**write a python program to simulate smoothening and sharpening operation on image using spatial filter**

import cv2

import numpy as np

from google.colab.patches import cv2\_imshow

# Load the image

image = cv2.imread('/content/smoothening data.png')

# Define the Gaussian kernel for smoothing

kernel\_size = (5, 5)

sigma = 1.5

gaussian\_kernel = cv2.getGaussianKernel(kernel\_size[0], sigma)

gaussian\_kernel = np.outer(gaussian\_kernel, gaussian\_kernel)

# Apply Gaussian smoothing

smoothed\_image = cv2.filter2D(image, -1, gaussian\_kernel)

# Define a sharpening kernel

sharpening\_kernel = np.array([[-1, -1, -1],

                              [-1, 9, -1],

                              [-1, -1, -1]])

# Apply sharpening

sharpened\_image = cv2.filter2D(image, -1, sharpening\_kernel)

# Display the original image, smoothed, and sharpened images

cv2\_imshow(image)

cv2\_imshow(smoothed\_image)

cv2\_imshow(sharpened\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

Output:



**write a python program to simulate smoothening using Averaging linear filters**

import cv2

import numpy as np

from google.colab.patches import cv2\_imshow

# Load the image

image = cv2.imread('/content/smoothening data.png')

# Define the size of the Averaging filter kernel

kernel\_size = (5, 5)  # You can adjust the size based on the desired smoothing level

# Create the Averaging filter kernel

kernel = np.ones(kernel\_size, dtype=np.float32) / (kernel\_size[0] \* kernel\_size[1])

# Apply the Averaging filter for smoothing

smoothed\_image = cv2.filter2D(image, -1, kernel)

# Display the original and smoothed images

cv2\_imshow(image)

cv2\_imshow(smoothed\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

output:



**write a python program to simulate smoothening using median filters**

import cv2

import numpy as np

from google.colab.patches import cv2\_imshow

# Load the image

image = cv2.imread('/content/smoothening data.png')

# Define the size of the median filter kernel (should be an odd number)

kernel\_size = 5  # You can adjust the size

# Apply the Median filter for smoothing

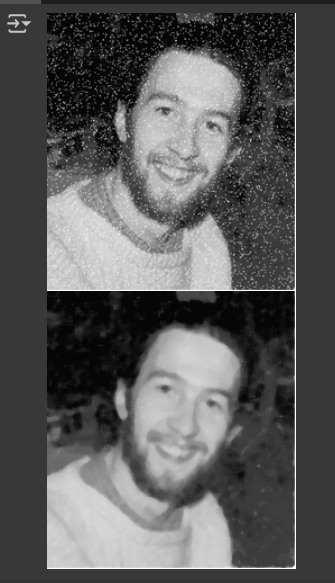
smoothed\_image = cv2.medianBlur(image, kernel\_size)

# Display the original and smoothed images using cv2\_imshow

cv2\_imshow(image)

cv2\_imshow(smoothed\_image)

Output:



**write a python program to simulate sharpening using spatial high pass filters**

import cv2

import numpy as np

from google.colab.patches import cv2\_imshow

# Load the image

image = cv2.imread('/content/smoothening data.png')

# Apply Gaussian smoothing to reduce noise (optional but recommended)

blurred\_image = cv2.GaussianBlur(image, (5, 5), 0)

# Create a Laplacian kernel for sharpening

laplacian\_kernel = np.array([[0, -1, 0],

                             [-1, 5, -1],

                             [0, -1, 0]], dtype=np.float32)

# Apply the Laplacian filter for sharpening

sharpened\_image = cv2.filter2D(blurred\_image, -1, laplacian\_kernel)

# Display the original image, blurred image, and sharpened image

cv2\_imshow(image)

cv2\_imshow(blurred\_image)

cv2\_imshow(sharpened\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

output:

