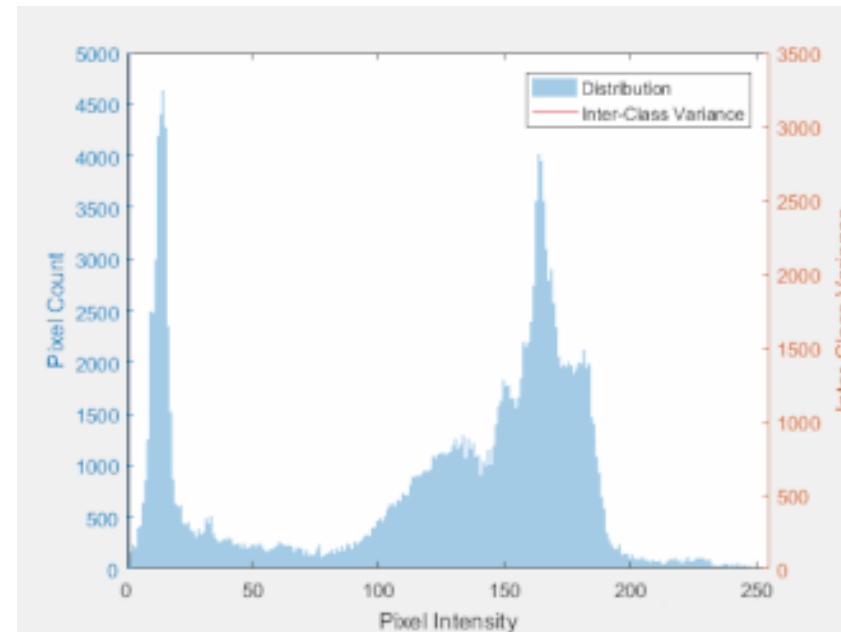
$$\sigma_b^2(t) = w_1(t)w_2(t)[\mu_1(t) - \mu_2(t)]^2$$

$$\mu_1(t) = \sum_{i=1}^t \frac{iP(i)}{w_1(t)}$$

$$w_1(t) = \sum_{i=1}^t P(i)$$

$$\mu_2(t) = \sum_{i=t+1}^I \frac{iP(i)}{w_2(t)}$$

$$w_2(t) = \sum_{i=t+1}^I P(i)$$



Otsu thresholding

- Definition: The method uses grey-value histogram of the given image I as input and aims at providing the best threshold (foreground/background)
- Otsu's algorithm selects a threshold that maximizes the between-class variance σ_b^2 or minimize within-class variance σ_w^2

Option 2: minimum of:

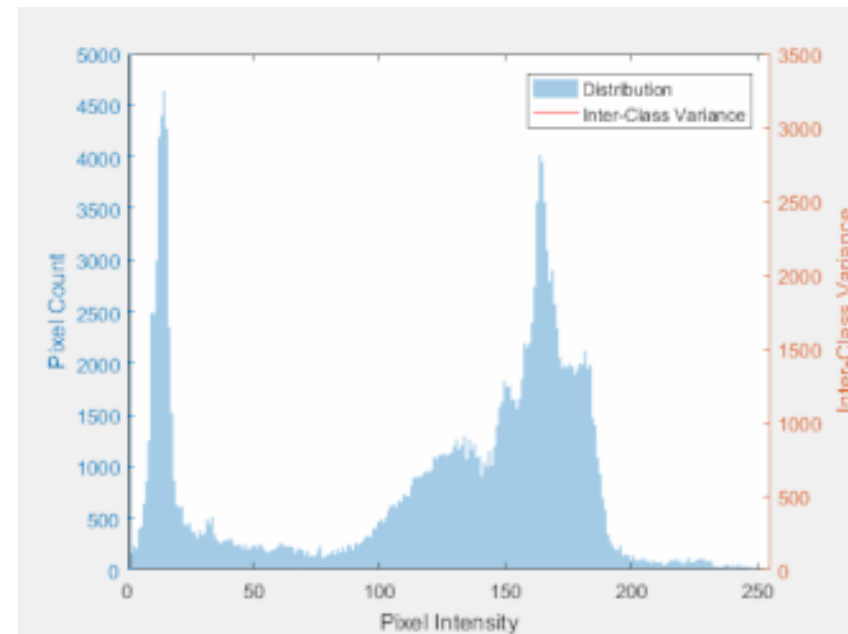
$$\sigma_w^2(t) = w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t)$$

$$w_1(t) = \sum_{i=1}^t P(i) \quad P(i) = \frac{n_i}{n}$$

$$w_2(t) = \sum_{i=t+1}^I P(i)$$

$$\sigma_1^2(t) = \sum_{i=1}^t [i - \mu_1(t)]^2 \frac{P(i)}{w_1(t)}$$

$$\sigma_2^2(t) = \sum_{i=t+1}^I [i - \mu_2(t)]^2 \frac{P(i)}{w_2(t)}$$



Step by step example

- Find Otsu threshold for this image

120	120	21	22
25	26	27	160
180	190	123	145
165	175	23	24

- By minimizing within-class variance

Step by step example

120	120	21	22
25	26	27	160
180	190	123	145
165	175	23	24

Thresholding in $t=100$



120	120		
			160
180	190	123	145
165	175		

Foreground

		21	22
25	26	27	
		23	24

Background

Step by step example

120	120	21	22
25	26	27	160
180	190	123	145
165	175	23	24

$$P_{all} = 16$$

Thresholding in $t=100$



Foreground

120	120		
			160
180	190	123	145
165	175		

$$P_{FG} = 9$$

$$\omega_{fg}(t) = \frac{P_{FG}(t)}{P_{all}} = 9/16 = 0.56$$

Background

		21	22
25	26	27	
		23	24

$$P_{BG} = 7$$

$$\omega_{bg}(t) = \frac{P_{BG}(t)}{P_{all}} = 7/16 = 0.44$$

Step by step example

120	120	21	22
25	26	27	160
180	190	123	145
165	175	23	24

$$P_{all} = 16$$

Thresholding in $t=100$



Foreground

120	120		
			160
180	190	123	145
165	175		

$$P_{FG} = 9$$

$$\omega_{fg}(t) = \frac{P_{FG}(t)}{P_{all}} = 9/16 = 0.56$$

$$\bar{x}_{fg} = \frac{120+120+160+180+190+123+145+165+175}{9} = 153.1$$

Background

		21	22
25	26	27	
		23	24

$$P_{BG} = 7$$

$$\omega_{bg}(t) = \frac{P_{BG}(t)}{P_{all}} = 7/16 = 0.44$$

$$\bar{x}_{bg} = \frac{21+22+25+26+27+23+24}{7} = 24$$

Step by step example

120	120	21	22
25	26	27	160
180	190	123	145
165	175	23	24

$$P_{all} = 16$$

Thresholding in $t=100$



Foreground

120	120		
			160
180	190	123	145
165	175		

$$P_{FG} = 9$$

$$\omega_{fg}(t) = \frac{P_{FG}(t)}{P_{all}} = 9/16 = 0.56$$

$$\bar{x}_{fg} = \frac{120+120+160+180+190+123+145+165+175}{9} = 153.1$$

$$\sigma_{fg}^2(t = 100) = \frac{(120-153.1)^2 + (120-153.1)^2 + \dots + (175-153.1)^2}{9} = 657.43$$

Background

		21	22
25	26	27	
		23	24

$$P_{BG} = 7$$

$$\omega_{bg}(t) = \frac{P_{BG}(t)}{P_{all}} = 7/16 = 0.44$$

$$\bar{x}_{bg} = \frac{21+22+25+26+27+23+24}{7} = 24$$

$$\sigma_{bg}^2(t = 100) = \frac{(21-24)^2 + (22-24)^2 + \dots + (24-24)^2}{7} = 4.0$$

Step by step example

within-class variance

$$w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t)$$

120	120	21	22
25	26	27	160
180	190	123	145
165	175	23	24

$$P_{all} = 16$$

Thresholding in $t=100$



Foreground

120	120		
			160
180	190	123	145
165	175		

$$P_{FG} = 9$$

$$\omega_{fg}(t) = \frac{P_{FG}(t)}{P_{all}} = 9/16 = 0.56$$

$$\bar{x}_{fg} = \frac{120+120+160+180+190+123+145+165+175}{9} = 153.1$$

$$\sigma_{fg}^2(t=100) = \frac{(120-153.1)^2 + (120-153.1)^2 + \dots + (175-153.1)^2}{9} = 657.43$$

Background

		21	22
25	26	27	
		23	24

$$P_{BG} = 7$$

$$\omega_{bg}(t) = \frac{P_{BG}(t)}{P_{all}} = 7/16 = 0.44$$

$$\bar{x}_{bg} = \frac{21+22+25+26+27+23+24}{7} = 24$$

$$\sigma_{bg}^2(t=100) = \frac{(21-24)^2 + (22-24)^2 + \dots + (24-24)^2}{7} = 4.0$$

Step by step example

within-class variance

$$w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t)$$

$$0.44 * 4.0 + 0.56 * 657.43 = 369.9208$$

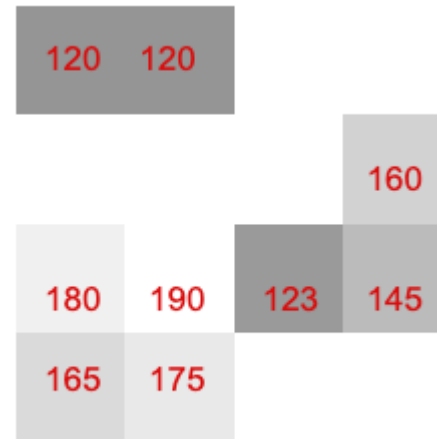


$$P_{all} = 16$$

Thresholding in $t=100$



Foreground



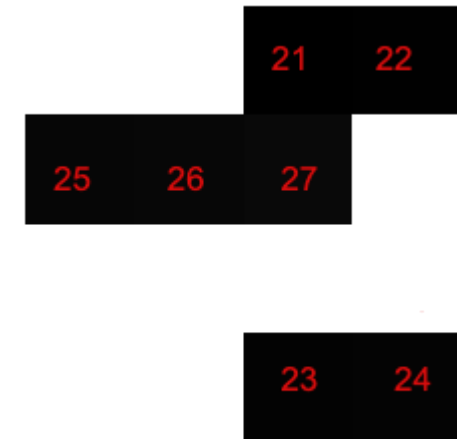
$$P_{FG} = 9$$

$$\omega_{fg}(t) = \frac{P_{FG}(t)}{P_{all}} = 9/16 = 0.56$$

$$\bar{x}_{fg} = \frac{120+120+160+180+190+123+145+165+175}{9} = 153.1$$

$$\sigma_{fg}^2(t = 100) = \frac{(120-153.1)^2 + (120-153.1)^2 + \dots + (175-153.1)^2}{9} = 657.43$$

Background





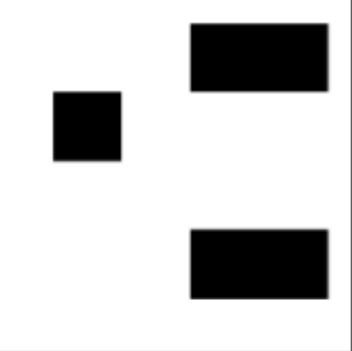
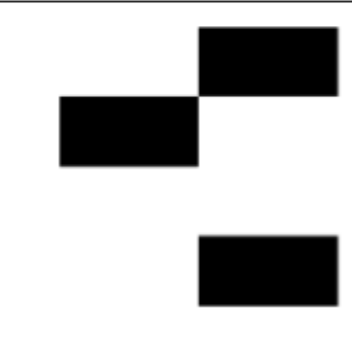
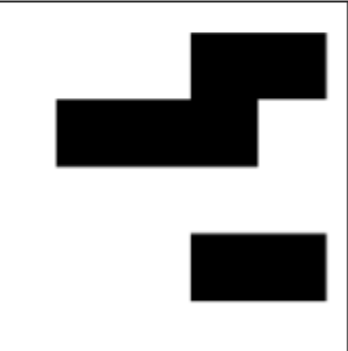

$$P_{BG} = 7$$

$$\omega_{bg}(t) = \frac{P_{BG}(t)}{P_{all}} = 7/16 = 0.44$$

$$\bar{x}_{bg} = \frac{21+22+25+26+27+23+24}{7} = 24$$

$$\sigma_{bg}^2(t = 100) = \frac{(21-24)^2 + (22-24)^2 + \dots + (24-24)^2}{7} = 4.0$$

Step by step (otsu thresholding)

		
$T=22, \sigma^2 = 4092.58$	$T=23, \sigma^2 = 3667.60$	$T=25, \sigma^2 = 2642.35$
		
$T=26, \sigma^2 = 2009.93$	$T=28, \sigma^2 = 371.55$	$T=124, \sigma^2 = 1316.48$

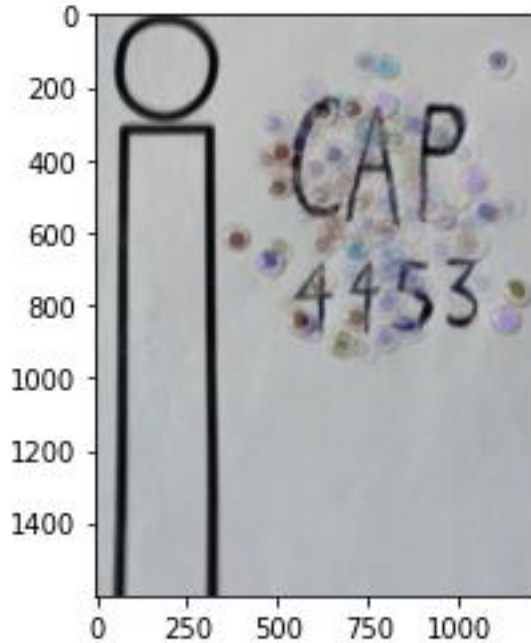
Minimize within-class variance

- The value of variance remains the same from 28 and 120.
- within-class variance is least at $t=28$ or more precisely between 28 to 120.
- Otsu threshold = 28.

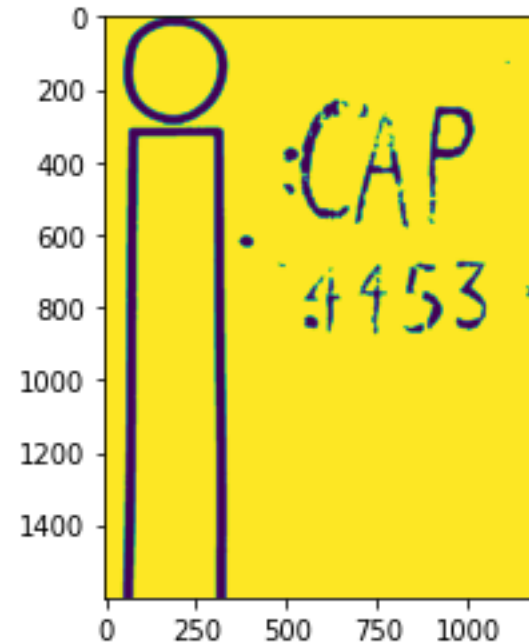
Otsu threshold implementation

```
1 # Set total number of bins in the histogram
2 bins_num = 256
3
4 # Get the image histogram
5 hist, bin_edges = np.histogram(image, bins=bins_num)
6
7 # Get normalized histogram if it is required
8 if is_normalized:
9     hist = np.divide(hist.ravel(), hist.max())
10
11 # Calculate centers of bins
12 bin_mids = (bin_edges[:-1] + bin_edges[1:]) / 2.
13
14 # Iterate over all thresholds (indices) and get the probabilities w1(t), w2(t)
15 weight1 = np.cumsum(hist)
16 weight2 = np.cumsum(hist[::-1])[::-1]
17
18 # Get the class means  $\mu_0(t)$ 
19 mean1 = np.cumsum(hist * bin_mids) / weight1
20 # Get the class means  $\mu_1(t)$ 
21 mean2 = (np.cumsum((hist * bin_mids)[::-1]) / weight2[::-1])[::-1]
22
23 inter_class_variance = weight1[:-1] * weight2[1:] * (mean1[:-1] - mean2[1:]) **
24 2
25 # Maximize the inter_class_variance function val
26 index_of_max_val = np.argmax(inter_class_variance)
27
28 threshold = bin_mids[:-1][index_of_max_val]
29 print("Otsu's algorithm implementation thresholding result: ", threshold)
```

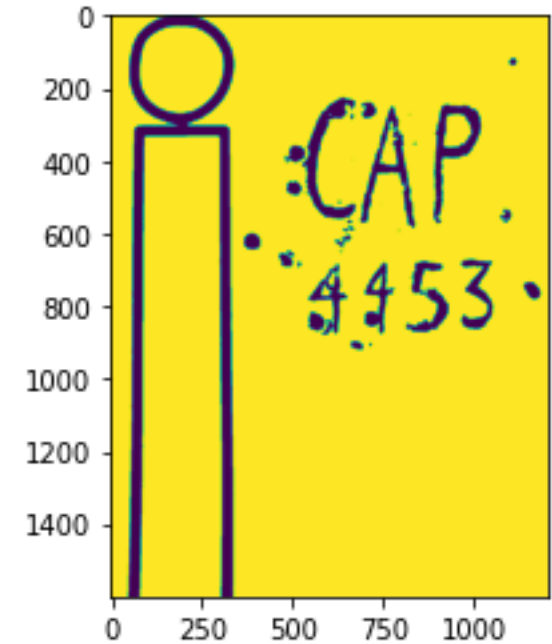
Otsu threshold implementation



Original image



Manually
Th = 90



otsu
Th = 127



Otsu threshold implementation

```
1 # Applying Otsu's method setting the flag value into cv.THRESH_OTSU.  
2 # Use a bimodal image as an input.  
3 # Optimal threshold value is determined automatically.  
4 otsu_threshold, image_result = cv2.threshold(  
5     image, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU,  
6 )  
7 print("Obtained threshold: ", otsu_threshold)
```

```
Obtained threshold: 132.0
```

Otsu thresholding example





The math !

$$\begin{aligned} \text{Var}(X) &= E[(X - E[X])^2] \\ &= E[X^2 - 2XE[X] + E[X]^2] \\ &= E[X^2] - 2E[X]E[X] + E[X]^2 \\ &= E[X^2] - E[X]^2 \end{aligned}$$

$$\sigma_{total}^2 = E[(X - E[X])^2] = E[X_{total}^2] - \mu_{total}^2 \quad (1)$$

$$P_i = \frac{n_i}{n_{total}} \quad \text{When } i \leq t$$

When $t < i < T$

$$P_i^1 = \frac{n_i}{n_1}$$

$$P_i^2 = \frac{n_i}{n_2}$$

$$P_i = \frac{n_1 P_i^1}{n_{total}} = w_1(t) P_i^1$$

$$P_i = \frac{n_2 P_i^2}{n_{total}} = w_2(t) P_i^2$$

$$E[X_{total}^2] = \sum_{i=1}^T P_i x_i^2 = \sum_{i=1}^t P_i x_i^2 + \sum_{i=t+1}^T P_i x_i^2 = w_1(t) \sum_{i=1}^t P_i^1 x_i^2 + w_2(t) \sum_{i=t+1}^T P_i^1 x_i^2 = w_1(t) E[X_1^2] + w_2(t) E[X_2^2] \quad (2)$$

$$\mu_{total}^2 = (w_1(t)\mu_1 + w_2(t)\mu_2)^2 = w_1^2\mu_1^2 + 2w_1w_2\mu_1\mu_2 + w_2^2\mu_2^2 = w_1(1 - w_2)\mu_1^2 + 2w_1w_2\mu_1\mu_2 + w_2(1 - w_1)\mu_2^2 \quad (3)$$



The math !

$$\begin{aligned}\text{Var}(X) &= E[(X - E[X])^2] \\ &= E[X^2 - 2XE[X] + E[X]^2] \\ &= E[X^2] - 2E[X]E[X] + E[X]^2 \\ &= E[X^2] - E[X]^2\end{aligned}$$

$$\sigma_{total}^2 = E[(X - E[X])^2] = E[X_{total}^2] - \mu_{total}^2 \quad (1)$$

$$E[X_{total}^2] = w_1(t)E[X_1^2] + w_2(t)E[X_2^2] \quad (2)$$

$$\mu_{total}^2 = w_1(1 - w_2)\mu_1^2 + 2w_1w_2\mu_1\mu_2 + w_2(1 - w_1)\mu_2^2 \quad (3)$$

$$\sigma_{total}^2 = w_1E[X_1^2] + w_2E[X_2^2] - [w_1\mu_1^2 + w_1w_2\mu_1^2 + 2w_1w_2\mu_1\mu_2 + w_2\mu_2^2 - w_1w_2\mu_2^2]$$

$$\sigma_{total}^2 = w_1E[X_1^2] - w_1\mu_1^2 + w_2E[X_2^2] - w_2\mu_2^2 + [-w_1w_2\mu_1^2 - 2w_1w_2\mu_1\mu_2 + w_1w_2\mu_2^2]$$

$$\sigma_{total}^2 = w_1(t)(E[X_1^2] - \mu_1^2) + w_2(t)(E[X_2^2] - \mu_2^2) + w_1(t)w_2(t)(\mu_2^2 - \mu_1^2) \quad (4)$$

The math !

$$\begin{aligned}
 \text{Var}(X) &= E[(X - E[X])^2] \\
 &= E[X^2 - 2XE[X] + E[X]^2] \\
 &= E[X^2] - 2E[X]E[X] + E[X]^2 \\
 &= E[X^2] - E[X]^2
 \end{aligned}$$

$$\sigma_{total}^2 = E[(X - E[X])^2] = E[X_{total}^2] - \mu_{total}^2 \quad (1)$$

$$\sigma_{total}^2 = w_1(t)(E[X_1^2] - \mu_1^2) + w_2(t)(E[X_2^2] - \mu_2^2) + w_1(t)w_2(t)(\mu_2^2 - \mu_1^2)$$

$$\sigma_{total}^2 = \underbrace{w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t)}_{\text{within-class variance}} + \underbrace{w_1(t)w_2(t)(\mu_2^2(t) - \mu_1^2(t))}_{\text{between-class variance}}$$

fixed

within-class variance

Minimize

between-class variance

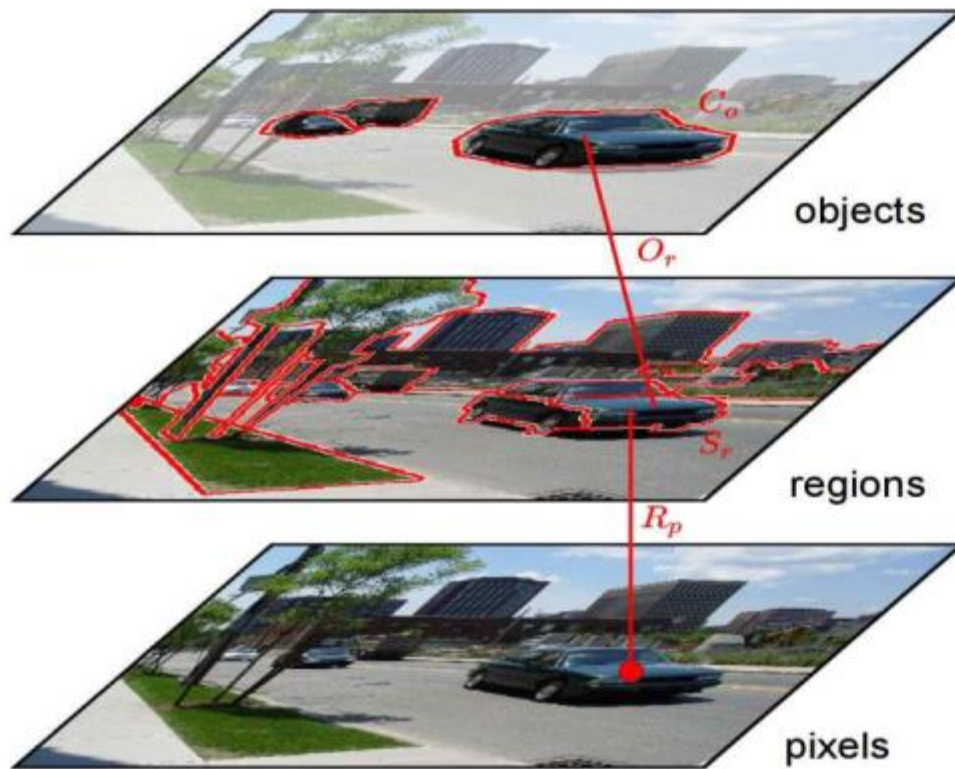
Maximize



Outline

- Image segmentation basics
- Thresholding based
 - Binarization
 - Otsu
- **Region based**
 - **Merging**
 - Splitting
- Clustering based
 - K-means (SLIC)

Region based segmentation



Region:

A group of connected pixels with similar properties

Closed boundaries

Computation of regions is based on similarity

Regions may correspond to Objects in a scene or parts of objects

Spatial proximity + similarity



Region growing

- For segment generation in grey-level or color images, we may start at one seed pixel $(x,y,I(x,y))$ and add recursively adjacent pixels that satisfy a “similarity criterion” with pixels contained in the so-far grown region around the seed pixel.
- Defining similarity criteria alone is not an effective basis for segmentation
- It is necessary to consider the adjacency spatial relationship between pixels



Region growing

- Algorithm

1. The absolute intensity difference between candidate pixel and the seed pixel must lie within a specified range
2. The absolute intensity difference between a candidate pixel and the running average intensity of the growing region must lie within a specified range;
3. The difference between the standard deviation in intensity over a specified local neighborhood of the candidate pixel and that over a local neighborhood of the candidate pixel must (or must not) exceed a certain threshold

Seeded segmentation. Region growing

1. Chose the seed pixel

0	0	5	6	7
1	1	5	8	7
0	<u>1</u>	6	<u>2</u>	7
2	0	7	6	6
0	1	5	6	5

(a)

Seeded segmentation. Region growing

1. Chose the seed pixel
2. Check the neighboring pixels and add them to the region if they are similar to the seed

0	0	5	6	7
1	1	5	8	7
0	<u>1</u>	6	<u>2</u>	7
2	0	7	6	6
0	1	5	6	5

(a)

Seeded segmentation. Region growing

1. Chose the seed pixel
2. Check the neighboring pixels and add them to the region if they are similar to the seed

0	0	5	6	7
1	1	5	8	7
0	<u>1</u>	6	<u>2</u>	7
2	0	7	6	6
0	1	5	6	5

(a)

a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b

(b)

Seeded segmentation. Region growing

1. Chose the seed pixel
2. Check the neighboring pixels and add them to the region if they are similar to the seed
3. Repeat step 2 for each of the newly added pixels; stop if no more pixels can be added

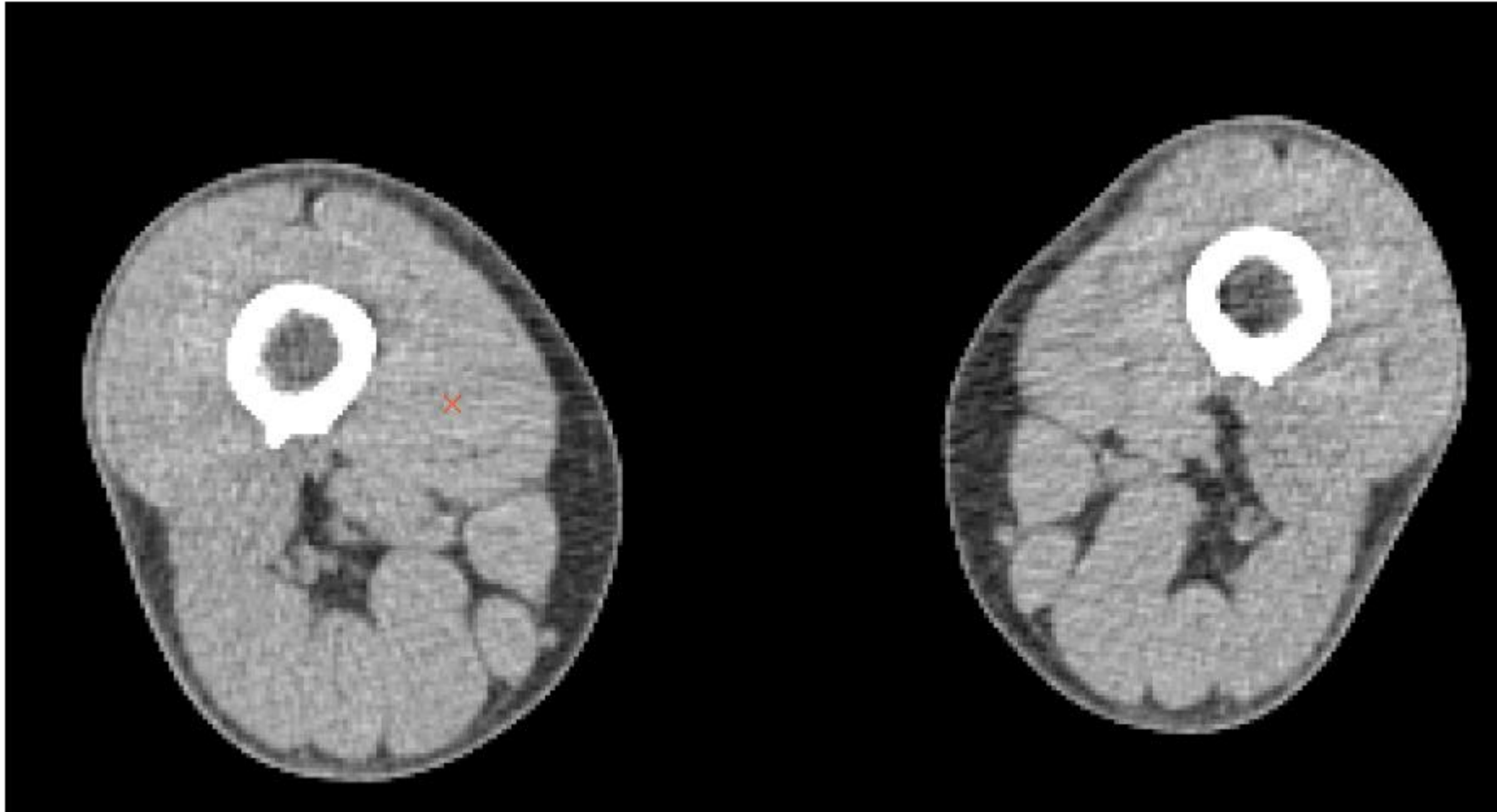
0	0	5	6	7
1	1	5	8	7
0	<u>1</u>	6	<u>2</u>	7
2	0	7	6	6
0	1	5	6	5

(a)

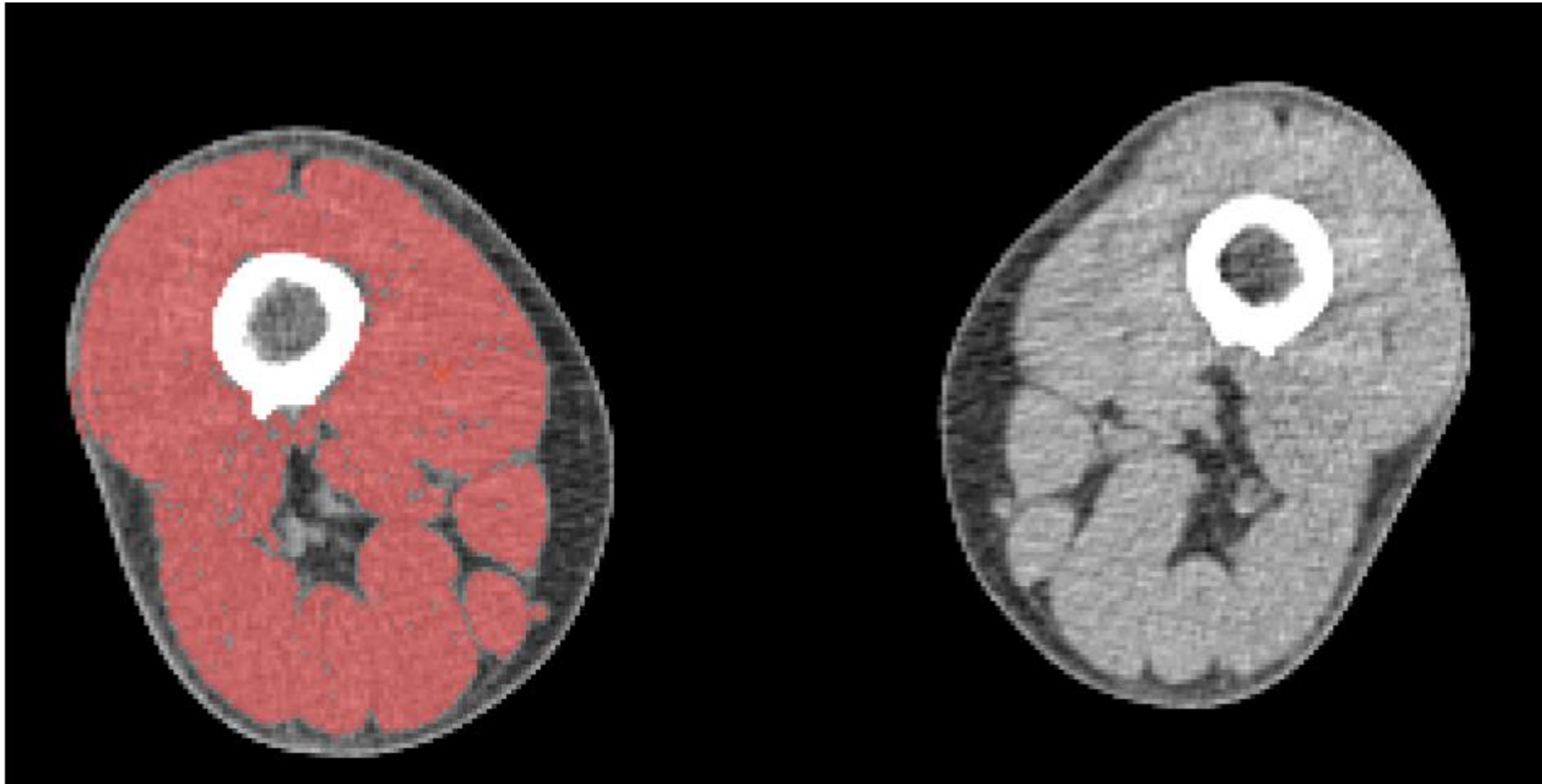
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b

(b)

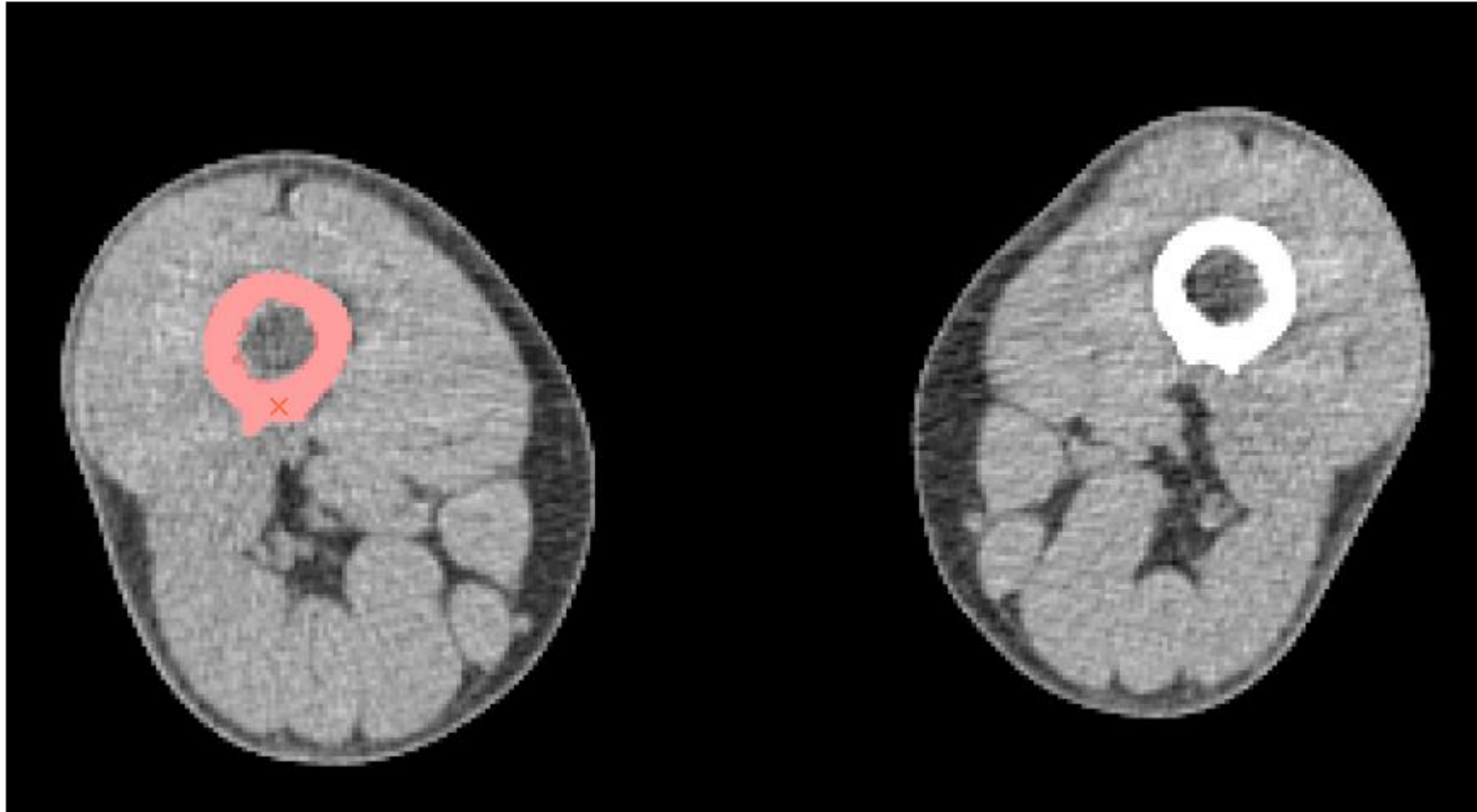
Ex: Muscle/Bone Segmentation in CT Scans



Ex: Muscle/Bone Segmentation in CT Scans



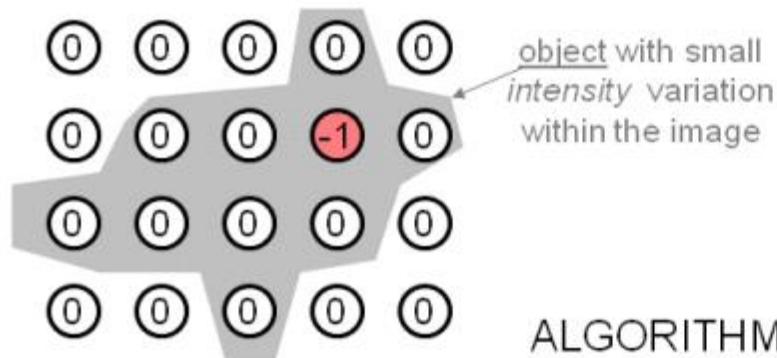
Ex: Muscle/Bone Segmentation in CT Scans



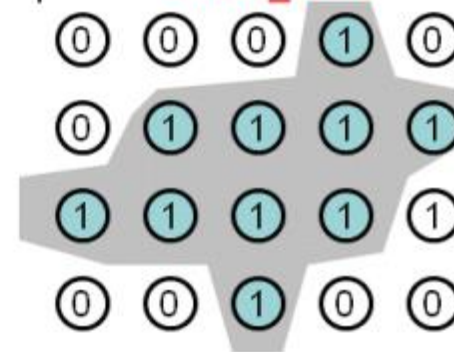
Region Growing Implementation

growRegion: red nodes are the "active_front" (queue or stack)

add seed into **active_front**

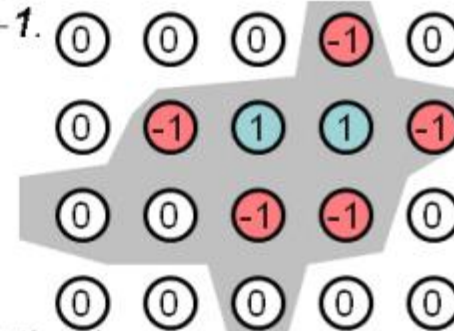
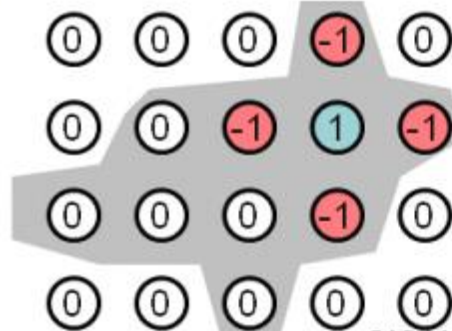


stop when **active_front** is empty



ALGORITHM:

Remove pixel p from **active_front**
and mark it as $region[p] = 1$.
Add all neighbors q such that:
 $region[q] == 0, |I_p - I_q| < T$
and set $region[q] = -1$.





Outline

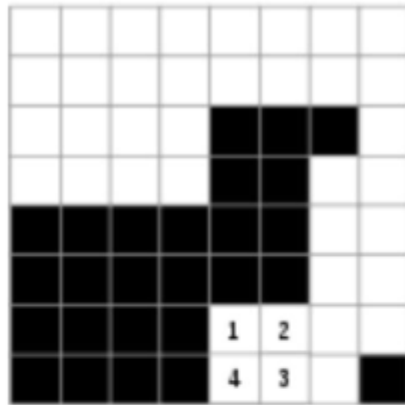
- Image segmentation basics
- Thresholding based
 - Binarization
 - Otsu
- **Region based**
 - Merging
 - **Splitting**
- Clustering based
 - K-means (SLIC)



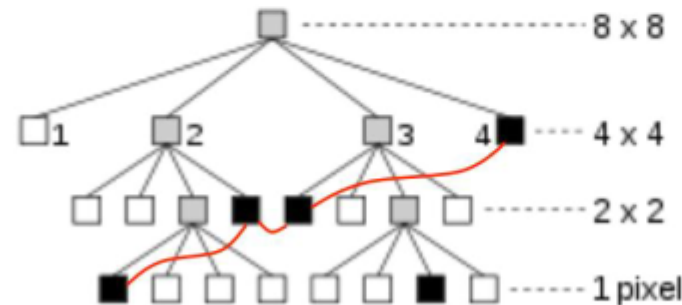
Region splitting and Merging Segmentation

- Region splitting:
 - Unlike region growing, which starts from a set of seed points, region splitting starts with the whole image as a single region and subdivides it into subsidiary regions recursively while a condition of homogeneity is not satisfied.
- Region merging:
 - Region merging is the opposite of splitting, and works as a way of avoiding over-segmentation
 - Start with small regions (2x2 or 4x4 regions) and merge the regions that have similar characteristics (such as gray level, variance).

Region splitting and Merging Segmentation



- RAG: region adjacency graph



- **Quadtree** for splitting (top-down) procedure

RAG with adjacency relations (in red) for big black region.

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 splitting and Merging Segmentation

0	1	0	0	7	7	7	7
1	0	2	2	7	7	7	7
0	2	2	2	7	7	7	7
4	4	2	2	7	7	7	7
0	0	1	1	3	3	7	7
1	1	2	2	3	7	7	7
2	4	3	0	5	7	7	7
2	3	3	5	5	0	7	7

original image

0	1	0	0	7	7	7	7
1	0	2	2	7	7	7	7
0	2	2	2	7	7	7	7
4	4	2	2	7	7	7	7
0	0	1	1	3	3	7	7
1	1	2	2	3	7	7	7
2	4	3	0	5	7	7	7
2	3	3	5	5	0	7	7

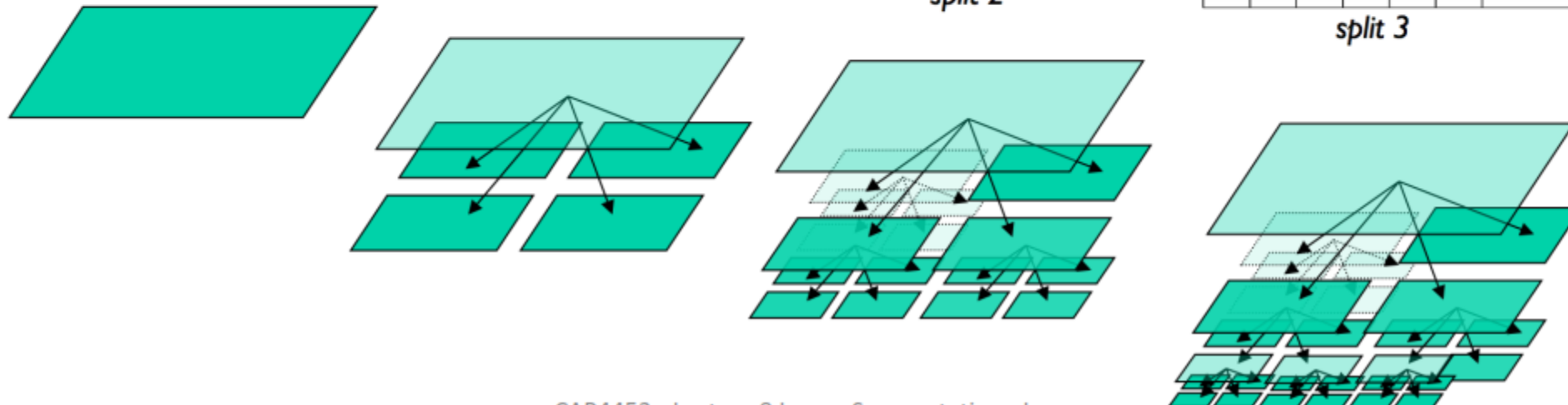
split 1

0	1	0	0	7	7	7	7
1	0	2	2	7	7	7	7
0	2	2	2	7	7	7	7
4	4	2	2	7	7	7	7
0	0	1	1	3	3	7	7
1	1	2	2	3	7	7	7
2	4	3	0	5	7	7	7
2	3	3	5	5	0	7	7

split 2

0	1	0	0	7	7	7	7
1	0	2	2	7	7	7	7
0	2	2	2	7	7	7	7
4	4	2	2	7	7	7	7
0	0	1	1	3	3	7	7
1	1	2	2	3	7	7	7
2	4	3	0	5	7	7	7
2	3	3	5	5	0	7	7

split 3



http://



Region splitting and Merging Segmentation

- **Region splitting:**
 - Unlike region growing, which starts from a set of seed points, region splitting starts with the whole image as a single region and subdivides it into subsidiary regions recursively while a condition of homogeneity is not satisfied.
- **Region merging:**
 - **Region merging is the opposite of splitting, and works as a way of avoiding over-segmentation**
 - **Start with small regions (2x2 or 4x4 regions) and merge the regions that have similar characteristics (such as gray level, variance).**

Depth of RAG – how many levels?

2	2	2	2	1	1	1	1
2	2	2	2	1	1	1	1
2	2	3	3	2	2	2	2
2	2	3	3	3	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2

Depth of RAG – how many levels? 4

2	2	2	2	1	1	1	1
2	2	2	2	1	1	1	1
2	2	3	3	2	2	2	2
2	2	3	3	3	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2
1	1	1	1	2	2	2	2



Questions?