Project Report

Project Based On Adaptive Thresholding: Edge Detection in Images

Name : HARI KRISHNAN

Course : AI and ML (Aug 2020)

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).
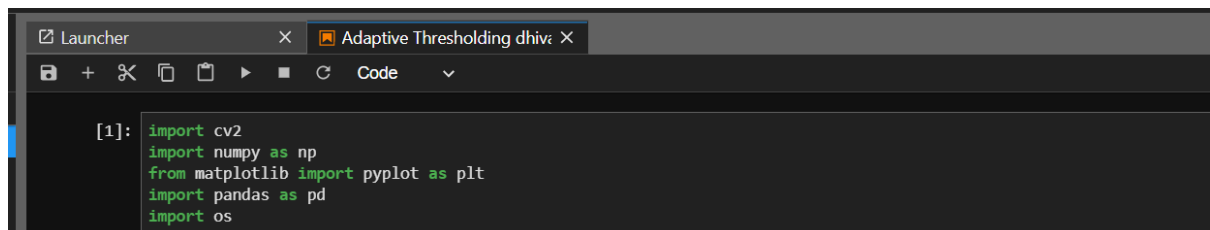
Prerequisites

What things you need to install the software and how to install them:

Python 3.6 This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/add-python-to-path-python-is-not-recognized-asan-internal-or-externalcommand/ . Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic. Second and easier option is to download anaconda and use its anaconda prompt to run the commands

To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.6 then run below commands in command prompt/terminal to install these packages:

Importing the libraries and loading dataset.

Reading the images

```python
#Reading the images
org_img_01 = cv2.imread('fruits.jpg',0)# reading the image in the grayscale image
org_img_02 = cv2.imread('im2.jpg',0)# reading the image in the grayscale image
org_img_03 = cv2.imread('im3.jpg',0)# reading the image in the grayscale image
```
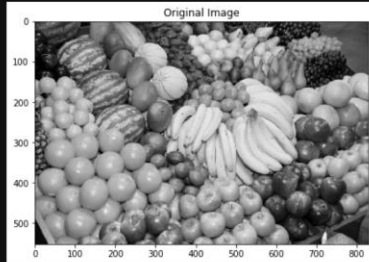
Visualizing the Images For Edge Detection

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

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

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

[3]: <matplotlib.image.AxesImage at 0x23714c0ac88>


```python
[4]: print(type(org_img_01))
     plt.figure(figsize=(7,6))
     plt.title("Original Image")
     plt.imshow(org_img_02,'gray')
```
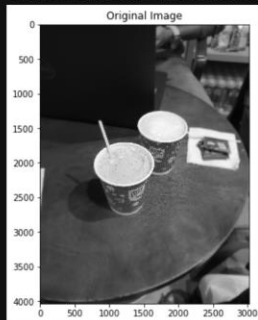
```
<class 'numpy.ndarray'>
```
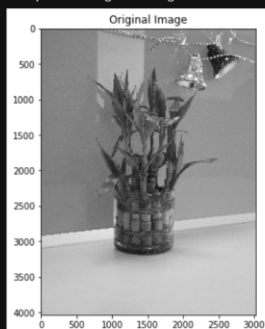
[4]: <matplotlib.image.AxesImage at 0x23714ccc2c8>


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

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

[5]: <matplotlib.image.AxesImage at 0x237168d6108>


```python
[6]: print(org_img_01.shape," ",org_img_02.shape," ",org_img_03.shape)
```

```
(553, 830)   (4032, 3024)   (4032, 3024)
```

Applying Thresholding on Image

```
[7]: 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()
```
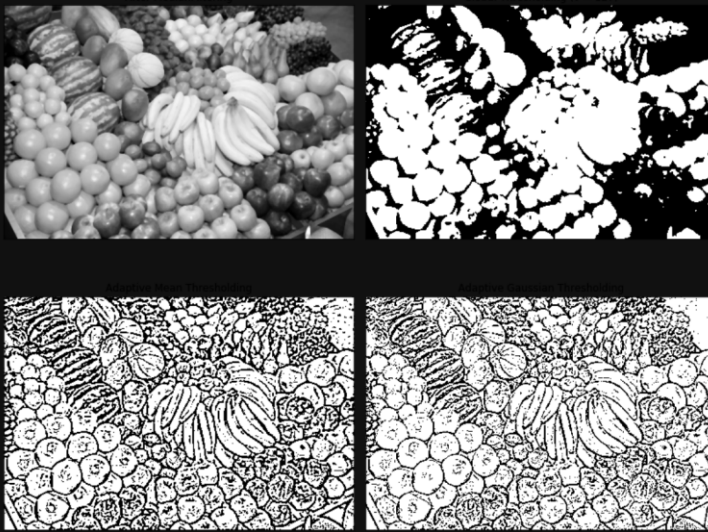

Otsu Method

```
[12]: 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()
```