

In [2]:

```
1 import cv2
2 import numpy as np
3 import matplotlib.pyplot as plt
4 %matplotlib inline
```

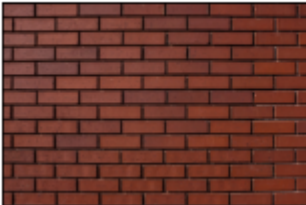
Question 1

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

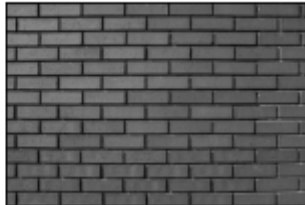
In [3]:

```
1 img = cv2.imread("wall.jpg")
2 gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
3
4 plt.subplot(121),plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
5 plt.title('Original Image'), plt.xticks([], plt.yticks([]))
6 plt.subplot(122),plt.imshow(gray, cmap='gray')
7 plt.title('Gray Image'), plt.xticks([], plt.yticks([]))
8 plt.show()
9
```

Original Image



Gray Image



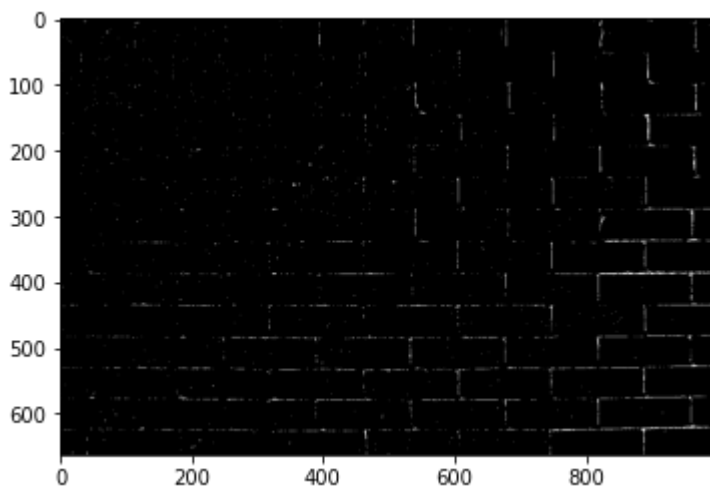
In [4]:

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

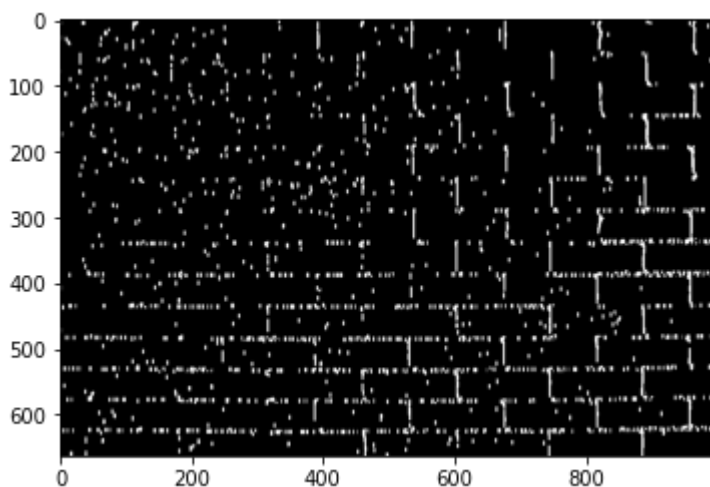
In [6]:

```
1
2 sobelx = cv2.Sobel(binary,-1,1,0,ksize=7)
3 sobely = cv2.Sobel(binary,-1,0,1,ksize=7)
4 sobelxy = cv2.Sobel(binary,-1,1,1,ksize=7)
5
6 print("BINARY")
7 plt.imshow(cv2.cvtColor(binary, cv2.COLOR_BGR2RGB))
8 plt.show()
9 print("SOBEL-X")
10 plt.imshow(sobelx,cmap="gray")
11 plt.show()
12 print("SOBEL-Y")
13 plt.imshow(sobely,cmap="gray")
14 plt.show()
15
16 print("SOBEL-XY")
17 plt.imshow(sobelxy,cmap="gray")
18 plt.show()
19
```

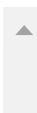
BINARY

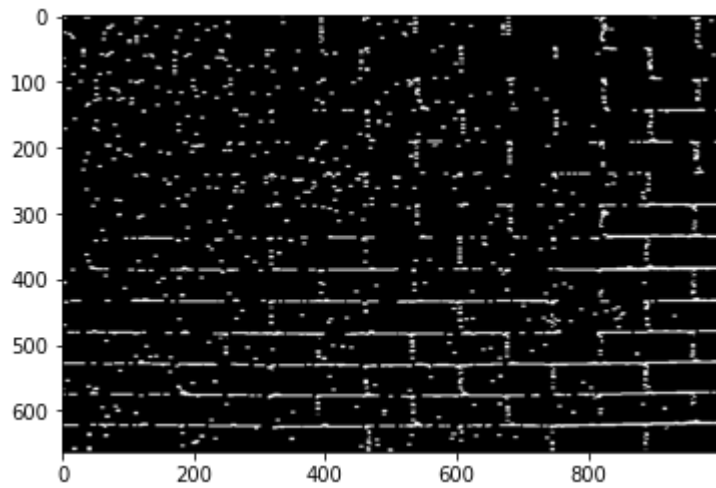


SOBEL-X

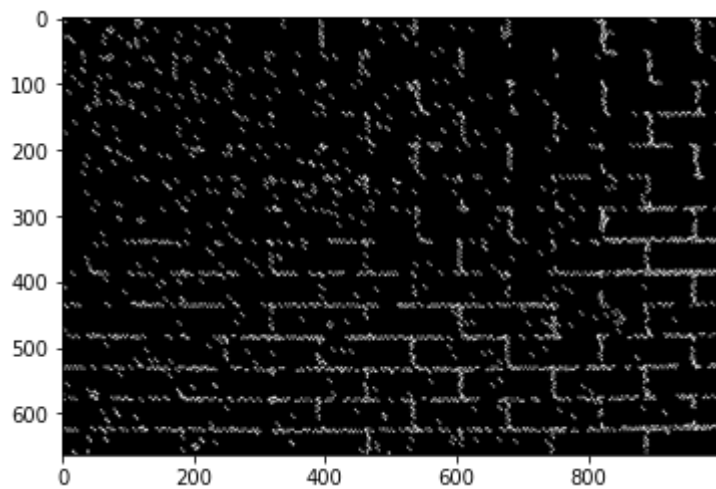


SOBEL-Y





SOBEL -XY



In [7]:

```
1
2 #binary.shape
```

In [8]:

```
1 edges_sobelx=0
2 for i in range(sobelx.shape[0]):
3     for j in range(sobelx.shape[1]):
4         if (sobelx[i,j]>0):
5             #print(sobelx[i][j])
6             edges_sobelx+=1
7 print("Sobelx shape :{}".format(sobelx.shape))
8 print("Edge pixels :{} out of {}".format(edges_sobelx,sobelx.size))
9
```

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

In [9]:

```

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

```

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

In [10]:

```

1 edges_sobelxy=0
2 for i in range(sobelxy.shape[0]):
3     for j in range(sobelxy.shape[1]):
4         if (sobelxy[i,j]>0):
5             edges_sobelxy+=1
6 print("Sobelxy shape :{}".format(sobelxy.shape))
7 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]:

```

1 sobelx = cv2.Sobel(gray,-1,1,0,ksize=5)
2 sobely = cv2.Sobel(gray,-1,0,1,ksize=5)
3
4 print("GRAY")
5 plt.imshow(cv2.cvtColor(gray, cv2.COLOR_BGR2RGB))
6 plt.show()
7 print("SOBEL-X")
8 plt.imshow(sobelx,cmap="gray")
9 plt.show()
10 print("SOBEL-Y")
11 plt.imshow(sobely,cmap="gray")
12 plt.show()
13
14
15

```

...

TREE IMAGE

In [110]:

```
1  img = cv2.imread("home_tree.jpg")
2  gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
3
4  plt.subplot(121),plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
5  plt.title('Original Image'), plt.xticks([], plt.yticks([]))
6  plt.subplot(122),plt.imshow(gray, cmap='gray')
7  plt.title('Gray Image'), plt.xticks([], plt.yticks([]))
8  plt.show()
9
10 ret, binary = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)
11
12 sobelx = cv2.Sobel(binary, -1,1,0, ksize=5)
13 sobely = cv2.Sobel(binary, -1,0,1, ksize=5)
14 sobelxy = cv2.Sobel(binary, -1,1,1, ksize=5)
15 print("BINARY")
16 plt.imshow(cv2.cvtColor(binary, cv2.COLOR_BGR2RGB))
17 plt.show()
18 print("SOBEL-X")
19 plt.imshow(sobelx, cmap="gray")
20 plt.show()
21 print("SOBEL-Y")
22 plt.imshow(sobely, cmap="gray")
23 plt.show()
24 print("SOBEL-XY")
25 plt.imshow(sobelxy, cmap="gray")
26 plt.show()
27
28
29 edges_sobelx=0
30 for i in range(sobelx.shape[0]):
31     for j in range(sobelx.shape[1]):
32         if (sobelx[i,j]>0):
33             edges_sobelx+=1
34 print("Sobelx shape :{}".format(sobelx.shape))
35 print("Edge pixels :{} out of {}".format(edges_sobelx,sobelx.size))
36
37 edges_sobely=0
38 for i in range(sobely.shape[0]):
39     for j in range(sobely.shape[1]):
40         if (sobely[i,j]>0):
41             edges_sobely+=1
42 print("Sobely shape :{}".format(sobelx.shape))
43 print("Edge pixels :{} out of {}".format(edges_sobely,sobely.size))
44
45 edges_sobelxy=0
46 for i in range(sobelxy.shape[0]):
47     for j in range(sobelxy.shape[1]):
48         if (sobelxy[i,j]>0):
49             #print(sobelx[i][j])
50             edges_sobelxy+=1
51 print("Sobelxy shape :{}".format(sobelxy.shape))
52 print("Edge pixels :{} out of {}".format(edges_sobelxy,sobelxy.size))
53
```

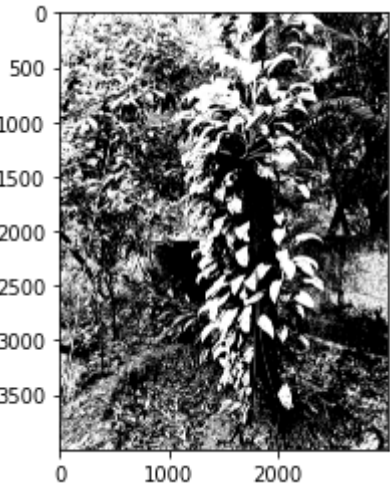
Original Image



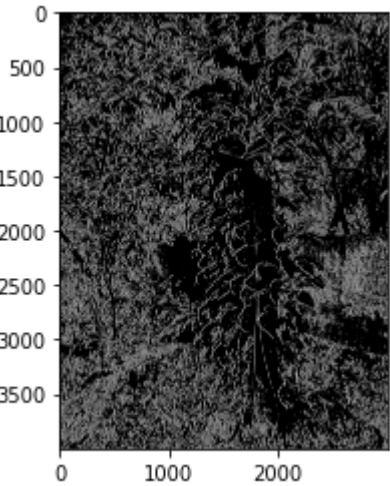
Gray Image



BINARY



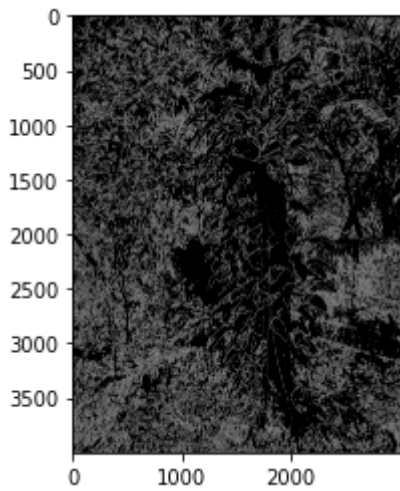
SOBEL -X



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 []:

1

In []:

1