**EC7212 – Computer Vision and Image Processing**

**Take Home Assignment 2**

Index No: EG/202/3886

Name: De Zoysa R.N.C.

Github Repo: <https://github.com/NisalDeZoysa/EC7212-CVIP-Assignment-02.git>

1. Consider an image with 2 objects and a total of 3-pixel values (1 for each object and one for the background). Add Gaussian noise to the image. Implement and test Otsu’s algorithm with this image

import numpy as np

import cv2

import matplotlib.pyplot as plt

image = cv2.imread("test\_shapes\_complex.png", 0)

if image is None:

    print("Image not found. Check filename.")

    exit()

def add\_gaussian\_noise(img, mean=0, sigma=25):

    noise = np.random.normal(mean, sigma, img.shape)

    noisy\_img = img + noise

    noisy\_img = np.clip(noisy\_img, 0, 255)  # Keep in [0, 255]

    return noisy\_img.astype(np.uint8)

noisy\_image = add\_gaussian\_noise(image)

def otsu\_threshold(img):

    if img.dtype != np.uint8:

        img = img.astype(np.uint8)

    \_, otsu\_img = cv2.threshold(img, 0, 255, cv2.THRESH\_BINARY + cv2.THRESH\_OTSU)

    return otsu\_img

otsu\_result = otsu\_threshold(noisy\_image)

plt.figure(figsize=(12, 4))

plt.subplot(1, 3, 1)

plt.title("Original Synthetic Image")

plt.imshow(image, cmap='gray')

plt.axis('off')

plt.subplot(1, 3, 2)

plt.title("Noisy Image")

plt.imshow(noisy\_image, cmap='gray')

plt.axis('off')

plt.subplot(1, 3, 3)

plt.title("Otsu Thresholded")

plt.imshow(otsu\_result, cmap='gray')

plt.axis('off')

plt.tight\_layout()

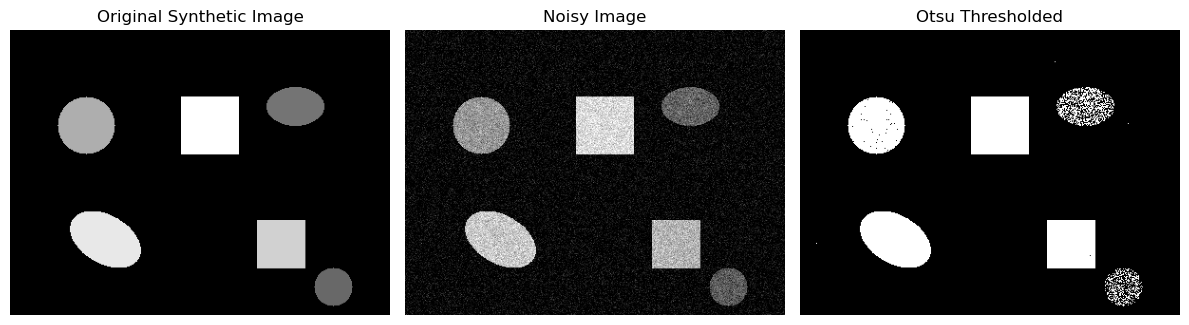
plt.show()

cv2.imwrite("otsu\_noisy\_input.png", noisy\_image)

cv2.imwrite("otsu\_result\_output.png", otsu\_result)

print("Images saved: 'otsu\_noisy\_input.png' and 'otsu\_result\_output.png'")

**Answer :**

****

2. Implement a region-growing technique for image segmentation. The basic idea is to start from a set of points inside the object of interest (foreground), denoted as seeds, and recursively add neighboring pixels as long as they are in a pre-defined range of the pixel values of the seeds.

import cv2

import numpy as np

import matplotlib.pyplot as plt

# ---------------------------

# Function: Region Growing

# ---------------------------

def region\_growing(image, seed\_points, threshold=15):

    rows, cols = image.shape

    visited = np.zeros\_like(image, dtype=bool)

    result = np.zeros\_like(image, dtype=np.uint8)

    for seed in seed\_points:

        x, y = seed

        region\_value = image[x, y]

        print(f"Seed at ({x},{y}) - Intensity: {region\_value}")

        stack = [(x, y)]

        visited[x, y] = True

        while stack:

            i, j = stack.pop()

            result[i, j] = 255

            for dx, dy in [(-1, 0), (1, 0), (0, -1), (0, 1)]:

                ni, nj = i + dx, j + dy

                if 0 <= ni < rows and 0 <= nj < cols and not visited[ni, nj]:

                    if abs(int(image[ni, nj]) - int(region\_value)) <= threshold:

                        visited[ni, nj] = True

                        stack.append((ni, nj))

    return result

# ---------------------------

# Load Image (Grayscale)

# ---------------------------

image = cv2.imread('test\_shapes\_complex.png', 0)  # <-- Replace with your filename if different

if image is None:

    print("Image not found. Check the path!")

    exit()

# ---------------------------

# Seed point & Threshold

# ---------------------------

seed\_points = [(120, 130)]  # You can change this manually

threshold = 25  # Try 20–40 depending on contrast

# ---------------------------

# Run Region Growing

# ---------------------------

segmented = region\_growing(image, seed\_points, threshold=threshold)

# ---------------------------

# Display Original + Result

# ---------------------------

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

plt.title("Original Image")

plt.imshow(image, cmap='gray')

plt.axis('off')

plt.subplot(1, 2, 2)

plt.title("Region-Growing Output")

plt.imshow(segmented, cmap='gray')

plt.axis('off')

plt.tight\_layout()

plt.show()

# ---------------------------

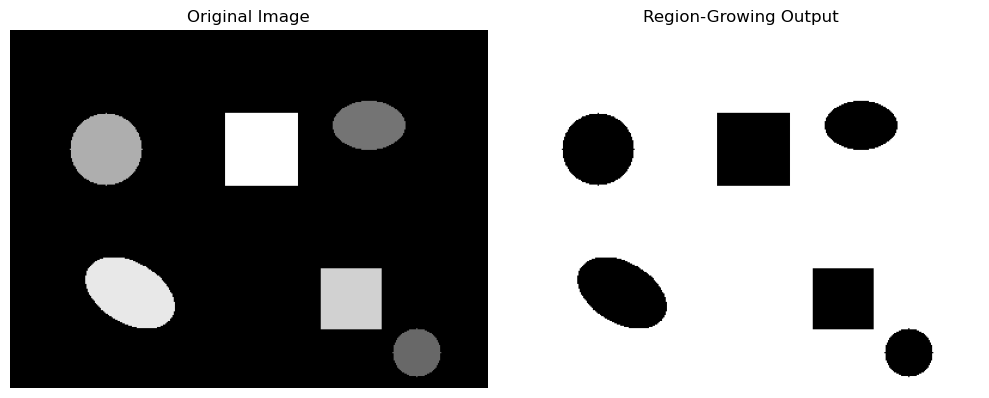
# Save the output

# ---------------------------

cv2.imwrite("region\_growing\_result.png", segmented)

print("Result saved as 'region\_growing\_result.png'")

**Answer :**

****