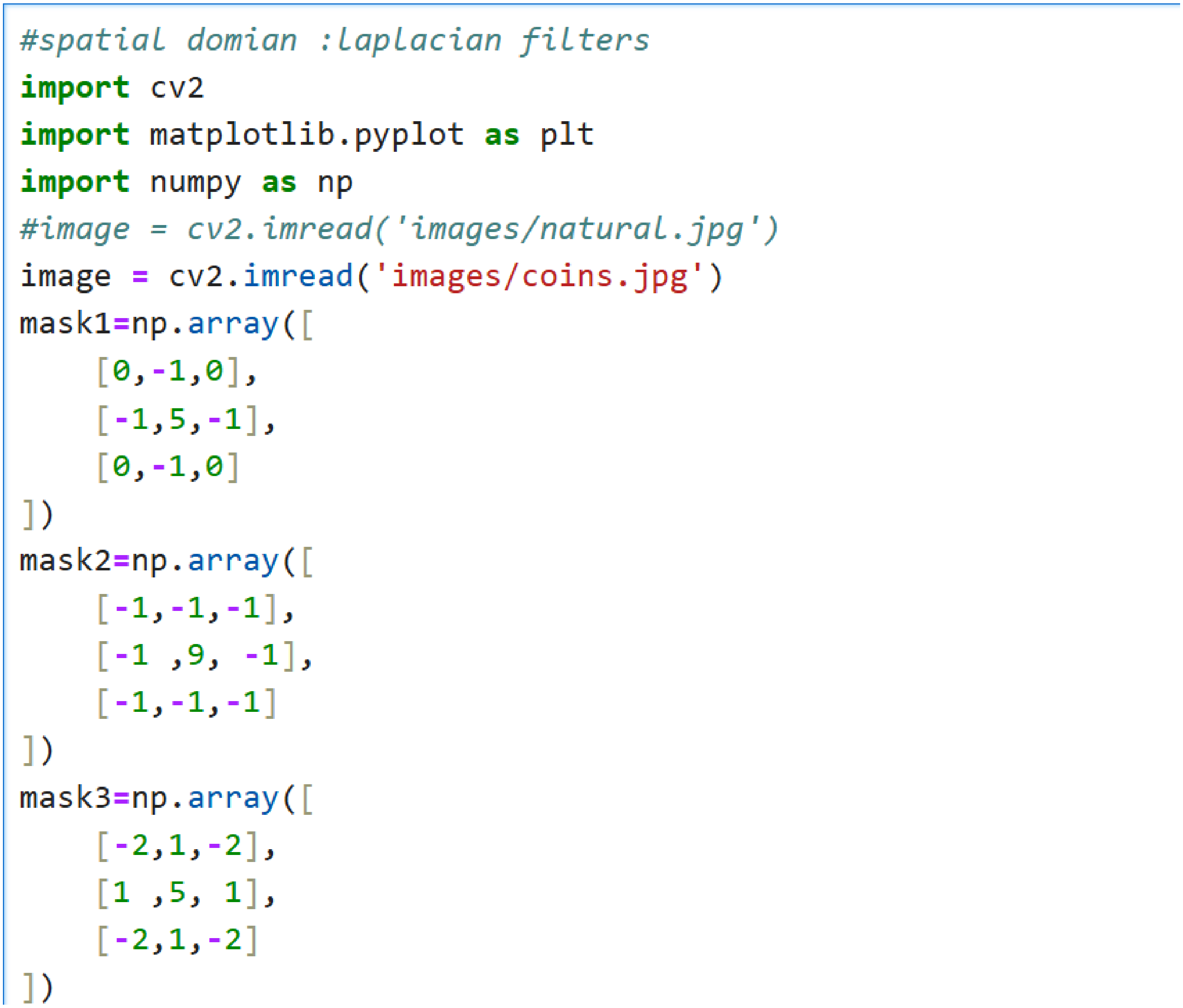
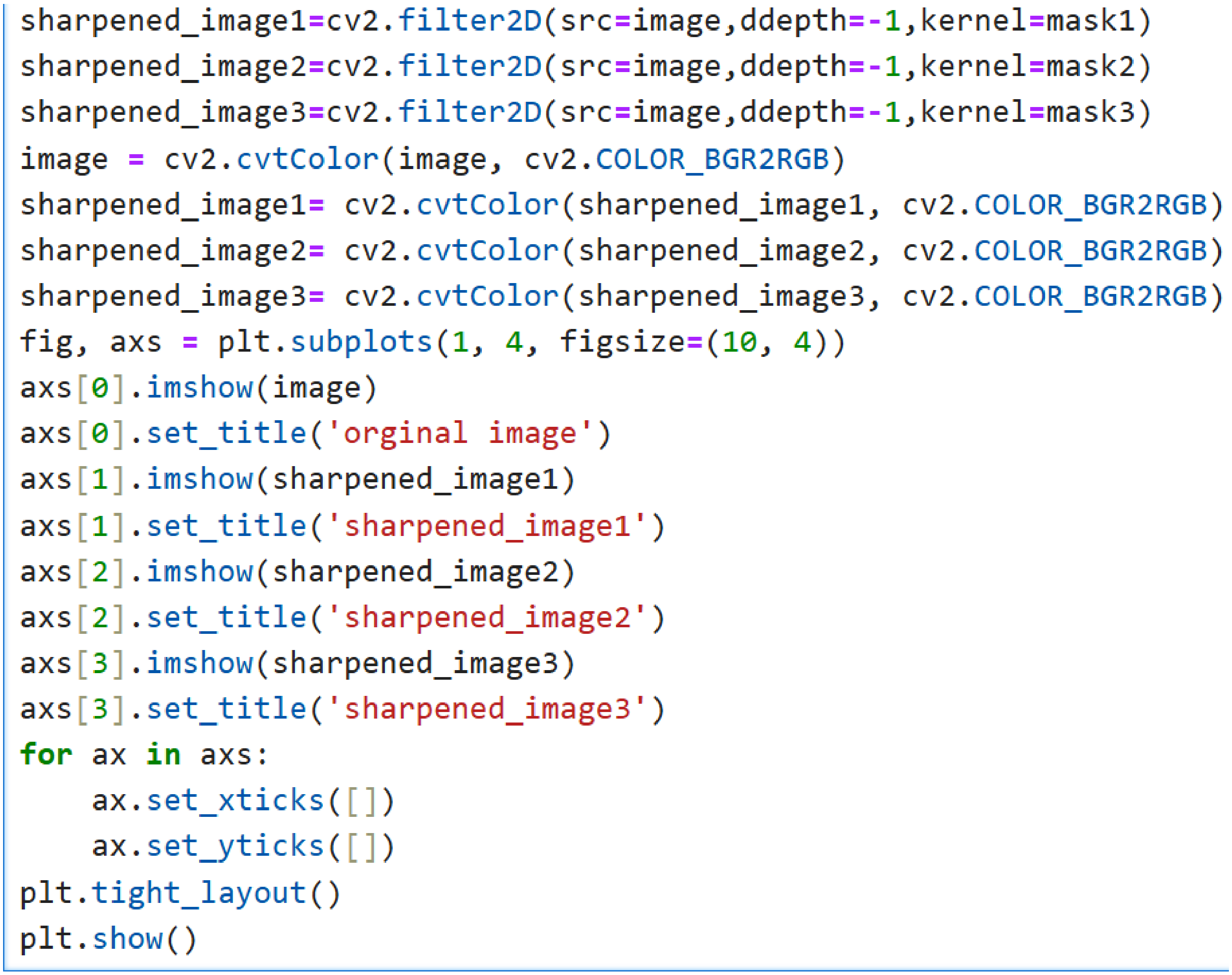
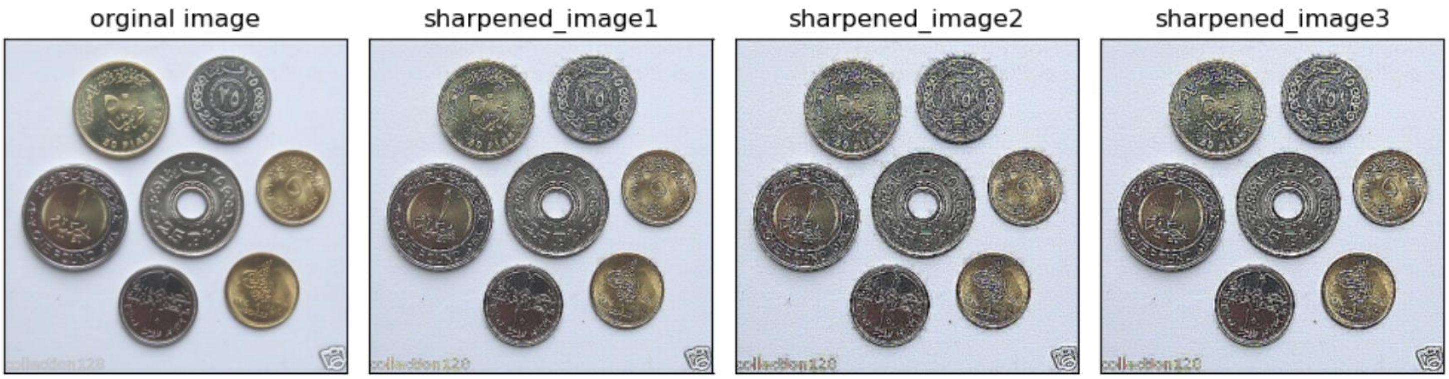
Spatial Domain- Sharpening Spatial Filters



```
#spatial domian :laplacian filters
import cv2
import matplotlib.pyplot as plt
import numpy as np
#image = cv2.imread('images/natural.jpg')
image = cv2.imread('images/coins.jpg')
mask1=np.array([
    [0,-1,0],
    [-1,5,-1],
    [0,-1,0]
])
mask2=np.array([
    [-1,-1,-1],
    [-1 ,9, -1],
    [-1,-1,-1]
])
mask3=np.array([
    [-2,1,-2],
    [1 ,5, 1],
    [-2,1,-2]
])
```



```
sharpened_image1=cv2.filter2D(src=image,ddepth=-1,kernel=mask1)
sharpened_image2=cv2.filter2D(src=image,ddepth=-1,kernel=mask2)
sharpened_image3=cv2.filter2D(src=image,ddepth=-1,kernel=mask3)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
sharpened_image1= cv2.cvtColor(sharpened_image1, cv2.COLOR_BGR2RGB)
sharpened_image2= cv2.cvtColor(sharpened_image2, cv2.COLOR_BGR2RGB)
sharpened_image3= cv2.cvtColor(sharpened_image3, cv2.COLOR_BGR2RGB)
fig, axs = plt.subplots(1, 4, figsize=(10, 4))
axs[0].imshow(image)
axs[0].set_title('orginal image')
axs[1].imshow(sharpened_image1)
axs[1].set_title('sharpened_image1')
axs[2].imshow(sharpened_image2)
axs[2].set_title('sharpened_image2')
axs[3].imshow(sharpened_image3)
axs[3].set_title('sharpened_image3')
for ax in axs:
    ax.set_xticks([])
    ax.set_yticks([])
plt.tight_layout()
plt.show()
```

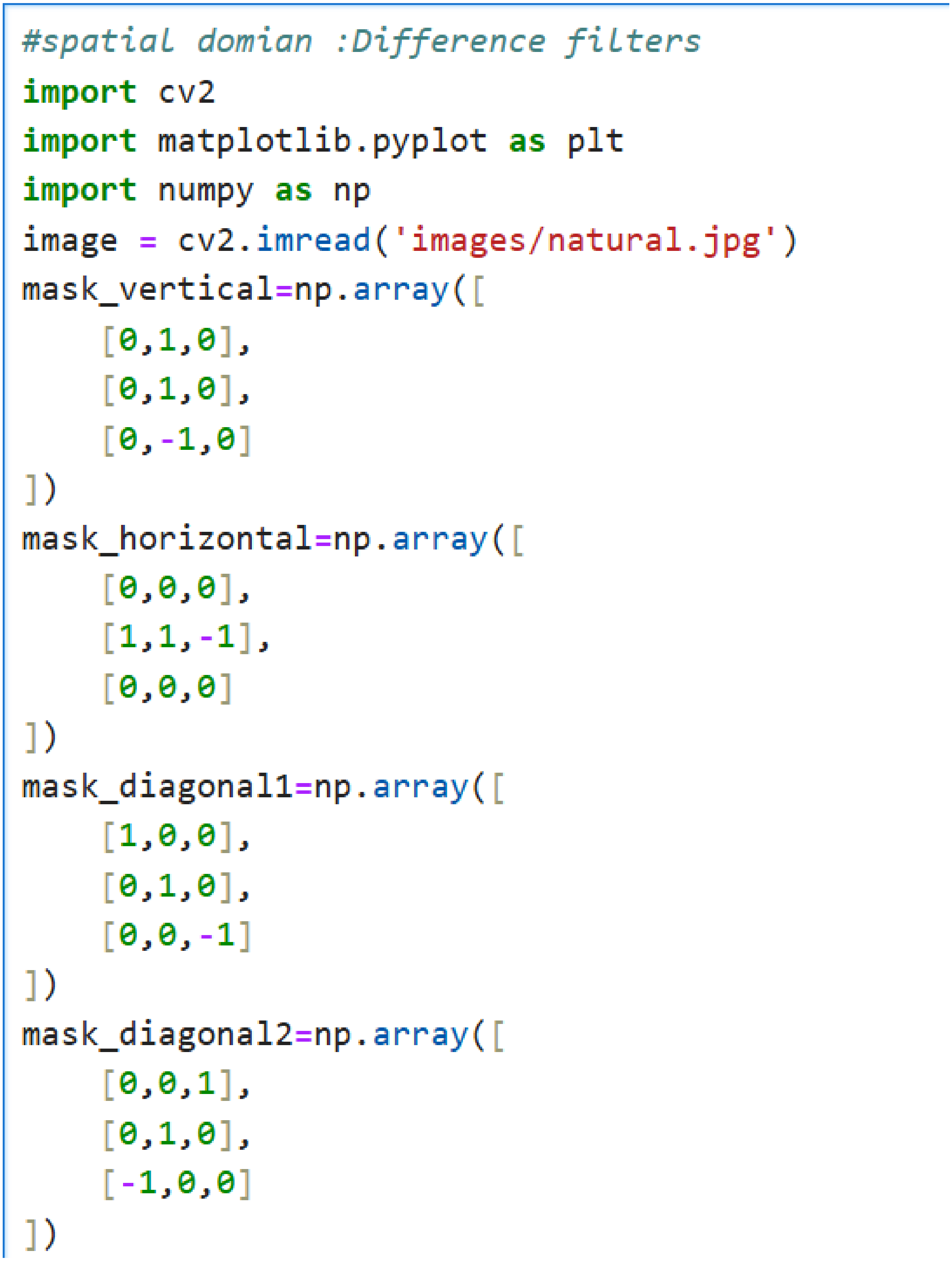


orginal image

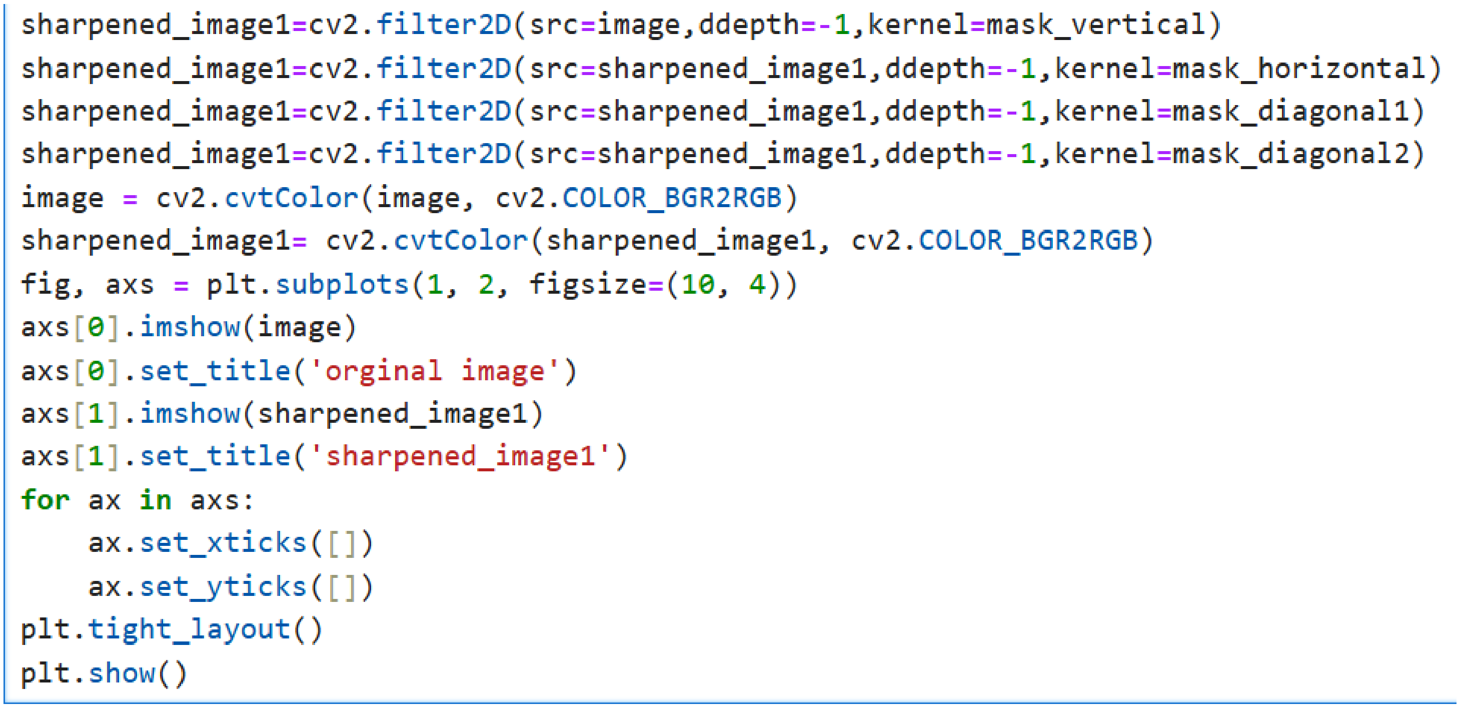
sharpened_image1

sharpened_image2

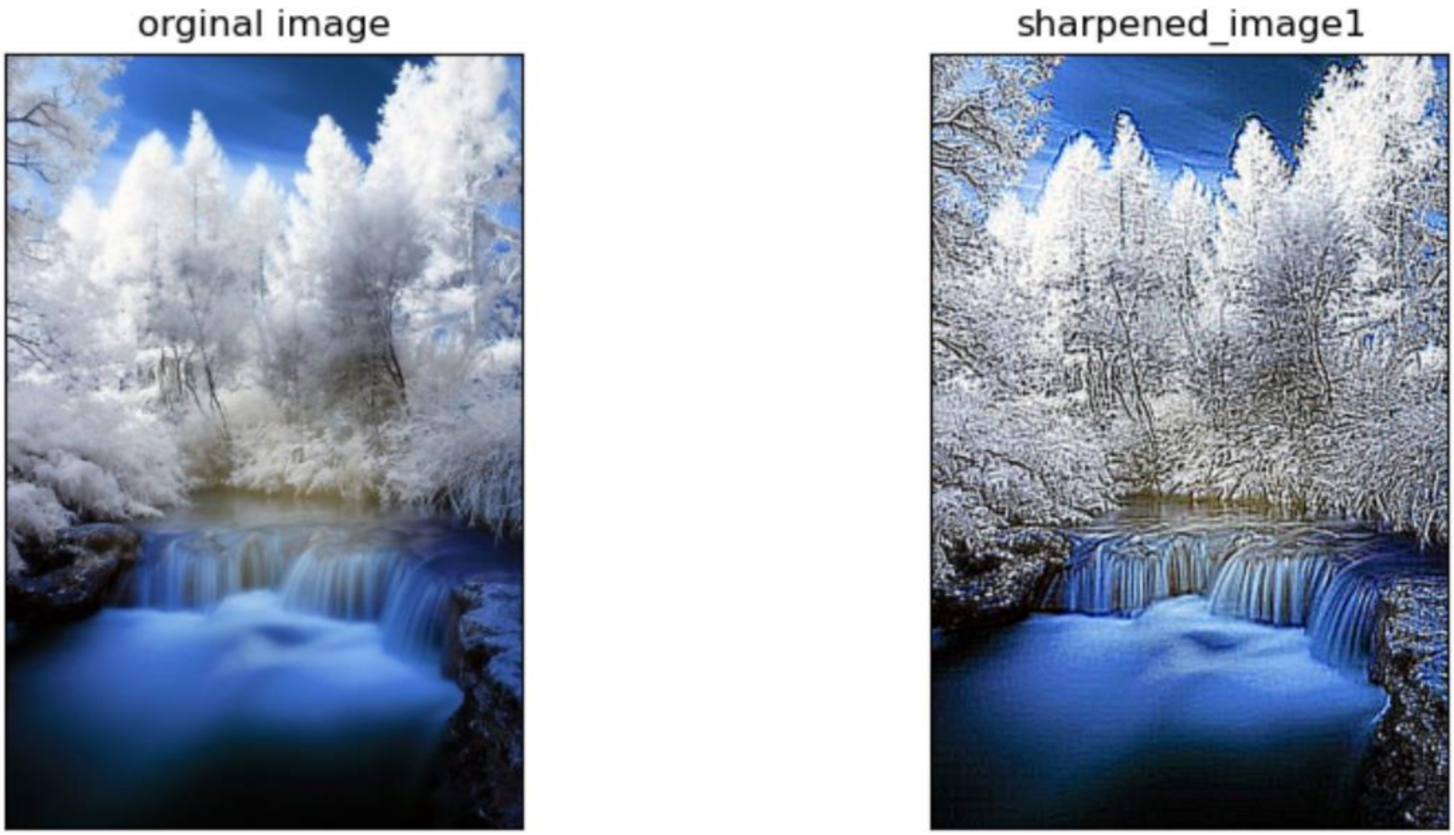
sharpened_image3



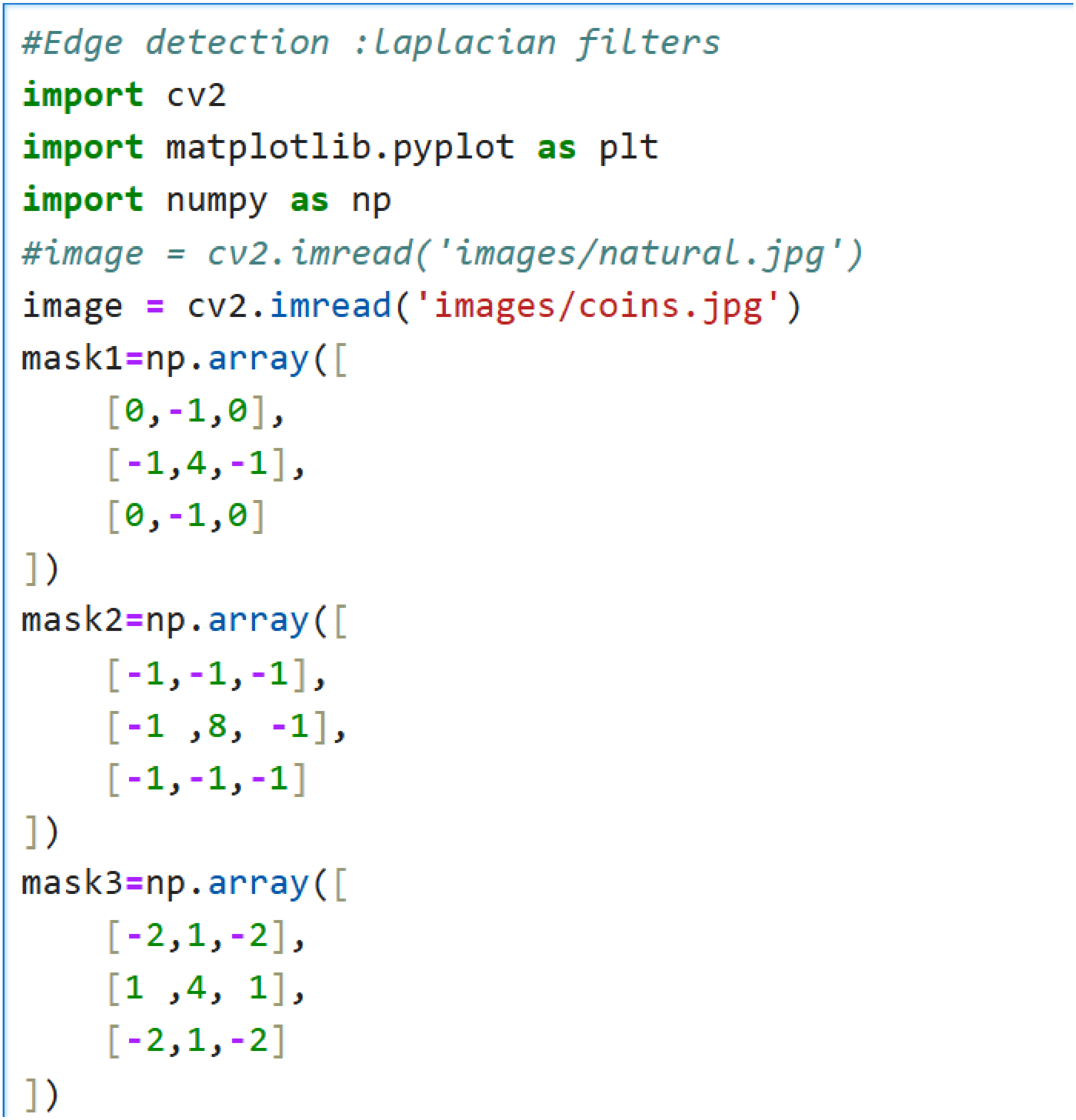
```
#spatial domian :Difference filters
import cv2
import matplotlib.pyplot as plt
import numpy as np
image = cv2.imread('images/natural.jpg')
mask_vertical=np.array([
    [0,1,0],
    [0,1,0],
    [0,-1,0]
])
mask_horizontal=np.array([
    [0,0,0],
    [1,1,-1],
    [0,0,0]
])
mask_diagonal1=np.array([
    [1,0,0],
    [0,1,0],
    [0,0,-1]
])
mask_diagonal2=np.array([
    [0,0,1],
    [0,1,0],
    [-1,0,0]
])
```



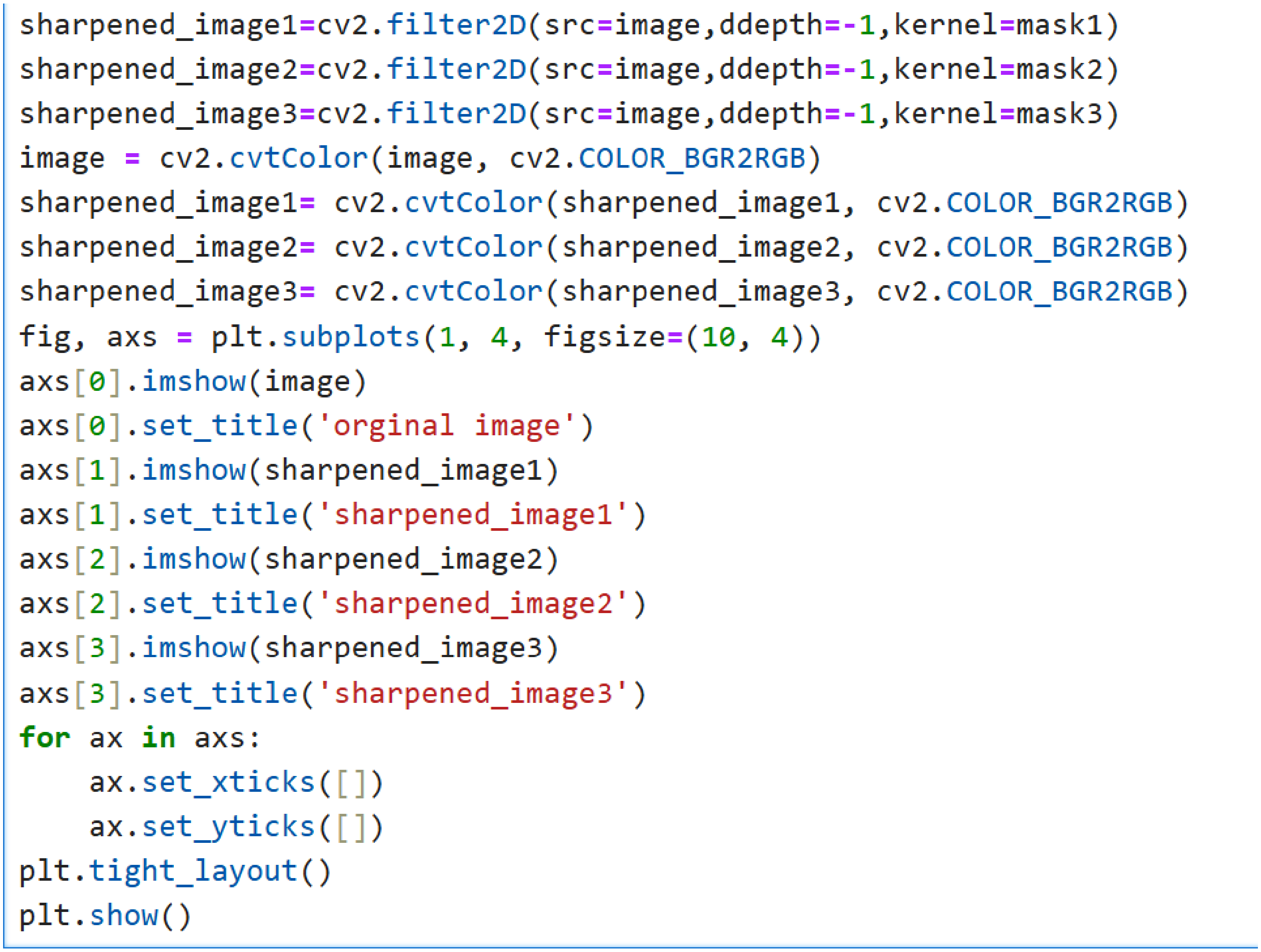
```
sharpened_image1=cv2.filter2D(src=image,ddepth=-1,kernel=mask_vertical)
sharpened_image1=cv2.filter2D(src=sharpened_image1,ddepth=-1,kernel=mask_horizontal)
sharpened_image1=cv2.filter2D(src=sharpened_image1,ddepth=-1,kernel=mask_diagonal1)
sharpened_image1=cv2.filter2D(src=sharpened_image1,ddepth=-1,kernel=mask_diagonal2)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
sharpened_image1= cv2.cvtColor(sharpened_image1, cv2.COLOR_BGR2RGB)
fig, axs = plt.subplots(1, 2, figsize=(10, 4))
axs[0].imshow(image)
axs[0].set_title('orginal image')
axs[1].imshow(sharpened_image1)
axs[1].set_title('sharpened_image1')
for ax in axs:
    ax.set_xticks([])
    ax.set_yticks([])
plt.tight_layout()
plt.show()
```



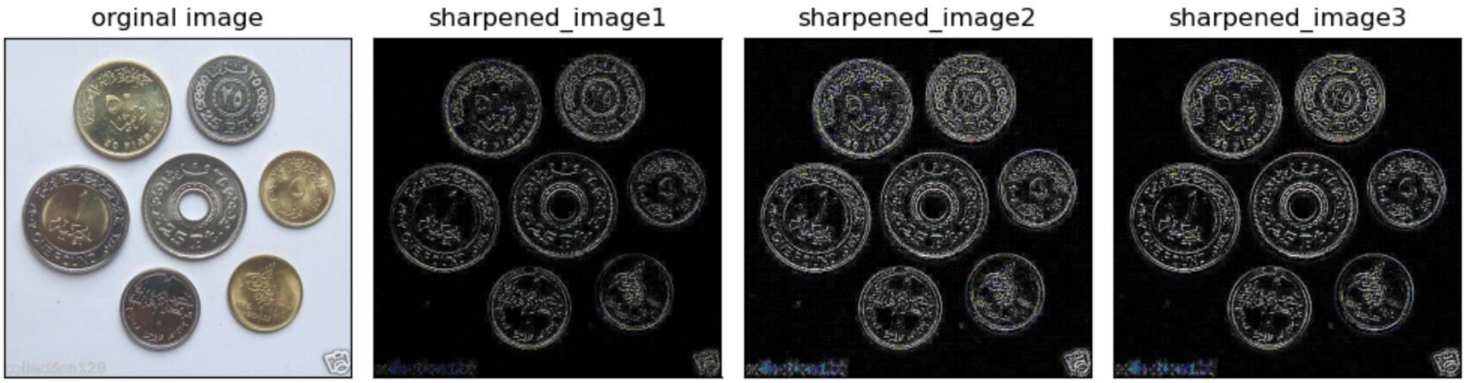
Edge Detection

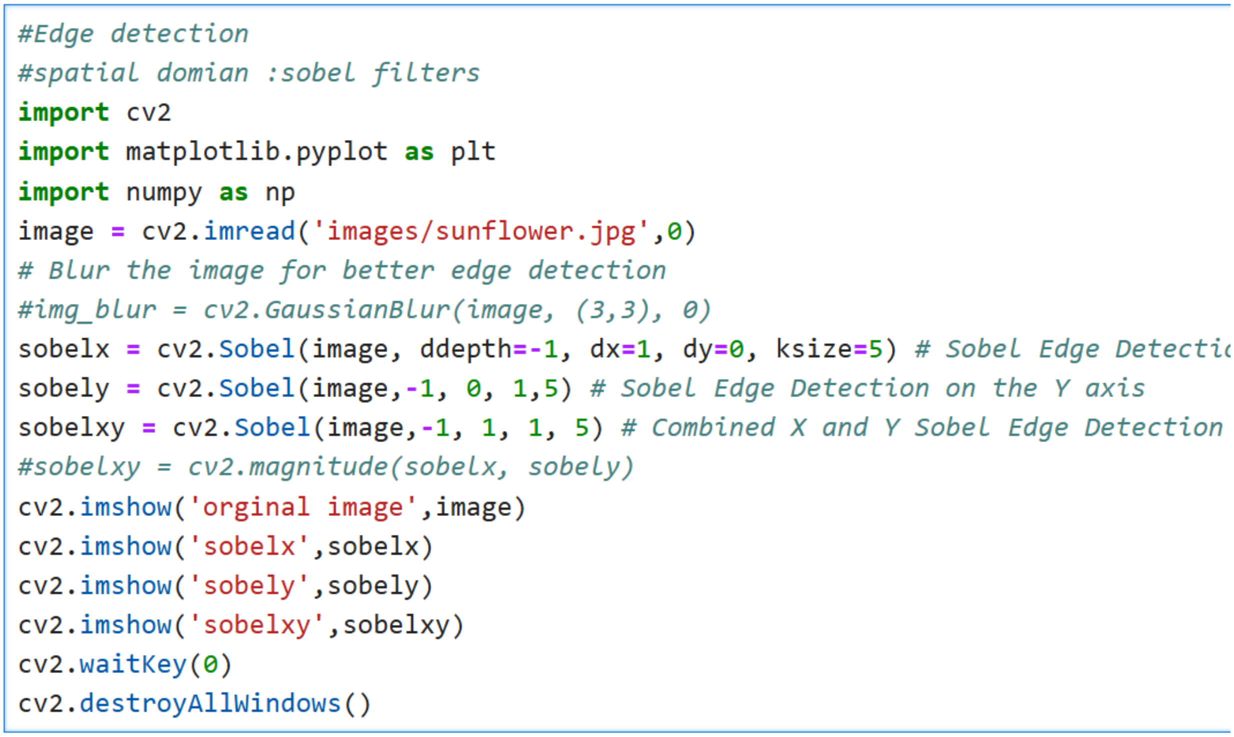


```
#Edge detection :laplacian filters
import cv2
import matplotlib.pyplot as plt
import numpy as np
#image = cv2.imread('images/natural.jpg')
image = cv2.imread('images/coins.jpg')
mask1=np.array([
    [0,-1,0],
    [-1,4,-1],
    [0,-1,0]
])
mask2=np.array([
    [-1,-1,-1],
    [-1 ,8, -1],
    [-1,-1,-1]
])
mask3=np.array([
    [-2,1,-2],
    [1 ,4, 1],
    [-2,1,-2]
])
```

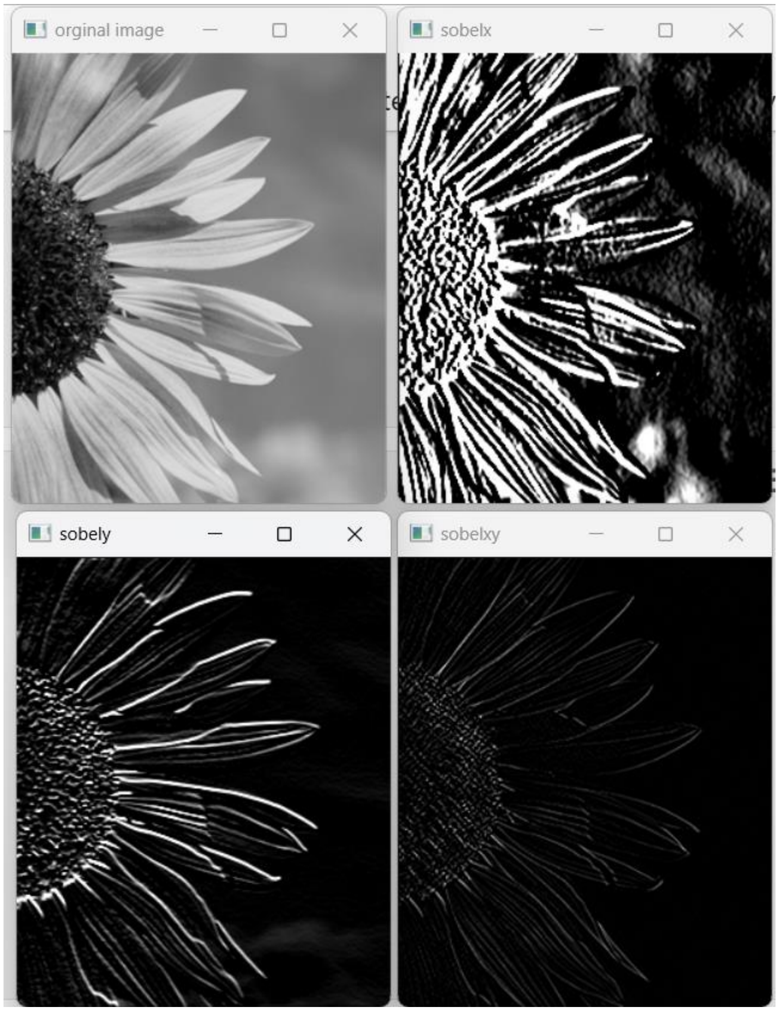


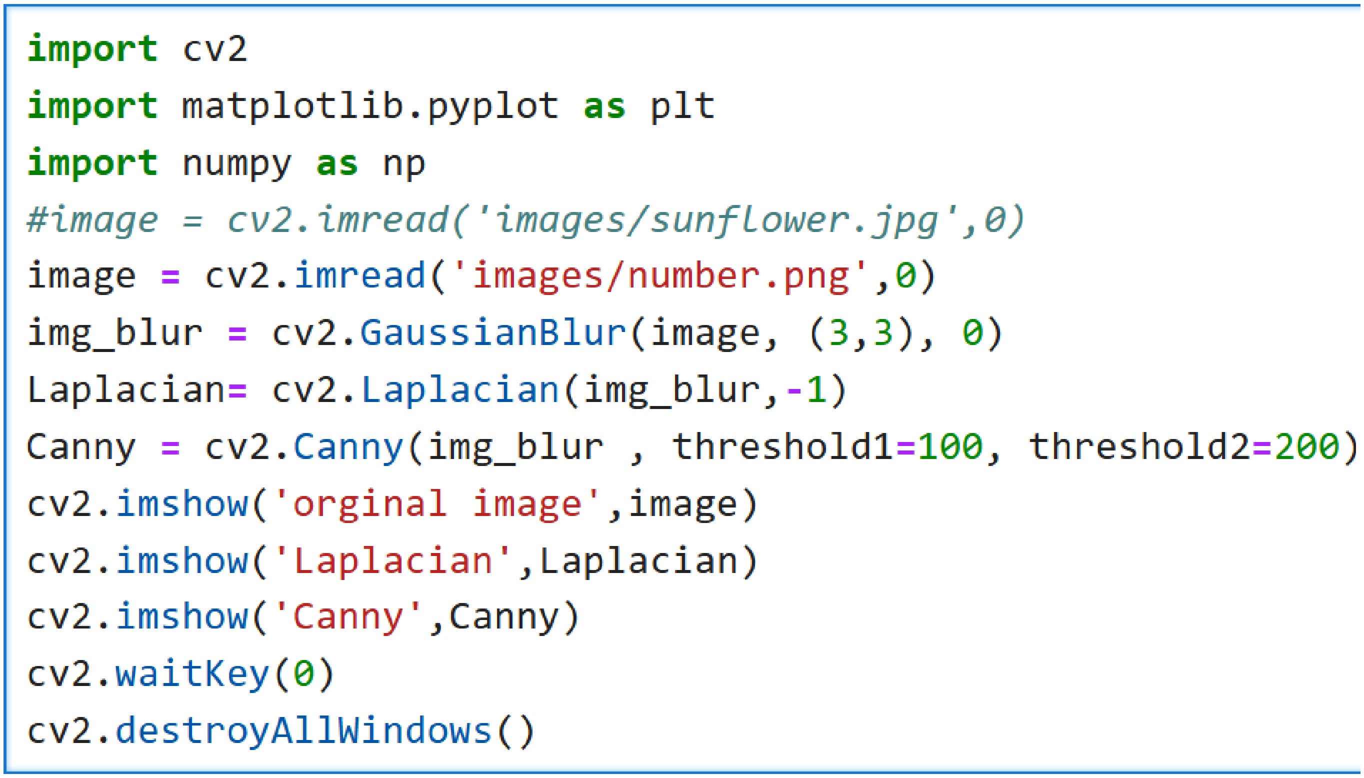
```
sharpened_image1=cv2.filter2D(src=image,ddepth=-1,kernel=mask1)
sharpened_image2=cv2.filter2D(src=image,ddepth=-1,kernel=mask2)
sharpened_image3=cv2.filter2D(src=image,ddepth=-1,kernel=mask3)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
sharpened_image1= cv2.cvtColor(sharpened_image1, cv2.COLOR_BGR2RGB)
sharpened_image2= cv2.cvtColor(sharpened_image2, cv2.COLOR_BGR2RGB)
sharpened_image3= cv2.cvtColor(sharpened_image3, cv2.COLOR_BGR2RGB)
fig, axs = plt.subplots(1, 4, figsize=(10, 4))
axs[0].imshow(image)
axs[0].set_title('orginal image')
axs[1].imshow(sharpened_image1)
axs[1].set_title('sharpened_image1')
axs[2].imshow(sharpened_image2)
axs[2].set_title('sharpened_image2')
axs[3].imshow(sharpened_image3)
axs[3].set_title('sharpened_image3')
for ax in axs:
    ax.set_xticks([])
    ax.set_yticks([])
plt.tight_layout()
plt.show()
```



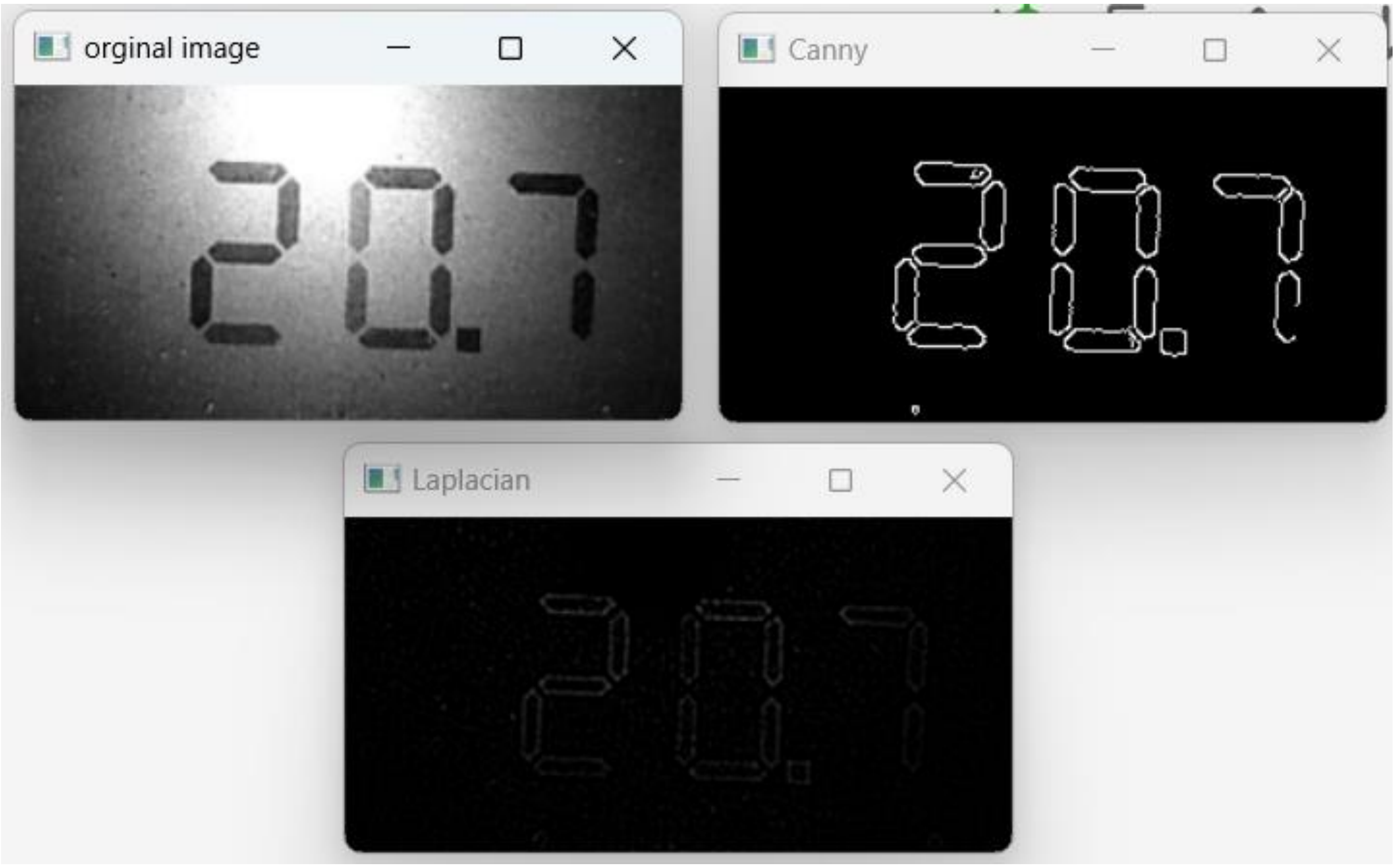


```
#Edge detection
#spatial domian :sobel filters
import cv2
import matplotlib.pyplot as plt
import numpy as np
image = cv2.imread('images/sunflower.jpg',0)
# Blur the image for better edge detection
#img_blur = cv2.GaussianBlur(image, (3,3), 0)
sobelx = cv2.Sobel(image, ddepth=-1, dx=1, dy=0, ksize=5) # Sobel Edge Detecti
sobely = cv2.Sobel(image,-1, 0, 1,5) # Sobel Edge Detection on the Y axis
sobelxy = cv2.Sobel(image,-1, 1, 1, 5) # Combined X and Y Sobel Edge Detection
#sobelxy = cv2.magnitude(sobelx, sobely)
cv2.imshow('orginal image',image)
cv2.imshow('sobelx',sobelx)
cv2.imshow('sobely',sobely)
cv2.imshow('sobelxy',sobelxy)
cv2.waitKey(0)
cv2.destroyAllWindows()
```





```
import cv2
import matplotlib.pyplot as plt
import numpy as np
#image = cv2.imread('images/sunflower.jpg',0)
image = cv2.imread('images/number.png',0)
img_blur = cv2.GaussianBlur(image, (3,3), 0)
Laplacian= cv2.Laplacian(img_blur,-1)
Canny = cv2.Canny(img_blur , threshold1=100, threshold2=200)
cv2.imshow('orginal image',image)
cv2.imshow('Laplacian',Laplacian)
cv2.imshow('Canny',Canny)
cv2.waitKey(0)
cv2.destroyAllWindows()
```


orginal image
Canny
Laplacian