HW1

110511010 楊育陞

```
python3 hw1.py -a -i <image_path>
```

Method

Rotation

使用旋轉矩陣，將圖片旋轉：

1. 首先要將圖片中心設為旋轉軸心，設為新坐標系中心，轉換矩陣M
2. 使用旋轉矩陣R 旋轉座標
3. 將坐標系轉回原圖坐標系，使用轉換矩陣M^{-1}

將三個矩陣依序相乘出轉換矩陣$T = M^{-1}RM$，我將這個矩陣乘上新圖片座標，就能逆推回對應的原圖座標，其中因為轉移出來的座標可能為浮點數，要再另外進行插值後，再填入新圖片中。

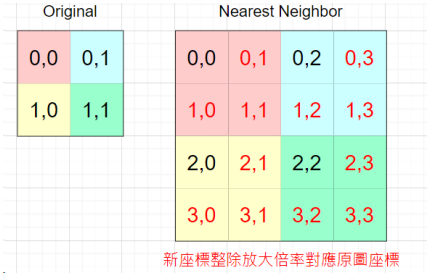
$$T = M^{-1}RM = \begin{bmatrix} 1 & 0 & x_{pivot} \\ 0 & 1 & y_{pivot} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(-30) & -sin(-30) & 0 \\ sin(-30) & cos(-30) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_{pivot} \\ 0 & 1 & -y_{pivot} \\ 0 & 0 & 1 \end{bmatrix}$$

Enlarge

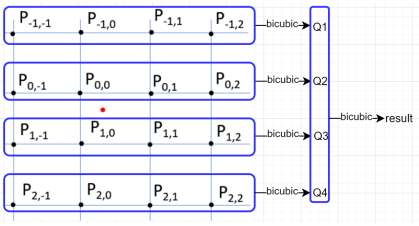
將原圖長寬乘以2作放大，再透過插值法填補原圖沒有的像素。

Interpolation

- Nearest Neighbor
  我取鄰居中最接近原點的像素：在圖片旋轉中，我直接將座標浮點數捨去；而在放大中，直接將座標整除放大倍率。
- Bilinear
  我先對X方向做兩次線性內插，再依兩個結果對y方向做內插。



- Bicubic
  因為三次內插會有overflow的問題，因此先將圖片資料類別換成int32便於計算，內插完成後再clip到0-255，並轉換回uint8。三次內插考慮內插該點周遭16個點，我首先對X方向做四遍三次內插，再依四個結果對y方向做三次內插。



Result

- 旋轉30度 - Nearest

- 旋轉30度 - Bilinear

- 旋轉30度 - Bicubic

- 放大2倍 - Nearest

- 放大2倍 - Bicubic

- 放大2倍 - Bilinear

- Nearest Neighbor
  像素方塊明顯

- Bilinear
  仍有些微顆粒感

- Bicubic
  效果最好，線條最平順

Feedback

這次作業我遇到的問題是，在三次插值中，我起初只有在計算完成後才做clip到0-255，因此圖片上產生許多黑點，後來才發現因為原本圖片資料型別是uint8，在計算過程中就會可能產生overflow，所以在計算前就要先轉換成較大範圍且有號的型別。

透過旋轉與插值的練習，我學會了opencv與numpy的基本用法，也重新熟悉了矩陣運算和邊界條件的考慮等等。另外這次作業的三次插值用了cubic spline interpolation，透過斜率近似導數，上課中沒有特別提到，我覺得是十分有趣且有效的做法。