In [2]:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

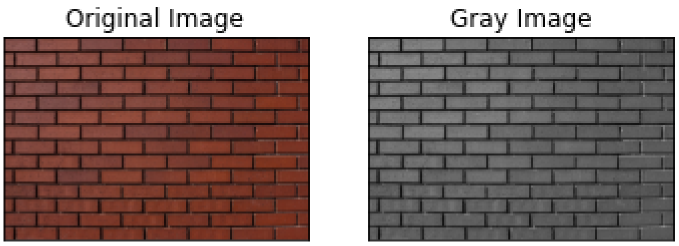
Question 1

1. a. Take an image of the wall of your house and apply a sobel filter and count the number of edge pixels in the edge map

b. Do the same operation for the image of a tree (capture your own image of the tree) and report your observation on the two images

In [3]:

```
img = cv2.imread("wall.jpg")
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

plt.subplot(121),plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title('Original Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(gray, cmap='gray')
plt.title('Gray Image'), plt.xticks([]), plt.yticks([])
plt.show()

```



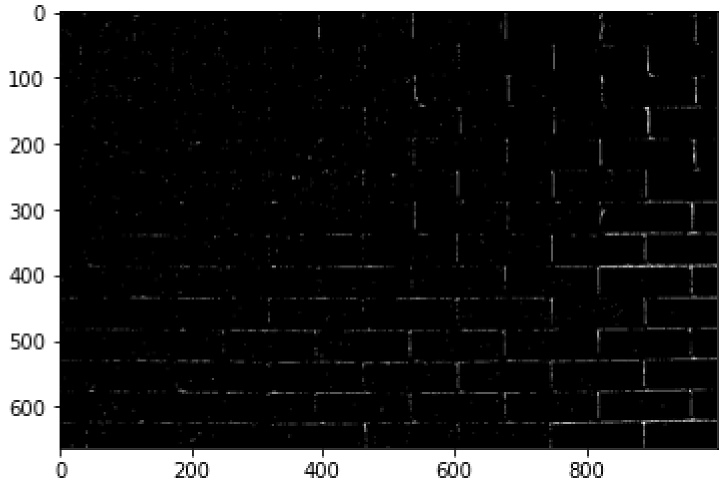
In [4]:

```
ret, binary = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
```

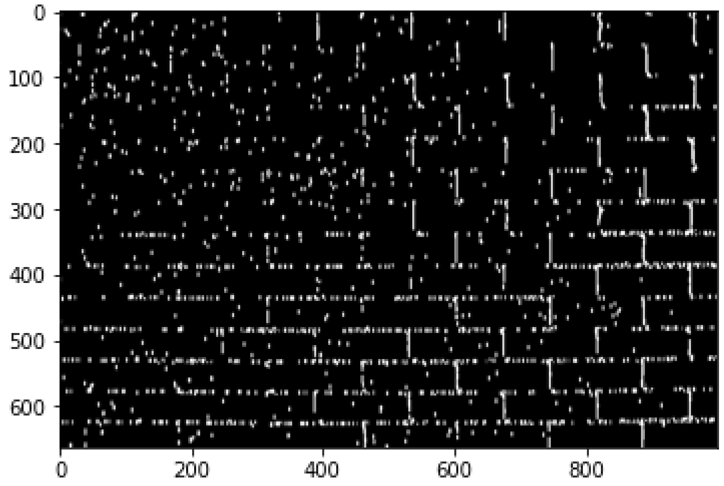
In [6]:

```

sobelx = cv2.Sobel(binary,-1,1,0,ksize=7)
sobely = cv2.Sobel(binary,-1,0,1,ksize=7)
sobelxy = cv2.Sobel(binary,-1,1,1,ksize=7)

print("BINARY")
plt.imshow(cv2.cvtColor(binary, cv2.COLOR_BGR2RGB))
plt.show()
print("SOBEL-X")
plt.imshow(sobelx,cmap="gray")
plt.show()
print("SOBEL-Y")
plt.imshow(sobely,cmap="gray")
plt.show()

print("SOBEL-XY")
plt.imshow(sobelxy,cmap="gray")
plt.show()

```

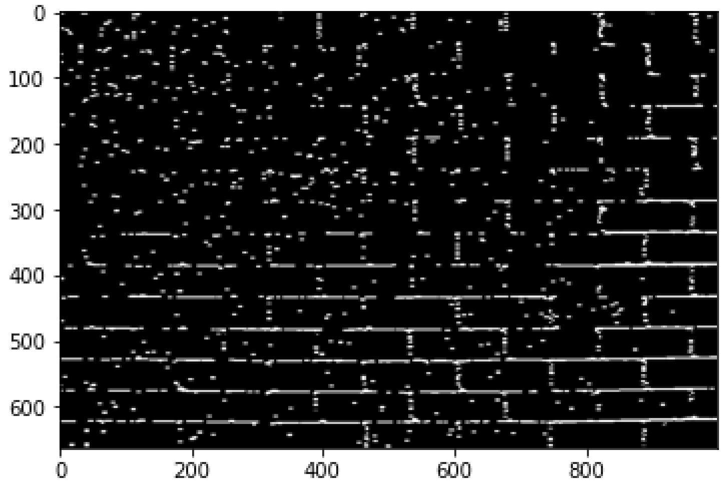
```
BINARY
```



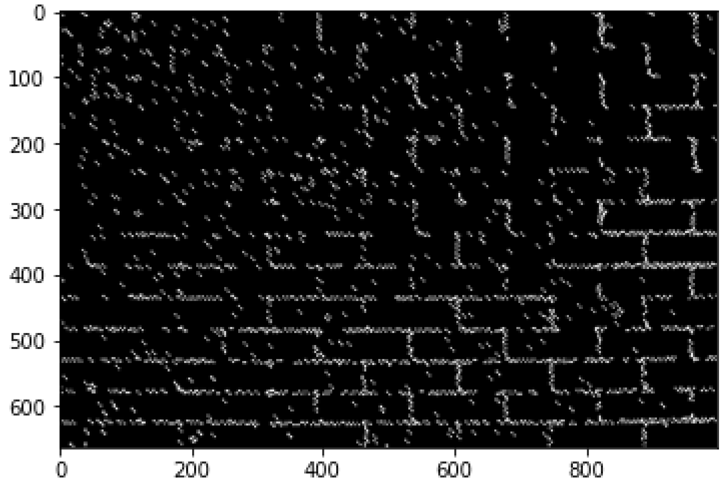
```
SOBEL-X
```



```
SOBEL-Y
```



SOBEL-XY



In [7]:

```

#binary.shape
```

In [8]:

```
edges_sobelx=0
for i in range(sobelx.shape[0]):
    for j in range(sobelx.shape[1]):
        if (sobelx[i,j]>0):
            #print(sobelx[i][j])
            edges_sobelx+=1
print("Sobelx shape :{}".format(sobelx.shape))
print("Edge pixels :{} out of {}".format(edges_sobelx,sobelx.size))

```

```
Sobelx shape :(663, 1000)
Edge pixels :37340 out of 663000
```

In [9]:

```
edges_sobely=0
for i in range(sobely.shape[0]):
    for j in range(sobely.shape[1]):
        if (sobely[i,j]>0):
            edges_sobely+=1
print("Sobely shape :{}".format(sobelx.shape))
print("Edge pixels :{} out of {}".format(edges_sobely,sobely.size))
```

```
Sobely shape :(663, 1000)
Edge pixels :37596 out of 663000
```

In [10]:

```
edges_sobelxy=0
for i in range(sobelxy.shape[0]):
    for j in range(sobelxy.shape[1]):
        if (sobelxy[i,j]>0):
            edges_sobelxy+=1
print("Sobelxy shape :{}".format(sobelxy.shape))
print("Edge pixels :{} out of {}".format(edges_sobelxy,sobelxy.size))
```

```
Sobelxy shape :(663, 1000)
Edge pixels :32621 out of 663000
```

Inference

the Sobel image in the x-direction predominantly identifies vertical edges

And that in the y-direction identifies horizontal edges

Sobel with grayscale input

In [109]:

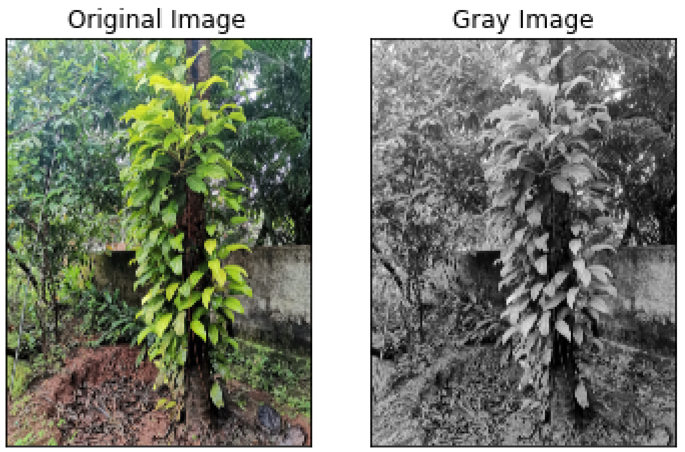
```
sobelx = cv2.Sobel(gray,-1,1,0,ksize=5)
sobely = cv2.Sobel(gray,-1,0,1,ksize=5)

print("GRAY")
plt.imshow(cv2.cvtColor(gray, cv2.COLOR_BGR2RGB))
plt.show()
print("SOBEL-X")
plt.imshow(sobelx,cmap="gray")
plt.show()
print("SOBEL-Y")
plt.imshow(sobely,cmap="gray")
plt.show()



```

...

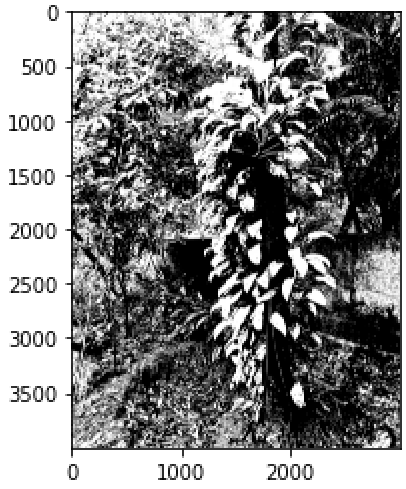
TREE IMAGE

In [110]:

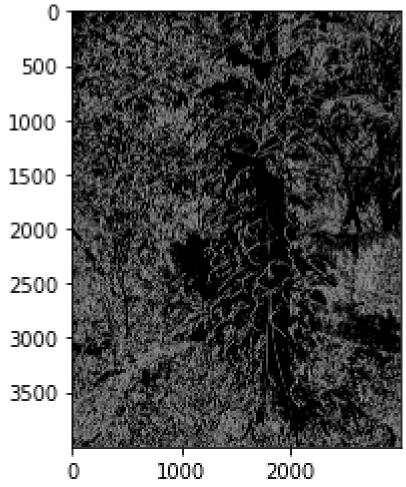
```
img = cv2.imread("home_tree.jpg")
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

plt.subplot(121),plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title('Original Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(gray, cmap='gray')
plt.title('Gray Image'), plt.xticks([]), plt.yticks([])
plt.show()

ret, binary = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)

sobelx = cv2.Sobel(binary,-1,1,0,ksize=5)
sobely = cv2.Sobel(binary,-1,0,1,ksize=5)
sobelxy = cv2.Sobel(binary,-1,1,1,ksize=5)
print("BINARY")
plt.imshow(cv2.cvtColor(binary, cv2.COLOR_BGR2RGB))
plt.show()
print("SOBEL-X")
plt.imshow(sobelx,cmap="gray")
plt.show()
print("SOBEL-Y")
plt.imshow(sobely,cmap="gray")
plt.show()
print("SOBEL-XY")
plt.imshow(sobelxy,cmap="gray")
plt.show()


edges_sobelx=0
for i in range(sobelx.shape[0]):
    for j in range(sobelx.shape[1]):
        if (sobelx[i,j]>0):
            edges_sobelx+=1
print("Sobelx shape :{}".format(sobelx.shape))
print("Edge pixels :{} out of {}".format(edges_sobelx,sobelx.size))

edges_sobely=0
for i in range(sobely.shape[0]):
    for j in range(sobely.shape[1]):
        if (sobely[i,j]>0):
            edges_sobely+=1
print("Sobely shape :{}".format(sobelx.shape))
print("Edge pixels :{} out of {}".format(edges_sobely,sobely.size))

edges_sobelxy=0
for i in range(sobelxy.shape[0]):
    for j in range(sobelxy.shape[1]):
        if (sobelxy[i,j]>0):
            #print(sobelx[i][j])
            edges_sobelxy+=1
print("Sobelxy shape :{}".format(sobelxy.shape))
print("Edge pixels :{} out of {}".format(edges_sobelxy,sobelxy.size))

```



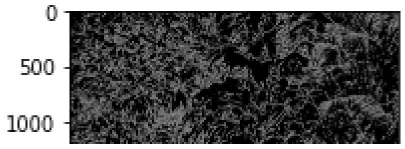
```
BINARY
```



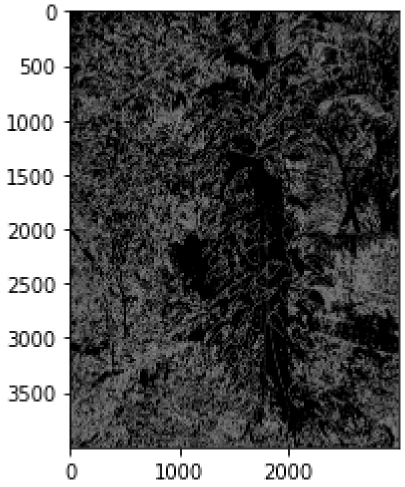
```
SOBEL-X
```



```
SOBEL-Y
```



```
SOBEL-XY
```



```
Sobelx shape :(4000, 3000)
Edge pixels :2215643 out of 12000000
Sobely shape :(4000, 3000)
Edge pixels :2202679 out of 12000000
Sobelxy shape :(4000, 3000)
Edge pixels :2023977 out of 12000000
```

In [ ]:

```

```

In [ ]:

```

```