Computer Vision HW1 楊閎喻 R09921012

Part 1

Write a program to do the following requirement.

(a). upside-down lena.bmp

```
int center_i = img.rows / 2;
int center_j = img.cols / 2;
for (int i = 0; i < center_i; i++) {
    for (int j = 0; j < img.cols; j++){
        uchar temp = img.ptr<uchar>(i)[j];
        img.ptr<uchar>(i)[j] = img.ptr<uchar>(img.rows - i - 1)[j];
        img.ptr<uchar>(img.rows - i - 1)[j] = temp;
    }
}
```

- 使用 center_i & center_j 去紀錄圖片中心點，作為翻轉依據。
- 利用雙重迴圈走訪每個上半部 pixel，將其與對應的下半部 pixel 交換

結果：

(b). right-side-left lena.bmp

```
center_i = img.rows / 2;
center_j = img.cols / 2;
for (int i = 0; i < img.rows; i++) { //change cols data
    for (int j = 0; j < center_j; j++) {
        uchar temp = img.ptr<uchar>(i)[j];
        img.ptr<uchar>(i)[j] = img.ptr<uchar>(i)[img.cols - j - 1];
```

```
            img.ptr<uchar>(i)[img.cols - j - 1] = temp;
        }
    }
```

- 使用 center_i & center_j 去紀錄圖片中心點，作為翻轉依據。
- 利用雙重迴圈走訪每個左半部 pixel，將其與對應的下半部 pixel 交換

結果：

(c). diagonally flip lena.bmp

```
    center_i = img.rows / 2;
    center_j = img.cols / 2;
    for (int i = 0; i < img.rows; i++) { //change rows&cols data
        for (int j = 0; j < i; j++) {
            uchar temp = img.ptr<uchar>(i)[j];
            img.ptr<uchar>(i)[j] = img.ptr<uchar>(j)[i];
            img.ptr<uchar>(j)[i] = temp;
        }
    }
```

- 使用 center_i & center_j 去紀錄圖片中心點，作為翻轉依據。
- 利用雙重迴圈走訪每個 pixel，將其與對應的 pixel 交換(亦即轉置矩陣)

結果(使用左上至右下的對角線翻轉)：

Part 2

Write a program or use software to do the following requirement.

(d). rotate lena.bmp 45 degrees clockwise

```
    Point2f a(399, 399); //rotate center
    Mat rot_mat(2, 3, CV_32FC1);
    rot_mat = getRotationMatrix2D(a, -45, 1); //find rotation tf matrix
    Size dsize(800, 800); //new img size

    Mat trans_mat = (Mat_<double>(2, 3) << 1, 0, 400-256,
                                           0, 1, 400-256); // set
transfermation matrix
    warpAffine(img, img, trans_mat, dsize); //transfer
    warpAffine(img, img, rot_mat, dsize); //rotate
```

- Point2f a 去選定圖片旋轉中心。
- 為避免圖片旋轉後被裁切，使用 Size dsize 設定新的大小
- rot_mat & trans_mat 分別對應旋轉與平移矩陣
- 使用 warpAffine 函式操作圖片

結果：

(e). shrink lena.bmp in half

```
resize(img, img,Size(img.rows/2,img.cols/2), 0, 0); //change img size to half
```

- 使用 resize 重新定義圖片大小

結果：

(f). binarize lena.bmp at 128 to get a binary image

```
for (int i = 0; i < img.rows; i++) {
        for (int j = 0; j < img.cols; j++) {
            if (img.ptr<uchar>(i)[j] > 128) { //set threadhold
                file.write("1", 1);
                img.ptr<uchar>(i)[j] = 255;
            }
            else {
                file.write("0", 1);
                img.ptr<uchar>(i)[j] = 0;
            }
        }
        file.write("\n", 1);
    }
```

- 走訪每個 pixel，使用 128 作為判斷更改黑白的標準，大於 128 則設為 255，小於 128 則設為 0

結果：