Computer Vision

Lab 1

Submitted By:

Name: Sana Naz

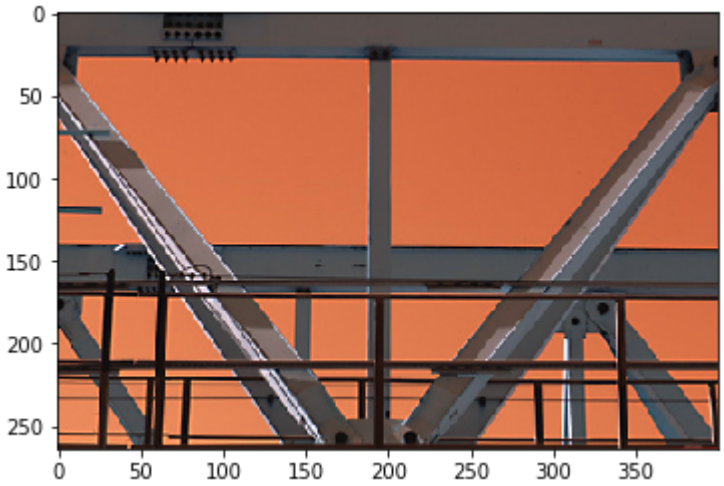
Class: BSCS-6B

Reg. No.: 197631

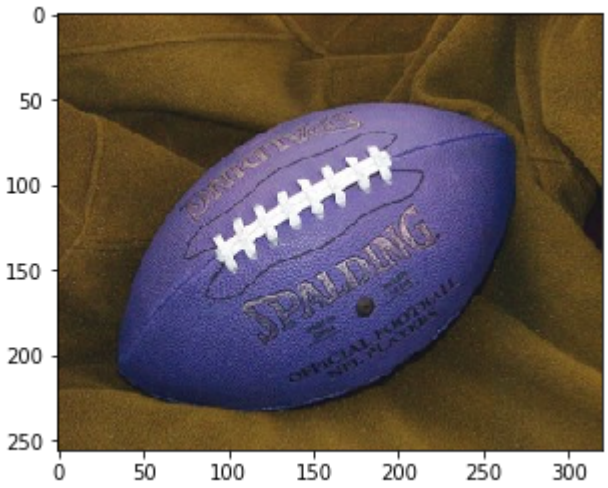
```
1 %matplotlib inline
2 import matplotlib.pyplot as plt
3 import matplotlib.image as mpimg
```

Display Images

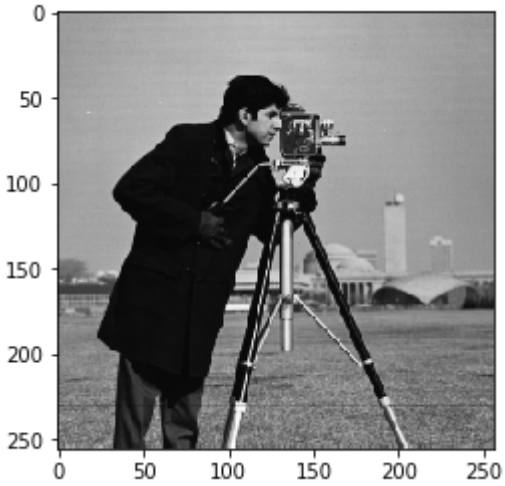
```
1 c_img1 = cv2.imread('/gantrycrane.png')
2 imgplot1 = plt.imshow(c_img1)
```




```
1 c_img2 = cv2.imread("/football.jpg")
2 imgplot2 = plt.imshow(c_img2)
```




```
1 c_img3 = cv2.imread("/cameraman.tif")
2 imgplot3 = plt.imshow(c_img3)
```



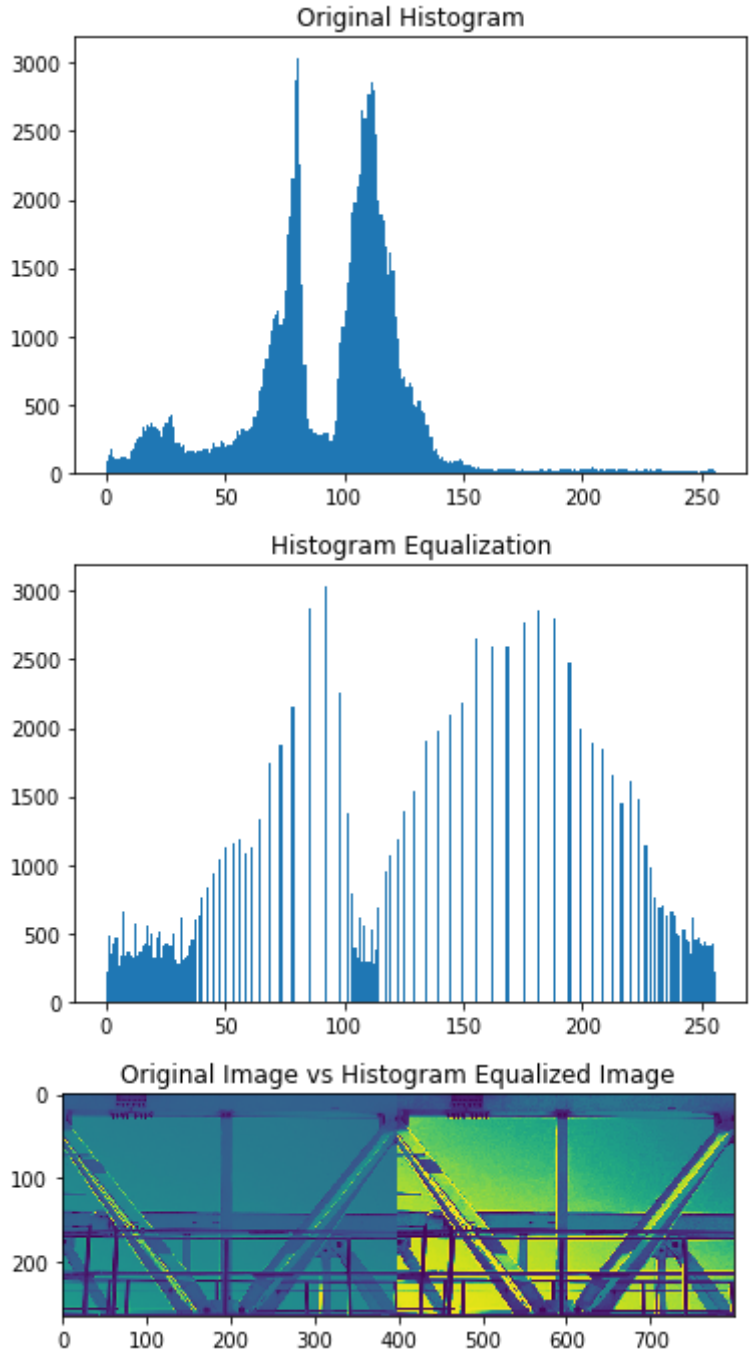
Histograms and Histogram Equalization

```
1 import cv2
2 import numpy as np
```

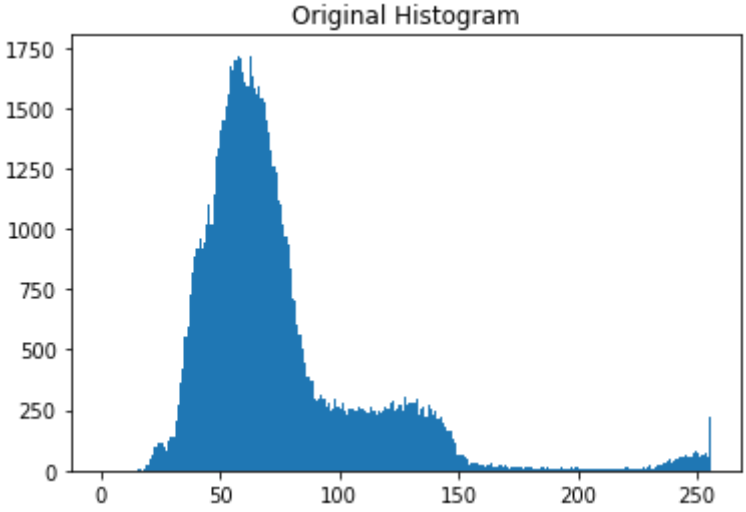
```
1 def img_hist(img_path):
2   img = cv2.imread(img_path,0)
3   equ = cv2.equalizeHist(img)
4
5   h1 = plt.hist(img.flatten(),256,[0,256])
6   plt.title("Original Histogram")
7   plt.show(h1)
8
9   h2 = plt.hist(equ.flatten(),256,[0,256])
10  plt.title("Histogram Equalization")
11  plt.show(h2)
12
13  res = np.hstack((img,equ)) #stacking images side-by-side
14  plt.title("Original Image vs Histogram Equalized Image")
```

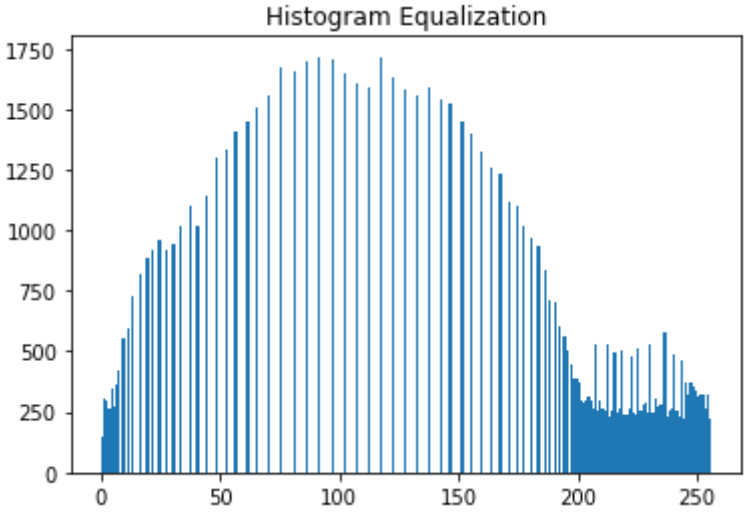
```
15   plt.imshow(res)
```

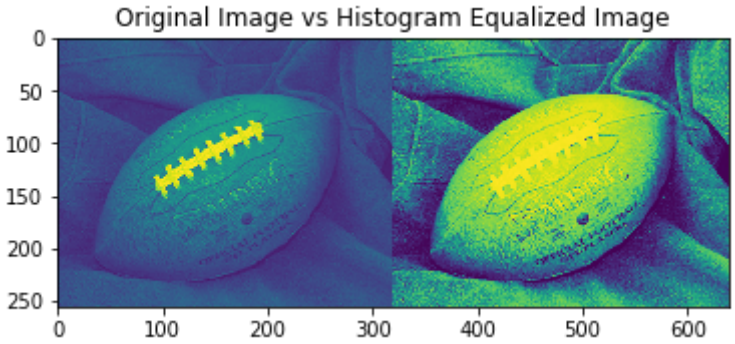
```
1 img_hist("/gantrycrane.png")
```



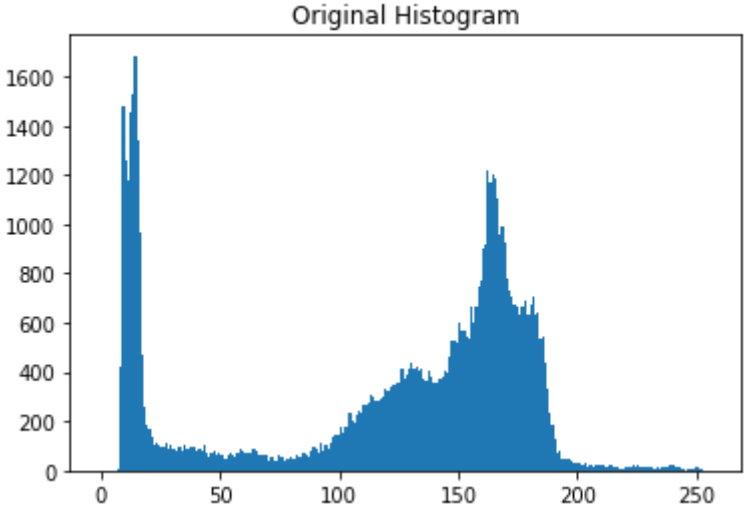

```
1 img_hist('/football.jpg')
```

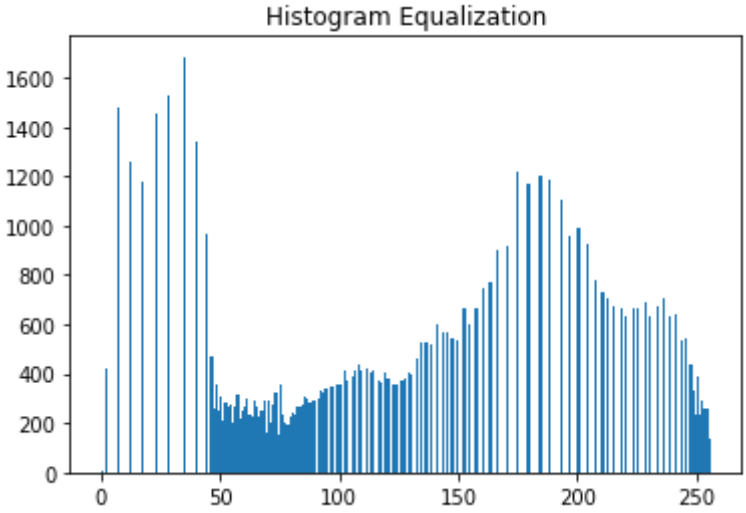


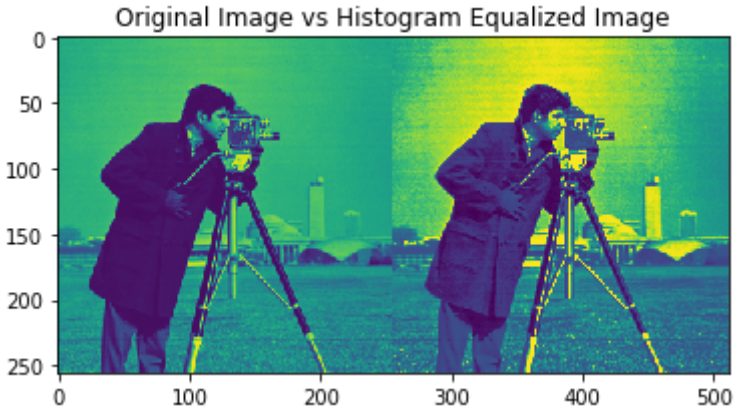





```
1 img_hist("/cameraman.tif")
```





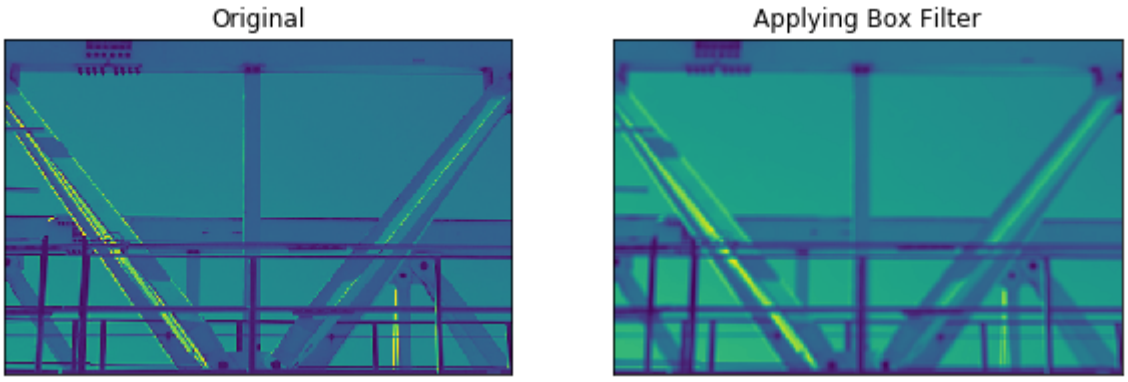


Box Filter

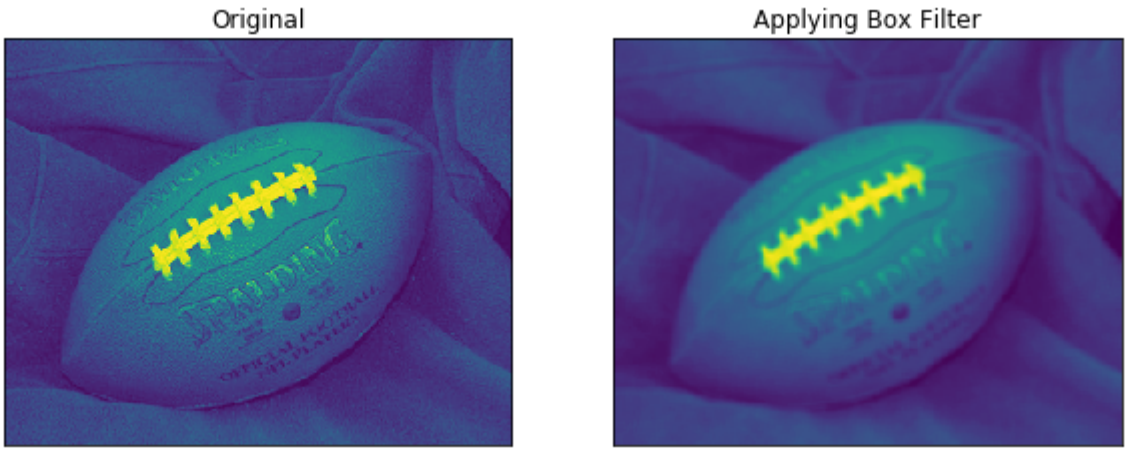
```
def display(img, filter_img, filter_text = ""):
  fig = plt.figure(figsize=(10,6))
  fig.add_subplot(1,2,1,xticks=[],yticks=[])
  plt.title('Original')
  plt.imshow(img)

  fig.add_subplot(1,2,2,xticks=[],yticks=[])
  plt.title(filter_text)
  plt.imshow(filter_img)
```

```
1 img1 = cv2.imread("/gantrycrane.png",0)
2 img2 = cv2.imread("/football.jpg",0)
3 img3 = cv2.imread("/cameraman.tif",0)
```

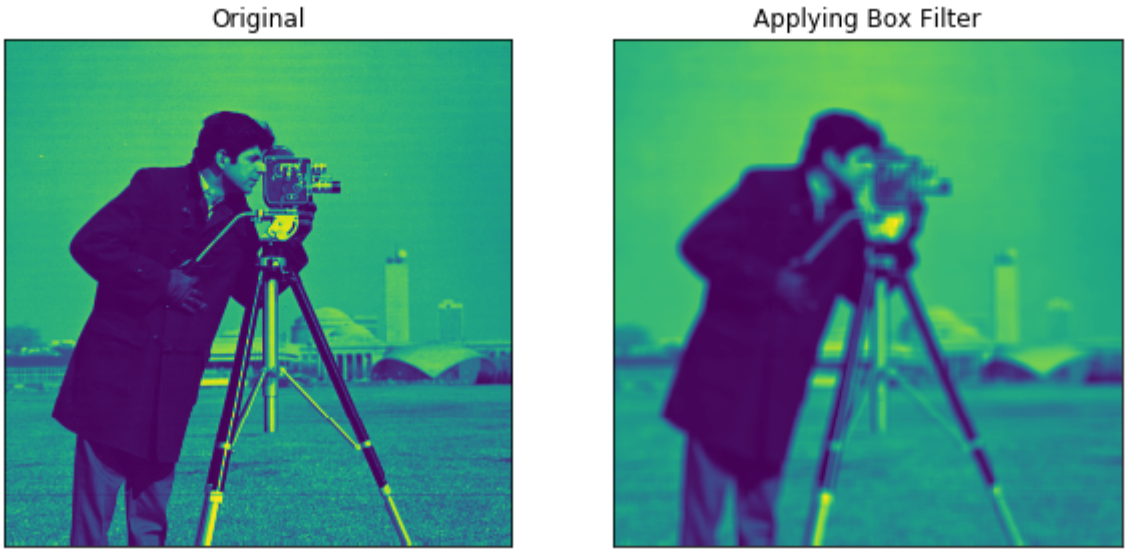
```
1 box_filter_1 = cv2.boxFilter(img1, 0, (5,5))
2 display(img1, box_filter_1, "Applying Box Filter")
```




```
1 box_filter_2 = cv2.boxFilter(img2, 0, (5,5))
2 display(img2, box_filter_2,"Applying Box Filter")
```

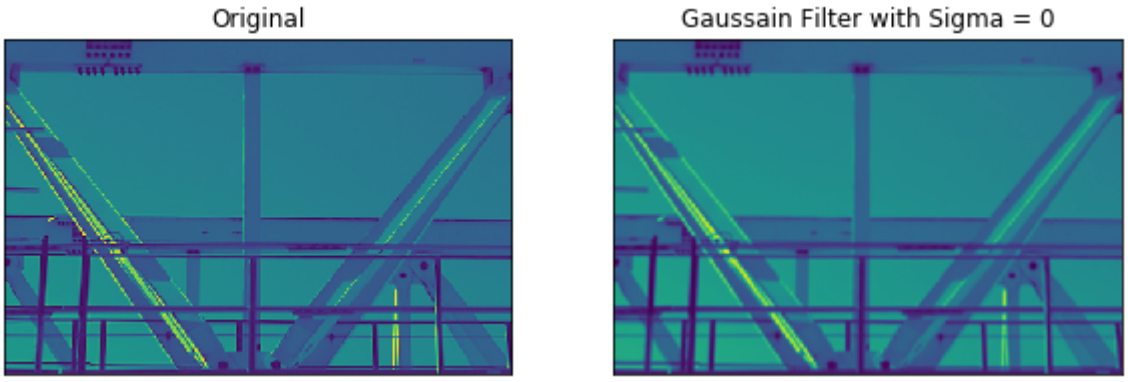



```
1 box_filter_3 = cv2.boxFilter(img3, 0, (5,5))
2 display(img3, box_filter_3,"Applying Box Filter")
```

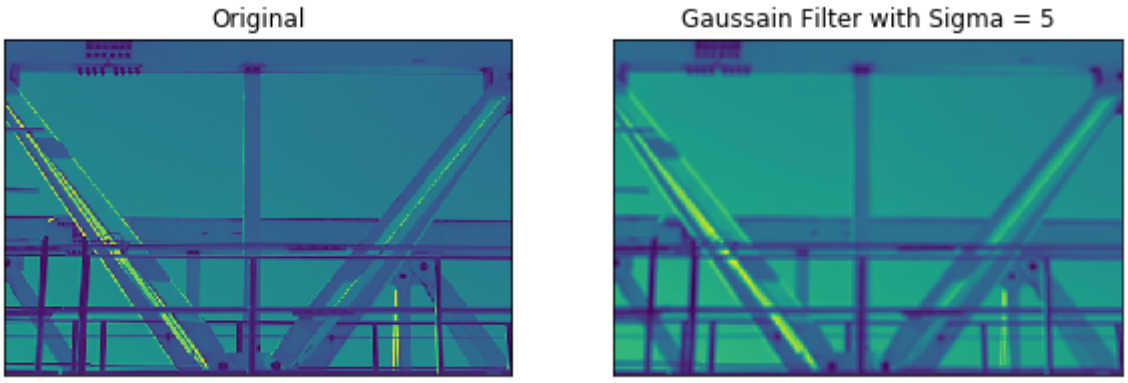


Gaussian Filter

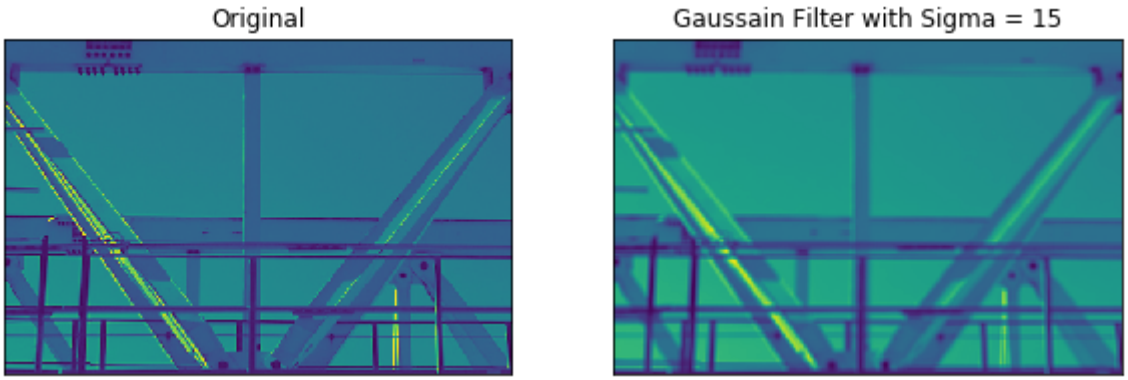
```
1 g1 = cv2.GaussianBlur(img1,(5,5),0)
2 display(img1, g1, "Gaussain Filter with Sigma = 0")
```



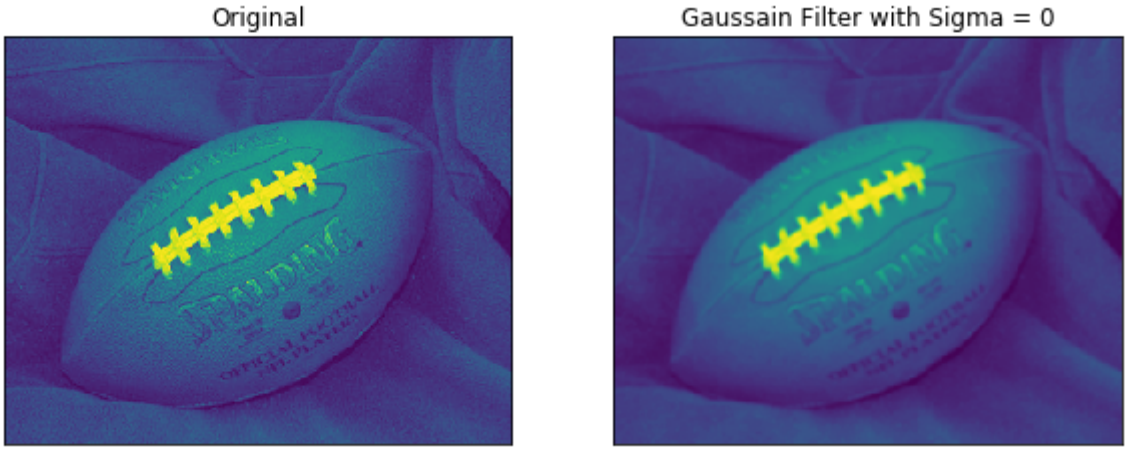

```
1 g1 = cv2.GaussianBlur(img1,(5,5),5)
2 display(img1, g1, "Gaussain Filter with Sigma = 5")
```



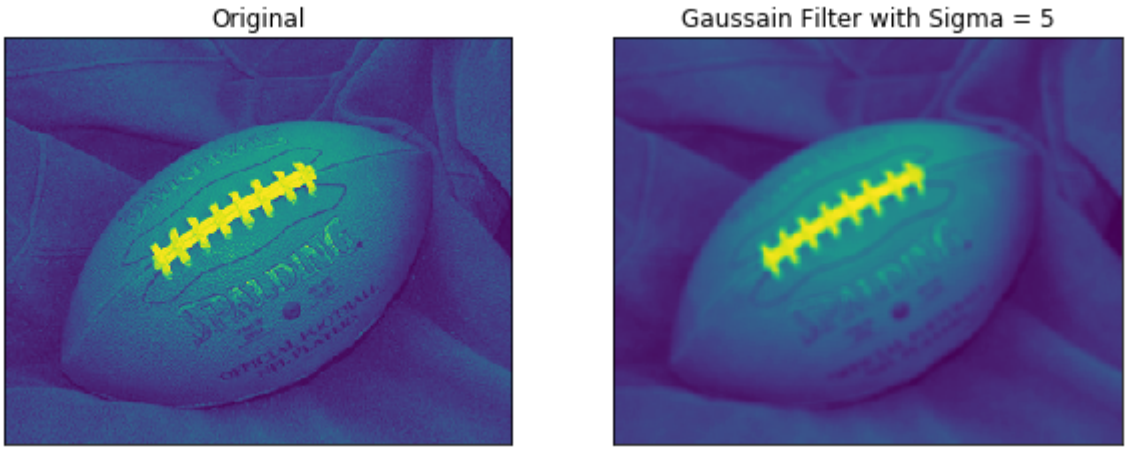

```
1 g1 = cv2.GaussianBlur(img1,(5,5),15)
2 display(img1, g1, "Gaussain Filter with Sigma = 15")
```



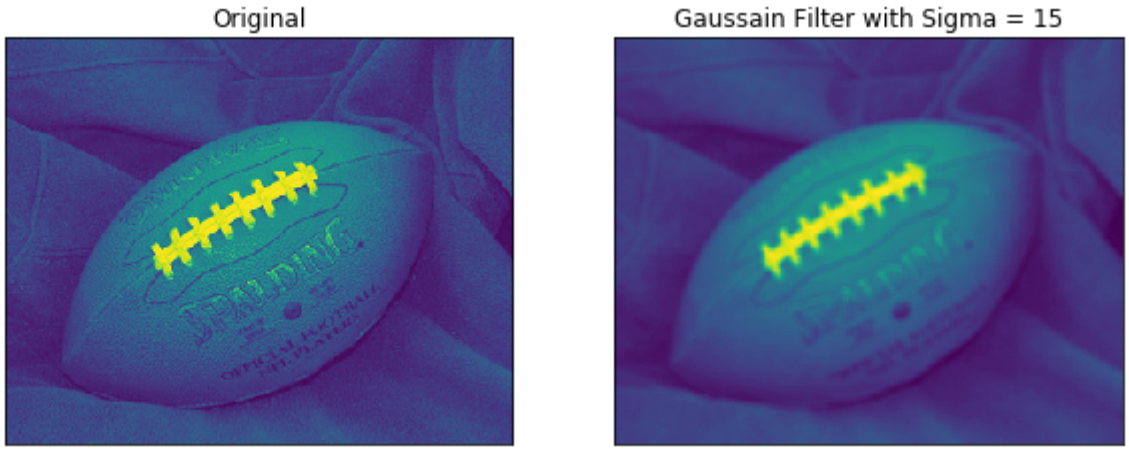

```
1 g2 = cv2.GaussianBlur(img2,(5,5),0)
2 display(img2, g2, "Gaussain Filter with Sigma = 0")
```




```
1 g2 = cv2.GaussianBlur(img2,(5,5),5)
2 display(img2, g2, "Gaussain Filter with Sigma = 5")
```




```
1 g2 = cv2.GaussianBlur(img2,(5,5),15)
2 display(img2, g2, "Gaussain Filter with Sigma = 15")
```




```
1 g3 = cv2.GaussianBlur(img3,(5,5),0)
2 display(img3, g3, "Gaussain Filter with Sigma = 0")
```


```
1 g3 = cv2.GaussianBlur(img3,(5,5),5)
2 display(img3, g3, "Gaussain Filter with Sigma = 5")
```

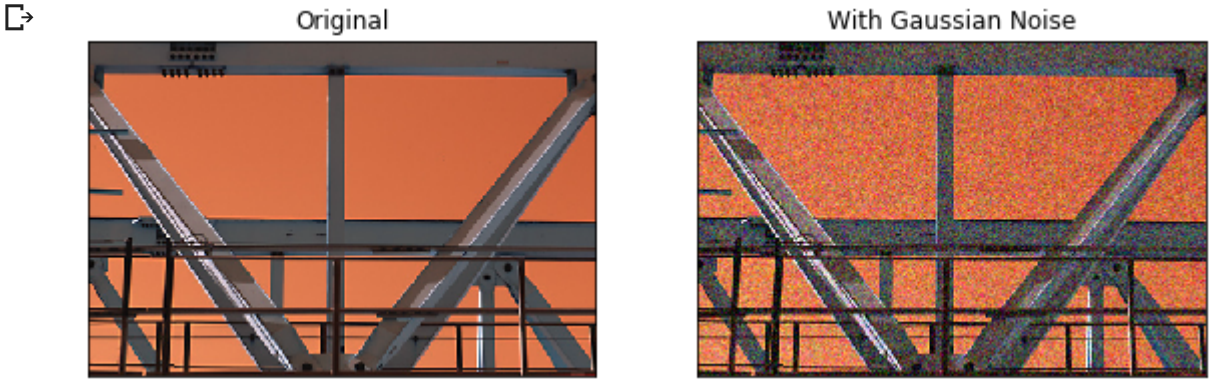

```
1 g3 = cv2.GaussianBlur(img3,(5,5),15)
```

```
2 display(img3, g3, "Gaussain Filter with Sigma = 15")
```

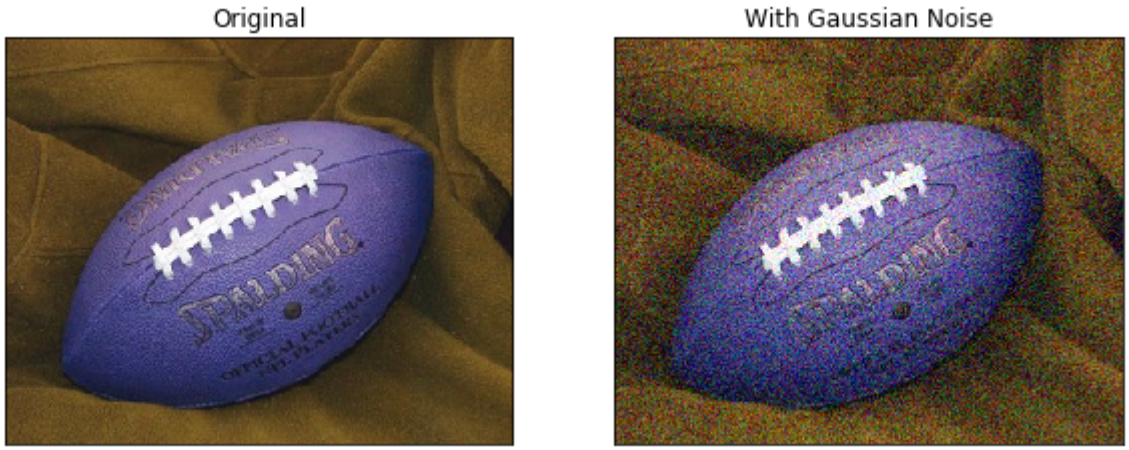
Add Gaussian Salt and Pepper Noise

```
1 from skimage.util import random_noise
```

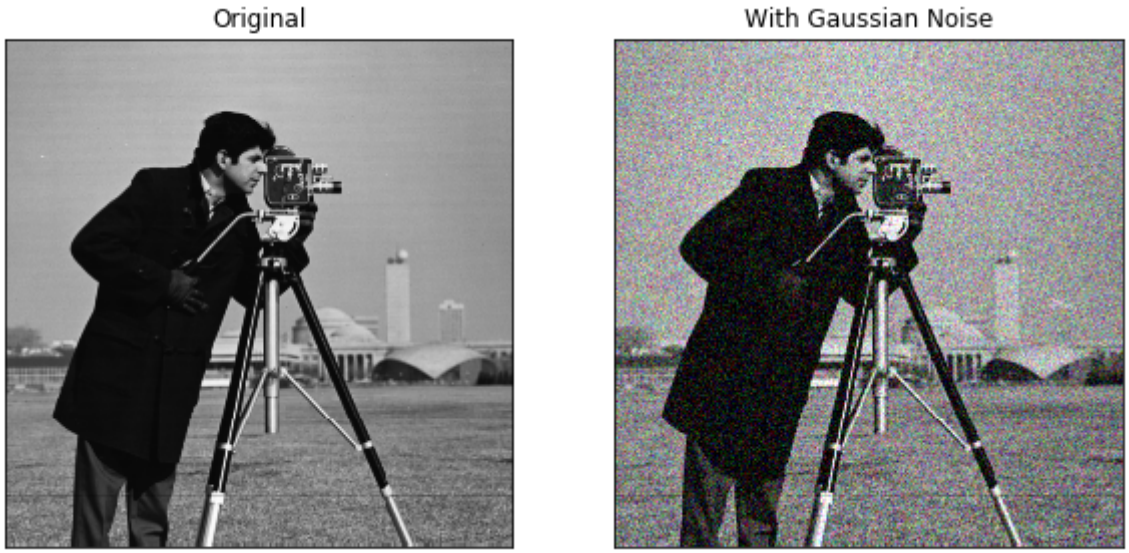
```
1 g_noise1 = random_noise(c_img1, mode='gaussian')
2 display(c_img1, g_noise1, "With Gaussian Noise")
```



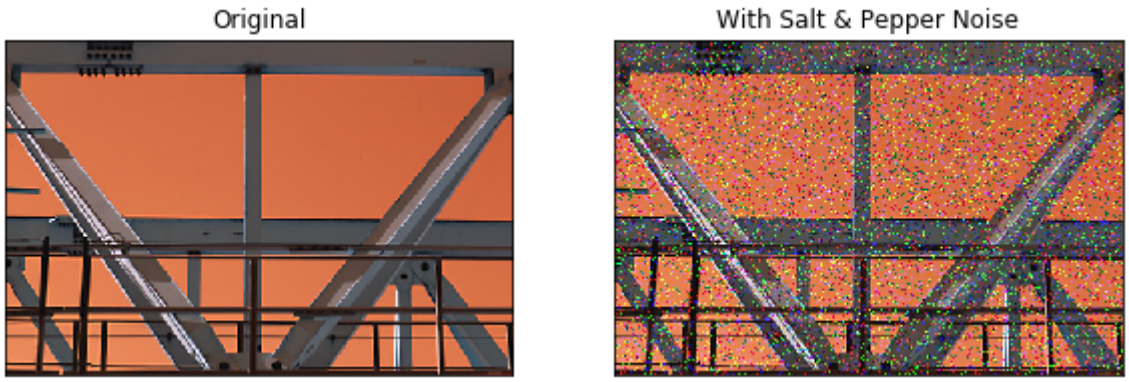

```
1 g_noise2 = random_noise(c_img2, mode='gaussian')
2 display(c_img2, g_noise2, "With Gaussian Noise")
```



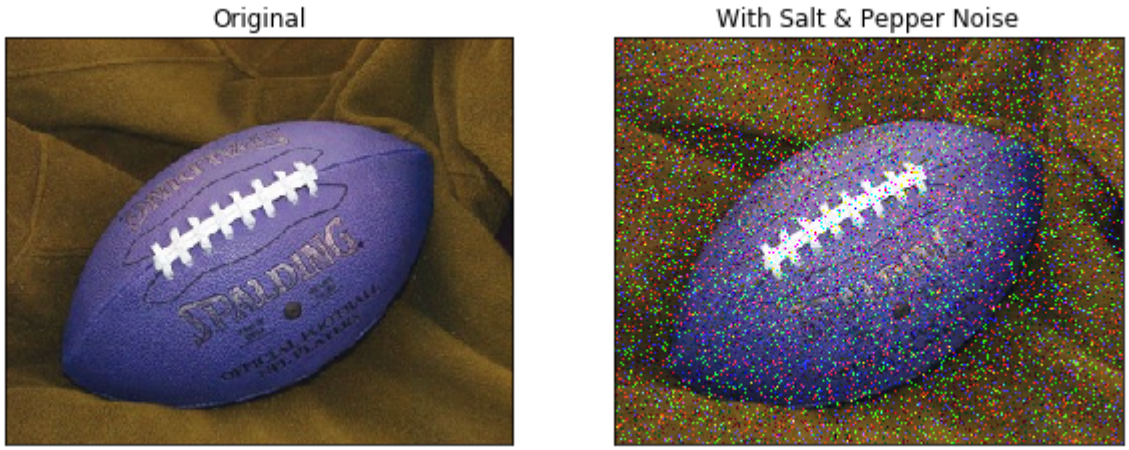

```
1 g_noise3 = random_noise(c_img3, mode='gaussian')
2 display(c_img3, g_noise3, "With Gaussian Noise")
```



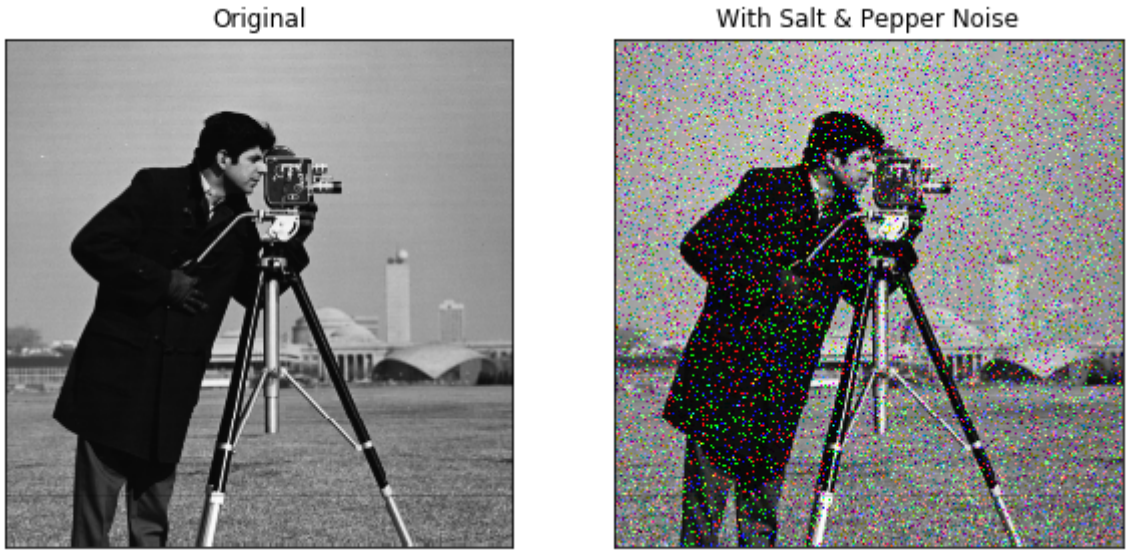

```
1 sp_1 = random_noise(c_img1, mode='s&p', amount = 0.1)
2 display(c_img1, sp_1, "With Salt & Pepper Noise")
```




```
1 sp_1 = random_noise(c_img2, mode='s&p', amount = 0.1)
2 display(c_img2, sp_1, "With Salt & Pepper Noise")
```




```
1 sp_1 = random_noise(c_img3, mode='s&p', amount = 0.1)
2 display(c_img3, sp_1, "With Salt & Pepper Noise")
```



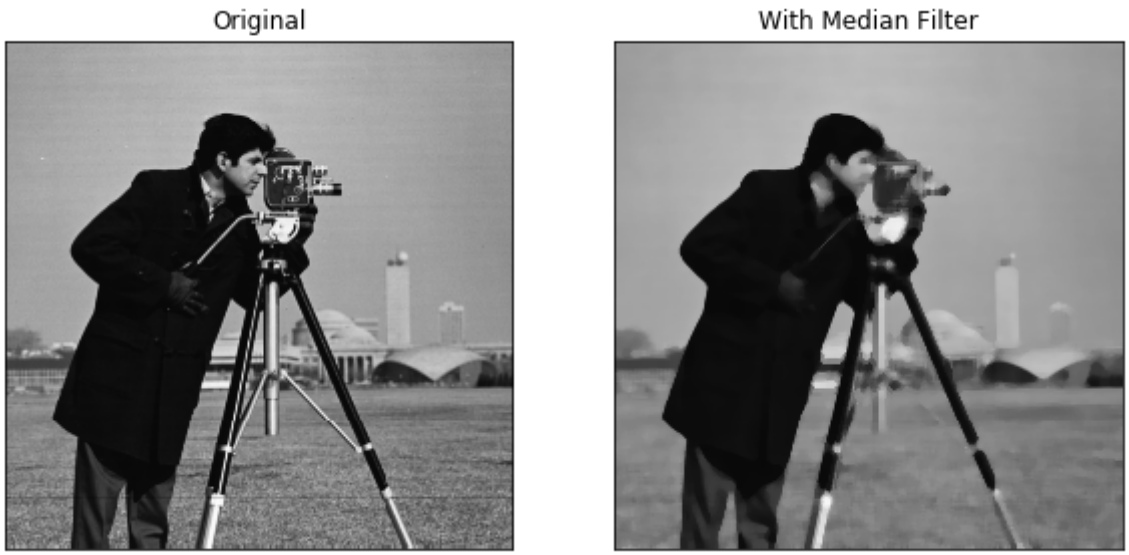
Median Filter

```
1 m_blur1 = cv2.medianBlur(c_img1,5)
2 display(c_img1, m_blur1, "With Median Filter")
```


```
1 m_blur1 = cv2.medianBlur(c_img2,5)
```

```
2 display(c_img2, m_blur1, "With Median Filter")
```


```
1 m_blur1 = cv2.medianBlur(c_img3,5)
2 display(c_img3, m_blur1, "With Median Filter")
```



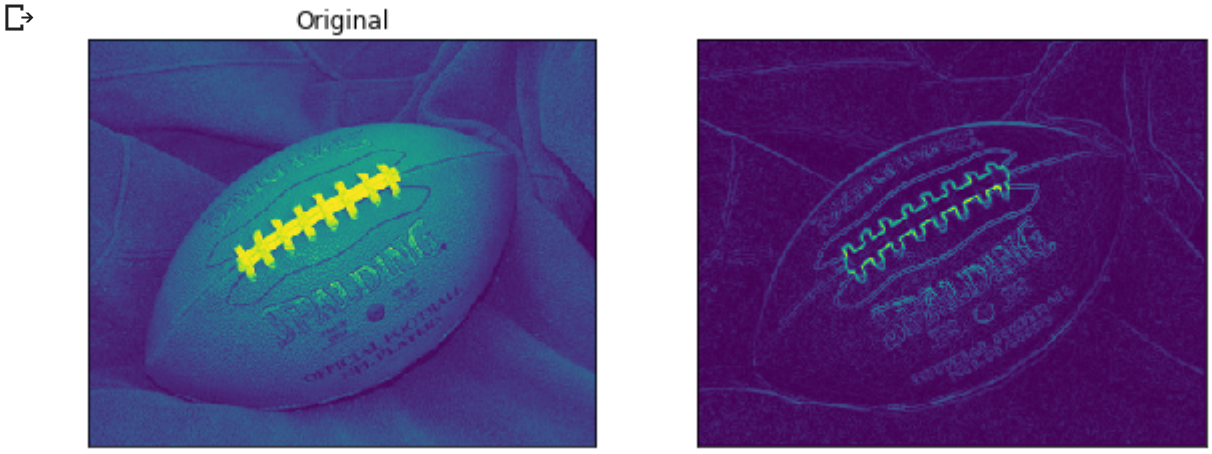
Sobel

```
1 from skimage import filters
```

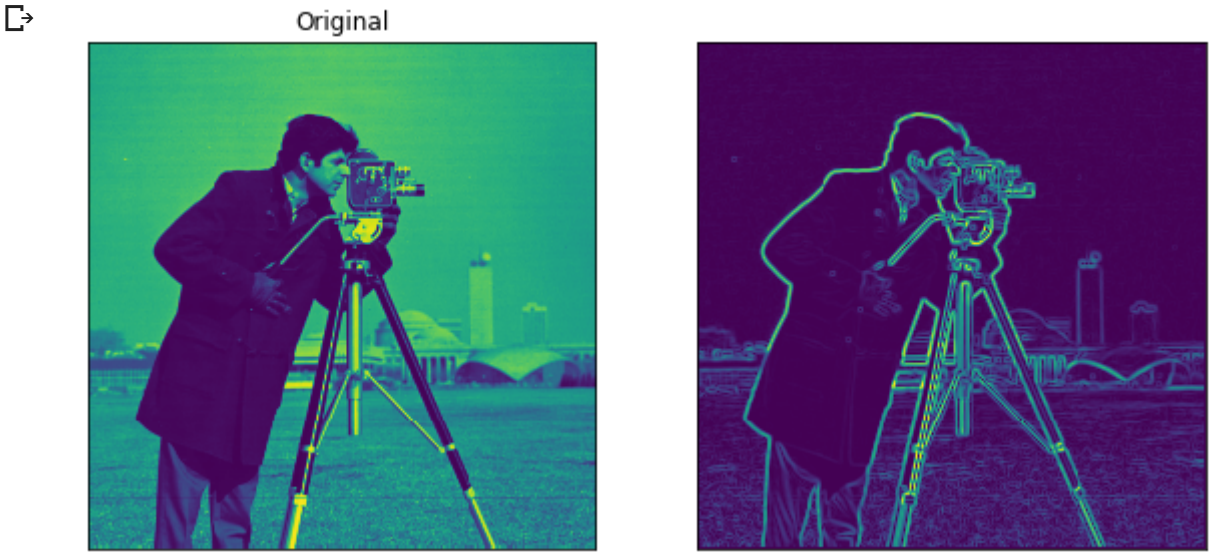
```
1 sobel1 = filters.sobel(img1)
2 display(img1, sobel1)
```

Original

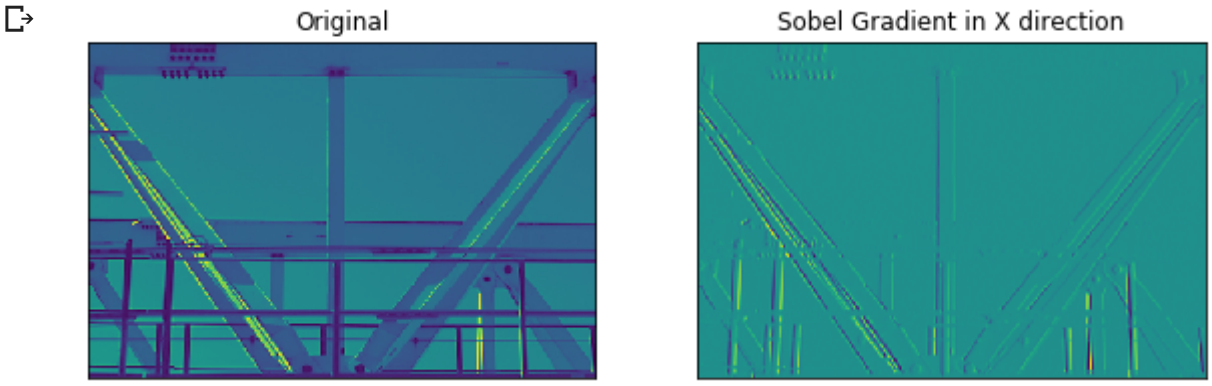
```
1 sobel1 = filters.sobel(img2)
2 display(img2, sobel1)
```



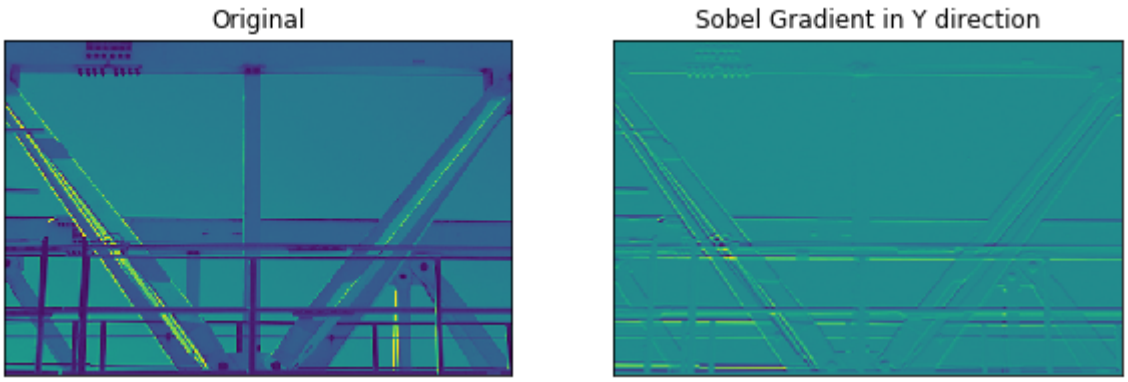

```
1 sobel1 = filters.sobel(img3)
2 display(img3, sobel1)
```



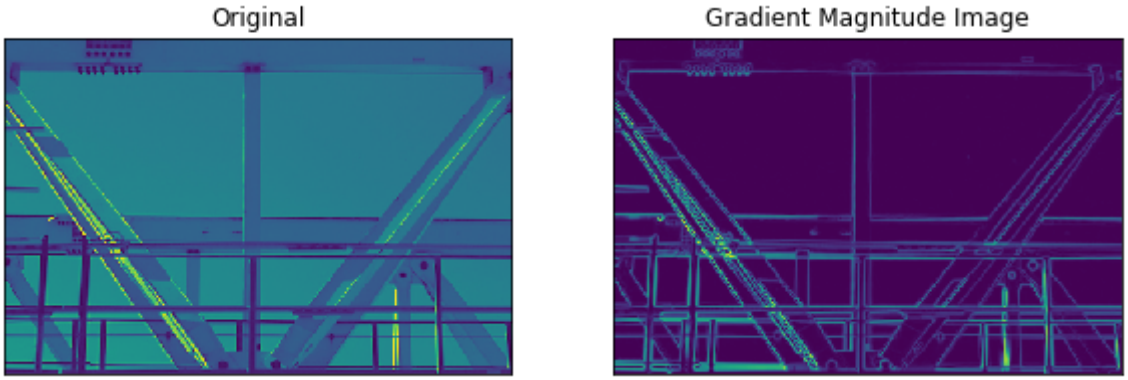

```
1 dX = cv2.Sobel(img1, cv2.CV_32F, 1, 0, (3,3))
2 display(img1, dX, "Sobel Gradient in X direction")
```




```
1 dY = cv2.Sobel(img1, cv2.CV_32F, 0, 1, (3,3))
2 display(img1, dY, "Sobel Gradient in Y direction")
```




```
1 mag= np.sqrt(dX*dX+dY*dY)
2 display(img1, mag, "Gradient Magnitude Image")
```



Canny Edge Detection

```
1 canny1 = cv2.Canny(img1, 100, 200)
2 display(img1, canny1, "Canny Edge Detection")
```