**Exercises**

# **Exercise 1:** Image Blending

## **Objective:** Learn how to use algebra operations to blend to images

**Task:** Write a Python function that takes two images and an alpha value as inputs and returns a blending image. The alpha value must be between 0 and 1 indicating the proportion of content keeped between image 1 and 2.

**Input:** Two images and an alpha value. img\_man.png and img\_lion.png

**Output:** A blended image



| import numpy as np import cv2  def image\_blending(image1, image2, alpha):  # Computed the image blending  blended\_image =   return blended\_image  # Test with a sample image  ‘’’   1. read image 1 and image 2 using opencv 2. Check image sizes 3. If the image sizes are different, resize image two equal size 4. Apply the image\_blending function 5. Show images. You can use opencv or matplotlib   ’’’ |
| --- |

# **Exercise 2:** Basic Thresholding

## **Objective:** Learn how to apply a basic threshold to a grayscale image.

**Task:** Write a Python function that takes a grayscale image and a threshold value as inputs and returns a binary image. The pixel value should be 1 if the original pixel value is greater than or equal to the threshold; otherwise, it should be 0.

**Input:** A grayscale image and a threshold value (e.g., 128).

**Output:** A binary image.

| import numpy as np import cv2  def basic\_thresholding(image, threshold):  # Apply the threshold  binary\_image =   return binary\_image  # Test with a sample image  ‘’’   1. read image using opencv 2. Convert image to grayscale 3. Select a threshold value # (e.g., 128) or plot the image histogram to determine the right value. 4. Apply the image thresholding function 5. Show original image and the Binary image. You can use opencv or matplotlib   ’’’ |
| --- |

# **Exercise 3:** Otsu's Thresholding

## **Objective:** Learn how to apply Otsu's thresholding to find the optimal threshold value automatically.

**Task:** Write a Python function that takes a grayscale image and applies Otsu's thresholding.

**Input:** A grayscale image.

**Output:** A binary image using Otsu's thresholding.

| import numpy as np import cv2  def otsu\_thresholding(image):  # Apply the threshold  binary\_image =   return binary\_image  # Test with a sample image  ‘’’   1. read image using opencv 2. Convert image to grayscale 3. Apply the image thresholding function 4. Show original image and the Binary image. You can use opencv or matplotlib   ’’’ |
| --- |

# **Exercise 4:** Comparison of Thresholding Techniques

## **Objective:** Compare different thresholding techniques on the same image.

**Task:** Write a Python script that applies basic thresholding, and Otsu's thresholding to the same image and displays the results side by side for comparison.

**Input:** A grayscale image.

**Output:** A display of the original image alongside the results of the different thresholding techniques.

| import numpy as np import cv2  def compare\_thresholding(image, threshold):  # Basic Thresholding  binary\_basic = ….  # Otsu's Thresholding  binary\_otsu = …    # Stack all images together for comparison  # Hint: use np.hstack  combined\_image = …    return combined\_image  # Test with a sample image  ‘’’   1. read image using opencv 2. Convert image to grayscale 3. Apply the image compare\_thresholding function 4. Show combined images   ’’’ |
| --- |
|  |

# **Exercise 5:** Multilevel Thresholding

## **Objective:** Compare different levels of thresholds using Otsu's Thresholding

**Task:** Write a Python script that applies Otsu's thresholding algorithm change the number of levels threshold

**Input:** The cameraman.jpg image

**Output:** A display of the original image alongside the segmented image applying different thresholding levels



| import numpy as np import cv2  def otsu\_thresholding(image, levels):  # Otsu's Thresholding  binary\_otsu = …    return binary\_otsu  # Test with a sample image  ‘’’   1. read image using opencv 2. Convert image to grayscale 3. Apply the image otsu\_thresholding function using 1, 2, 3 .. levels 4. Show segmented images   ’’’ |
| --- |
|  |

# **Exercise 6:** Change Detection

## **Objective:** Detect and visualize differences between two images

**Task:** Write a Python script that applies change detection techniques to determine the regions with changes.

**Input:** Two colored images: ImageL.jpg and ImageR.jpg.

**Output:** A display of the original images alongside the results areas presenting differences among images

# 

| import numpy as np import cv2  def change\_detection(image1, image2, method):  if method==”diff”:  # Basic Thresholding  binary\_basic = ….  elif method==”diff”:  # Otsu's Thresholding  binary\_otsu = …    # Stack all images together for comparison  # Hint: use np.hstack  combined\_image = …  return combined\_image  # Test with a sample image  ‘’’   1. read image using opencv 2. Convert image to grayscale 3. Apply the image change\_detection function 4. Show combined images   ’’’ |
| --- |
|  |
|  |

# **Exercise 7:** Segmentation using HSV Color Space

## **Objective**: Learn how to segment an image based on color in the HSV color space.

**Task:** Write a Python function that segments an image by a specified color range in the HSV color space.

**Input:** The vegetable image and a color range in HSV.

**Output:** A binary mask where the specified color range is segmented.



| import numpy as np import cv2  def segment\_hsv(image, lower\_hsv, upper\_hsv):  return segmented\_image, mask  # Test with a sample image  ‘’’   1. read image using opencv 2. Convert image to hsv space 3. Select the lower and upper color ranges 4. Apply the image segment\_hsv function 5. Show combined images   ’’’ |
| --- |
|  |
|  |