CV HW1

R11922196 林佑鑫

Part 1:

(a)

Brief discription, algorithm: 以 for 迴圈經過影像上每個 row，將原始影像的第 h 個 row 寫進新影像的倒數第 h 個 row。

Parameters: None

Principal code fragment:

```
4  def upside_down(img):
5      height, width, _ = img.shape
6      upside_down_img = np.zeros_like(img)
7
8      for h in range(height):
9          upside_down_img[height-1-h] = img[h]
10
11     return upside_down_img
```

Resulting image:

(b)

Brief discription, algorithm: 以 for 迴圈經過影像上每個 column，將原始影像的第 w 個 column 寫進新影像的倒數第 w 個 row。

Parameters: None

Principal code fragment:

```python
def right_side_left(img):
    height, width, _ = img.shape
    right_side_left_img = np.zeros_like(img)

    for w in range(width):
        right_side_left_img[:, width-1-w] = img[:, w]

    return right_side_left_img
```

Resulting image:

(c)

Brief discription, algorithm: 以 for 迴圈經過影像上每個 pixel，將原始影像的(h, w) pixel 的值寫進新影像的(w, h) pixel 中。

Parameters: None

Principal code fragment:

```python
def diagonally_flip(img):
    height, width, _ = img.shape
    diagonally_flip_img = np.zeros_like(img)

    for h in range(height):
        for w in range(width):
            diagonally_flip_img[w, h] = img[h, w]

    return diagonally_flip_img
```

Resulting image:

Part 2:

(d)

Brief discription, algorithm: 以 cv2.getRotationMatrix2D 得到順時針旋轉 45 度的旋轉矩陣，再以 cv2.warpAffine 乘上，得到旋轉後的影像。

Parameters: degree = 旋轉的角度、以逆時針爲基準，故順時針轉 45 度參數爲-45。

Principal code fragment:

```python
def rotate_degree(img, degree):
    height, width, _ = img.shape
    center = (height // 2, width // 2)
    M = cv2.getRotationMatrix2D(center, degree, 1.0)

    rotate_img = cv2.warpAffine(img, M, (width, height))
    return rotate_img
```

Resulting image:

(e)

Brief discription, algorithm: 建立 size 爲 1/2 的小圖，小圖中每個 pixel 都是取原圖的 2*h, 2*w 的數值

Parameters: ratio 爲縮小比率，此爲 2

Principal code fragment:

```python
def shrink(img, ratio):
    height, width, _ = img.shape
    new_height = height // ratio
    new_width = width // ratio
    shrink_img = np.zeros((new_height, new_width, 3))

    for h in range(new_height):
        for w in range(new_width):
            shrink_img[h, w] = img[h * ratio, w * ratio]
```

Resulting image:

(f)

Brief discription, algorithm: 對影像中每個 pixel，若 pixel 數值大於等於 128，則在新圖上同位置值爲 255，反之爲 0

Parameters: None

Principal code fragment:

```
52  def binarize(img):
53      height, width, _ = img.shape
54      binarize_img = np.zeros_like(img)
55
56      for h in range(height):
57          for w in range(width):
58              if((img[h, w] >= 128).any()):
59                  binarize_img[h, w] = 255
60              else:
61                  binarize_img[h, w] = 0
62      return binarize_img
```

Resulting image: