Homework No.8

Student ID: 41047902S Name: 鄭淮薰

Problem Statement

Implement Otsu's thresholding method

Input / Output

Using python to run this program, and here are some parameters user can use to specify the input and output

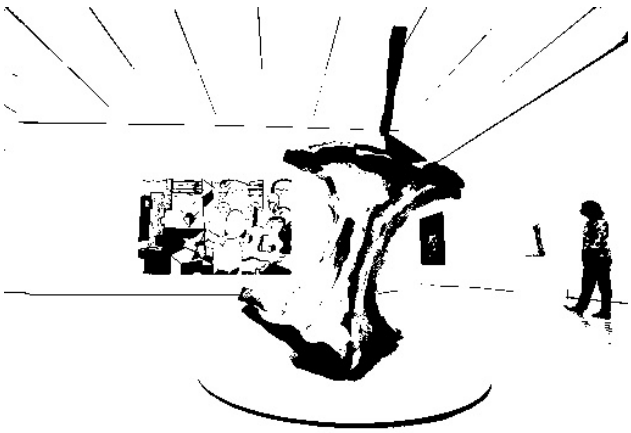
```
usage: hw8.py [-h] [-i IMAGE]

options:
  -h,         --help              | show this help message and exit
  -i IMAGE,   --image IMAGE       | path to the image
```

Results

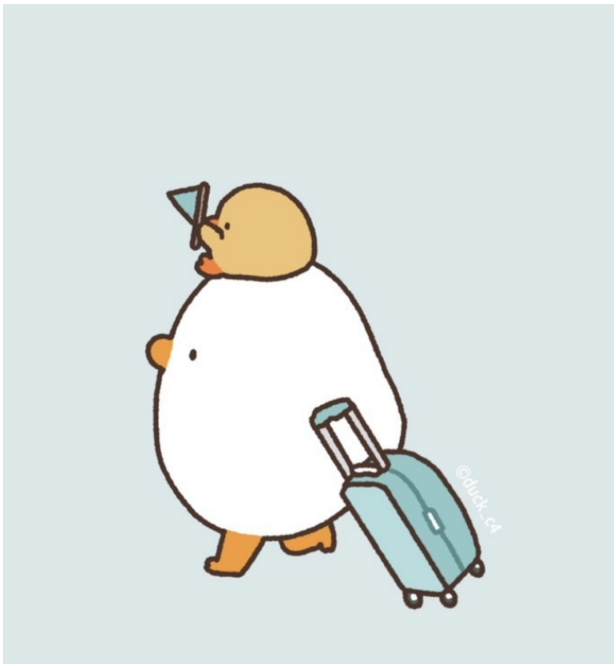
Test Case 1

Fig1.1 是一張 `apple.jpeg` 作為 input，將其轉為灰階圖片後做 Otsu's thresholding method 的結果則為 Fig1.2。

Fig1.1 Input Image	Fig1.2 Thresholded Image
	

Test Case 2

Fig2.1 是一張 `duck.png` 作為 input，將其轉為灰階圖片後做 Otsu’s thresholding method 的結果則為 Fig2.2。由於原圖邊界鮮明，在經過 Otsu's thresholding 轉換後途中物件很完整得被分割出來。

Fig2.1 Input Image	Fig2.2 Thresholded Image
	

Comment

本次作業要實作 Otsu’s thresholding method，因此我將一輸入圖片以 Otsu’s thresholding 方法處理後輸出的方式來呈現題目需求。

首先我將輸入輸數圖片轉為灰階圖片，計算影像的 histogram，接著尋遍所有可能閾值 (0 ~ ~255)，並選擇最佳的閾值作為 threshold 值。最後將灰階圖片以上述最佳值做 thresholding 處理得出最終結果。

其中影像的讀取、histogram 的計算，以及平均數、變異數等運算皆是使用 `cv2` 與 `numpy` 函式庫來處理。

從結果中可觀察到經過 Otsu’s thresholding 方法能有效的將圖片切割，擷取出圖中的物件。邊界若越完整鮮明，效果越佳。

Source Code

```
import argparse
import numpy as np
import cv2

def otsu_thresholding(image_file):
    # load image and convert to grayscale
    img = cv2.imread(image_file, cv2.IMREAD_GRAYSCALE)

    # get histogram of image
    hist = cv2.calcHist([img], [0], None, [256], [0, 256])

    # get total number of pixels
    total = img.shape[0] * img.shape[1]

    # initialize variables
    varMax = 0  # maximum variance
    threshold = 0

    # loop through all possible thresholds
    for t in range(256):
        # calculate weights
        wB = np.sum(hist[:t]) / total
        wF = np.sum(hist[t:]) / total

        # calculate means
        mB = np.sum([i * hist[i] for i in range(t)]) /
np.where(np.sum(hist[:t]) == 0, 1, np.sum(hist[:t]))
        mF = np.sum([i * hist[i] for i in range(t, 256)]) /
np.where(np.sum(hist[t:]) == 0, 1, np.sum(hist[t:]))

        # calculate variance
        var = wB * wF * (mB - mF) ** 2

        # update threshold if variance is greater than current maximum
        if var > varMax:
            varMax = var
            threshold = t

    # threshold image
    ret, thresh_img = cv2.threshold(img, threshold, 255, cv2.THRESH_BINARY)

    return thresh_img
```

```
if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    parser.add_argument('-i', '--image', type=str, default='apple.jpeg',
help='path to the image')
    args = parser.parse_args()
    input_file = args.image
    output_file = input_file.split('.')[0] + '_thresh.' + input_file.split('.')
[1]

    thresh_img = otsu_thresholding(args.image)
    cv2.imwrite(output_file, thresh_img)
```