PROJECT REPORT

Assignment Project Report Adaptive Thresholding: Edge Detection in Images
NAME: RISHI RAJ SINGH
AI AND ML B-4

Problem Statement:

Using OpenCV, first convert any image with varying High condition to a grayscale image. Now implement edge detection first using the canny edge detection. Then apply simple thresholding and also Adaptive/OTSU thresholding using OpenCV to see the working of each of these methods. Once you obtain good results, use the obtained edge detection result as a mask to give color to all the edges (if edges use the color from the original image, else leave it black only).

Prerequisites:
- Software: Python 3
Tools:
- Pandas
- Numpy
- Matplotlib
- Seaborn
- OpenCv

Implementation:
- Load all required libraries

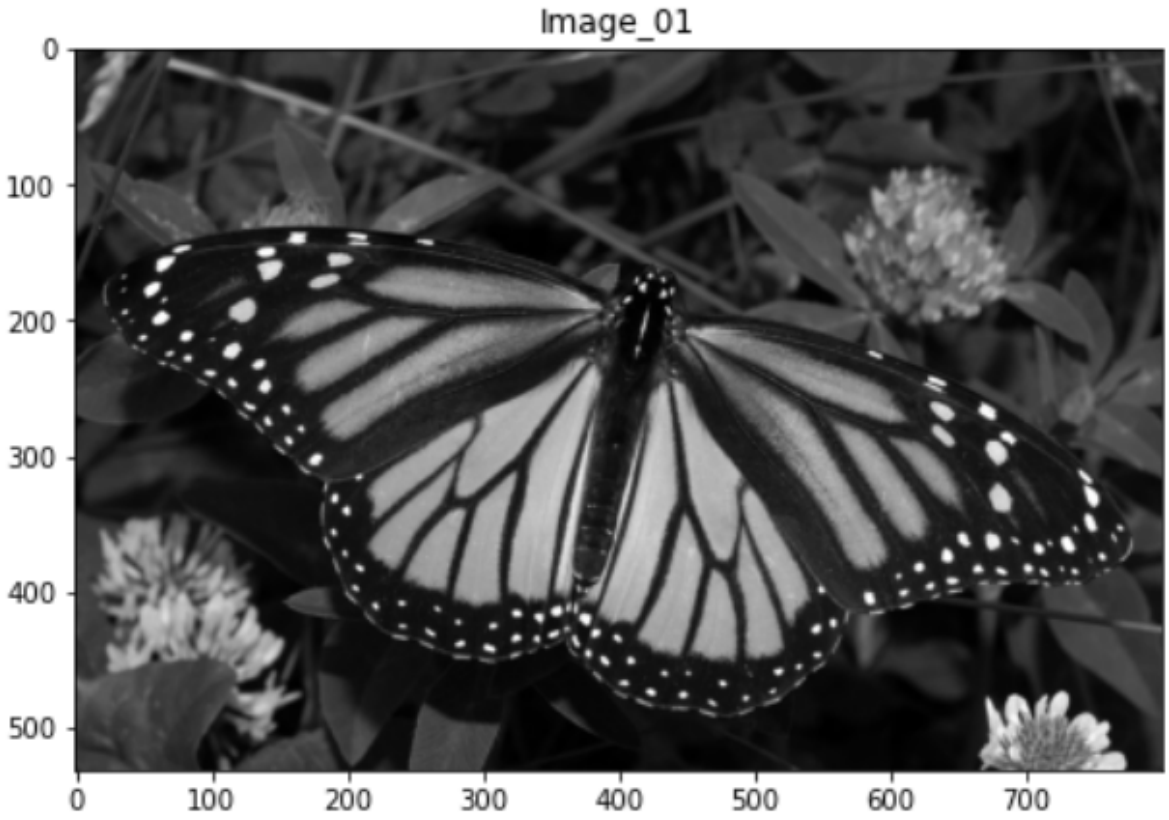
In [1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import pandas as pd
import os
```

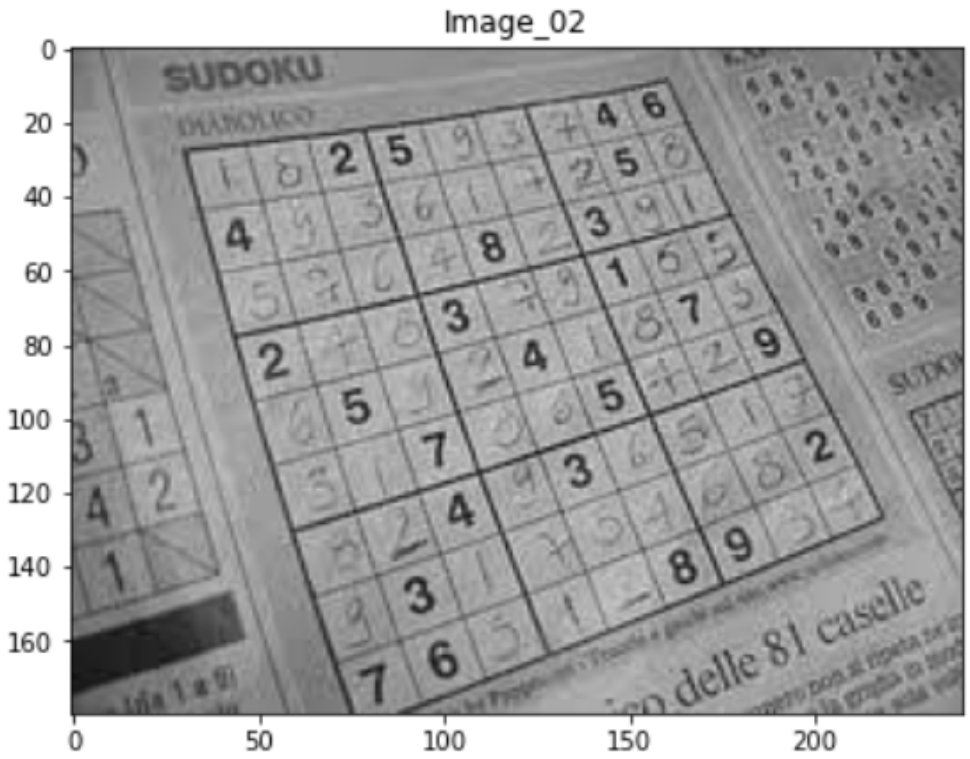
- Loading Dataset

```
#reading the image in the grayscale image
img_01 = cv2.imread('Test_01.jpg',0)
img_02 = cv2.imread('Test_02.jfif',0)
img_03 = cv2.imread('Test_03.jpg',0)
```

- Visualizing Data

```
plt.figure(figsize=(10,5))
plt.title("Image_01")
plt.imshow(img_01,'gray')
```



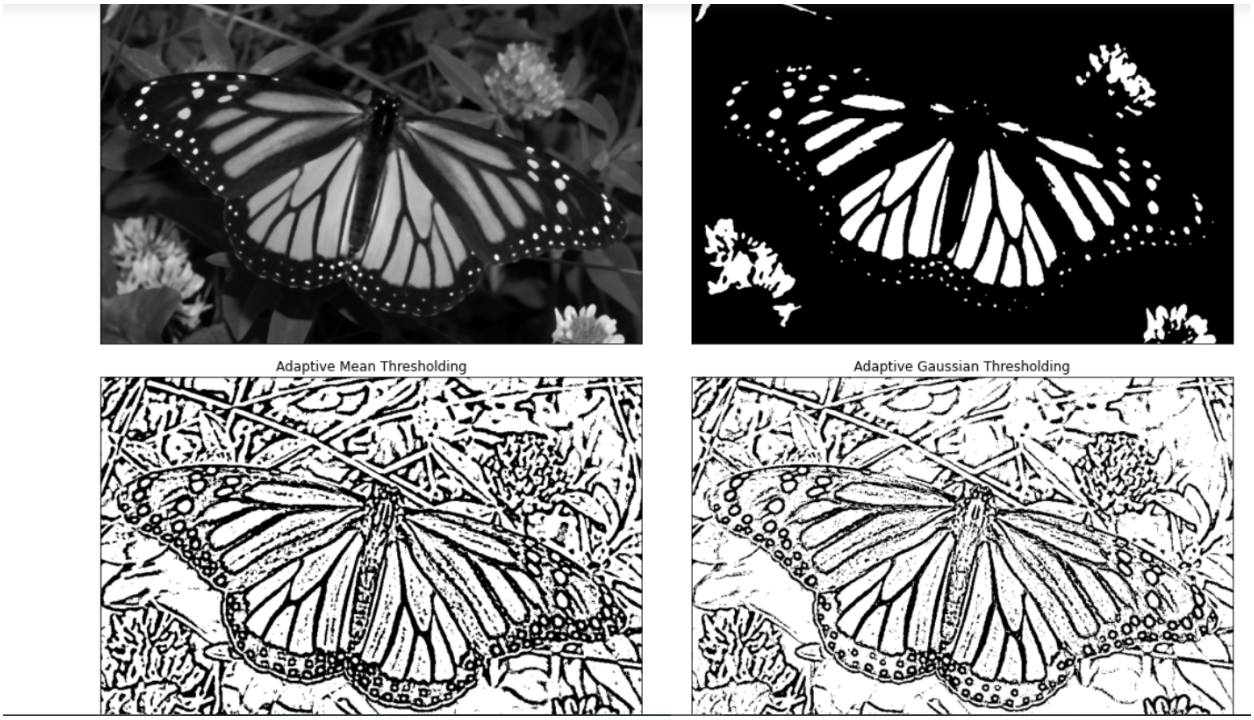


- Applying Thresholding On Image

```
#removing  noise using median blur
img = cv2.medianBlur(img_01,5)
#2nd argument is threshold,3rd argument - Value assigned if pixel is greater than threshold
ret,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
#threshold value will be 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=(15,10))
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()
```

- Output After Thresholding



- Otsu Method

```
img = cv2.medianBlur(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()
```

- Result After Otsu Method

Image After Median Filtering

Image after Otsu's Thresholding