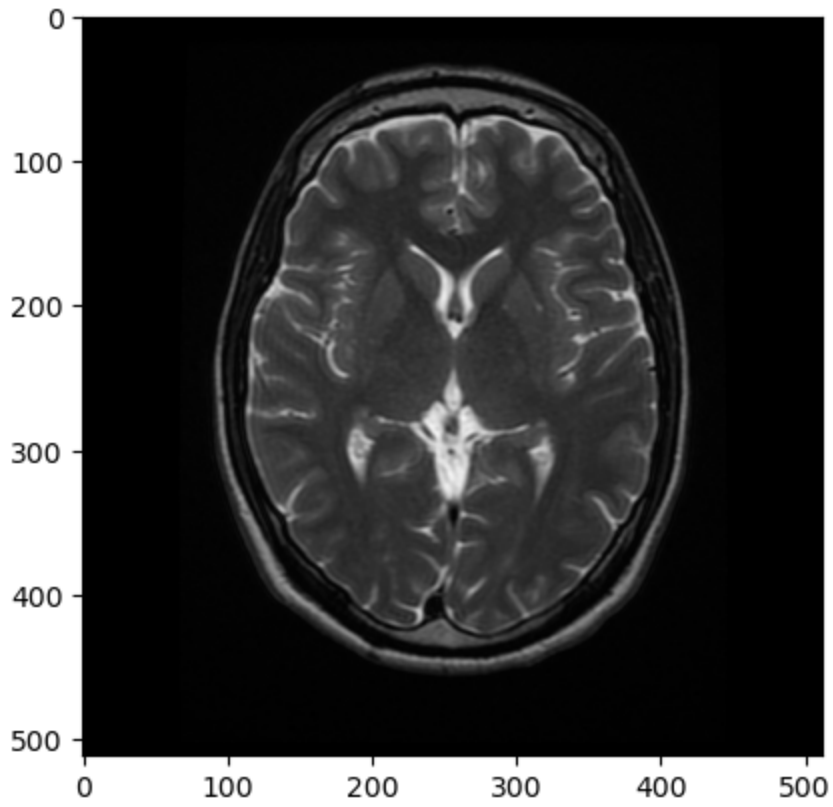
Canny Edge Detection

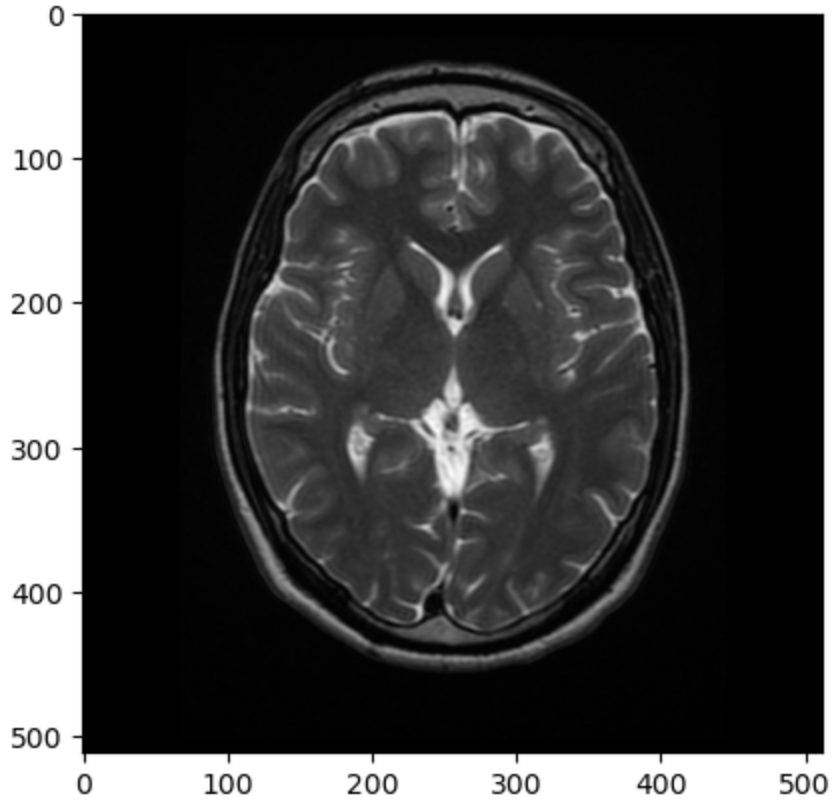
In [6]:
```
import numpy as np
import matplotlib.pyplot as plt
import cv2

%matplotlib inline

# Read in the image
image = cv2.imread('D:/Udacity/Jupyter/brain_MR.jpg')

# Change color to RGB (from BGR)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.imshow(image)
```

Out[6]:
```
<matplotlib.image.AxesImage at 0x1e1be531f10>
```



In [2]:
```
# Convert the image to grayscale for processing
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)

plt.imshow(gray, cmap='gray')
```

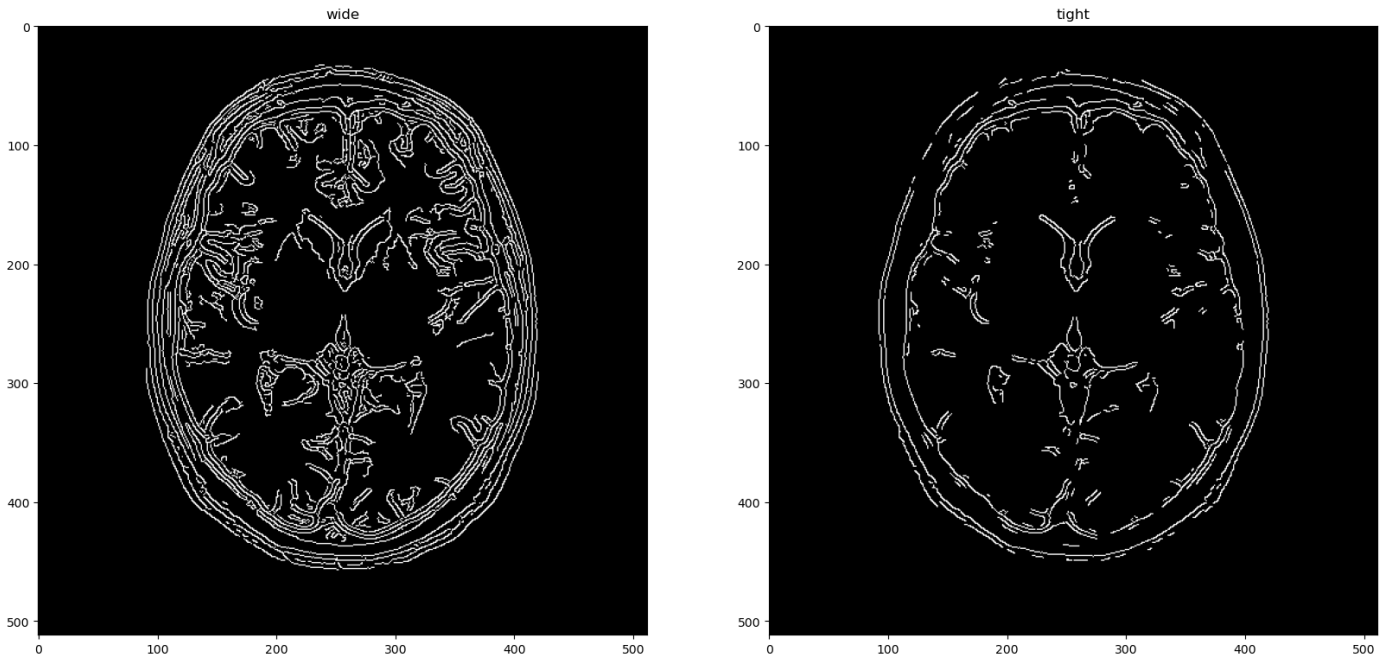
Out[2]:
```
<matplotlib.image.AxesImage at 0x1e1bd896710>
```



Implementing Canny edge detection

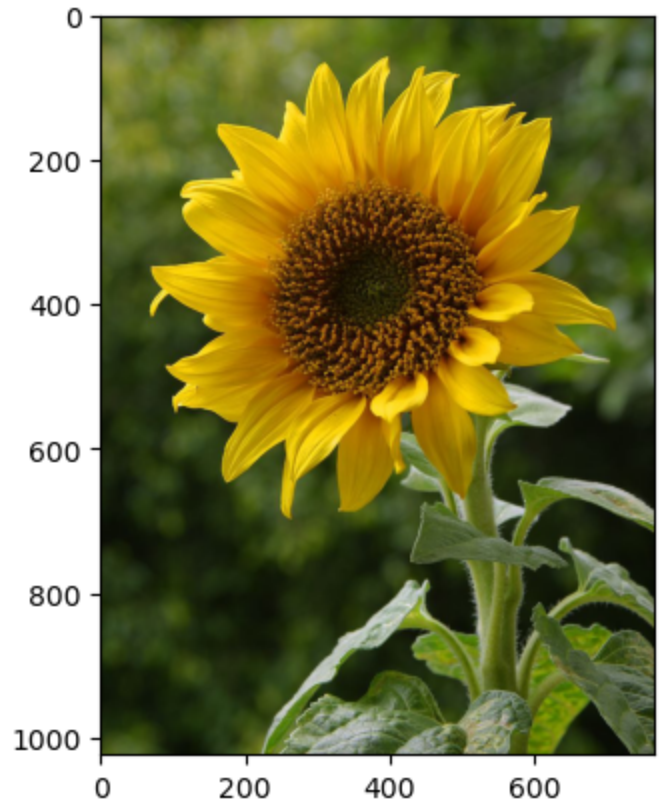
In [3]:
```
# Canny using "wide" and "tight" thresholds

wide = cv2.Canny(gray, 30, 100)
tight = cv2.Canny(gray, 200, 240)


# Display the images
f, (ax1, ax2) = plt.subplots(1, 2, figsize=(20,10))

ax1.set_title('wide')
ax1.imshow(wide, cmap='gray')

ax2.set_title('tight')
ax2.imshow(tight, cmap='gray')
```

Out[3]:
```
<matplotlib.image.AxesImage at 0x1e1bd8e4f90>
```



In [7]:
```
# Read in the image
image = cv2.imread('D:/Udacity/Jupyter/sunflower.jpg')

# Changing color to RGB (from BGR)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.imshow(image)
```

Out[7]:
```
<matplotlib.image.AxesImage at 0x1e1be5a1f10>
```



In [8]:
```
# Converting the image to grayscale
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)

# right now the threshold is so small and low that it will pick up a lot of noise
lower = 50
upper = 100
```

```
edges = cv2.Canny(gray, lower, upper)

plt.figure(figsize=(20,10))
plt.imshow(edges, cmap='gray')
```

Out[8]: <matplotlib.image.AxesImage at 0x1e1be8b6090>