Code for Gaussian Kernel -

```
def gaussian_kernel(sigma_x,sigma_y):
    k_sizex = int (5*sigma_x)
    k_sizey = int (5*sigma_y)
    if(k_sizex % 2 == 0):
        k_sizex+=1
    if(k_sizey % 2 == 0):
        k_sizey+=1
    norm = 1/( 2*3.141592*sigma_x*sigma_y
    gaussian = np.zeros((k_sizex,k_sizey),np.float32
    for x in range(k_sizex):
        for y in range(k_sizey):
            px = (x**2)/(sigma_x**2)
            py =  (y**2)/(sigma_y**2)
            p = (px + py )/2
            p = math.exp(-p)
            gaussian[x,y]= p*norm
    print(gaussian)
    return gaussian
```

Gaussian Filter (Grayscale)

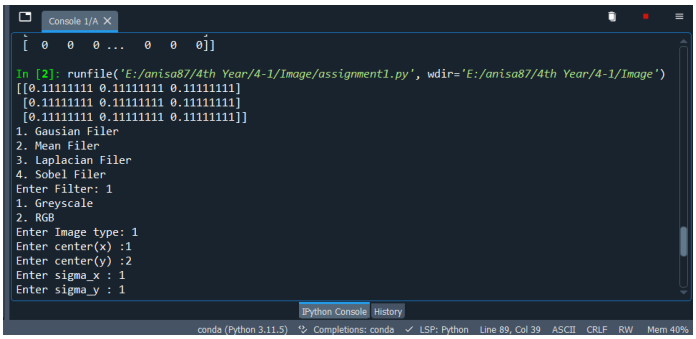


Fig 1.1 : Selection of Gaussian Filter and Grayscale image

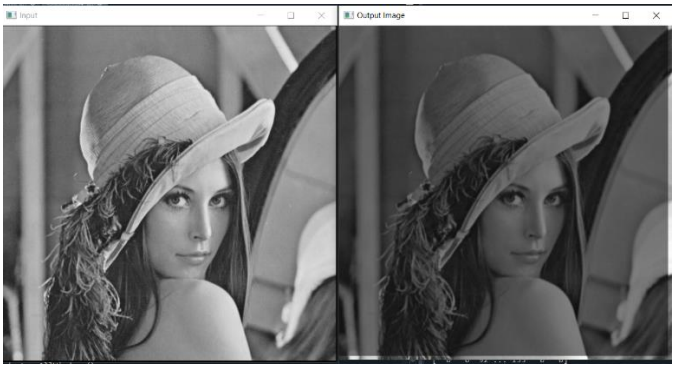


Fig 1.2 : Grayscale image before and after convolution

Gaussian Filter (RGB)

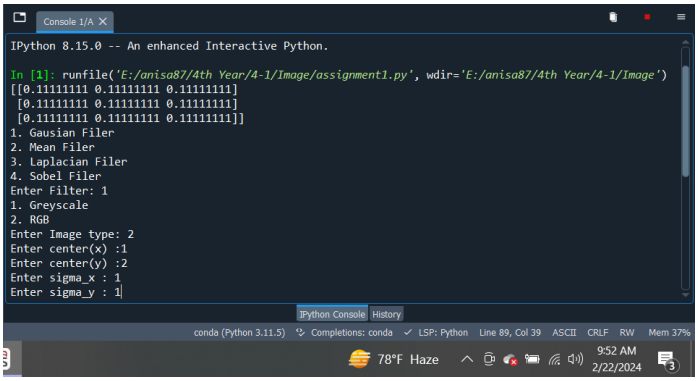


Fig 1.3 : Selection of Gaussian Filter and RGB image

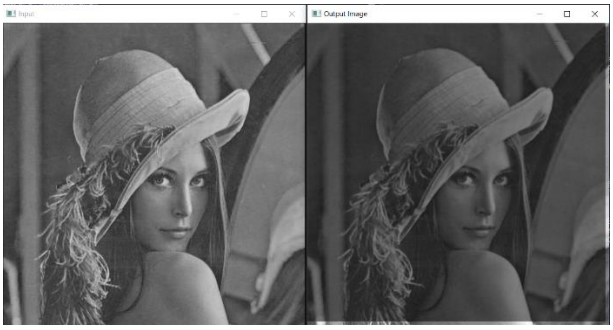
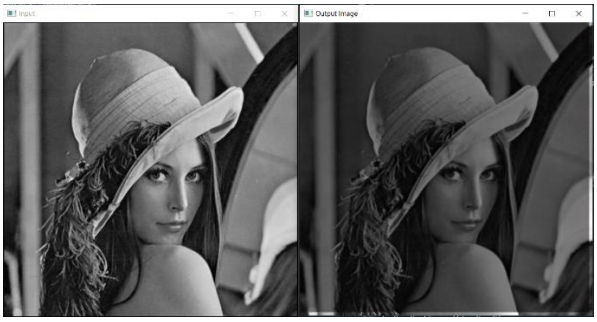


Fig 1.4: Blue Image for RGB

Fig 1.5: Blue Image for HSV



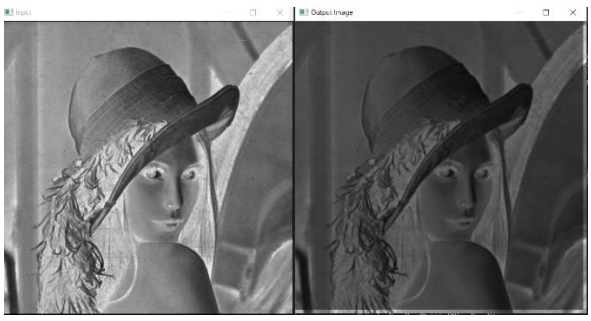


Fig 1.6 : Green Image for RGB

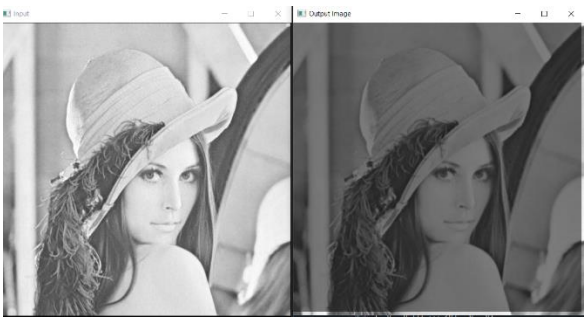


Fig 1.7 : Green Image for HSV

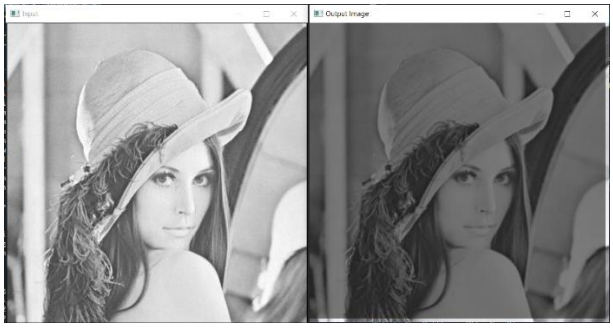


Fig 1.8 : Red Image for RGB

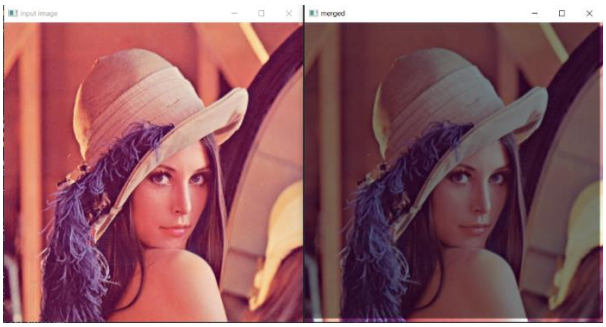


Fig 1.9 : Red Image for HSV

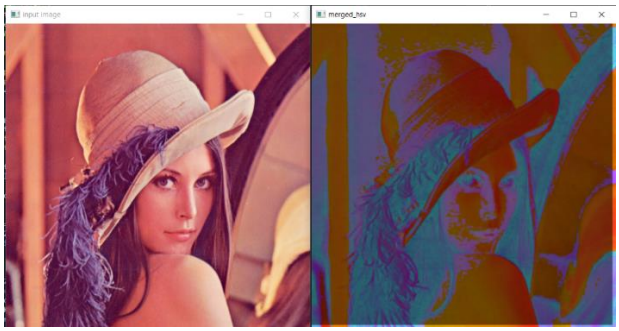


Fig 1.10 : Merge after convolution

Fig 1.11 : Merge after Concolution HSV

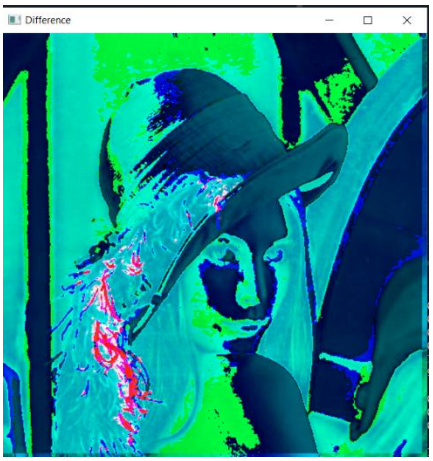


Fig 1.12: Difference of RGB and HSV convoluted image

Code for Mean Kernel -

```
def mean_kernel(row,col):
    meann = (1 / (row * col)) * np.ones((row,col), dtype=np.uint8
    print(meann)
    return meann
```

Mean Filter (Grayscale)

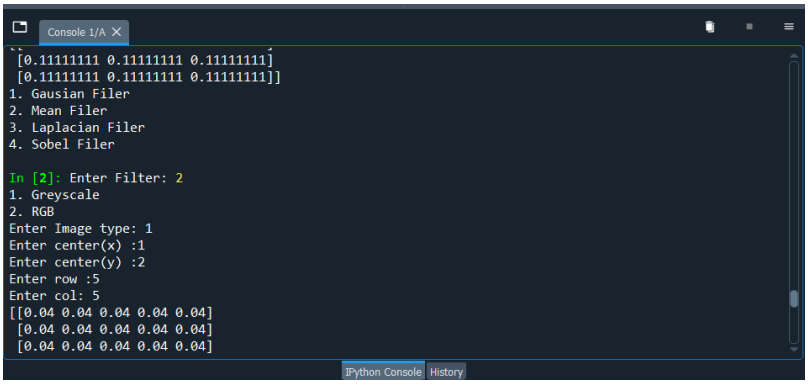


Fig 1.12: Selection of Mean filter in Grayscale image

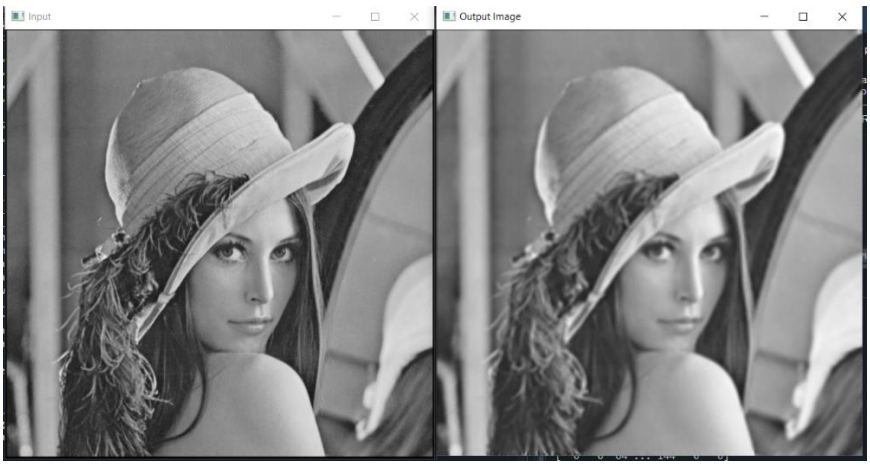


Fig 1.13: Grayscale image

Mean Filter (RGB)

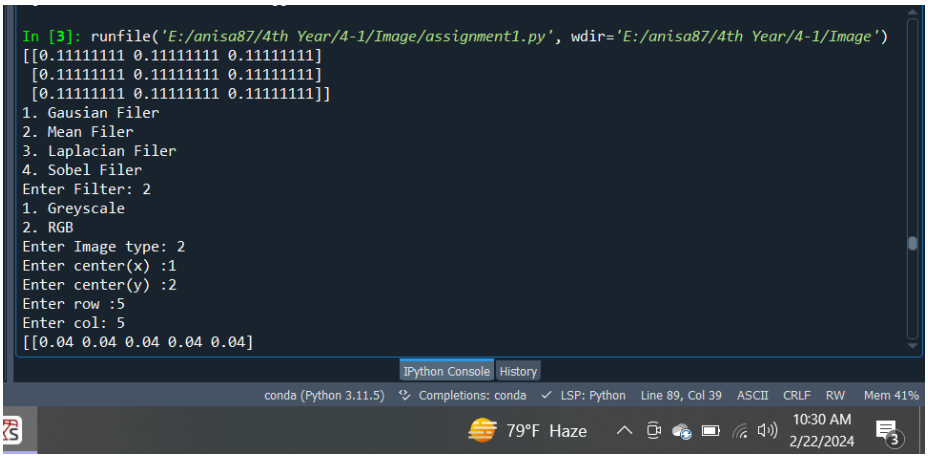


Fig 1.14: Selection of Mean filter in RGB image

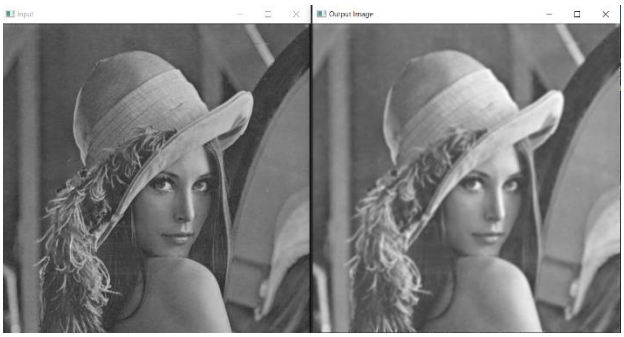


Fig 1.15: Blue Image

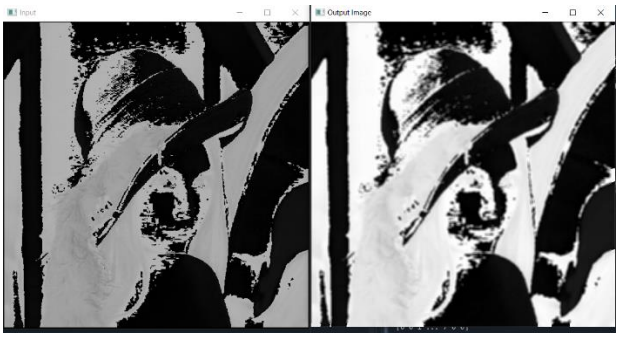


Fig 1.16: Blue Image HSV

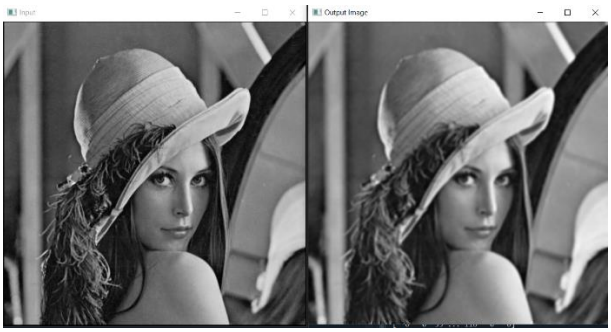


Fig 1.17: Green Image

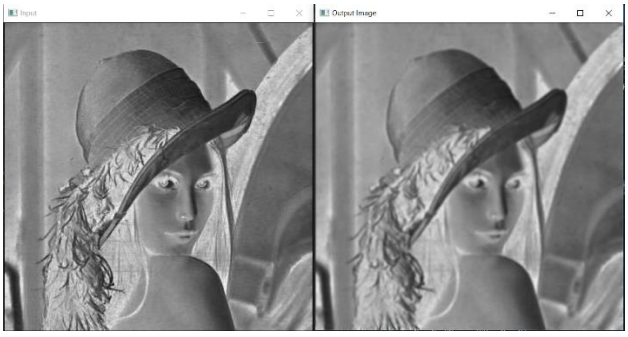


Fig 1.18 : Green Image HSV

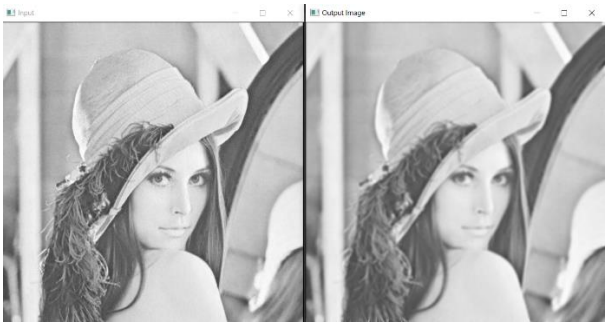


Fig 1.19: Red Image

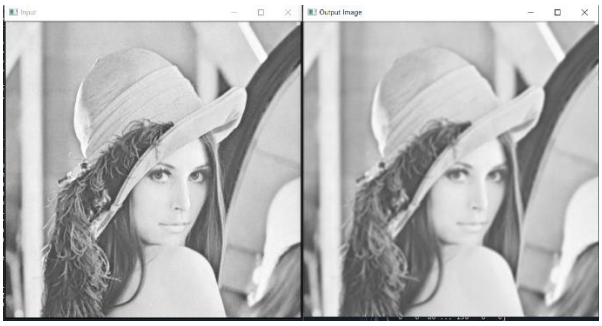


Fig 1.20 :Red Image HSV

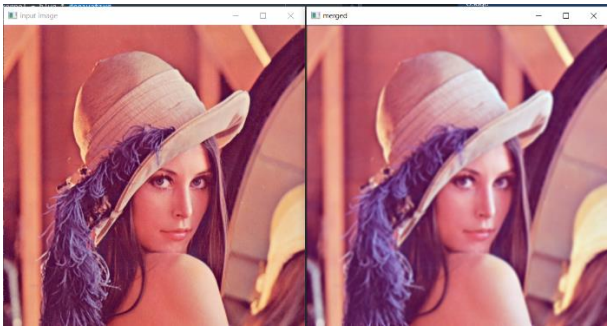


Fig 1.21: Merged Image after Convolution

Fig 1.22: Merged Image after Convolution HSV

Fig 1.23 :Difference of RGB and HSV convoluted image

Code for Laplacian Kernel -

```
def laplacian_kernel():
    laplacian = np.array([[0,1,0],
                          [1,-4,1],
                          [0,1,0]],np.float32)
    print(laplacian)
    return laplacian
```

Laplacian Filter(Grayscale)

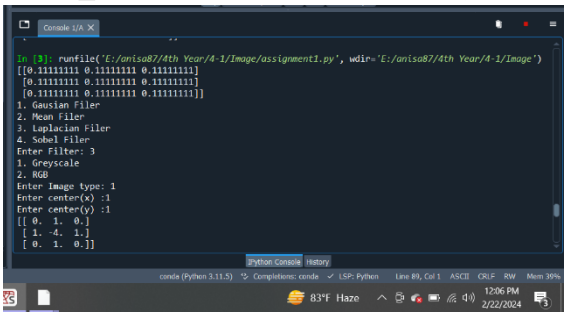


Fig 1.24: Selection of Mean filter in Grayscale image

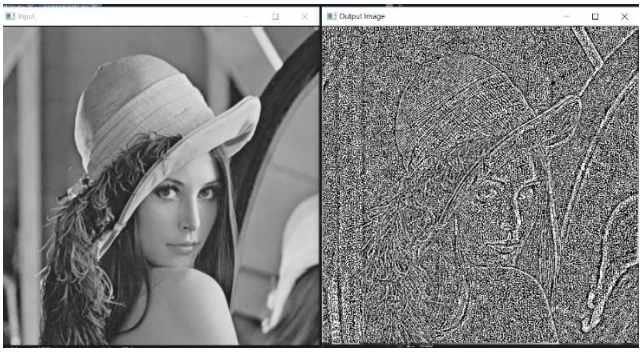


Fig 1.25: Grayscale image

Laplacian Filter (RGB)

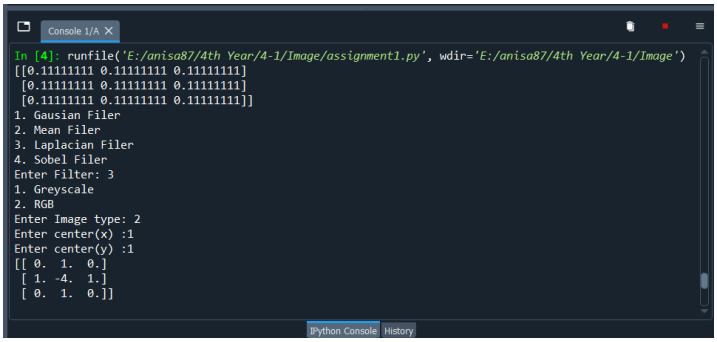


Fig 1.26: Selection of Laplacian filter in RGB image

Fig 1.27: Blue Image

Fig 1.28: Blue Image HSV

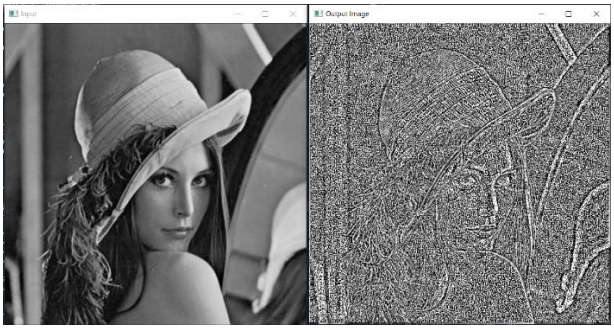


Fig 1.29: Green Image

Fig 1.30: Green Image HSV

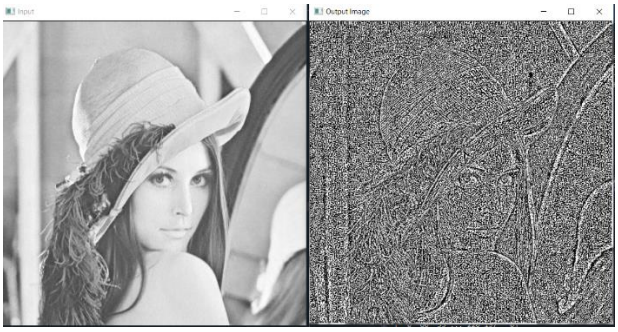


Fig 1.31: Red Image

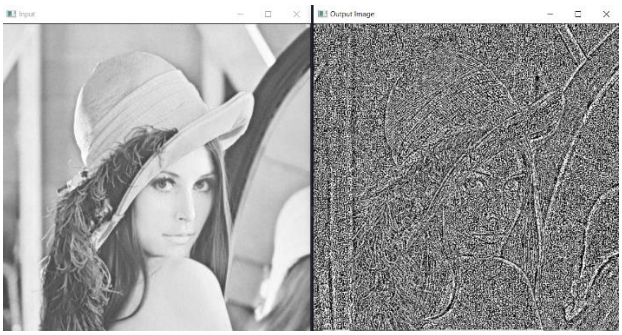


Fig 1.32: Red Image HSV

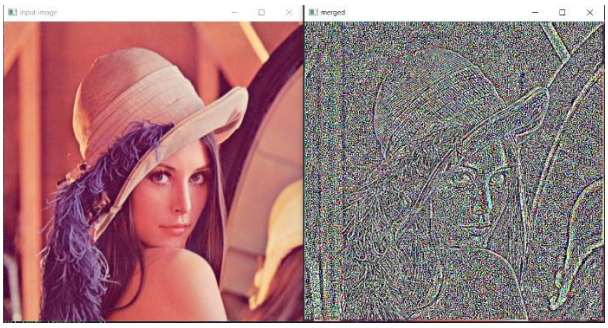


Fig 1.33: Merged Image after Convolution

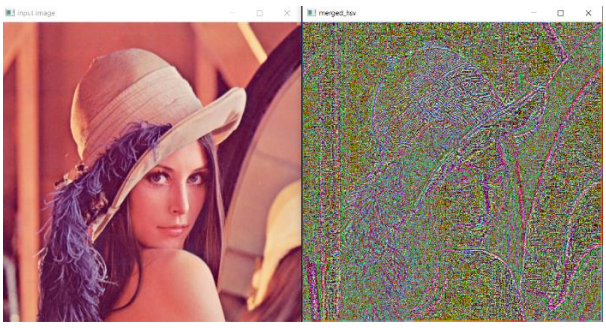


Fig 1.34: Merged Image after Convolution HSV

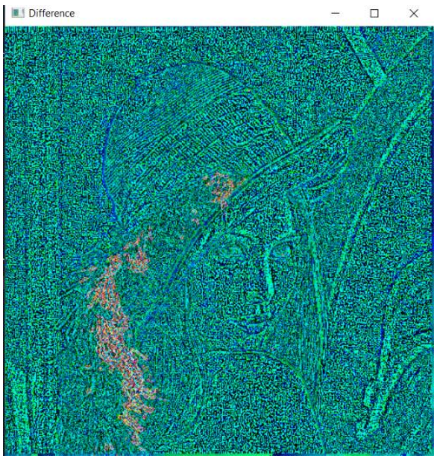


Fig 1.35 :Difference of RGB and HSV convoluted image

Code for Sobel Kernel -

```
def sobel_kernel():
    blur = np.array([[1],
                     [2],
                     [1]], dtype=np.float32)

    derivative = np.array([1, 0, -1], dtype=np.float32)
    h_sobel_kernel = blur * derivative
    return h_sobel_kernel
```

Sobel Filter (Grayscale)

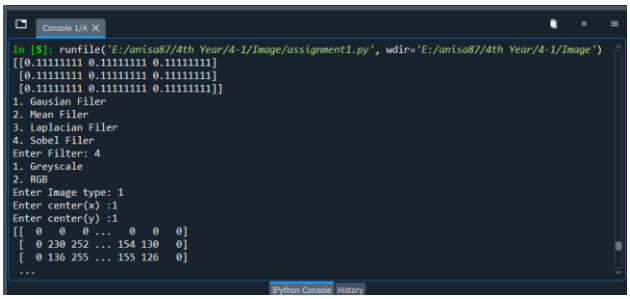


Fig 1.36: Selection of Sobel filter in Grayscale image

Fig 1.37:Grayscale image

Sobel Filter (RGB)

Fig 1.38: Fig 1.26: Selection of Sobel filter in RGB image

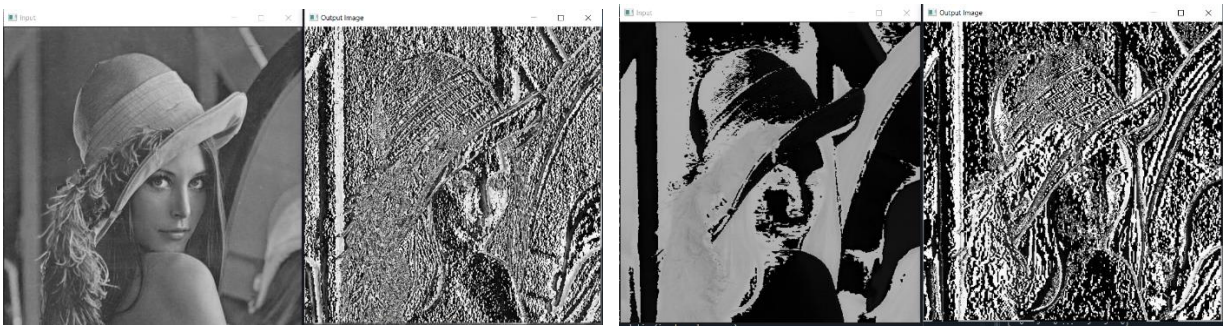


Fig 1.39: Blue Image

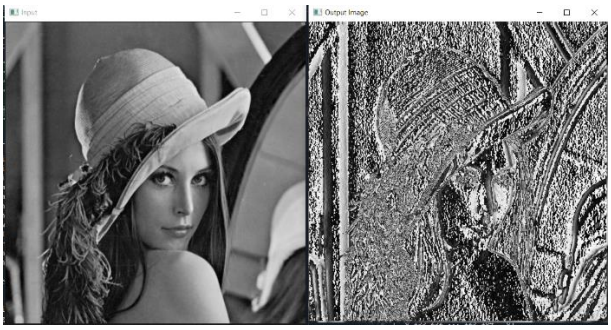


Fig 1.40: Blue Image HSV

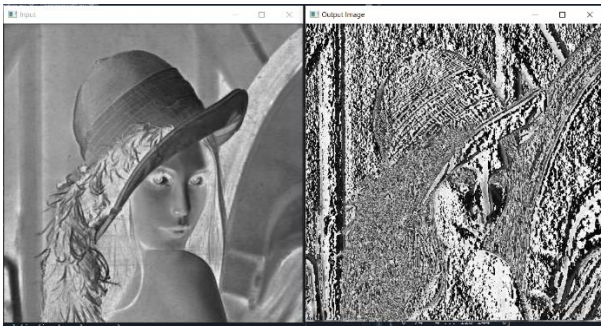


Fig 1.41: Green Image

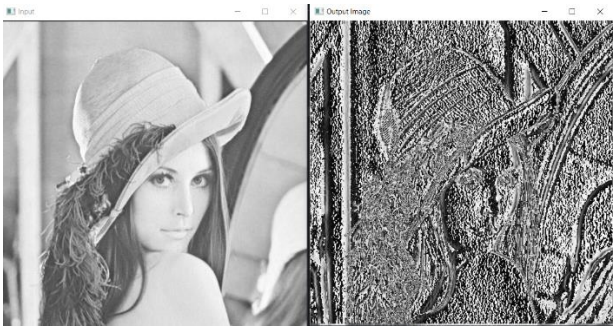


Fig 1.42: Green Image HSV

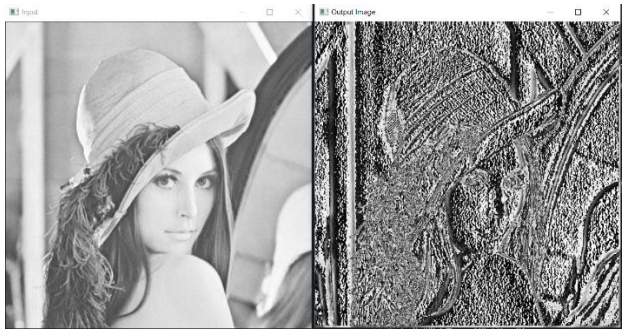


Fig 1.43: Red Image

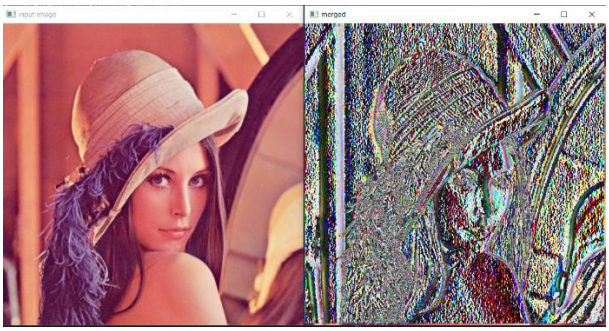


Fig 1.44: Red Image HSV

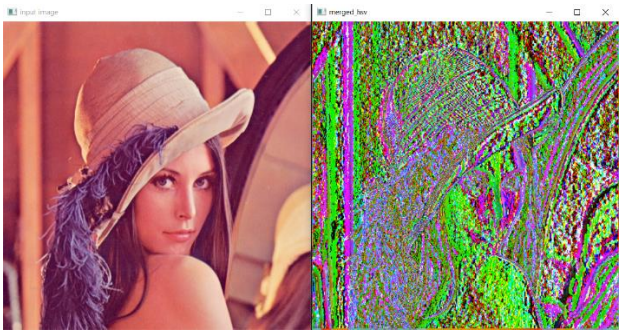


Fig 1.45: Merged Image after Convolution

Fig 1.46: Merged Image after Convolution HSV

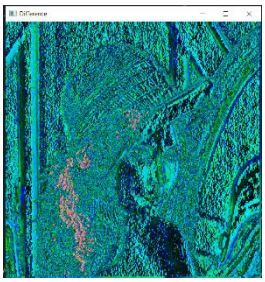


Fig 1.47: Difference of RGB and HSV convoluted image