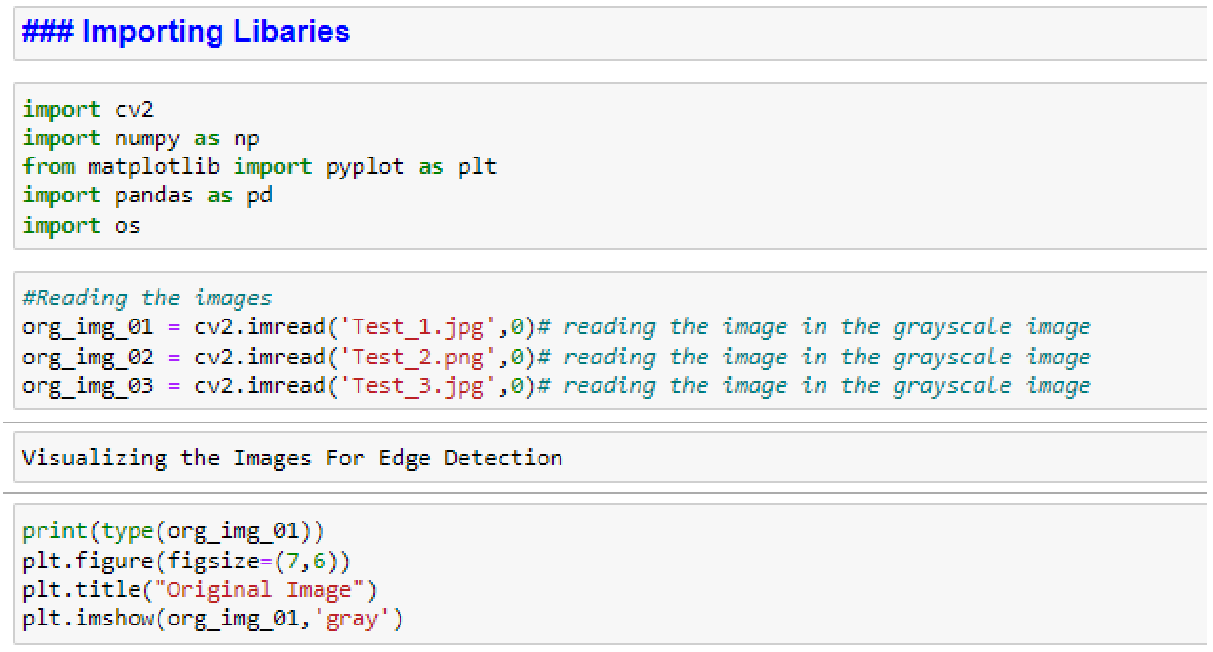
Govardhan Seshasayee

Image Edge detection using Adaptive Thresholding

Edges define the boundaries between different regions in an image, which helps in matching the pattern, segment, and recognize an object. In simple thresholding, the threshold value is global, which is prone to fail in many cases. Adaptive thresholding is a modified method where the threshold value is calculated for each pixel based on a smaller region around it. Therefore, there will be different threshold values for different regions which gives better results for images with varying illumination.

Implementation: Image edge detection



Importing Libaries

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import pandas as pd
import os
```

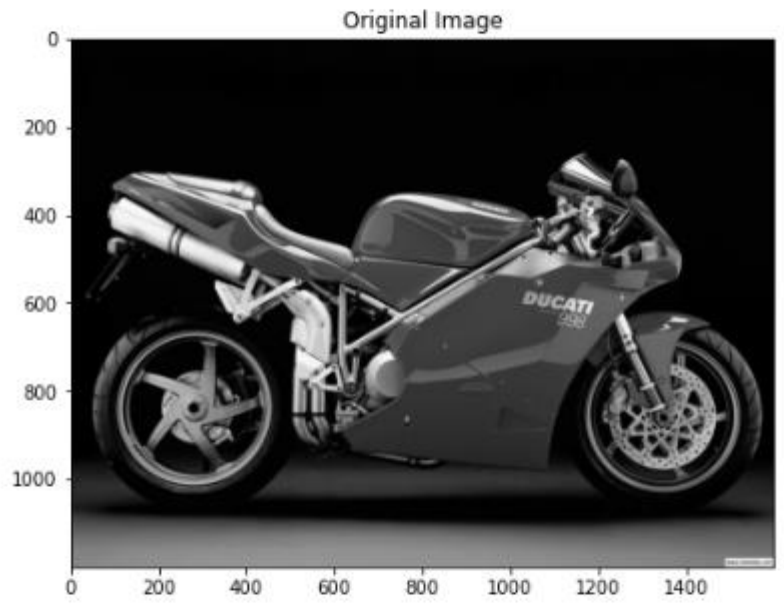
```
#Reading the images
org_img_01 = cv2.imread('Test_1.jpg',0)# reading the image in the grayscale image
org_img_02 = cv2.imread('Test_2.png',0)# reading the image in the grayscale image
org_img_03 = cv2.imread('Test_3.jpg',0)# reading the image in the grayscale image
```

```
Visualizing the Images For Edge Detection
```

```
print(type(org_img_01))
plt.figure(figsize=(7,6))
plt.title("Original Image")
plt.imshow(org_img_01,'gray')
```

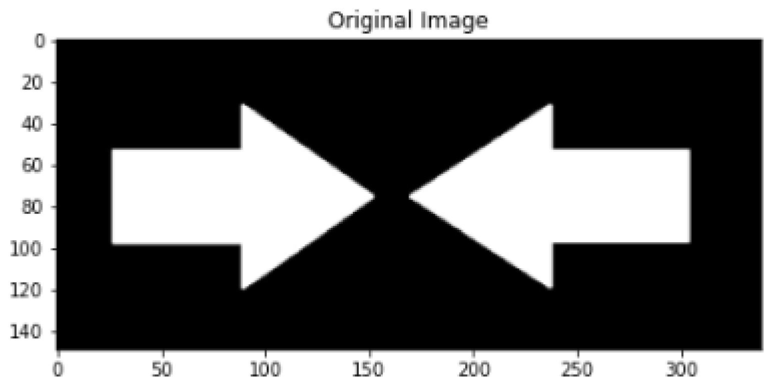
```
<class 'numpy.ndarray'>
```

```
<matplotlib.image.AxesImage at 0x26e7825c9a0>
```



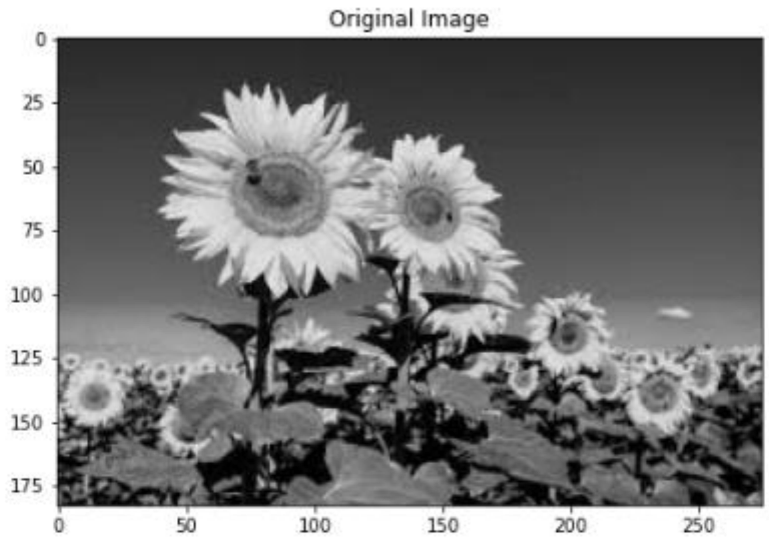
```python
print(type(org_img_01))
plt.figure(figsize=(7,6))
plt.title("Original Image")
plt.imshow(org_img_02,'gray')
```

```
<class 'numpy.ndarray'>

<matplotlib.image.AxesImage at 0x26e7856ca00>
```



```python
print(type(org_img_01))
plt.figure(figsize=(7,6))
plt.title("Original Image")
plt.imshow(org_img_03,'gray')
```

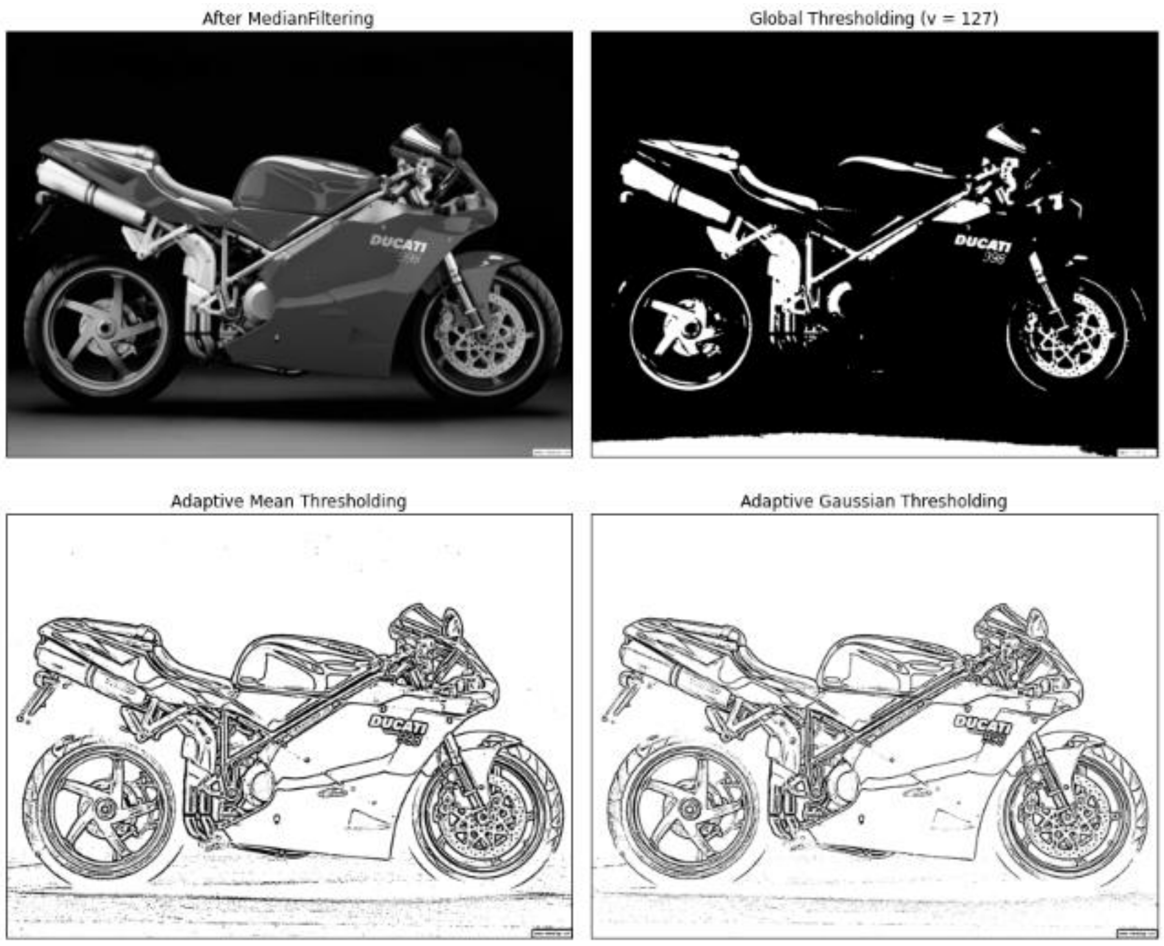
```
<class 'numpy.ndarray'>

<matplotlib.image.AxesImage at 0x26e789ef040>
```



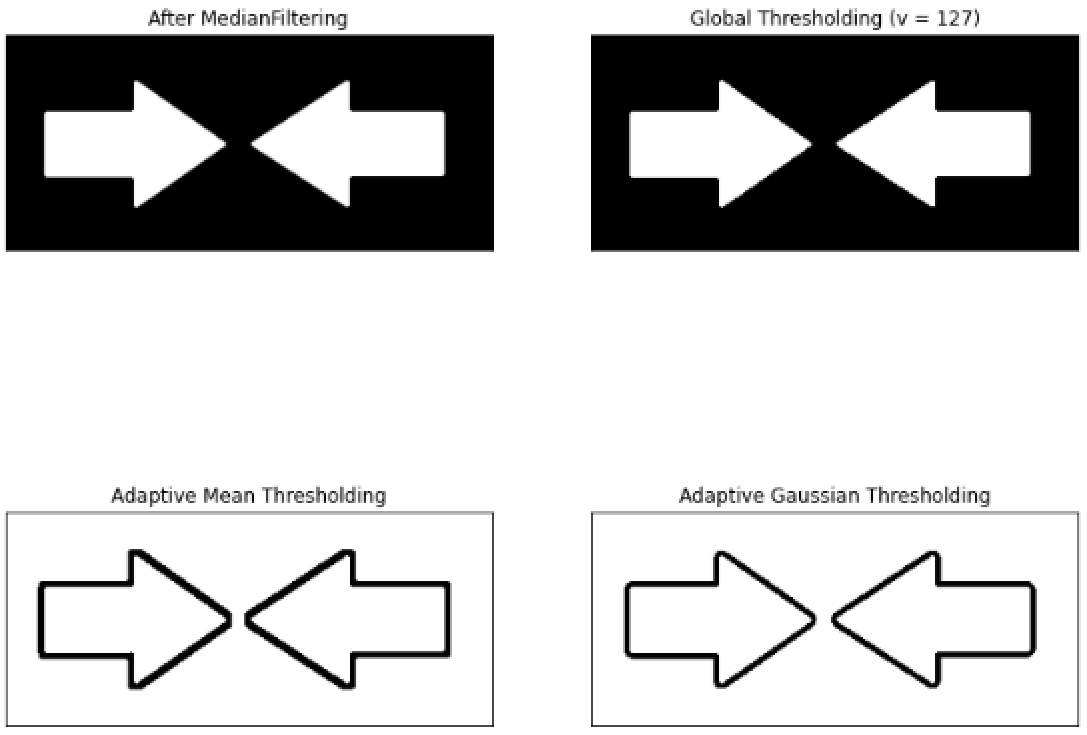
```python
print(org_img_01.shape," ",org_img_02.shape," ",org_img_03.shape)
```

```
(1200, 1600)   (149, 339)   (183, 275)
```

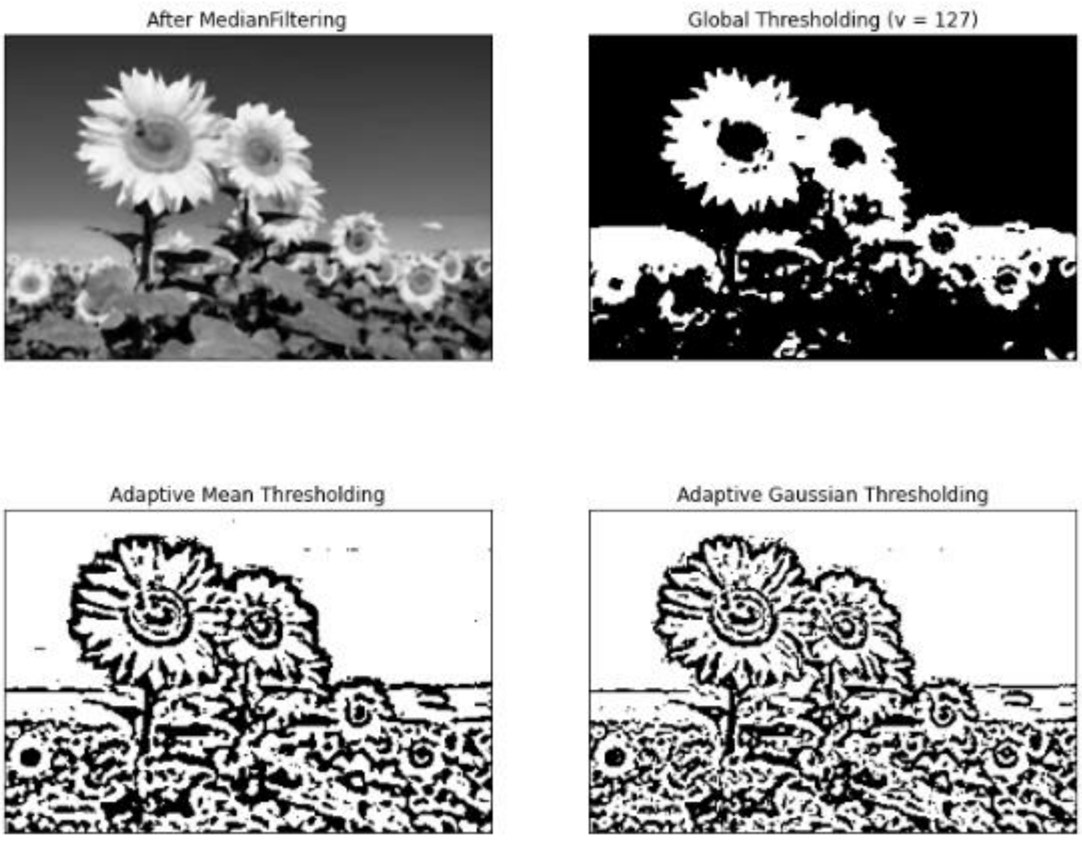
```
img = cv2.medianBlur(org_img_01,5) ## Remove noise using median Blur

## 2nd argument - threshold, 3rd Argument - Value assigned if pixel is greater than threshold
ret,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)

## threshold value is the mean of neighbourhood area
th2 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_MEAN_C,cv2.THRESH_BINARY,9,2)

## threshold value is the weighted sum of neighbourhood values where weights are a gaussian window.
th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,9,2)

titles = ['After MedianFiltering','Global Thresholding (v = 127)',
            'Adaptive Mean Thresholding', 'Adaptive Gaussian Thresholding']

images = [img, th1, th2, th3]

plt.figure(figsize=(12,10))
for i in range(4):
    plt.subplot(2,2,i+1)
    plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

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


After MedianFiltering
Global Thresholding (v = 127)
Adaptive Mean Thresholding
Adaptive Gaussian Thresholding

```python
img = cv2.medianBlur(org_img_02,3)## Remove noise from the source image

## 2nd argument - threshold, 3rd Argument - Value assigned if pixel is greater than threshold
ret,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)

## threshold value is the mean of neighbourhood area
th2 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_MEAN_C,cv2.THRESH_BINARY,9,2)

## threshold value is the weighted sum of neighbourhood values where weights are a gaussian window.
th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,9,2)

titles = ['After MedianFiltering','Global Thresholding (v = 127)',
            'Adaptive Mean Thresholding', 'Adaptive Gaussian Thresholding']

images = [img, th1, th2, th3]

plt.figure(figsize=(12,10))
for i in range(4):
    plt.subplot(2,2,i+1)
    plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])
```


After MedianFiltering
Global Thresholding (v = 127)
Adaptive Mean Thresholding
Adaptive Gaussian Thresholding

```
img = cv2.medianBlur(org_img_03,3)## Remove noise from the source image

## 2nd argument - threshold, 3rd Argument - Value assigned if pixel is greater than threshold
ret,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)

## threshold value is the mean of neighbourhood area
th2 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_MEAN_C,cv2.THRESH_BINARY,9,2)

## threshold value is the weighted sum of neighbourhood values where weights are a gaussian window.
th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,9,2)

titles = ['After MedianFiltering','Global Thresholding (v = 127)',
            'Adaptive Mean Thresholding', 'Adaptive Gaussian Thresholding']

images = [img, th1, th2, th3]

plt.figure(figsize=(12,10))
for i in range(4):
    plt.subplot(2,2,i+1)
    plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])
```


After MedianFiltering
Global Thresholding (v = 127)
Adaptive Mean Thresholding
Adaptive Gaussian Thresholding

Result after Otsu's thresholding

```
img = cv2.medianBlur(org_img_01,5)
ret2,th2 = cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

fig = plt.figure(figsize=(15,10))
plt.subplot(1,2,1)
plt.imshow(img,'gray')
plt.xticks([]),plt.yticks([])
plt.title("Image After Median Filtering")

plt.subplot(1,2,2)
plt.imshow(th2,'gray')
plt.xticks([]),plt.yticks([])
plt.title("Image after Otsu's Thresholding")
plt.show()
```

Image After Median Filtering

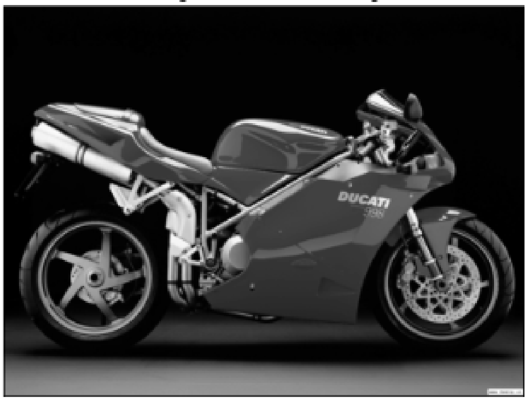
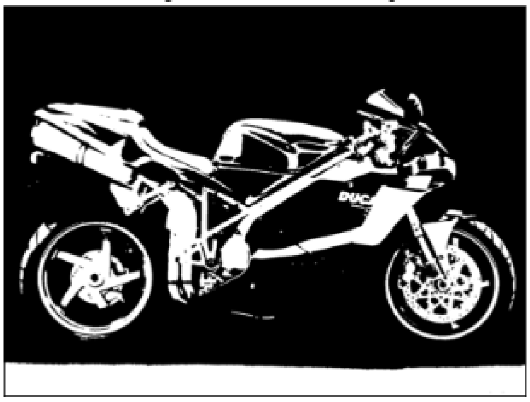


Image after Otsu's Thresholding



```
img = cv2.medianBlur(org_img_02,5)
ret2,th2 = cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

fig = plt.figure(figsize=(15,10))
plt.subplot(1,2,1)
plt.imshow(img,'gray')
plt.xticks([]),plt.yticks([])
plt.title("Image After Median Filtering")

plt.subplot(1,2,2)
plt.imshow(th2,'gray')
plt.xticks([]),plt.yticks([])
plt.title("Image after Otsu's Thresholding")
plt.show()
```

Image After Median Filtering

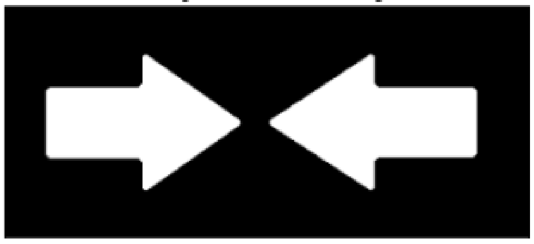
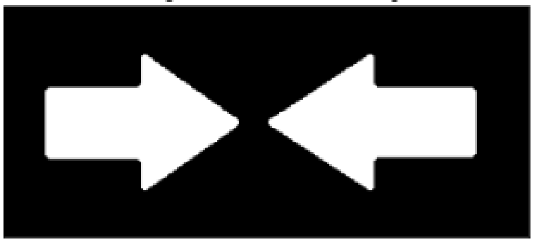


Image after Otsu's Thresholding



```
img = cv2.medianBlur(org_img_03,5)
ret2,th2 = cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

fig = plt.figure(figsize=(15,10))
plt.subplot(1,2,1)
plt.imshow(img,'gray')
plt.xticks([]),plt.yticks([])
plt.title("Image After Median Filtering")

plt.subplot(1,2,2)
plt.imshow(th2,'gray')
plt.xticks([]),plt.yticks([])
plt.title("Image after Otsu's Thresholding")
plt.show()
```

Image After Median Filtering

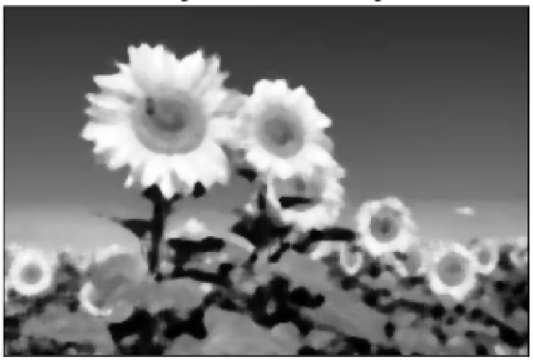


Image after Otsu's Thresholding