**PAPER – III         PRACTICAL NO. : 02**

**AIM : WRITE PROGRAM TO DEMONSTRATE THE FOLLOWING ASPECTS OF SIGNAL ON SOUND /IMAGE DATA :**

1. **CONVOLUTION OPERATION**
2. **TEMPLATE MATCHING**

**ROLL NO. : 02                                BATCH : M.SC PART-I**

**DATE : 25/08/22**

**1] CONVOLUTION OPERATION :**

**CODE :**

from skimage.io import imread,imshow,show

from skimage.color import rgb2gray

import numpy as np

from scipy import ndimage,misc,signal

import matplotlib.pylab as pylab

im = rgb2gray(imread('E:/Msc Roll no. 02/Image processing/tiger3.jpg')).astype(float)

print(np.max(im))

print(im.shape)

blur\_box\_kernel=np.ones((3,3))/9

edge\_laplace\_kernel =np.array([[0,1,0],[1,-4,1],[0,1,0]])

im\_blurred=signal.convolve2d(im,blur\_box\_kernel)

im\_edges=np.clip(signal.convolve2d(im,edge\_laplace\_kernel),0,1)

fig,axes=pylab.subplots(ncols=3,sharex=True,sharey=True,figsize=(18,6))

axes[0].imshow(im,cmap=pylab.cm.gray)

axes[0].set\_title('OriginalImage',size=20) # can use imshow instead of axes

axes[1].imshow(im\_blurred,cmap=pylab.cm.gray)

axes[1].set\_title('BoxBlur',size=20)

axes[2].imshow(im\_edges,cmap=pylab.cm.gray)

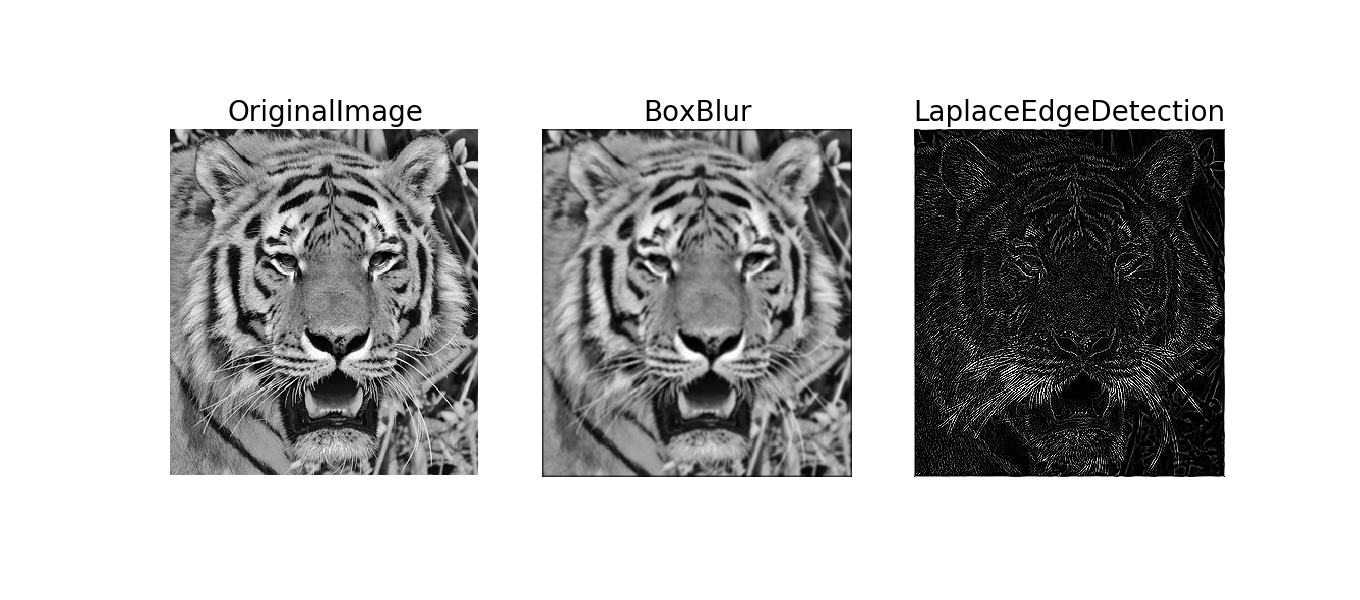
axes[2].set\_title('LaplaceEdgeDetection',size=20)

for ax in axes:

ax.axis('off')

pylab.show()

**OUTPUT :**



**2] TEMPLATE MATCHING :**

**CODE :**

from skimage.io import imread,imshow,show

from skimage.color import rgb2gray

import numpy as np

from scipy import ndimage,misc,signal

import matplotlib.pylab as pylab

#Applyingconvolutiontoacolor(RGB)image

im = imread('E:/Msc Roll no. 02/Image processing/tiger3.jpg').astype(np.float)

#readasfloat

print(np.max(im))

sharpen\_kernel=np.array([0,-1,0,-1,5,-1,0,-1,0]).reshape((3,3,1))

emboss\_kernel=np.array(np.array([[-2,-1,0],[-1,1,1],[0,1,2]])).reshape((3,3,1))

im\_sharp=ndimage.convolve(im,sharpen\_kernel,mode='nearest')

im\_sharp=np.clip(im\_sharp,0,255).astype(np.uint8)

im\_emboss=ndimage.convolve(im,emboss\_kernel,mode='nearest')

im\_emboss=np.clip(im\_emboss,0,255).astype(np.uint8)

pylab.figure(figsize=(10,15))

pylab.subplot(131),pylab.imshow(im.astype(np.uint8)),pylab.axis('off')

pylab.title('Original Image',size=25)

pylab.subplot(132),pylab.imshow(im\_sharp),pylab.axis('off')

pylab.title('Sharpened Image',size=25)

pylab.subplot(133),pylab.imshow(im\_emboss),pylab.axis('off')

pylab.title('Embossed Image',size=25)

pylab.tight\_layout()

pylab.show()

face\_image=misc.face(gray=True)-misc.face(gray=True).mean()

template\_image=np.copy(face\_image[300:365,670:750])

#righteye

template\_image-=template\_image.mean()

face\_image=face\_image+np.random.randn(\*face\_image.shape)\*50

#addrandomnoise

correlation=signal.correlate2d(face\_image,template\_image,boundary='symm',mode='same')

y,x=np.unravel\_index(np.argmax(correlation),correlation.shape)

fig,(ax\_original,ax\_template,ax\_correlation)=pylab.subplots(3,1,figsize=(6,15))

ax\_original.imshow(face\_image,cmap='gray')

ax\_original.set\_title('Original',size=20)

ax\_original.set\_axis\_off()

ax\_template.imshow(template\_image,cmap='gray')

ax\_template.set\_title('Template',size=20)

ax\_template.set\_axis\_off()

ax\_correlation.imshow(correlation,cmap='afmhot')

ax\_correlation.set\_title('Cross-correlation',size=20)

ax\_correlation.set\_axis\_off()

ax\_original.plot(x,y,'ro')

fig.show()

**OUTPUT :**

