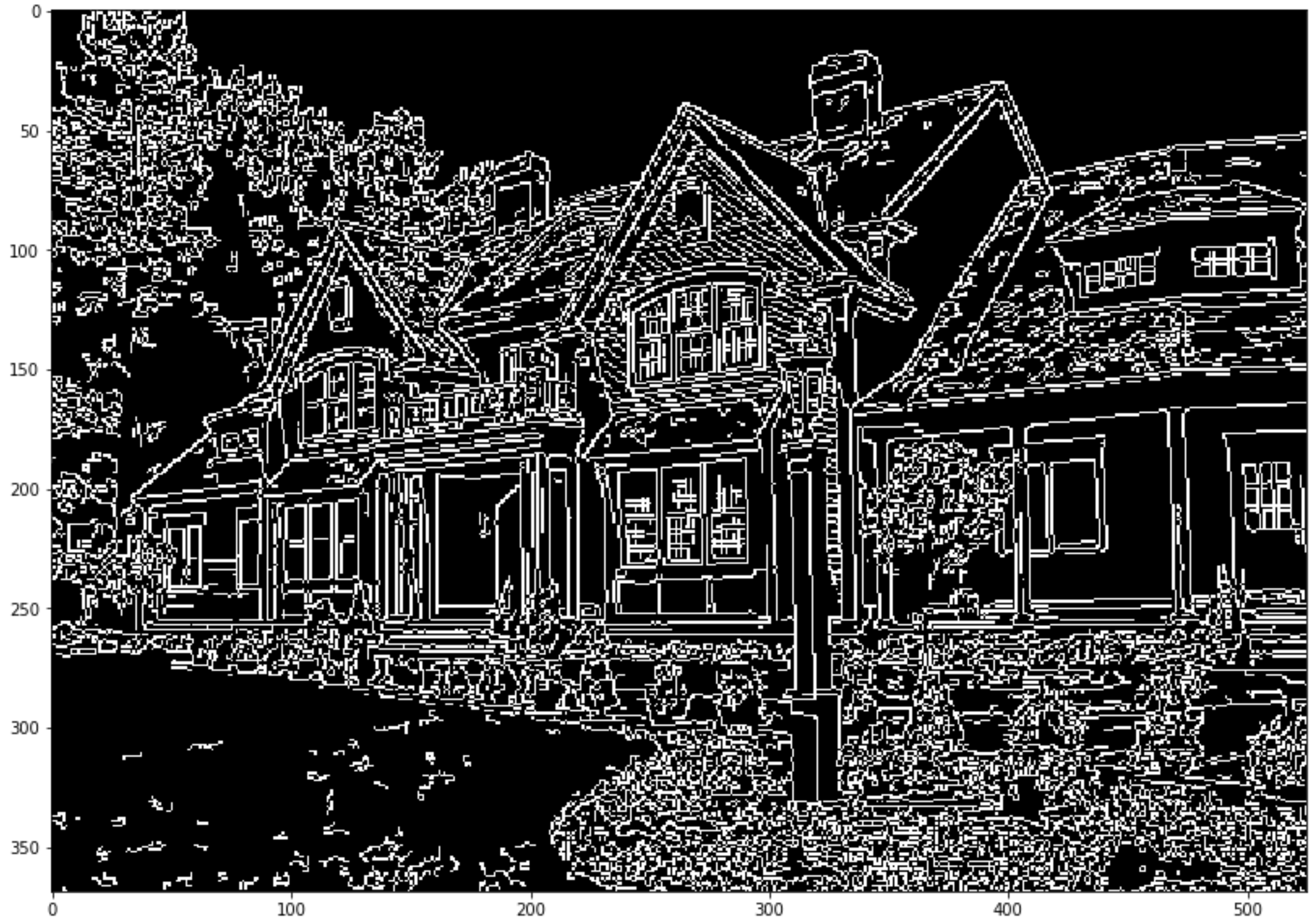
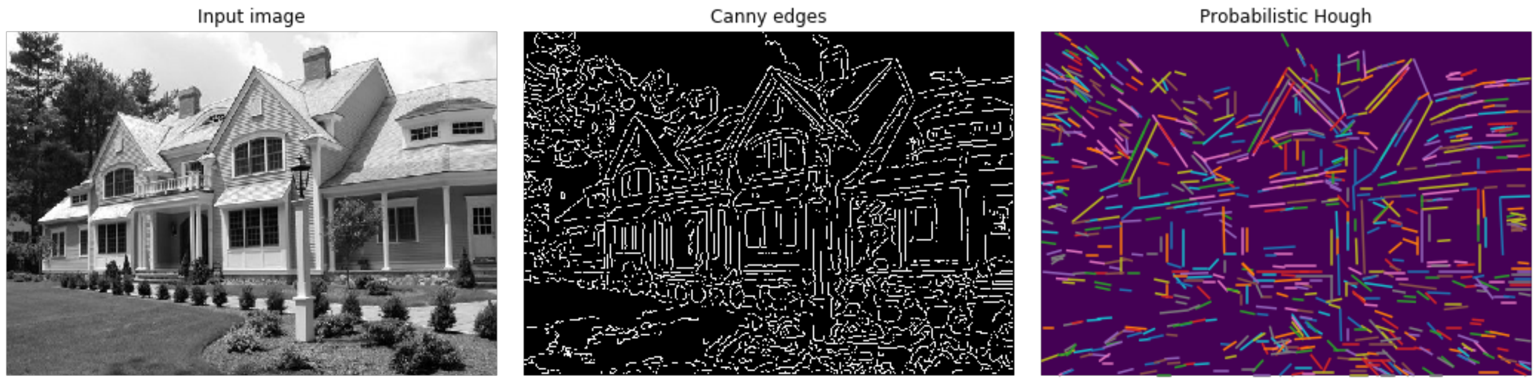
In [60]:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
img=cv2.imread('house.jpg')
v = np.median(img)

#---- apply automatic Canny edge detection using the computed median----
lower = int(max(0, (1.0 - 0.3) * v))#0.3 is the standard deviation
upper = int(min(255, (1.0 + 0.3) * v))
edged = cv2.Canny(img, lower, upper)
plt.figure(figsize=(20,10))
plt.imshow(edged,cmap='gray')
plt.show()
```



In [62]:

```
from skimage.transform import(hough_line,hough_line_peaks,probabilistic_hough_line)
from skimage.feature import canny
from matplotlib import cm
image=cv2.imread('house.jpg',0)
#canny takes(image,sigma,low_threshold,high_threshold)
edges = canny(image, 2, 1, 25)
lines = probabilistic_hough_line(edges, threshold=10, line_length=10,
                                 line_gap=3)

# Generating figure 2
fig, axes = plt.subplots(1, 3, figsize=(15, 5), sharex=True, sharey=True)
ax = axes.ravel()

ax[0].imshow(image, cmap=cm.gray)
ax[0].set_title('Input image')

ax[1].imshow(edges, cmap=cm.gray)
ax[1].set_title('Canny edges')

ax[2].imshow(edges * 0)
for line in lines:
    p0, p1 = line
    ax[2].plot((p0[0], p1[0]), (p0[1], p1[1]))
ax[2].set_xlim((0, image.shape[1]))
ax[2].set_ylim((image.shape[0], 0))
ax[2].set_title('Probabilistic Hough')

for a in ax:
    a.set_axis_off()

plt.tight_layout()
plt.show()
```



Input image

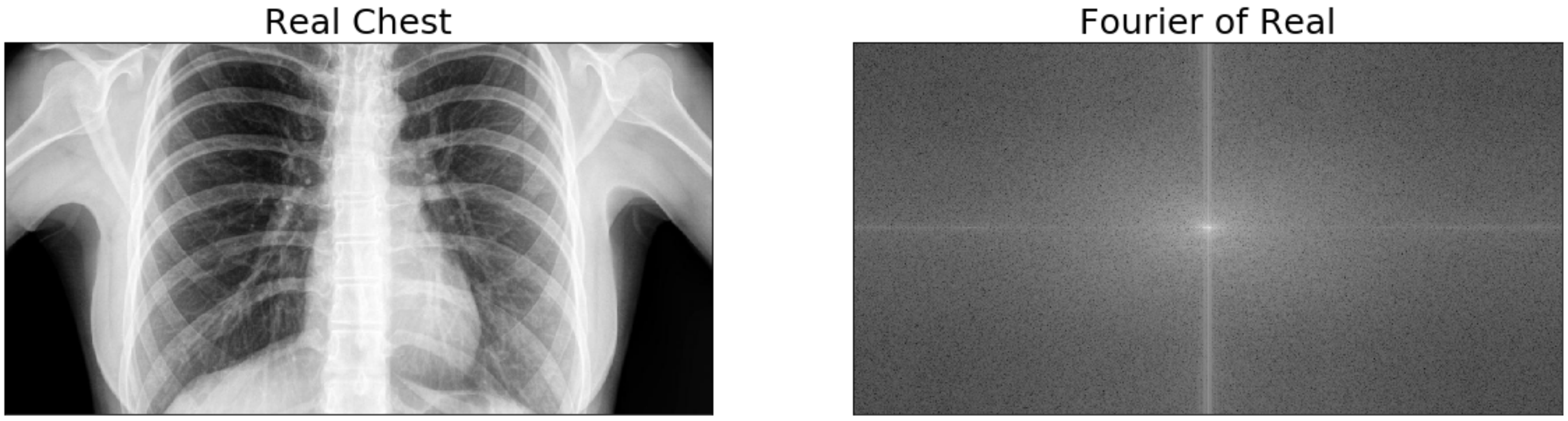
Canny edges

Probabilistic Hough

Extra Credit

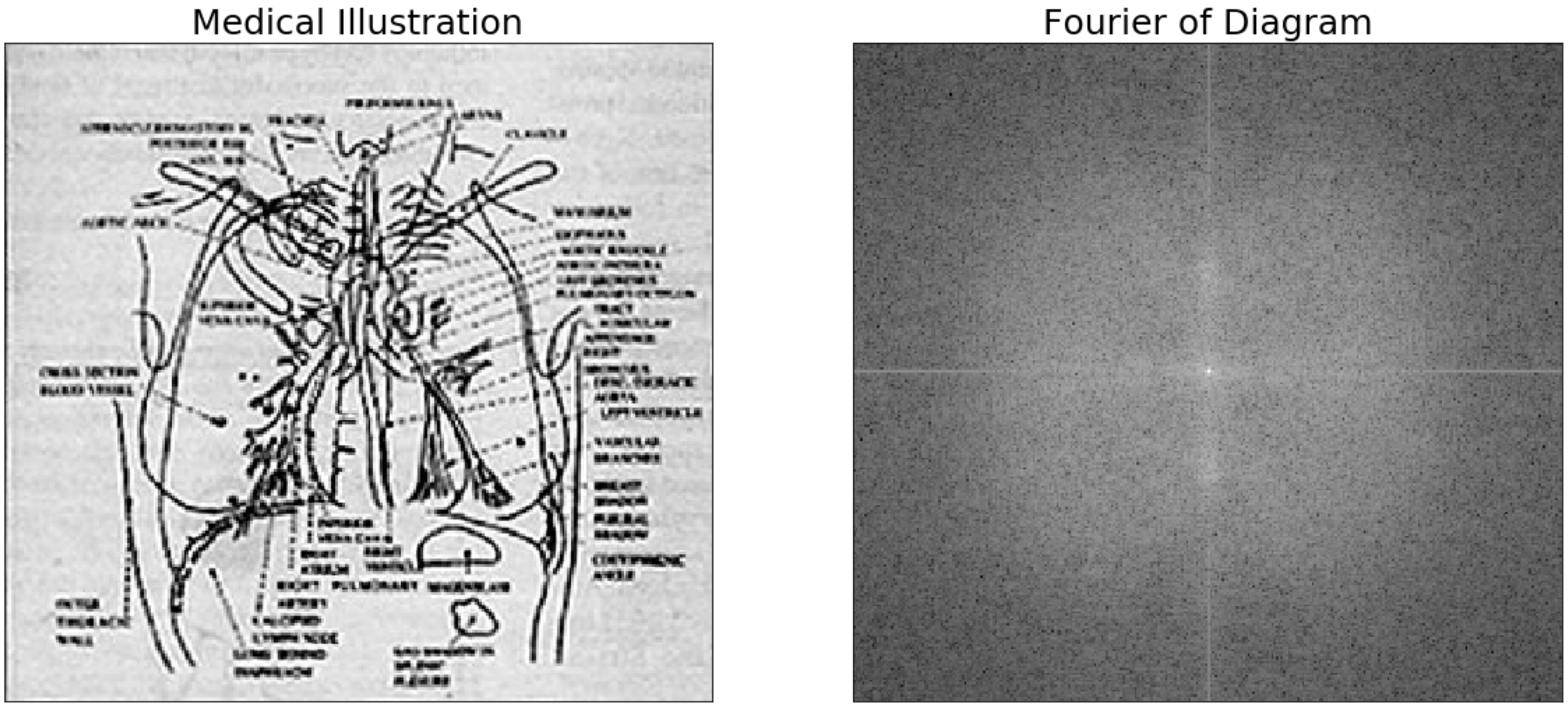
In [9]:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
from matplotlib import pyplot as plt
#read and display real image and its Fourier
path='/Users/cheryl/Desktop/real.jpg'
img = cv2.imread(path,0)
f = np.fft.fft2(img)
fshift = np.fft.fftshift(f)
magnitude_spectrum = np.log(np.abs(fshift))
plt.figure(figsize=(20,10))
plt.subplot(121),plt.imshow(img, cmap = 'gray')
plt.title('Real Chest',fontsize=25), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(magnitude_spectrum, cmap = 'gray')
plt.title('Fourier of Real',fontsize=25), plt.xticks([]), plt.yticks([])
plt.show()
#read and display fake image and its Fourier
path='/Users/cheryl/Desktop/fake.jpg'
img = cv2.imread(path,0)
f = np.fft.fft2(img)
fshift = np.fft.fftshift(f)
magnitude_spectrum = np.log(np.abs(fshift))
plt.figure(figsize=(20,10))
plt.subplot(121),plt.imshow(img, cmap = 'gray')
plt.title('Medical Illustration',fontsize=25), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(magnitude_spectrum, cmap = 'gray')
plt.title('Fourier of Diagram',fontsize=25), plt.xticks([]), plt.yticks([])
plt.show()
```



Real Chest

Fourier of Real



Medical Illustration

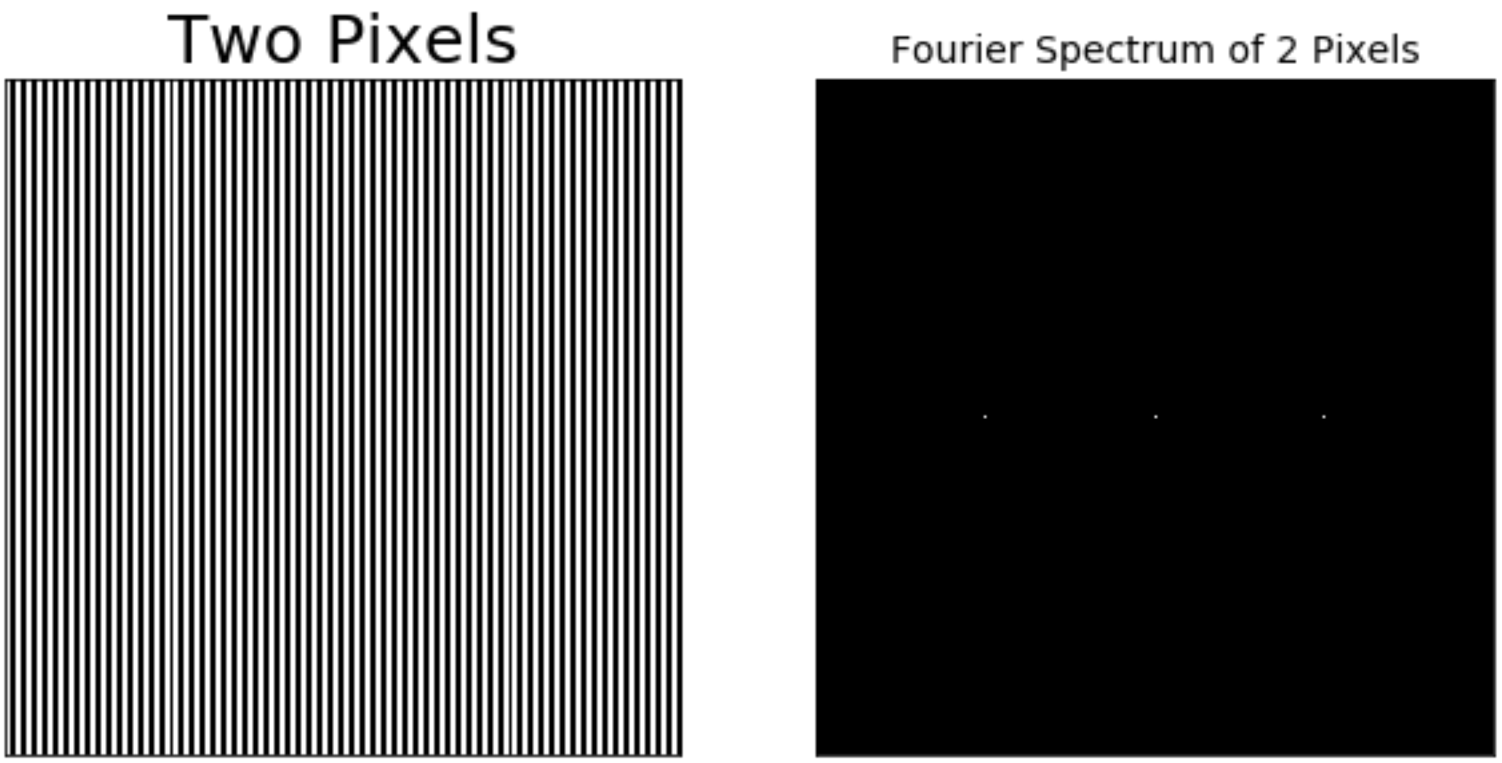
Fourier of Diagram

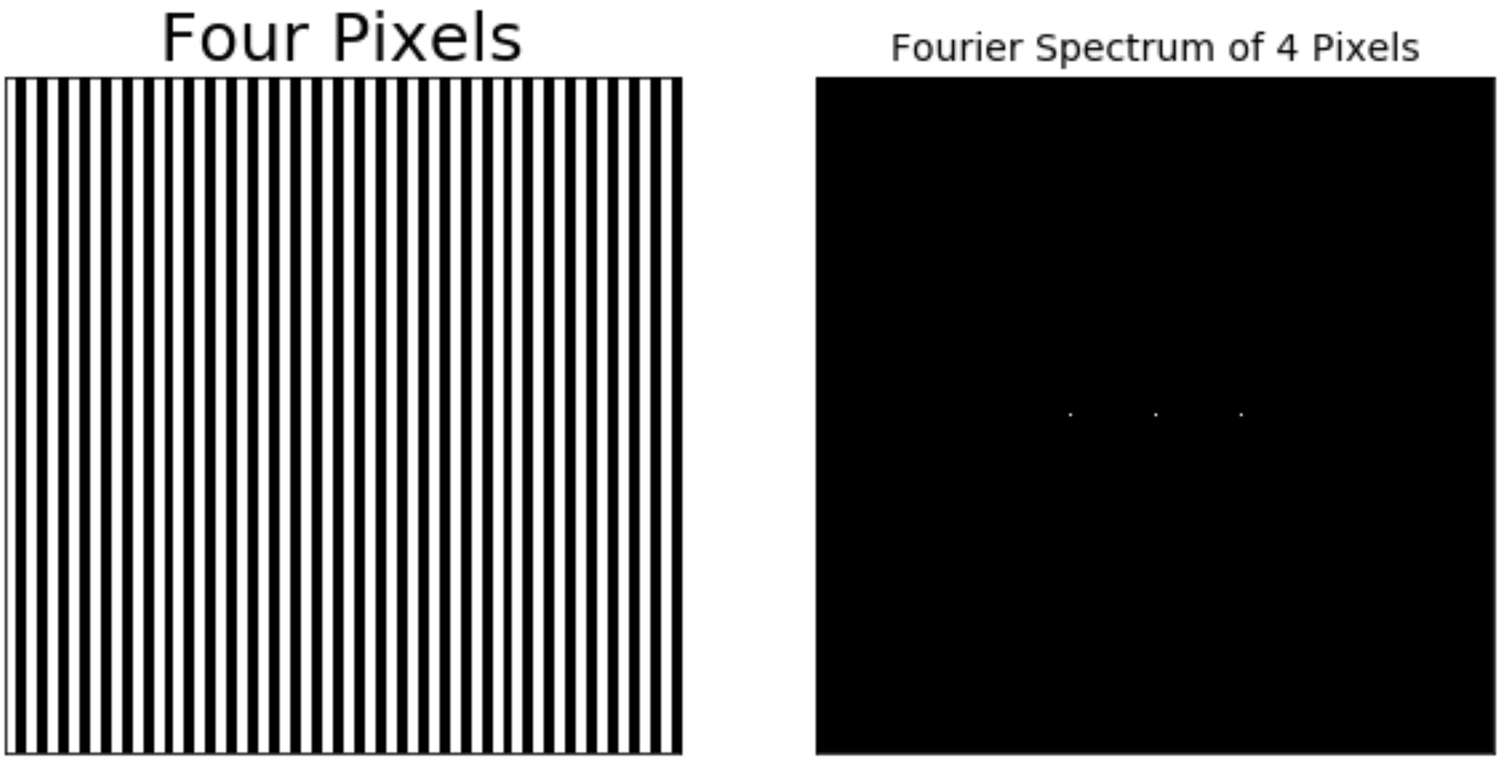
Problem 3

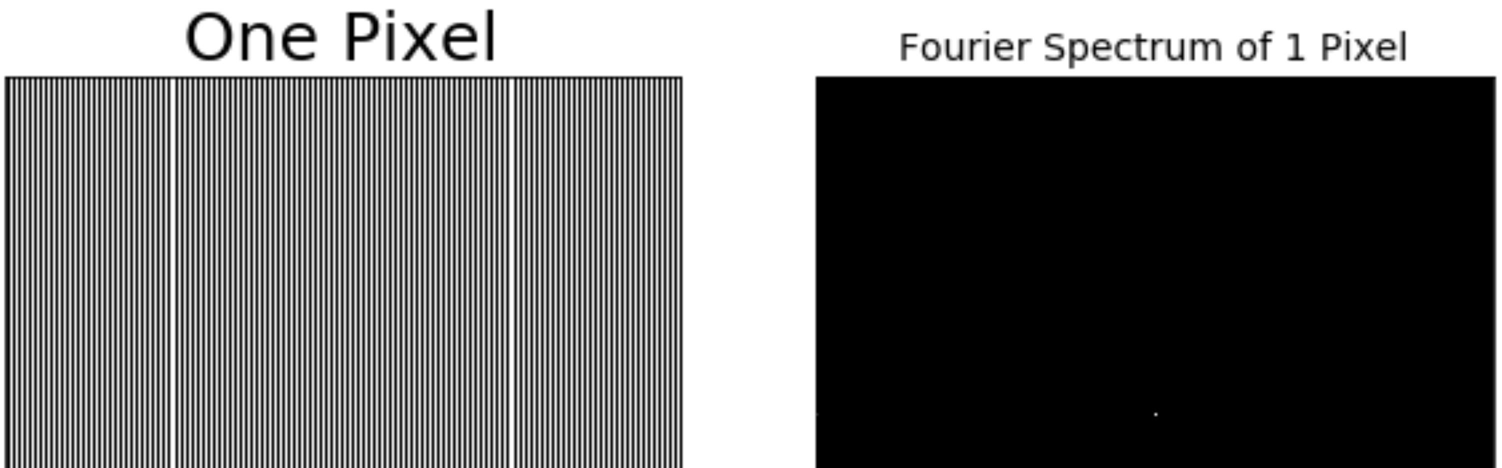
In [44]:

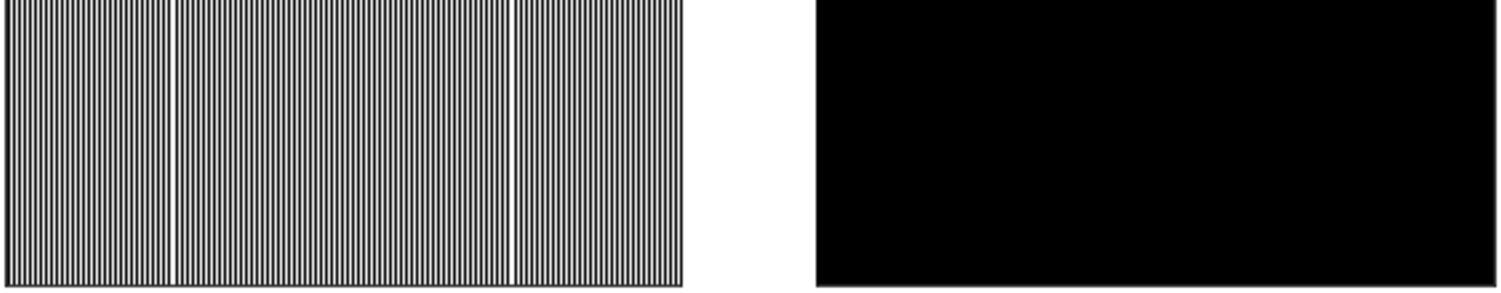
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import scipy.ndimage
#2 pixels wide
a=np.zeros([256,256])
for i in range(0,256,4):

    a[:,i:i+2]=1

f = np.fft.fft2(a)
fshift = np.fft.fftshift(f)
loga = np.log(1+np.abs(fshift))
plt.figure(figsize=(10,5))
plt.subplot(121),plt.imshow(a, cmap = 'gray')
plt.title('Two Pixels',fontsize=25), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(loga, cmap = 'gray')
plt.title('Fourier Spectrum of 2 Pixels',fontsize=14), plt.xticks([]), plt.yticks([
#plt.show()
#4 pixels
b=np.zeros([256,256])
for i in range(0,256,8):

    b[:,i:i+4]=1

f = np.fft.fft2(b)
fshift = np.fft.fftshift(f)
logb = np.log(1+np.abs(fshift))
#getting rid of the outer two values
logb[:,0:85]=0
logb[:,170:256]=0
plt.figure(figsize=(10,5))
plt.subplot(121),plt.imshow(b, cmap = 'gray')
plt.title('Four Pixels',fontsize=25), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(logb, cmap = 'gray')
plt.title('Fourier Spectrum of 4 Pixels',fontsize=14), plt.xticks([]), plt.yticks([
plt.show()

#1 pixel wide
c=np.zeros([256,256])
for i in range(0,256,2):

    c[:,i:i+1]=1

f = np.fft.fft2(c)
fshift = np.fft.fftshift(f)
logc = np.log(1+np.abs(fshift))
```

```
plt.figure(figsize=(10,5))
plt.subplot(121),plt.imshow(c, cmap = 'gray')
plt.title('One Pixel',fontsize=25), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(logc, cmap = 'gray')
plt.title('Fourier Spectrum of 1 Pixel',fontsize=14), plt.xticks([]), plt.yticks([]
plt.show()
```









In [111]:

In [148]:

In [149]:

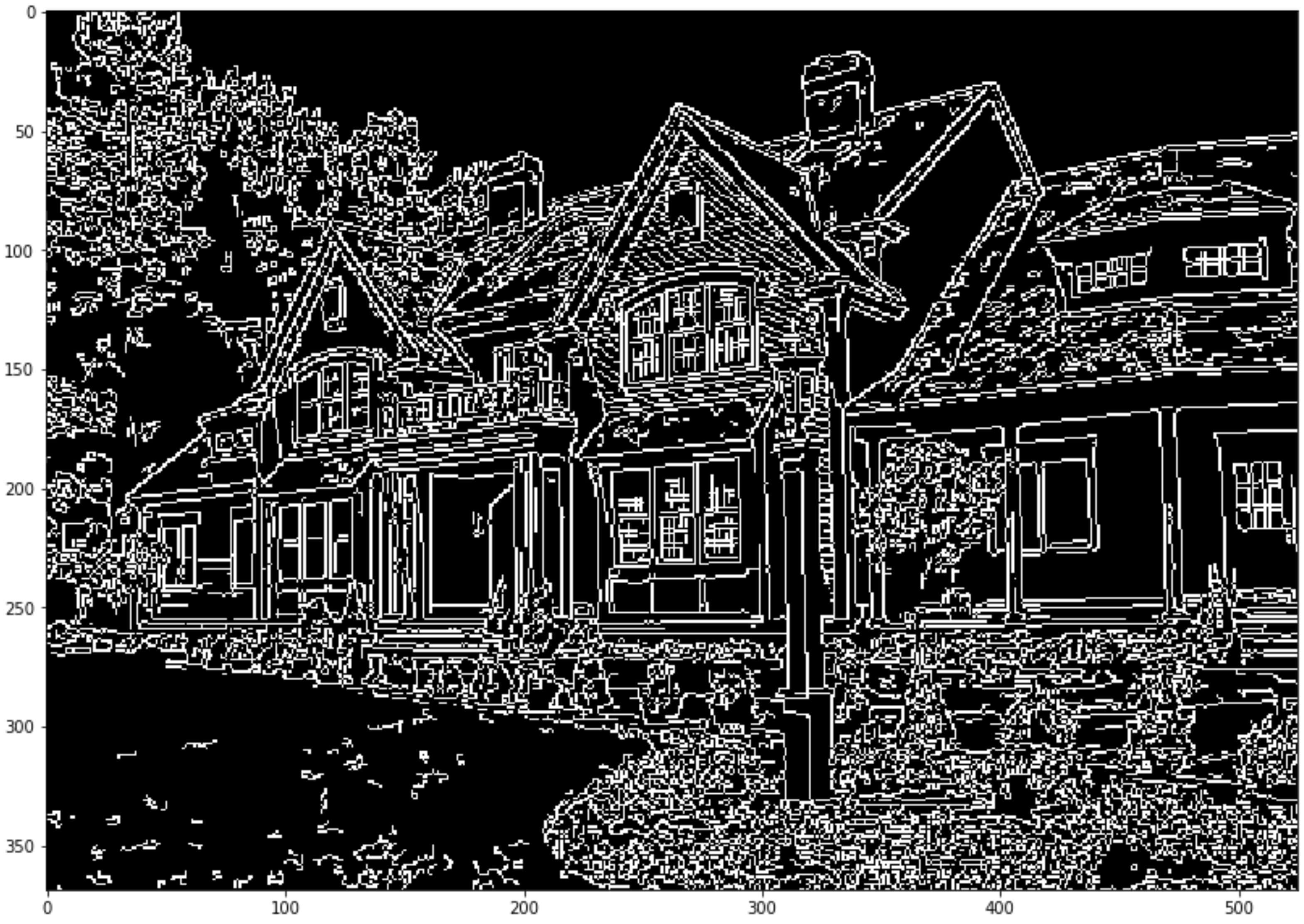
In [ ]:

In [ ]:

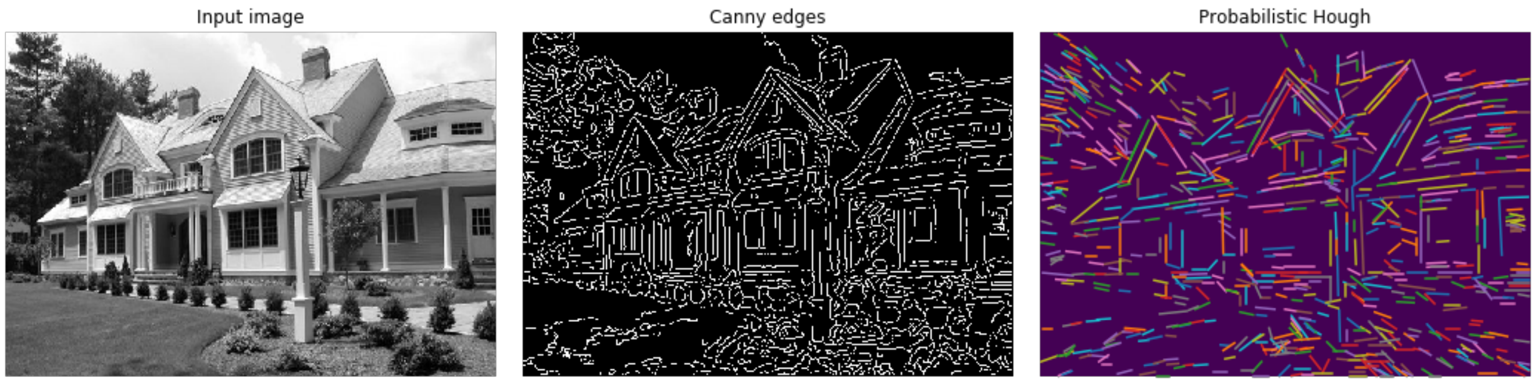
In [ ]:

In [ ]:

In [60]:

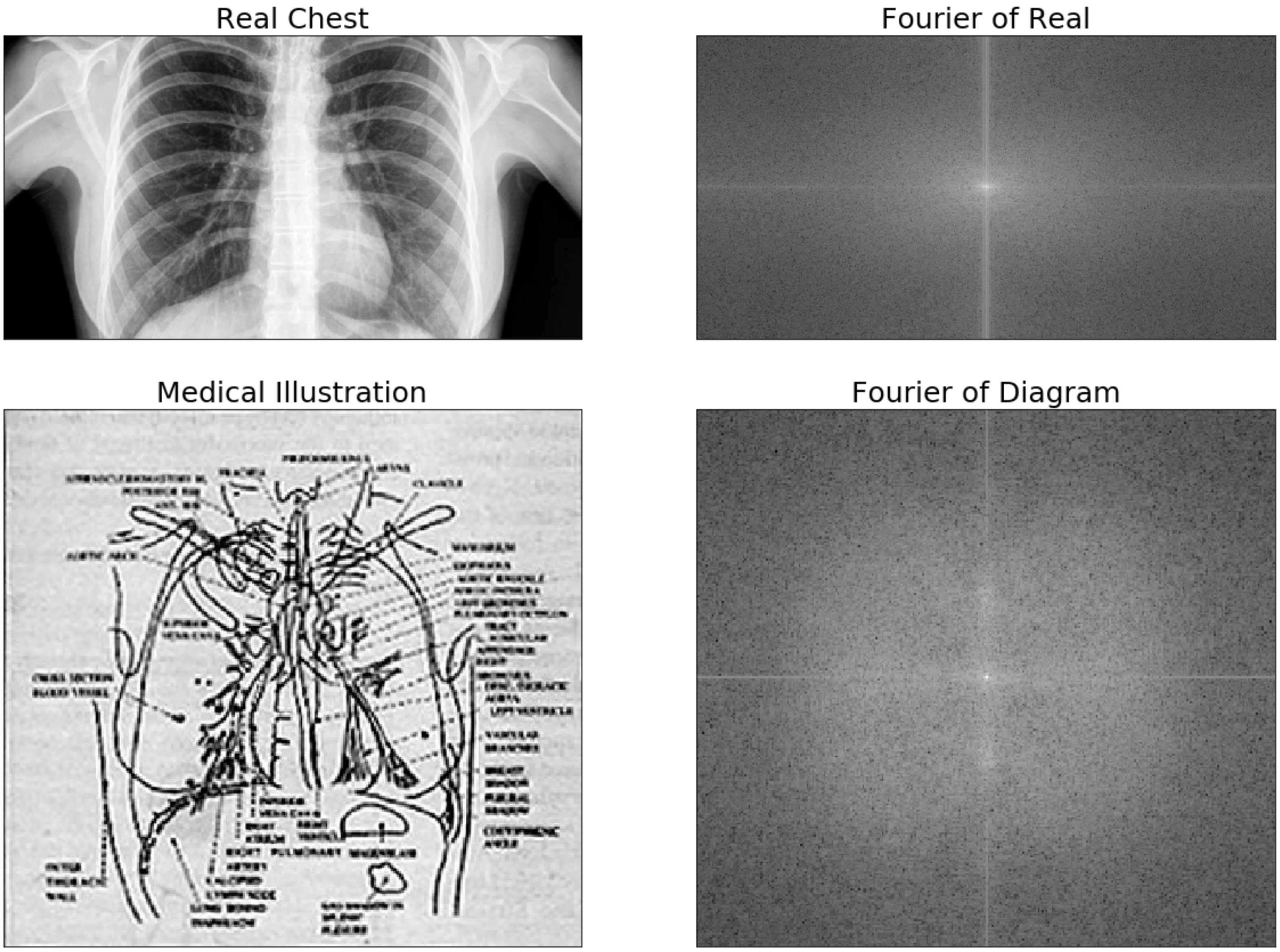


In [62]:



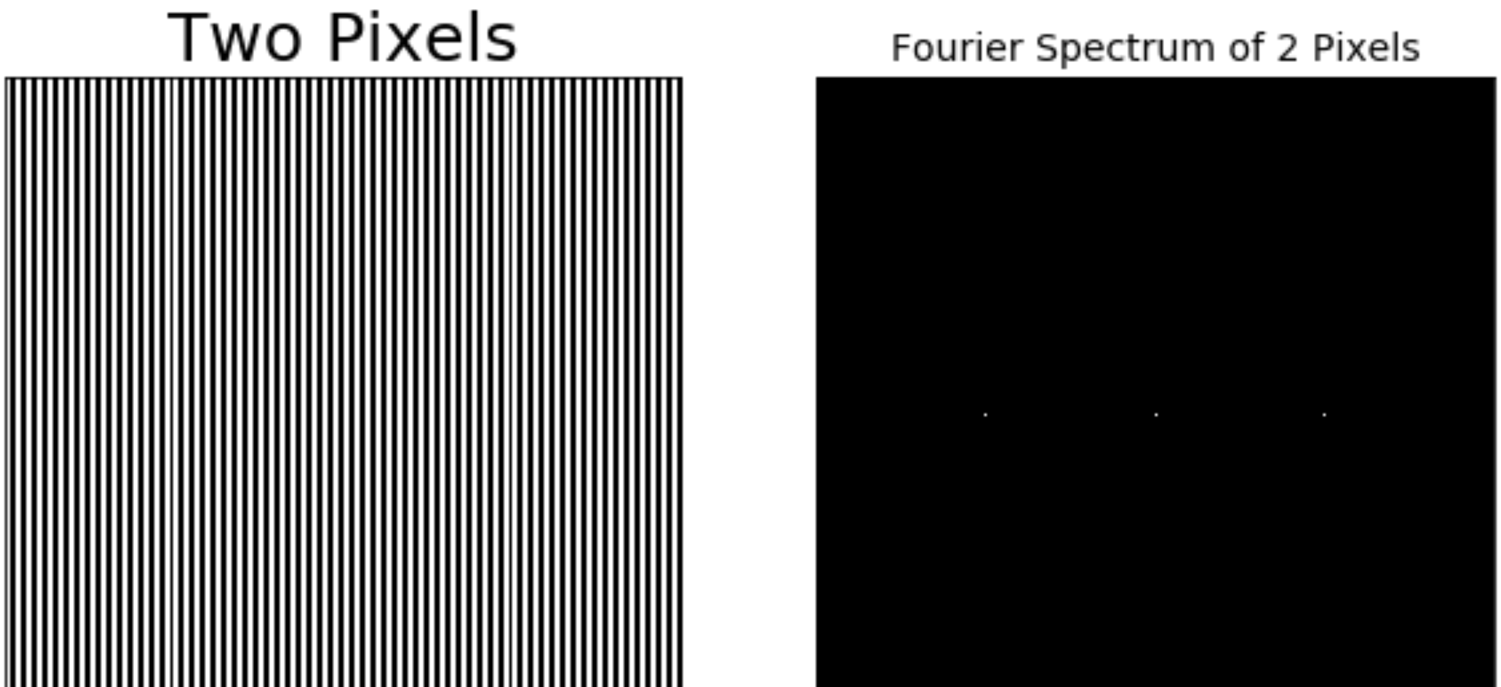
Extra Credit

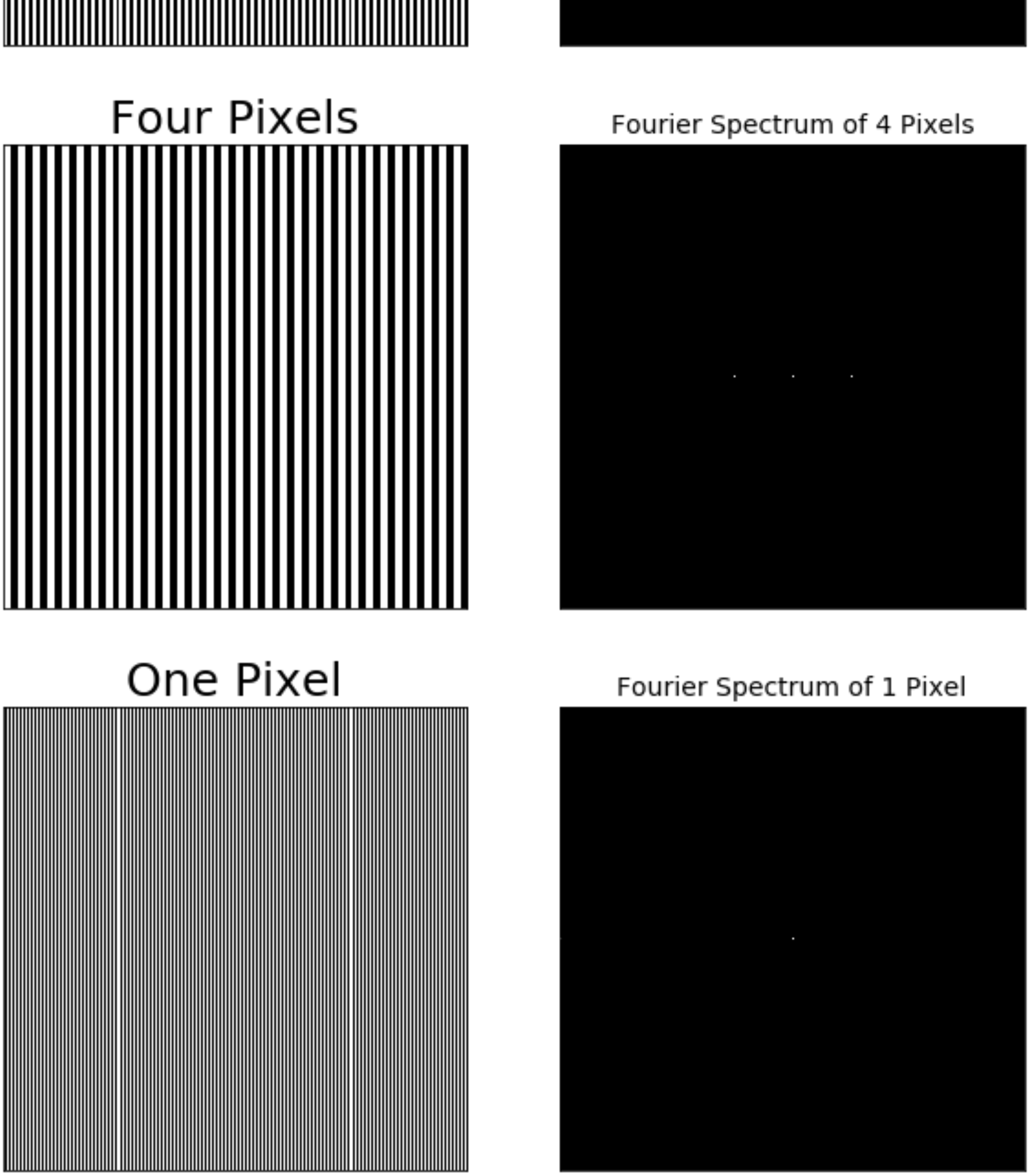
```
In [9]:
```



Problem 3

```
In [44]:
```





Four Pixels

Fourier Spectrum of 4 Pixels

One Pixel

Fourier Spectrum of 1 Pixel

In [111]:

In [148]:

In [149]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: