Computer Vision HW4

111598066 資工碩一 許哲維

I、Explain Program and Method

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

Import three Python libraries.

```
def RGB2Gray(image):
    return np.dot(image[...,:3], [0.21, 0.72, 0.07]).astype(np.uint8)
```

Convert every RGB image’s bit to [0.21, 0.72, 0.07] to get the grayscale image. (Base on luminosity method)

```
def Gaussian_Blur(image):
    x, y = np.mgrid[-1:2, -1:2]
    kernel = np.exp(-(x * x + y *y))
    kernel = kernel / kernel.sum()
    imagepad = np.zeros((int(image.shape[0]+2), int(image.shape[1]+2)))
    imagepad[1:image.shape[0]+1, 1:image.shape[1]+1] = image
    result = np.zeros((int(image.shape[0]), int(image.shape[1])))

    for y in range(0, result.shape[1]):
        for x in range(0, result.shape[0]):
            result[x, y] = (kernel * imagepad[x: x + 3, y: y + 3]).sum()

    return result.astype(np.uint8)
```

The Gaussian Blur operation is using a 3*3 box to take turns create the image blur.

```
def Set_Initial_Points (center, radius, n):
    t = np.linspace(0, 2 * np.pi, n+1)
    x = center[0] + radius[0] * np.cos(t)
    y = center[1] + radius[1] * np.sin(t)

    return np.array([x, y])
```

The Set_Initial_Points is drawing a circle to initial the points.

```
def getDiagCycleMat(n):
    a = 2 * 0.015 + 6 * 10
    b = -(0.015 + 4 * 10)
    c = 10
    diag_mat_a = a * np.eye(n)
```

```
    diag_mat_b = b * np.roll(np.eye(n), 1, 0) + b * np.roll(np.eye(n), -1, 0)
    diag_mat_c = c * np.roll(np.eye(n), 2, 0) + c * np.roll(np.eye(n), -2, 0)
    return diag_mat_a + diag_mat_b + diag_mat_c

def getGaussianPE(src):
    #blur = Gaussian_Blur(src)
    blur = cv2.GaussianBlur(src, ksize=(5, 5), sigmaX=3)
    dx = cv2.Sobel(blur, cv2.CV_16S, 1, 0)
    dy = cv2.Sobel(blur, cv2.CV_16S, 0, 1)
    E = dx**2 + dy**2
    return E

def Find_The_Contour (img, init):
    x, y, errs = init[0].copy(), init[1].copy(), []
    n = len(x)
    A = getDiagCycleMat(n)
    inv = np.linalg.inv(A + 0.001 * np.eye(n))
    e = -getGaussianPE(img)
    fx = cv2.Sobel(e, cv2.CV_16S, 1, 0)
    fy = cv2.Sobel(e, cv2.CV_16S, 0, 1)
    T = np.max([abs(fx), abs(fy)])
    fx, fy = fx / T, fy / T
    for g in range(10000):
        x_pre, y_pre = x.copy(), y.copy()
        i, j = np.uint8(y), np.uint8(x)
        try:
            xn = inv @ (0.001 * x + fx[i, j])
            yn = inv @ (0.001 * y + fy[i, j])
        except Exception as e:
            print("Error")
        x, y = xn, yn
        err = np.mean(0.5 * np.abs(x_pre - x) + 0.5 * np.abs(y_pre - y))
        errs.append(err)
        if err < 0.03:
            break
    return x, y
```

The function will find the contour, but unfortunately it still cannot find it before deadline.

```
def main():
    src = cv2.imread("D:/CV/CV_HW4/test_images/pic3.jpg") #改成圖片路徑
    img = RGB2Gray(src)
    img = Gaussian_Blur(img)
    W, H = img.shape

    init = Set_Initial_Points((H//2, W//2), (H//2, W//2), 500)

    x, y = Find_The_Contour(img, init)

    plt.figure()
    plt.imshow(img, cmap="gray")
    plt.plot(init[0], init[1], '--r', lw=1)
    plt.plot(x, y, 'g', lw=1)
    plt.xticks([]), plt.yticks([]), plt.axis("off")
    plt.show()

main()
```

The program will perform grayscale conversion, draw initial point, and draw contour point on image.

II、Result Images

1. pic1.jpg

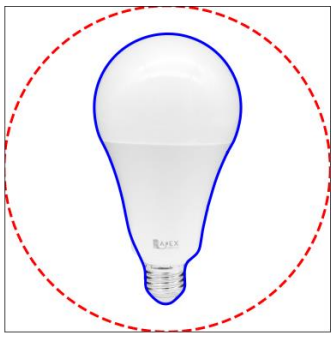
Origin	Set initial points	Find the contour
		

2. pic2.jpg

Origin	Set initial points	Find the contour
		

3. pic3.jpg

Origin	Set initial points	Find the contour
		

pic1.jpg	pic2.jpg	pic3.jpg
		

You can see that we cannot find contour point, but if I use library it can find it.
My Code : shorturl.at/eEPY4

※ All the original outputs can be seen from the attached file.