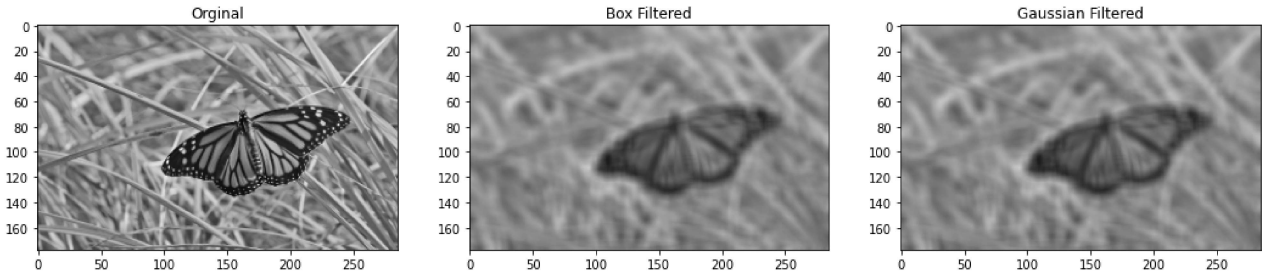
Index : 190144D

Name: Dilshan J.V.A.P

Exercise 3

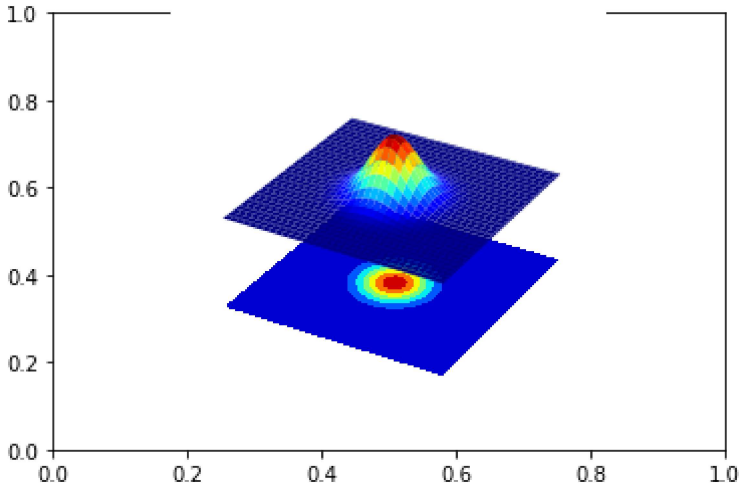
In [ ]:
```
import numpy as np
import cv2 as cv
import matplotlib.pyplot as plt
```

In [ ]:
```
im =cv.imread(r'butterfly.jpg',cv.IMREAD_REDUCED_GRAYSCALE_4)
assert im is not None

k_size = 9
sigma =4
box_kernal = 1./81*np.ones((9,9))
im_avg = cv.filter2D(im,-1,box_kernal)
im_gaussian = cv.GaussianBlur(im,(k_size,k_size),sigma)

fig,ax = plt.subplots (1,3,figsize = (18,6))
ax[0].imshow(im,cmap='gray',vmin = 0 ,vmax = 255)
ax[0].set_title('Orginal')
ax[1].imshow(im_avg,cmap='gray',vmin = 0 ,vmax = 255)
ax[1].set_title('Box Filtered')
ax[2].imshow(im_avg,cmap='gray',vmin = 0 ,vmax = 255)
ax[2].set_title('Gaussian Filtered')
plt.show()
```



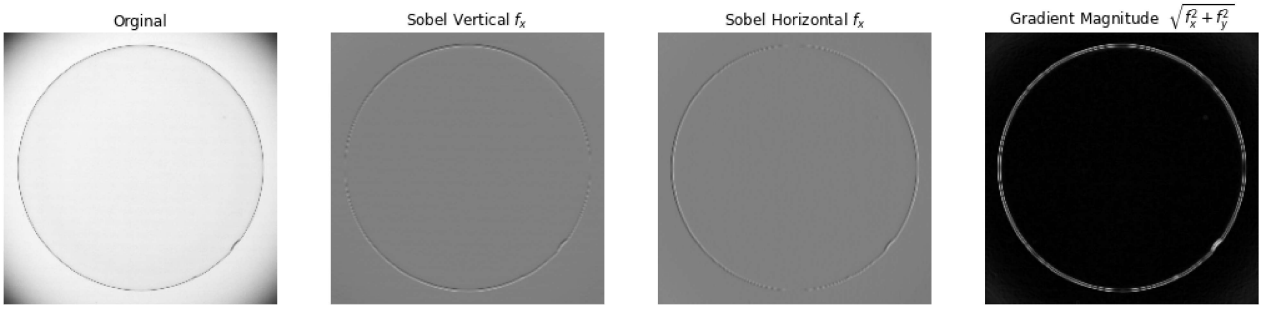
In [ ]:
```
#(2)

from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm

fig,ax = plt.subplots()
ax = fig.add_subplot(111,projection = '3d')

step = 0.1
sigma =1.
X = np.arange(-5,5 + step , step)
Y = np.arange(-5,5 + step , step)
XX,YY = np.meshgrid(X,Y)
g= np.exp(-(XX**2 +YY**2)/(2*sigma**2))

surf =ax.plot_surface(XX,YY,g,cmap = cm.jet)

cset = ax.contourf(XX,YY,g,zdir = 'z',offset = np.min(g) -1.5, cmap = cm.jet)
```

```
ax.set_zlim(np.min(g) -2,np.max(g))
plt.axis('off')
plt.show()
```



In [ ]:

```
#(3)

img = cv.imread(r'contact_lens.tif',cv.IMREAD_REDUCED_GRAYSCALE_4).astype(np.float32);
assert img is not None;
sobel_vertical = np.array([[-1,-2,-1],
 [0,0,0],
 [1,2,1]],dtype=np.float32);
sobel_horizontal = np.array([
 [-1,0,1],
 [-2,0,2],
 [-1,0,1]],dtype=np.float32);
im_x = cv.filter2D(img,-1,sobel_vertical)
im_y = cv.filter2D(img,-1,sobel_horizontal)
grad_mag = np.sqrt(im_x**2 + im_y**2)

fig,ax = plt.subplots (1,4,figsize = (18,6))
ax[0].imshow(img,cmap='gray',vmin = 0 ,vmax = 255)
ax[0].set_title('Orginal')
ax[1].imshow(im_x,cmap='gray',vmin = -1020 ,vmax =1020)
ax[1].set_title(r'Sobel Vertical $f_x$')
ax[2].imshow(im_y,cmap='gray',vmin = -1020 ,vmax =1020)
ax[2].set_title(r'Sobel Horizontal $f_x$')
ax[3].imshow(grad_mag,cmap = 'gray')
ax[3].set_title(r'Gradient Magnitude  $\sqrt{f_x^2 + f_y^2}$')
for i in range(4):
    ax[i].axis('off')
plt.show()
```



In [ ]:

```
#(4)

f =cv.imread(r'tom.jpg',cv.IMREAD_GRAYSCALE).astype(np.float32)
assert im is not None

sigma =2
gaussian_1d = cv.getGaussianKernel(5,sigma)
f_lp = cv.sepFilter2D(f,-1,gaussian_1d,gaussian_1d)
f_hp = f- f_lp
f_sharpened = cv.addWeighted(f,1.0,f_hp,2.0,0)

fig,ax = plt.subplots (1,4,figsize = (18,6))
ax[0].imshow(f,cmap='gray')
ax[0].set_title('Orginal')
ax[1].imshow(f_lp,cmap='gray')
ax[1].set_title(r'f_{lp}')
ax[2].imshow(f_hp,cmap='gray')
ax[2].set_title(r'f_{hp}')
ax[3].imshow(f_sharpened,cmap = 'gray')
ax[3].set_title(r'Sharpened')
for i in range(4):
    ax[i].axis('off')
plt.show()
```