Name : Rishikeh Vadodaria

Roll No. : C114

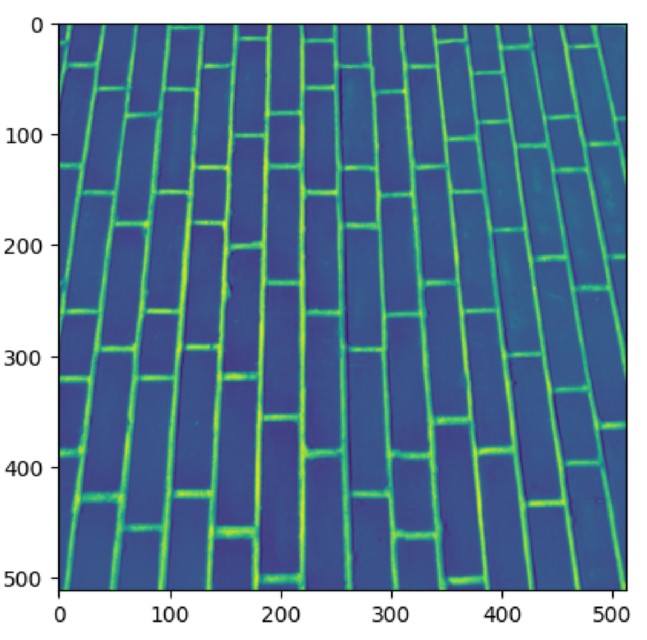
Aim : To analyze and enhance edge detection in images by applying Sobel filters, calculating gradient properties, and studying the effects of varying filter sizes.

In [25]:
```
import cv2 as cv
import matplotlib.pyplot as plt
from skimage import data
import numpy as np
```

In [26]:
```
image = data.brick()
```

In [27]:
```
plt.imshow(image)
```

Out[27]:
```
<matplotlib.image.AxesImage at 0x2b5a52aa290>
```



In [28]:
```
[rw,col] = image.shape
```

In [29]:
```
rw
```

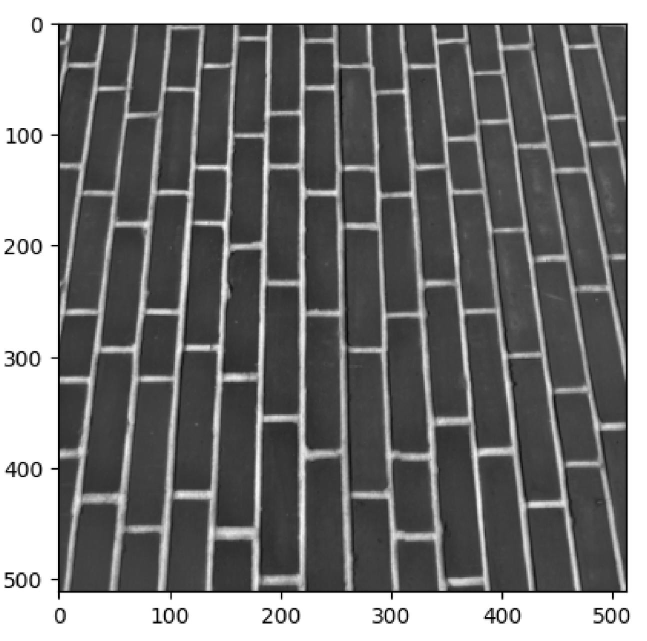
Out[29]: 512

In [30]:
```
col
```

Out[30]: 512

In [31]:
```
plt.imshow (image,cmap='gray')
```

Out[31]:
```
<matplotlib.image.AxesImage at 0x2b5a50efdd0>
```



In [32]:
```
gx = cv.Sobel(image, ddepth=cv.CV_64F, dx=1, dy=0, ksize=31)
```

In [33]:
```
gy = cv.Sobel(image,ddepth = cv.CV_64F,dx=0, dy=1, ksize=31)
```

In [34]:
```
gx
```

Out[34]:
```
array([[0.00000000e+00, 2.84951314e+17, 7.19846925e+17, ...,
        6.35484085e+17, 3.43303037e+17, 0.00000000e+00],
       [0.00000000e+00, 2.91446217e+17, 7.33232238e+17, ...,
        6.70436694e+17, 3.65148336e+17, 0.00000000e+00],
       [0.00000000e+00, 3.09986742e+17, 7.71259690e+17, ...,
        7.60170615e+17, 4.21524334e+17, 0.00000000e+00],
       ...,
       [0.00000000e+00, 1.66250568e+18, 3.14234324e+18, ...,
        4.28277273e+18, 2.26005346e+18, 0.00000000e+00],
       [0.00000000e+00, 1.69802695e+18, 3.19628811e+18, ...,
        4.41957927e+18, 2.34406919e+18, 0.00000000e+00],
       [0.00000000e+00, 1.71066192e+18, 3.21548896e+18, ...,
        4.46357533e+18, 2.37121804e+18, 0.00000000e+00]])
```

In [35]:
```
gy
```

Out[35]:
```
array([[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
         0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
       [-1.47647076e+16, -8.26980443e+15,  1.16104110e+16, ...,
        -1.81301072e+18, -1.75621281e+18, -1.73436751e+18],
       [-1.92981742e+16, -7.25255253e+15,  2.94352089e+16, ...,
        -3.28204759e+18, -3.19273558e+18, -3.15820488e+18],
       ...,
       [ 2.29283508e+17,  2.52169807e+17,  3.09800123e+17, ...,
         7.03120790e+16,  2.19989437e+17,  2.76856313e+17],
       [ 1.28349546e+17,  1.40984517e+17,  1.72820339e+17, ...,
         1.13467558e+16,  8.24916660e+16,  1.09640517e+17],
       [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
         0.00000000e+00,  0.00000000e+00,  0.00000000e+00]])
```

In [36]:
```
gx_cv2Abs = cv.convertScaleAbs(gx)
gy_cv2Abs = cv.convertScaleAbs(gy)
```

In [37]:
```
gx_cv2Abs
```

Out[37]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=uint8)
```

In [38]:
```
gy_cv2Abs
```

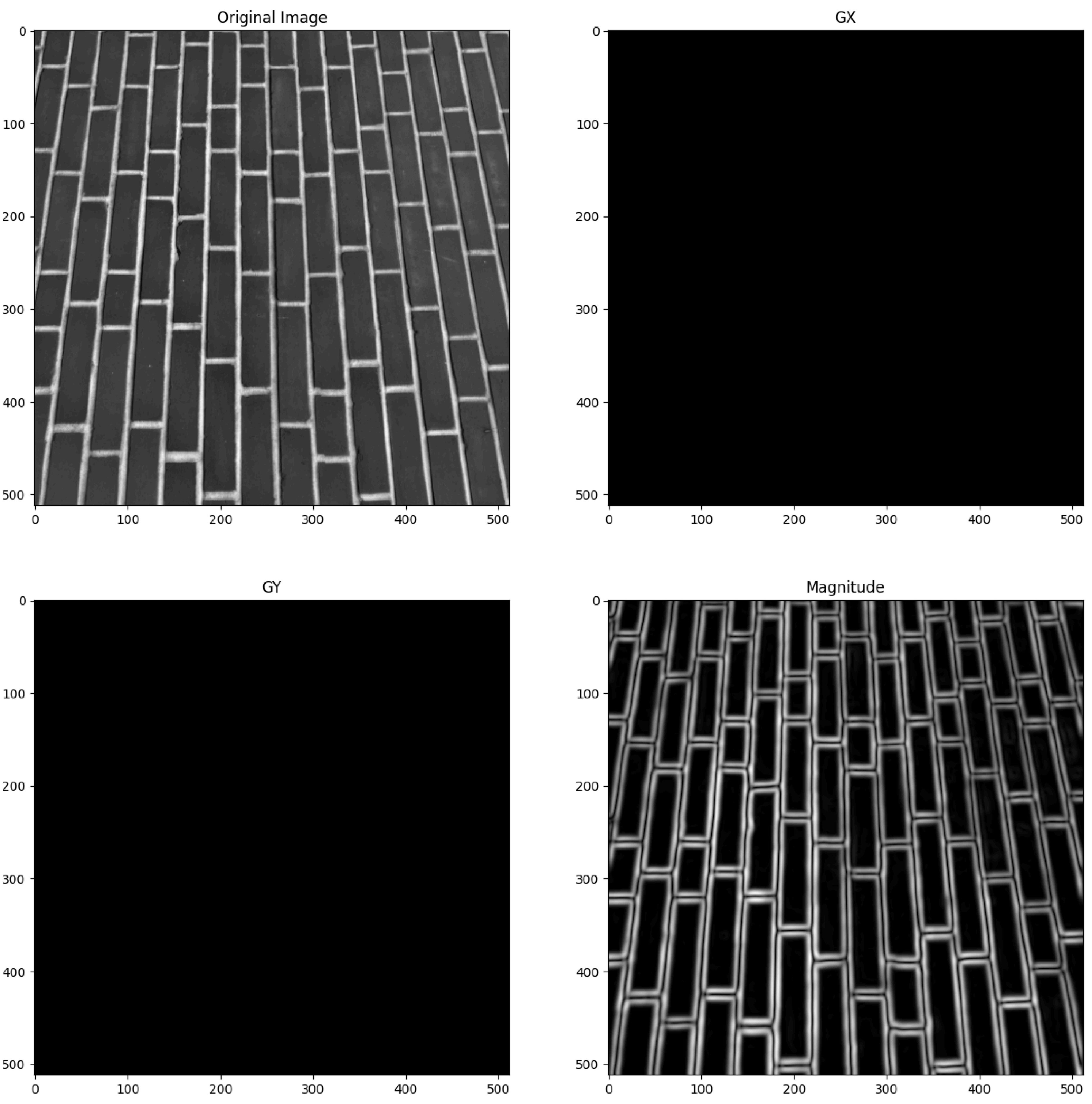
Out[38]:
```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=uint8)
```

In [39]:
```
g_mag = np.sqrt((gx)**2 + (gy)**2)
```

In [40]:
```
g_mag
```

Out[40]:
```
array([[0.00000000e+00, 2.84951314e+17, 7.19846925e+17, ...,
        6.35484085e+17, 3.43303037e+17, 0.00000000e+00],
       [1.47647076e+16, 2.91563521e+17, 7.33324155e+17, ...,
        1.93300109e+18, 1.79377165e+18, 1.73436751e+18],
       [1.92981742e+16, 3.10071571e+17, 7.71821184e+17, ...,
        3.36893095e+18, 3.22044147e+18, 3.15820488e+18],
       ...,
       [2.29283508e+17, 1.68152156e+18, 3.15757774e+18, ...,
        4.28334986e+18, 2.27073490e+18, 2.76856313e+17],
       [1.28349546e+17, 1.70386976e+18, 3.20095682e+18, ...,
        4.41959383e+18, 2.34552025e+18, 1.09640517e+17],
       [0.00000000e+00, 1.71066192e+18, 3.21548896e+18, ...,
        4.46357533e+18, 2.37121804e+18, 0.00000000e+00]])
```

In [41]:
```
plt.figure(figsize=(15, 15))

plt.subplot(2, 2, 1)
plt.imshow(image, cmap='gray')
plt.title('Original Image')

plt.subplot(2, 2, 2)
plt.imshow(gx_cv2Abs, cmap='gray')
plt.title('GX')

plt.subplot(2, 2, 3)
plt.imshow(gy_cv2Abs, cmap='gray')
plt.title('GY')

plt.subplot(2, 2, 4)
plt.imshow(g_mag, cmap='gray')
plt.title('Magnitude')

plt.show()
```



In [42]:
```
angle = np.arctan2(gy, gx) * (180/np.pi) % 180
```

In [43]:
```
angle
```

Out[43]:
```
array([[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [9.00000000e+01, 1.78374665e+02, 9.07177734e-01, ...,
        1.10293986e+02, 1.01745483e+02, 9.00000000e+01],
       [9.00000000e+01, 1.78659733e+02, 2.18563881e+00, ...,
        1.03040603e+02, 9.75210385e+01, 9.00000000e+01],
       ...,
       [9.00000000e+01, 8.62491297e+00, 5.63053170e+00, ...,
        9.40564383e-01, 5.55955129e+00, 9.00000000e+01],
       [9.00000000e+01, 4.74629226e+00, 3.09491615e+00, ...,
        1.47099928e-01, 2.01550135e+00, 9.00000000e+01],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00]])
```

In [44]:
```
angle1D = np.reshape(angle, (rw*col,1))
```

In [45]:
```
angle1D
```

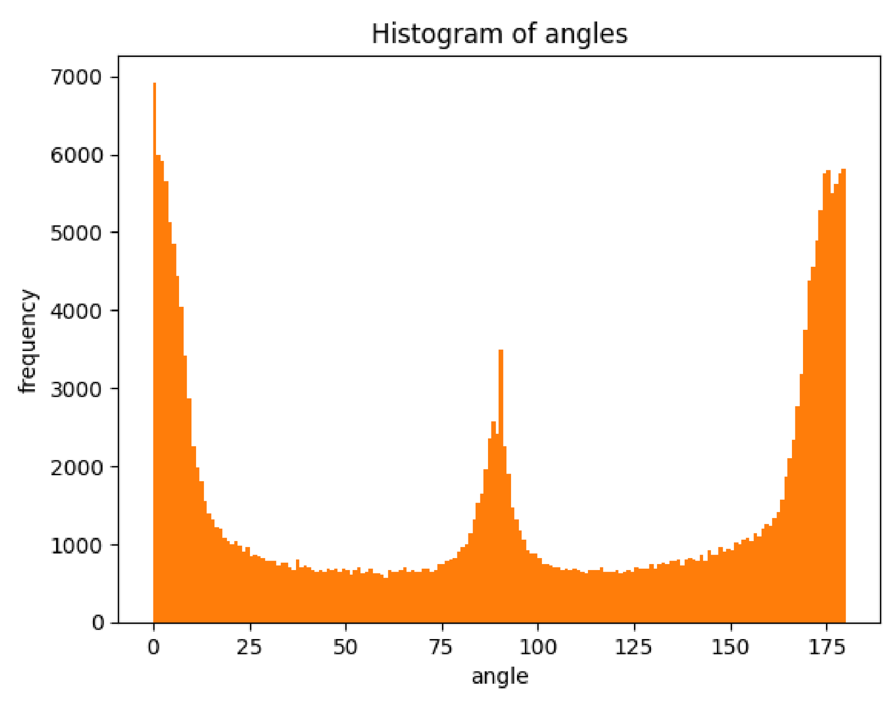
Out[45]:
```
array([[0.],
       [0.],
       [0.],
       ...,
       [0.],
       [0.],
       [0.]])
```

In [46]:
```
angle1D.shape
```

Out[46]: (262144, 1)

In [47]:
```
plt.hist(angle1D, bins = 180)
plt.hist(angle1D, bins = 180)
plt.xlabel('angle')
plt.ylabel('frequency')
plt.title('Histogram of angles')
```

Out[47]:
```
Text(0.5, 1.0, 'Histogram of angles')
```



Conclusion:

The Sobel filter is utilized to determine the gradient along the X and Y directions, the magnitude of the gradient, and the orientation of the edges in an image. For a small image size, such as 3x3, the filter is effective in detecting small edges. In contrast, for larger image sizes, such as 15x15 or greater, the filter detects larger edges more effectively. Additionally, with a larger filter size, the angles of edge pixels are more uniformly distributed. For a 3x3 Sobel filter, the identified angles are typically limited to discrete values like 0°, 45°, 90°, 135°, and 180°. Therefore, it can be concluded that a smaller filter size is more suitable for detecting fine, small edges, while a larger filter size is better suited for detecting broader, larger edges.