



# Computer Vision



## Task (2)

Name	Section	Bench Number
Amr Mohamed	2	7
Adham Mohamed	1	9
Belal Mohamed	1	23
Mahmoud Emad	2	28
Ezzat Hegazy	1	56
Mina Azer	1	58

**Submitted To:**

**Prof. Dr. Ahmed Badawi**

**TA. Peter Emad**

**Systems and Biomedical Engineering Dept.**

**Cairo University**



# Computer Vision



**Image segmentation** using thresholding is a simple and widely used technique for segmenting images into regions based on their pixel intensity values. The basic idea is to select a threshold value, which separates the pixels in the image into two groups: those with pixel intensities above the threshold and those below it. This creates a binary image where each pixel is either assigned a value of 1 or 0, depending on whether it falls above or below the threshold value.

## 1) Thresholding “binary image”:

Each thresholding method (optimal and otsu), we give the user the option to choose to apply **global or local mode**. Where in the local mode we divide the image to number of parts and apply threshold using optimal or ostu then recombine the image.

**Thresholding**

Optimal

Otsu

Spectral\_Otsu

Thresholding Mode: 

global  
local

 Number Of Slices: 4

**Optimal Thresholding “2 clusters only”:** Here are the general steps for implementing an optimal thresholding technique:

- I. Load the image: Load the image you want to segment into your program or code.
- II. Initiate background array and object array



# Computer Vision

- III. Initially take the 4 corners of the image as background, append to background array and the rest of the image as object
- IV. Calculate the mean for both the background and object gray level
- V. Calculate the thresholding using average of two means.
- VI. Threshold the image make above the threshold as object and under as background.
- VII. Repeat from step 4 “IV” and iterate until the new threshold equal the old threshold.  $T' = T$
- VIII. Post-processing: Perform any post-processing steps, such as noise reduction or morphology operations, to refine the segmentation result.

**Note** that these steps may vary slightly depending on the specific optimal thresholding technique being used.

\*The following images show the differences between global and local thresholding in optimal thresholding

Thresholding Mode: global ▼ Number Of Slices: 4





# Computer Vision



Thresholding Mode: local ▼ Number Of Slices: 4



**Otsu's algorithm "2 clusters only":** is a simple and effective method for thresholding and image segmentation, and it can be used in a wide range of applications. It is particularly useful for images with bi-modal intensity distributions, where there are two distinct groups of pixels with different intensities.

The Otsu's algorithm involves the following steps:

- I. Compute the histogram of the image: A histogram is a graphical representation of the distribution of pixel intensities in an image.
- II. Compute the probabilities of each intensity level: The probability of each intensity level is calculated by dividing the number of pixels with that intensity by the total number of pixels in the image.
- III. Compute the cumulative sums and means: The cumulative sums and means are computed based on the probabilities.
- IV. Compute the inter-class variance for each threshold: The inter-class variance is calculated for each threshold value



# Computer Vision



V. Find the maximum inter-class variance: The threshold value that corresponds to the maximum inter-class variance is selected as the optimal threshold.

VI. Threshold the image: The image is thresholded based on the optimal threshold value.

\*The following image shows the otsu's global segmentation.

Thresholding Mode: global Number Of Slices: 15



\*The following image shows the otsu's local segmentation.

Optimal

Otsu

Spectral\_Otsu

Spectral Prepare For Kmean



Thresholding Mode: local Number Of Slices: 15





# Computer Vision

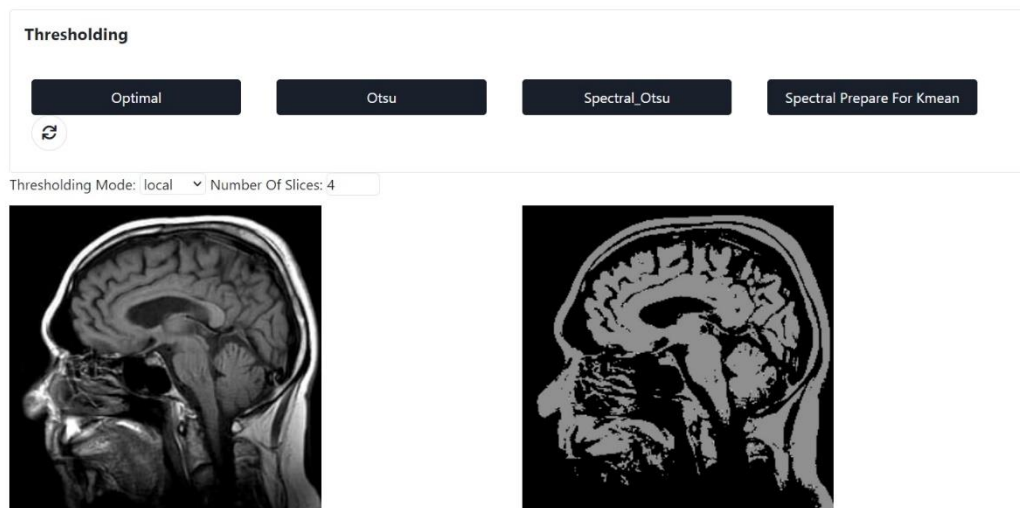


in spectral thresholding their different interpretation of it online where one say it is thresholding with multiple mode “more than 2 clusters” using thechince like modify Otsu and other say that is image segmentation that uses the frequency components and Fourier transform

**Spectral segmentation using Otsu modify method** can be performed using the following the same algorithm of Otsu with some updates:

- I. Compute the histogram of the image: A histogram is a graphical representation of the distribution of pixel intensities in an image.
- II. Compute the probabilities of each intensity level: The probability of each intensity level is calculated by dividing the number of pixels with that intensity by the total number of pixels in the image.
- III. Compute the cumulative sums and means: The cumulative sums and means are computed based on the probabilities.
- IV. Compute multiple thresholding peaks.
- V. Threshold the that it have number of clusters different gray level

\*The following image shows the Spectral thresholding using otsu’s method





# Computer Vision



**Spectral Thresholding using Fourier:** is a technique for image segmentation that uses the frequency components of the image to partition it into multiple regions. The Fourier transform is used to obtain the frequency components of the image, and spectral thresholding is applied on the frequency components to obtain a binary image. Here are the steps for spectral thresholding using Fourier transform for image segmentation:

- I. Load the image and convert it to grayscale.
- II. Compute the two-dimensional Fourier transform of the image using a fast Fourier transform (FFT) algorithm.
- III. Compute the magnitude spectrum of the Fourier transform, which represents the frequency components of the image.
- IV. Apply spectral thresholding on the magnitude spectrum to obtain a binary image. This can be done by choosing a threshold value based on the application requirements or by using an automatic thresholding algorithm, such as Otsu's method.
- V. Inverse Fourier transform the thresholded magnitude spectrum to obtain the segmented image.
- VI. Post-process the binary image to refine the segmentation result. This can involve applying morphological operations, such as erosion or dilation, to remove small objects or fill in gaps between segmented regions.



# Computer Vision



\*The following image shows the Spectral thresholding using Fourier transform.

**Thresholding**

Optimal

Otsu

Spectral\_Otsu

Spectral Prepare For Kmean

Thresholding Mode: local ▼ Number Of Slices: 4



## 2) Unsupervised segmentation:

**K-means Clustering:** a widely used unsupervised machine learning algorithm for data clustering and segmentation. In the context of image processing, K-means clustering can be used for unsupervised image segmentation, where pixels in the image are grouped into clusters based on their similarity in color or intensity.

Here are the steps for implementing K-means clustering for image segmentation:

- I. Load the image and reshape it to a two-dimensional array, where each row represents a pixel and each column represents a color channel or intensity.
- II. Choose the number of clusters K, which represents the number of segments in the image. This can be done manually or using an automatic method, such as the elbow method or silhouette score.
- III. Initialize K cluster centroids randomly.





# Computer Vision



- IV. Assign each pixel to the nearest cluster centroid based on its distance in color or intensity space.
- V. Update the cluster centroids to be the mean color or intensity of the pixels assigned to each cluster.
- VI. Repeat steps 4 and 5 until convergence or a maximum number of iterations is reached.
- VII. Reshape the clustered pixel values back into the shape of the original image.
- VIII. Post-process the segmented image, if necessary, using morphological operations or other image processing techniques.



# Computer Vision

## K Mean segmentation in RGB

Segmentation

K-Mean	Region Growing	Agglomerative	Mean Shift
K: <input type="text" value="7"/>	Seed-X: <input type="text" value="72"/> Seed-Y: <input type="text" value="90"/>	Number Of Clusters: <input type="text" value="2"/>	Raduis: <input type="text" value="80"/>
Max Iteration: <input type="text" value="50"/>	Threshold: <input type="text" value="1"/>		Threshold: <input type="text" value="10"/> %

Color Space: RGB▼





Figure: Screenshot from the website



Figure: K-mean output for k=7 in RGB

## K Mean segmentation in LUV



# Computer Vision



## Segmentation

K-Mean

K:

Max Iteration:

Region Growing

Seed-X:  Seed-Y:

Threshold:

Agglomerative

Number Of Clusters:

Mean Shift

Radius:

Threshold:  %

Color Space: LUV

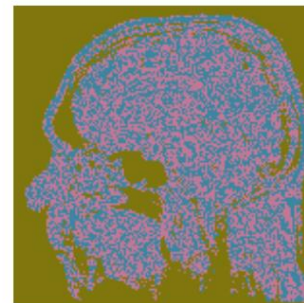
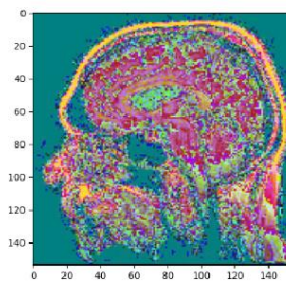


Figure Screenshot from the website

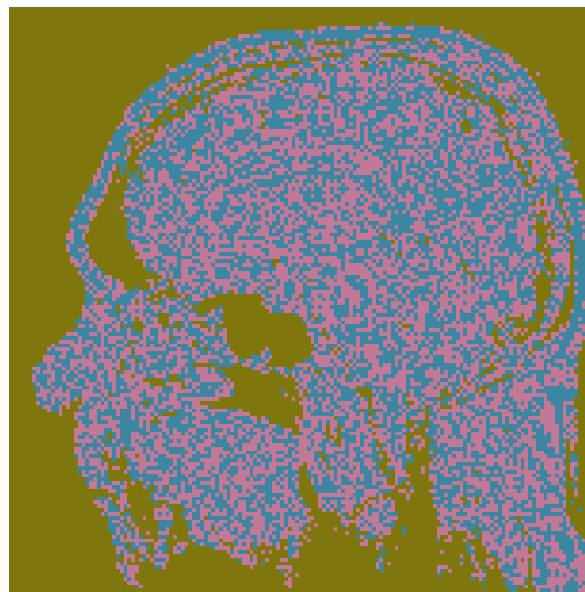


Figure 1 K-mean output for  $k=3$  in LUV



# Computer Vision



**Region growing** is a method of image segmentation that groups together neighboring pixels or regions that have similar properties, such as intensity or color. The algorithm for region growing can be summarized in the following steps:

- I. Choose a starting pixel or region as a seed.
- II. Define a similarity criterion based on the pixel intensity, color, texture, or other features.
- III. Add the neighboring pixels or regions that satisfy the similarity criterion to the current region.
- IV. Repeat step 3 for all the newly added pixels or regions until the region stops growing or reaches a predefined size or shape.
- V. Repeat steps 1-4 for all unassigned pixels or regions until all pixels or regions are assigned to a region.

Segmentation

K-Mean	Region Growing	Agglomerative	Mean Shift
K: <input type="text" value="7"/>	Seed-X: <input type="text" value="140"/> Seed-Y: <input type="text" value="90"/>	Number Of Clusters: <input type="text" value="2"/>	Radius: <input type="text" value="80"/>
Max Iteration: <input type="text" value="50"/>	Threshold: <input type="text" value="10"/>		Threshold: <input type="text" value="10"/> %

Color Space: RGB

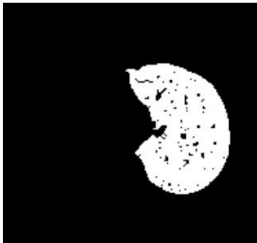
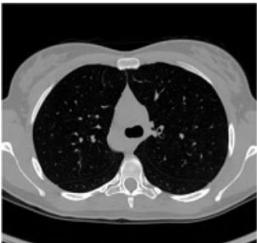
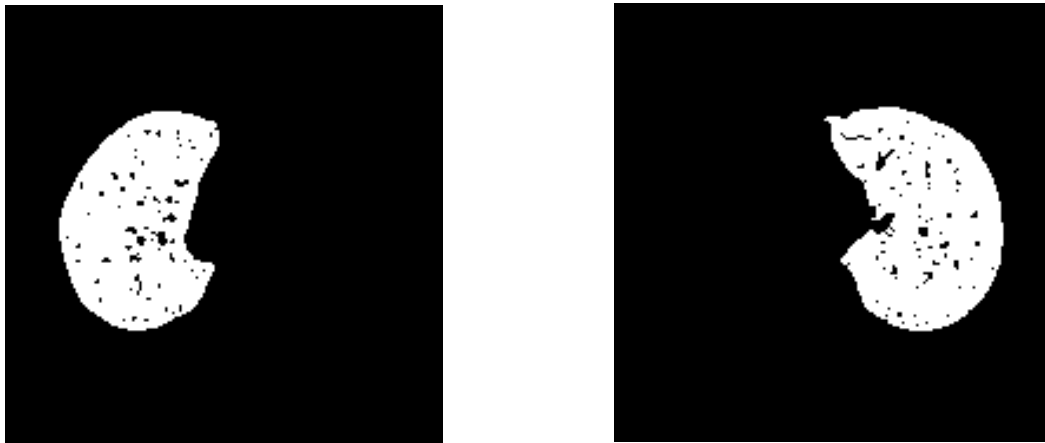


Figure: Screenshot from website



# Computer Vision



*Figure: Region Growing for different seeds*

**Agglomerative clustering:** is a bottom-up approach to clustering that starts with each data point as a separate cluster and then iteratively merges the closest pairs of clusters until a stopping criterion is met. The algorithm for agglomerative clustering can be summarized in the following steps:

- I. Assign each data point to a separate cluster.
- II. Compute the pairwise distance or similarity matrix between all clusters.
- III. Find the closest pair of clusters based on a distance or similarity metric.
- IV. Merge the two closest clusters into a single new cluster.
- V. Update the distance or similarity matrix to reflect the new cluster.
- VI. Repeat steps 3-5 until a stopping criterion is met, such as a fixed number of clusters or a distance threshold.





# Computer Vision



Segmentation

<p><b>K-Mean</b></p> <p>K: <input type="text" value="7"/></p> <p>Max Iteration: <input type="text" value="7"/></p>	<p><b>Region Growing</b></p> <p>Seed-X: <input type="text" value="72"/> Seed-Y: <input type="text" value="90"/></p> <p>Threshold: <input type="text" value="1"/></p>	<p><b>Agglomerative</b></p> <p>Number Of Clusters: <input type="text" value="100"/></p>	<p><b>Mean Shift</b></p> <p>Raduis: <input type="text" value="80"/></p> <p>Threshold: <input type="text" value="10"/> %</p>
--	--	---	---

Color Space: RGB▼



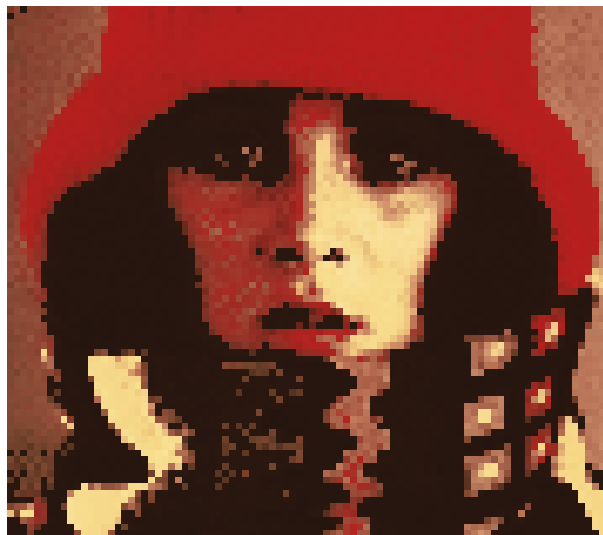
**Mean Shift:** is a non-parametric clustering algorithm that can also be used for image segmentation. The algorithm is based on the idea of shifting each data point towards the direction of the local mean in its neighborhood until convergence. Here is the algorithm for Mean Shift segmentation:

- I. Choose a window size and a similarity metric.
- II. For each pixel in the image, define a window around it.
- III. Compute the similarity between each pair of pixels within the window using the chosen similarity metric.
- IV. Compute the mean shift vector for the pixel by averaging the differences between the pixel and its neighbors weighted by their similarity scores.



# Computer Vision

- V. Shift the pixel towards the direction of the mean shift vector.
- VI. Repeat steps 3-5 until the pixel converges to a local maximum.
- VII. Assign each pixel to the nearest mode of the converged regions.
- VIII. Merge regions with similar modes.



*Mean Shift Output*