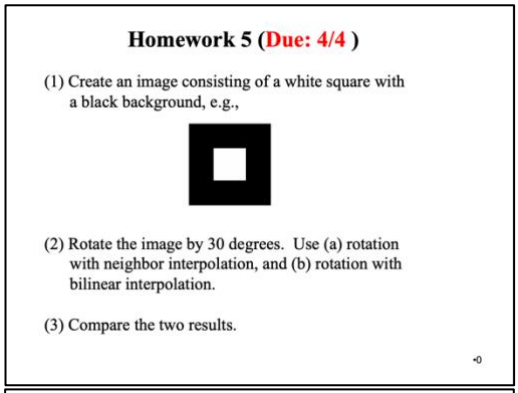
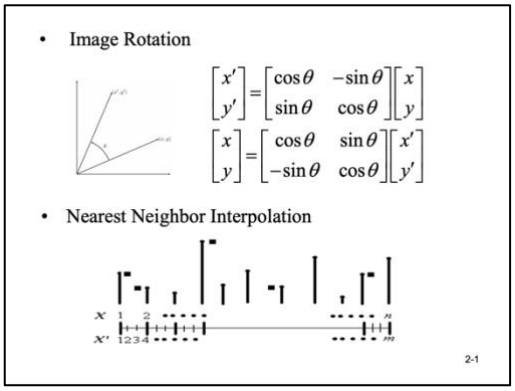
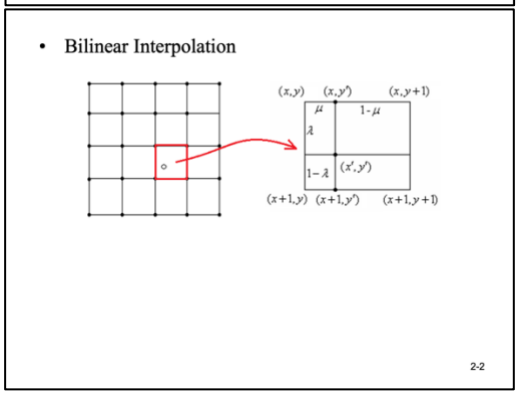
41047011S 資工 114 趙安庭

(1) Assignment statement





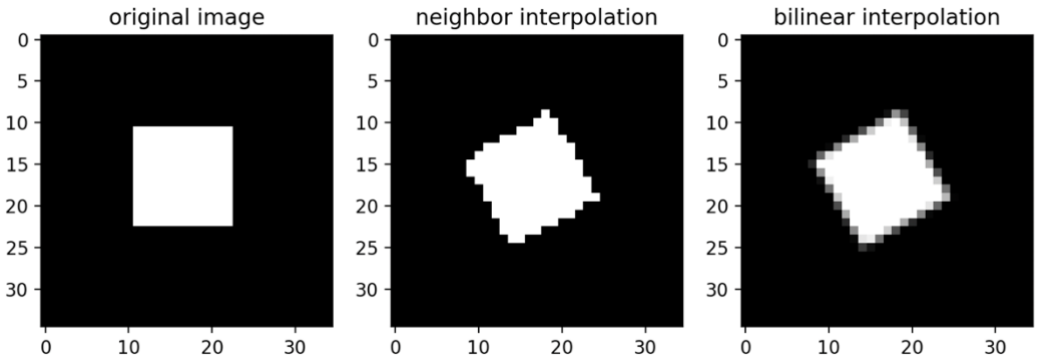


(2)

(a) Input/output images

Input: 此份作業程式碼沒有 user input。

Output:



(b) Source code

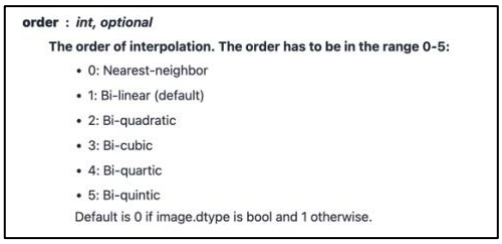
hw05.py > ...

```python
import sys
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import skimage.io as io
import skimage.transform as trans
side = 35
img = np.zeros([side, side])

for y in range(side//3, side*2//3):
    for x in range(side//3, side*2//3):
        img[y][x] = 255

neighbor_interp = trans.rotate(img, 30, order=0)
bilinear_interp = trans.rotate(img, 30, order=1)

plt.subplot(1, 3, 1)
plt.imshow(img, cmap='gray')
plt.title("original image")

plt.subplot(1, 3, 2)
plt.imshow(neighbor_interp, cmap='gray')
plt.title("neighbor interpolation")

plt.subplot(1, 3, 3)
plt.imshow(bilinear_interp, cmap='gray')
plt.title("bilinear interpolation")

plt.show()
```

(c) Comments

這次作業首先創造一個 35*35 的黑白圖片（黑底、正中央有約 1/9 的區域是白色正方形），再使用 scikit-image 的 transform.rotate() 函式來旋轉圖片，order=0 代表使用 neighbor interpolation，order=1 則是表示使用 bilinear interpolation 旋轉圖片（下圖中包含 order 的值分別代表的含義，來源自 scikit-image 官方 document）。

儘管目前已有現成函式庫可以用來處理圖像，就算不知道原理也能順利處理圖片，並且使用上相當簡潔方便，但若是能了解這些轉換背後的含義及原理，使用別人寫好的函式時心裡會比較踏實，因為知道這個函式實際做了什麼，另外，可能會遇到某些情況下需要自行撰寫處理、旋轉圖片的程式，這時就必須了解圖片旋轉投射的原理。


order : *int, optional*

The order of interpolation. The order has to be in the range 0-5:

- 0: Nearest-neighbor
- 1: Bi-linear (default)
- 2: Bi-quadratic
- 3: Bi-cubic
- 4: Bi-quartic
- 5: Bi-quintic

Default is 0 if image.dtype is bool and 1 otherwise.