LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.03**

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No.: C026 | Name: Anirbaan Ghatak |
| Class : B | Batch : B1 |
| Date of Experiment: | Date of Submission |
| Grade : |  |

**B.1 Software Code written by student:**

# Aim: Implementation of Point Processing image enhancement Operations in Spatial Domain.

# Name: Anirbaan Ghatak

# Roll No.: C026

import cv2

import numpy as np

def show\_image(image, title='image'):

    cv2.imshow(title, image)

    cv2.waitKey(0)

    cv2.destroyAllWindows()

image\_path = 'IMG\_2458.jpg'

image = cv2.imread(image\_path)

image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

image = cv2.resize(image, (500, 500))

show\_image(image)

# Find the maximum gray level pixel and its location

max\_ins = 0

max\_loc = (0, 0)

height, width = image.shape[:2]

for r in range(height):

    for c in range(width):

        ins = image[r, c]

        if ins > max\_ins:

            max\_ins = ins

            max\_loc = (r, c)

print(f"Max gray level pixel loc: {max\_loc}")

# Create and display the negative image

neg = max\_ins - image

show\_image(neg, 'Negative image')

# Apply thresholding to create a binary image

threshold = 127.5

thresholded\_image = (image > threshold) \* 255

show\_image(thresholded\_image, "Thresholded Image")

# Apply contrast stretching

img2 = image.copy()

s1, s2 = 63.75, 127.5

r1, r2 = np.min(image) + 1, np.max(image)

l = max\_ins

alpha = s1 / r1

beta = (s2 - s1) / (r2 - r1)

gamma = (l - 1 - s2) / (l - 1 - r2)

for r in range(height):

    for c in range(width):

        if img2[r, c] <= r1:

            img2[r, c] = alpha \* img2[r, c]

        elif img2[r, c] <= r2 and img2[r, c] > r1:

            img2[r, c] = beta \* (img2[r, c] - r1) + s1

        else:

            img2[r, c] = gamma \* (img2[r, c] - r2) + s2

# Display the original image and the contrast stretched image

show\_image(image)

show\_image(img2)

# Gray level slicing without and with background

min\_t = 63.75

max\_t = 127.5

highlight\_value = 255

# Create a mask for the highlighted region

mask = np.logical\_and(image >= min\_t, image <= max\_t)

# Create copies of the original image

wb\_without\_bg = image.copy()

wb\_with\_bg = image.copy()

# Apply the highlight value to the pixels within the mask (without background)

wb\_without\_bg[mask] = highlight\_value

show\_image(wb\_without\_bg, 'Without Background')

# Apply a different value to the background pixels (with background)

wb\_with\_bg[~mask] = 50

show\_image(wb\_with\_bg, 'With Background')

# Bit plane slicing and display

for bp in range(8):

    bit\_plane = np.bitwise\_and(image, 2 \*\* bp)

    nbp = (bit\_plane \* 255).astype(np.uint8)

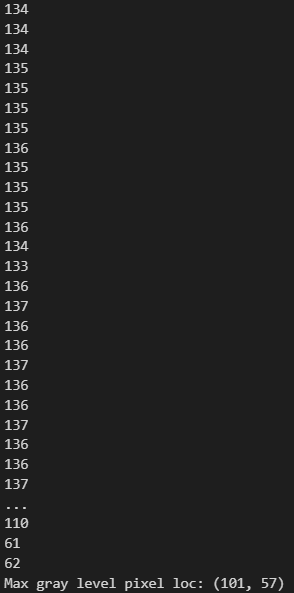
    show\_image(nbp, f"Bit Plane {bp + 1}")

**B.2 Input and Output:**

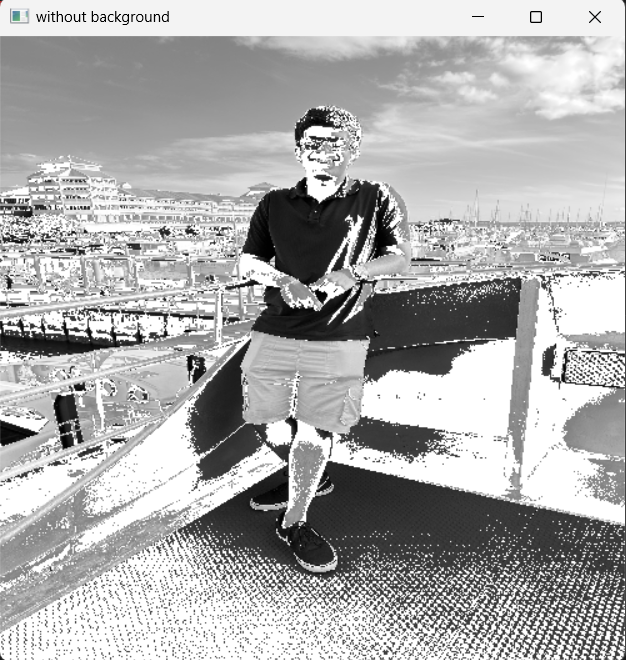
**Input Images: Your photographs**

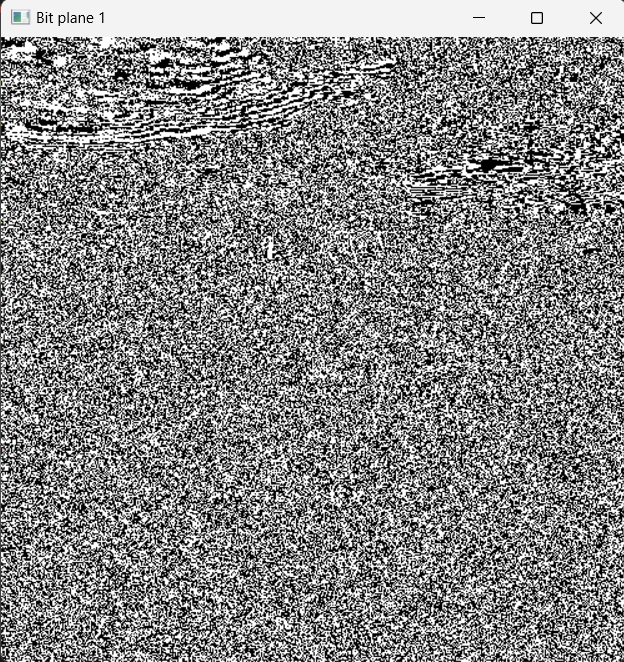
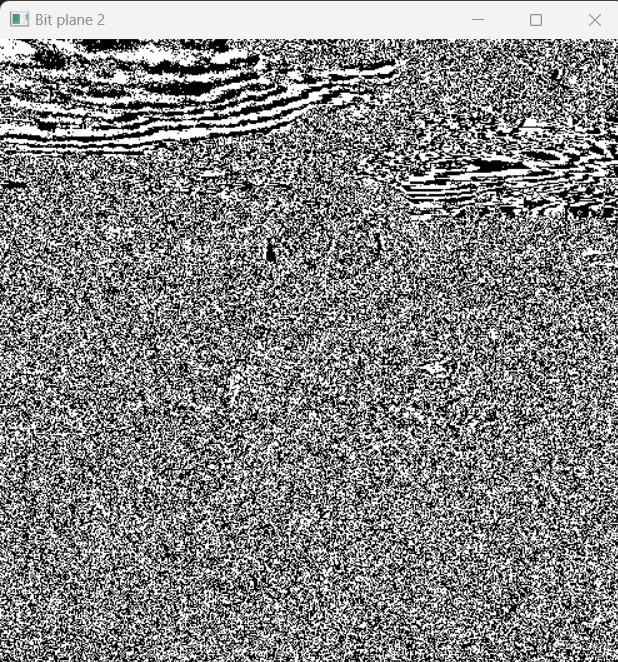
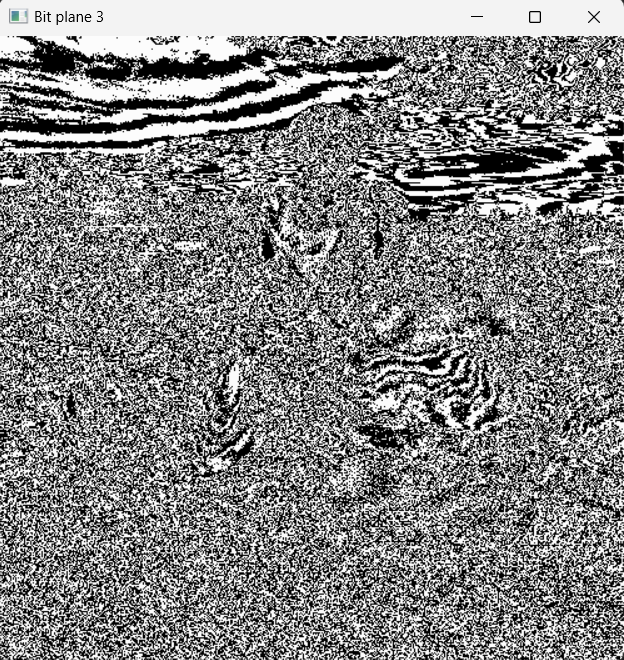
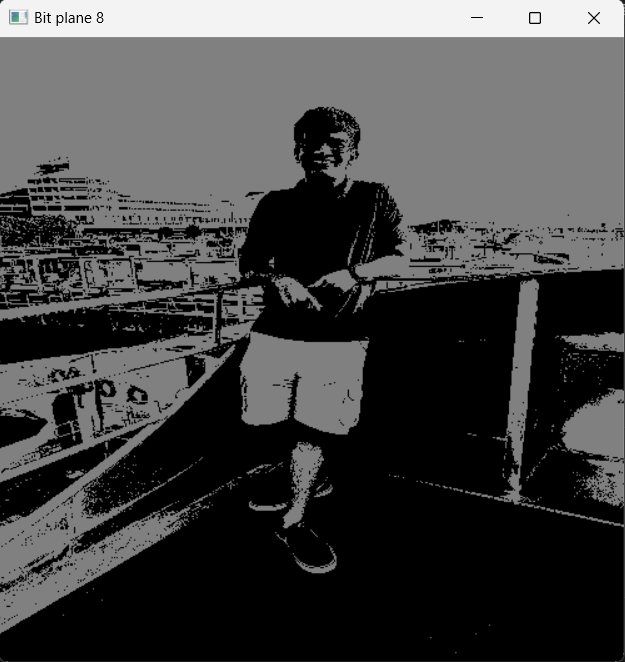
****

**Output:**

****

**** ****

**** ****

**** ****      

**B.3 Observations and learning:**

*Point processing techniques show the impact on image enhancement based on pixel intensity. Negative transformation, thresholding, and contrast stretching improve image quality, while gray level and bit plane slicing provide distinct ways to emphasize regions and analyze binary contributions. Overall, the significance of pixel-level operations, enhancing image analysis capabilities for diverse applications and increasing the understanding of fundamental concepts. Point processing techniques important for image enhancement and information extraction.*

**B.4 Conclusion:**

*Power of point processing techniques in enhancing images, highlighting specific intensities, and revealing hidden information. The methods showcased the utility of pixel-level manipulations for various real-world applications, from image enhancement to data extraction. These techniques help us to manipulate and visualize images based on pixel values.*

**B.5 Question of Curiosity**

***(To be answered by student based on the practical performed and learning/observations)***

Q1: List out real life applications of all point processing techniques performed.

Real Life applications of point processing techniques are

Image Enhancement

Image Compression

Highlight Objects in Medical Imaging

Object Segmentation in Medical Imaging

Brightness correcting to beautify images

Etc.

Q2: How relevant these techniques are for diagnosis procedures followed in hospitals?

Point processing techniques are crucial in hospitals for diagnosing medical conditions. They enhance image quality, reducing noise and highlighting important details in X-rays, MRI, and CT scans. These techniques aid in segmenting organs, detecting abnormalities, and aligning images for tracking disease progression. Visualization tools like bit plane slicing assist in complex data interpretation. By improving image clarity, standardizing intensities, and aiding research and education, point processing techniques empower medical professionals to make accurate diagnoses and informed decisions for patient care.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*