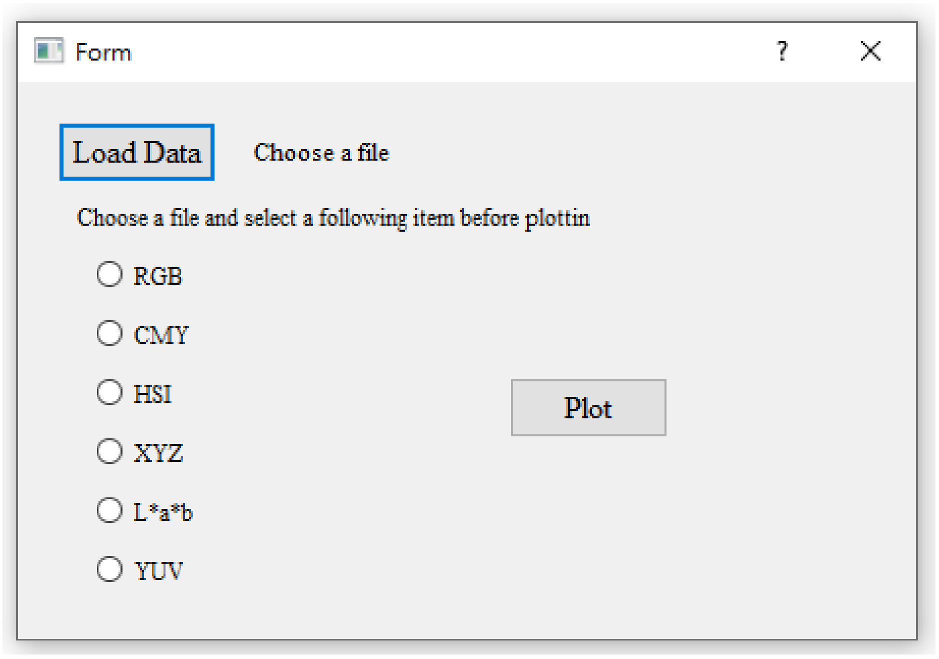
Homework5

R08525116 吳承哲

Program 請執行 part1.py、part2.py、part3.py

Part 1

先點選一個模式，然後按 Plot



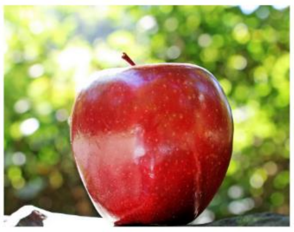
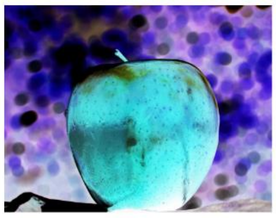
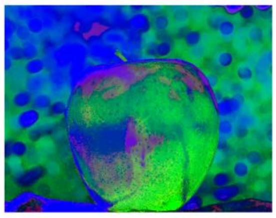
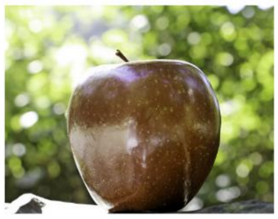
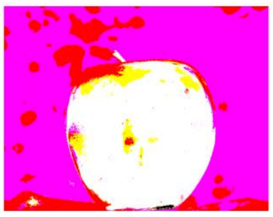
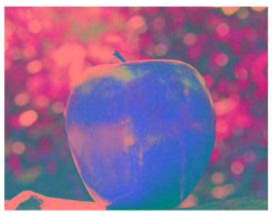
使用 opencv 讀檔會是 BGR 的影像，因此先轉換成 RGB

```
self.original_img = cv2.cvtColor(
                self.original_img, cv2.COLOR_BGR2RGB)  # convert BGR to RGB
```

依照公式去做轉換

```
def RGB2CMY(img):
    new_img = np.zeros((img.shape))
    RGB_scale = 255.0

    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            R = float(img[i, j, 0])
            G = float(img[i, j, 1])
            B = float(img[i, j, 2])

            C = 1 - R / RGB_scale
            M = 1 - G / RGB_scale
            Y = 1 - B / RGB_scale
```

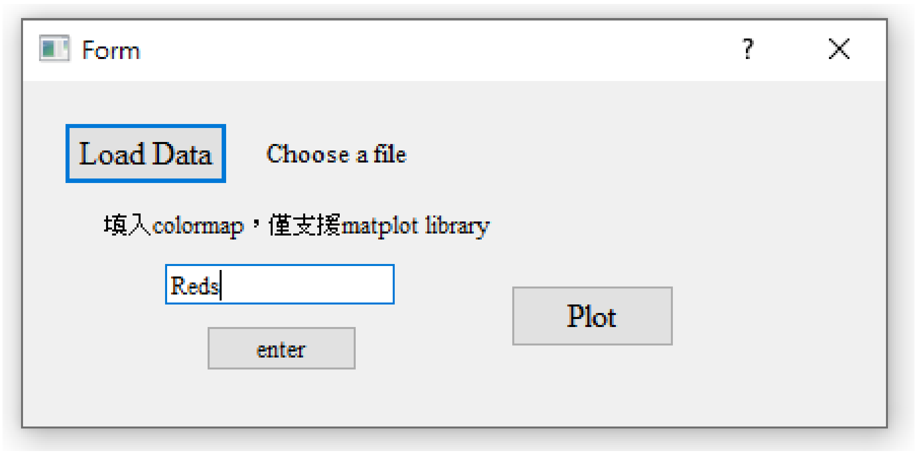
```python
            new_img[i, j, 0] = C
            new_img[i, j, 1] = M
            new_img[i, j, 2] = Y
    return new_img

def RGB2HSI(img):
    new_img = np.zeros((img.shape))

    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            R = float(img[i, j, 0])
            G = float(img[i, j, 1])
            B = float(img[i, j, 2])

            fraction = ((R-G) + (R-B)) / 2
            denominator = np.sqrt((R-G)**2 + (R-B)*(G-B))
            if denominator == 0:
                H = 0
            else:
                theta = np.arccos(fraction / denominator)
                if B <= G:
                    H = theta
                else:
                    H = 360 - theta

            S = 1 - (3 / (R+G+B+1e-7))*np.min((R, G, B))
            I = (R+G+B) / 3

            new_img[i, j, 0] = H
            new_img[i, j, 1] = S*255
            new_img[i, j, 2] = I
    new_img = new_img.astype(np.uint8)
    return new_img

def RGB2XYZ(img):
    M = np.array([[0.412453, 0.357580, 0.180423],
                  [0.212671, 0.715160, 0.072169],
                  [0.019334, 0.119193, 0.950227]])
    new_img = np.zeros((img.shape))

    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            R = float(img[i, j, 0])
            G = float(img[i, j, 1])
            B = float(img[i, j, 2])
            rgb = np.array([R, G, B])
            XYZ = np.dot(M, rgb.T)
            XYZ = XYZ / 255.0
            X = XYZ[0] / 0.95047
            Y = XYZ[1] / 1.0
            Z = XYZ[2] / 1.08883
            new_img[i, j, 0] = X
            new_img[i, j, 1] = Y
            new_img[i, j, 2] = Z
    return new_img

def h(q):
    return np.power(q, 1 / 3) if q > 0.008856 else 7.787 * q + 0.137931
def RGB2Lab(img):
    M = np.array([[0.412453, 0.357580, 0.180423],
                  [0.212671, 0.715160, 0.072169],
                  [0.019334, 0.119193, 0.950227]])
    new_img = np.zeros((img.shape))

    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            R = float(img[i, j, 0])
            G = float(img[i, j, 1])
            B = float(img[i, j, 2])
            rgb = np.array([R, G, B])
            XYZ = np.dot(M, rgb.T)
            XYZ = XYZ / 255.0

            h_XYZ = [h(x) for x in XYZ]
            L = 116 * h_XYZ[1] - 16 if XYZ[1] > 0.008856 else 903.3 * XYZ[1]
            a = 500 * (h_XYZ[0] - h_XYZ[1])
            b = 200 * (h_XYZ[1] - h_XYZ[2])

            new_img[i, j, 0] = L
            new_img[i, j, 1] = a
            new_img[i, j, 2] = b
    return new_img

def RGB2YUV(img):
    m = np.array([[0.29900, -0.16874,  0.50000],
                  [0.58700, -0.33126, -0.41869],
                  [0.11400, 0.50000, -0.08131]])
    new_img = np.zeros((img.shape))

    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            R = float(img[i, j, 0])
            G = float(img[i, j, 1])
            B = float(img[i, j, 2])
            rgb = np.array([R, G, B])
            yuv = np.dot(rgb, m)
            yuv[1:] += 128.0

            Y = yuv[0]
            U = yuv[1]
            V = yuv[2]

            new_img[i, j, 0] = Y
            new_img[i, j, 1] = U
            new_img[i, j, 2] = V

    new_img = new_img.astype(np.uint8)
    return new_img
```

RGB	
CMY 類似負片的感覺，CMY 影像跟 RGB 影像友互補的關係。	
HSI 使用 Hue、Saturation、Intensity 來描述顏色。	
XYZ 用來表達人類眼睛看到的全部色彩。	
Lab 使用數值化的方式來表達人眼所見的影像，適合用來分離前景和背景。	
YUV 亮度存在 Y channel，而色度信號存在 U、V channel。	

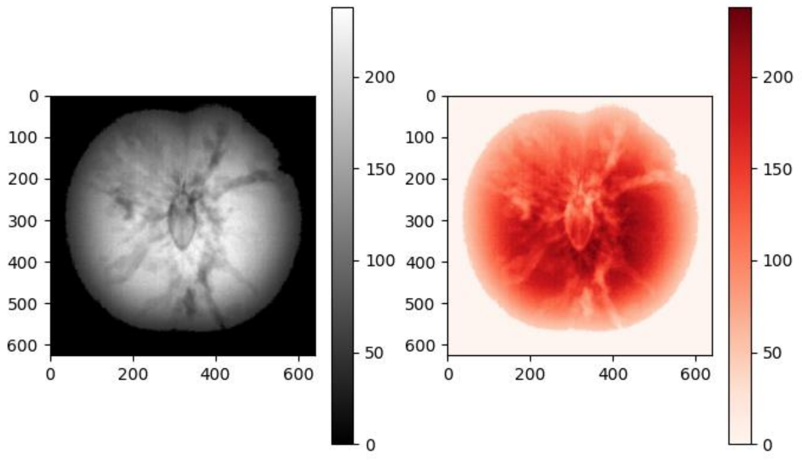
Part 2

```
 # convert BGR to GRAY
self.original_img = cv2.cvtColor(
                self.original_img, cv2.COLOR_BGR2GRAY)

def plot(img, colormap="Reds"):
    plt.subplot(121)
    plt.imshow(img, cmap="gray")
    plt.colorbar()
    plt.subplot(122)
    plt.imshow(img, cmap=colormap)
    plt.colorbar()
    plt.show()

def choose_color(self):
        text = str(self.ui.lineEdit.text())
        self.color = text
```

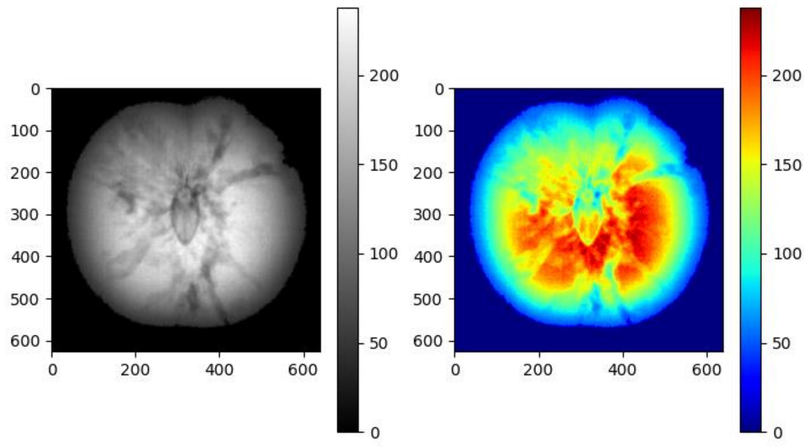
空格內可以輸入 matplot library 的 colormap
e.g. Reds, Greens, Blues, jet, summer, autumn, winter…



Colormap : Reds



Colormap : jet

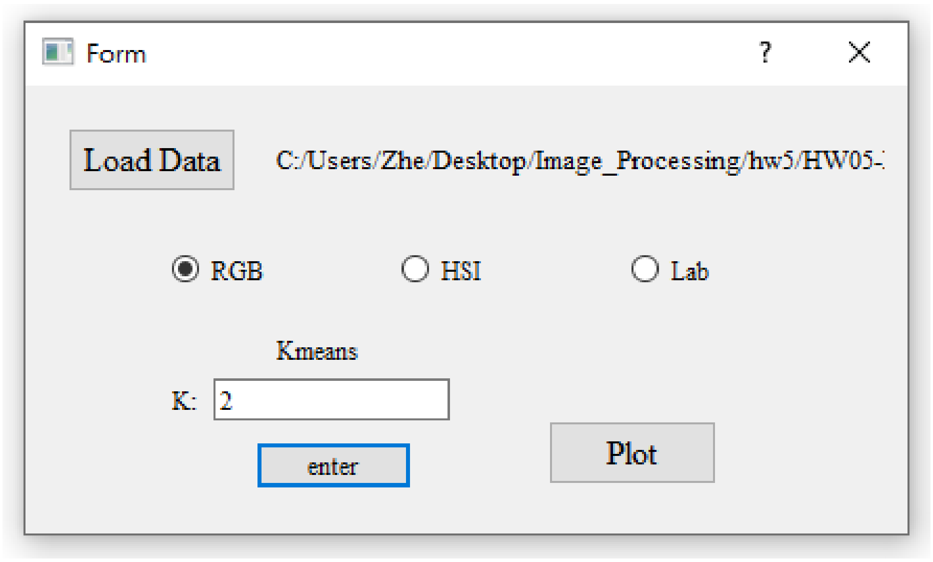


由上面結果可看出，設置更多的顏色，圖片分離的效果越好，更能清楚分辨不同灰階的部分。

Part 3

```
def segmentation(img, K=2):
    img_tmp = img.reshape((img.shape[0]*img.shape[1], 3))
    img_tmp = img_tmp.astype(np.float32)

    criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0
)
    ret, label, center = cv2.kmeans(
        img_tmp, K, None, criteria, 10, cv2.KMEANS_RANDOM_CENTERS)

    new_img = label.reshape((img.shape[0], img.shape[1]))
    return new_img

def plot(img, K=2):
    new_img = segmentation(img, K)
    plt.subplot(121)
    plt.imshow(img, cmap="gray")
    plt.subplot(122)
    plt.imshow(new_img, cmap="gray")
    plt.show()

def choose_K(self):
        text = int(self.ui.lineEdit.text())
        self.K = text
```

先選擇一個模式，再輸入 K 值，最後點選 Plot



C:/Users/Zhe/Desktop/Image_Processing/hw5/HW05-

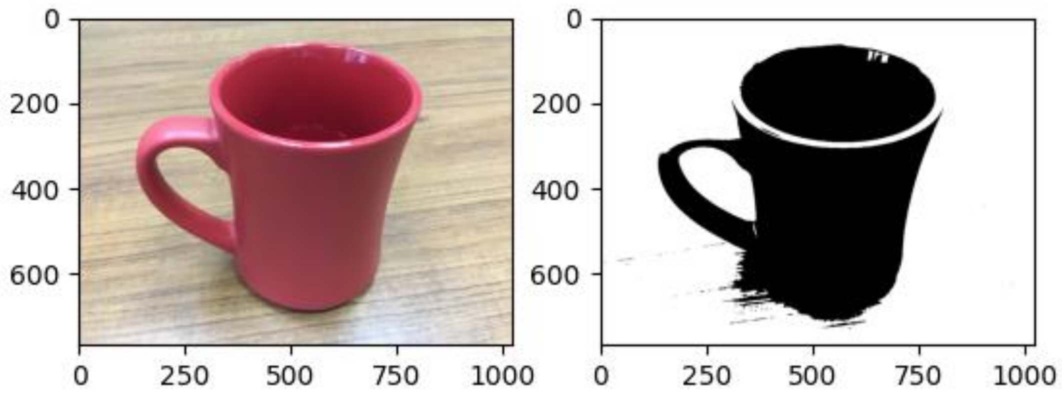
[✓ RGB] HSI Lab

Kmeans

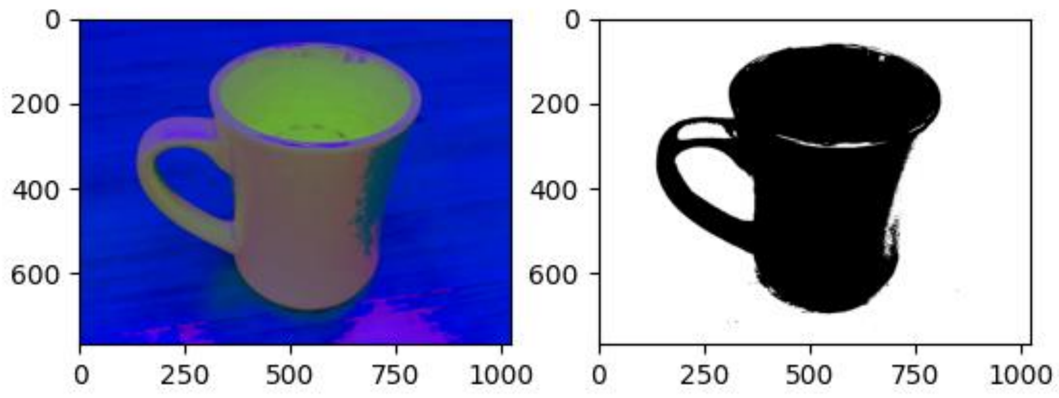
K: 2

K = 2

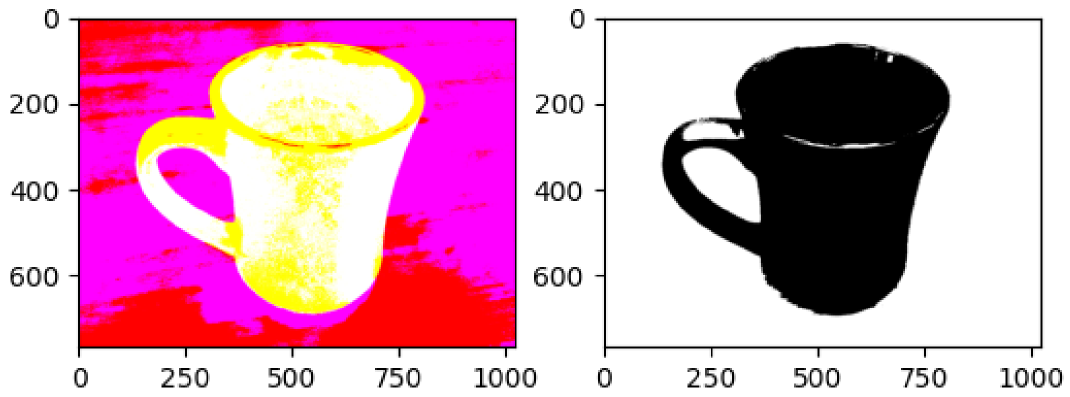
RGB



HSI



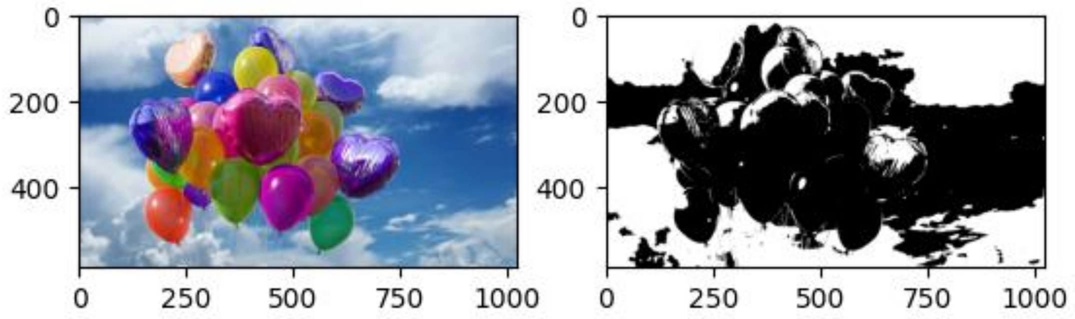
Lab



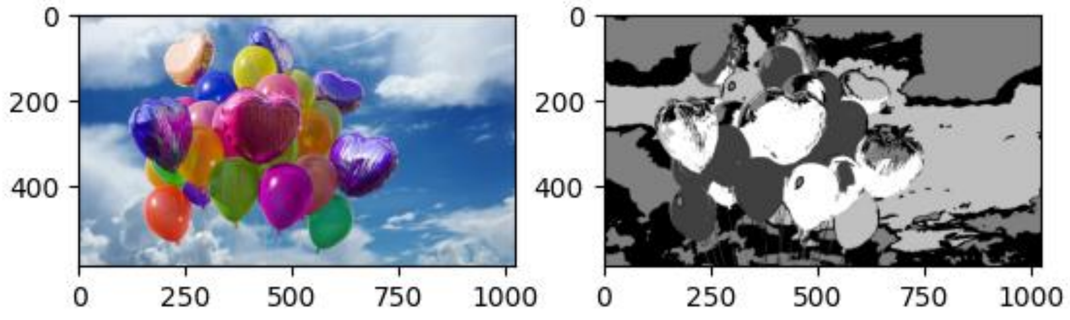
由上面的比較可以看出，Lab 的分離效果最好，而 RGB 跟 HIS 都有些微的雜訊。

RGB

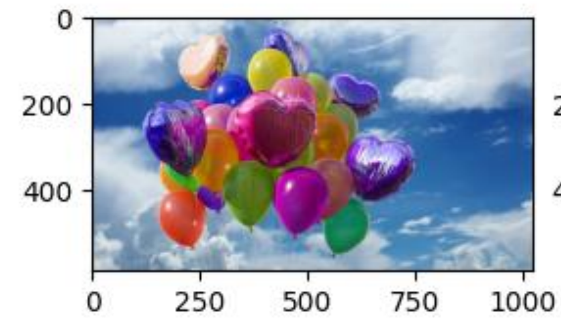
K = 2

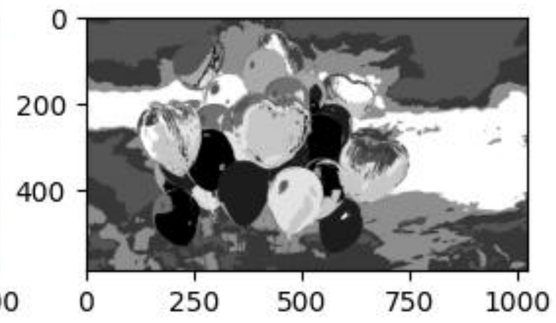


K = 3



K = 4





K 值越大，可以分離的物體越多。