第10章
影像的幾何變換

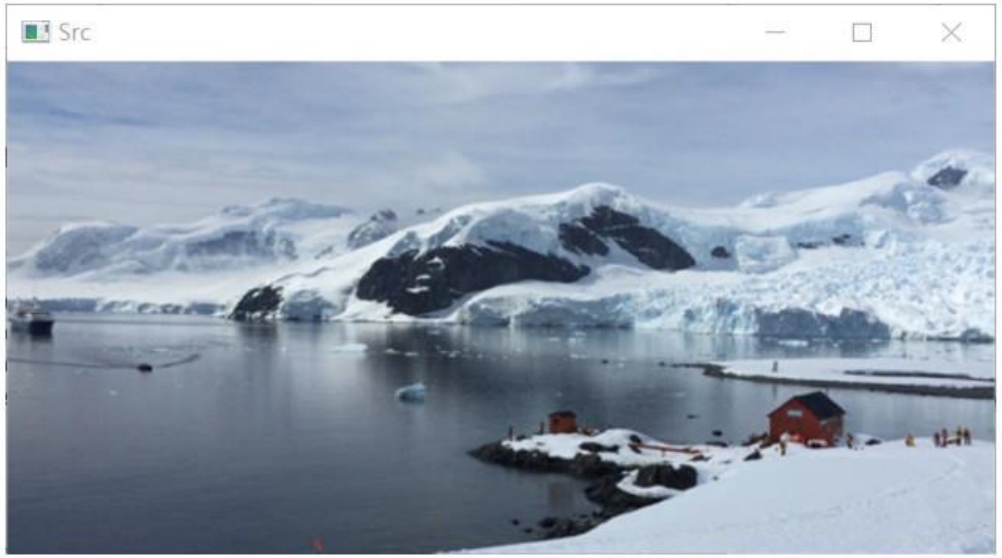
10-1：影像縮放效果

- dst = cv2.resize(src, dsize, fx, fy, interpolation)

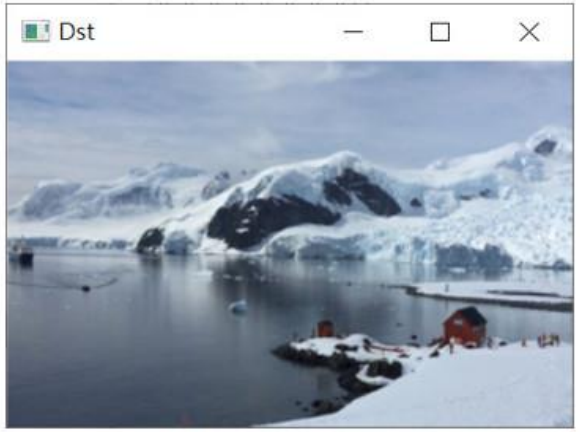
具名常數	值	說明
INTER_NEAREST	0	最近插值法
INTER_LINEAR	1	雙線性插值法，在插入點選擇 4 個點進行插值處理，這是預設的方法
INTER_CUBIC	2	雙三次插值法，可以創造更平滑的邊緣影像
INTER_AREA	3	對影像縮小重新採樣的首選方法，但是影像放大時類似最近插值法
INTER_LENCZOS4	4	Lencz 的插值方法，這個方法會在 x 和 y 的方向分別對 8 個點進行插值

10-1-1：使用dsize參數執行影像縮放

- 程式實例ch10_1.py：使用dsize參數將影像更改為width = 300，height = 200的實例。



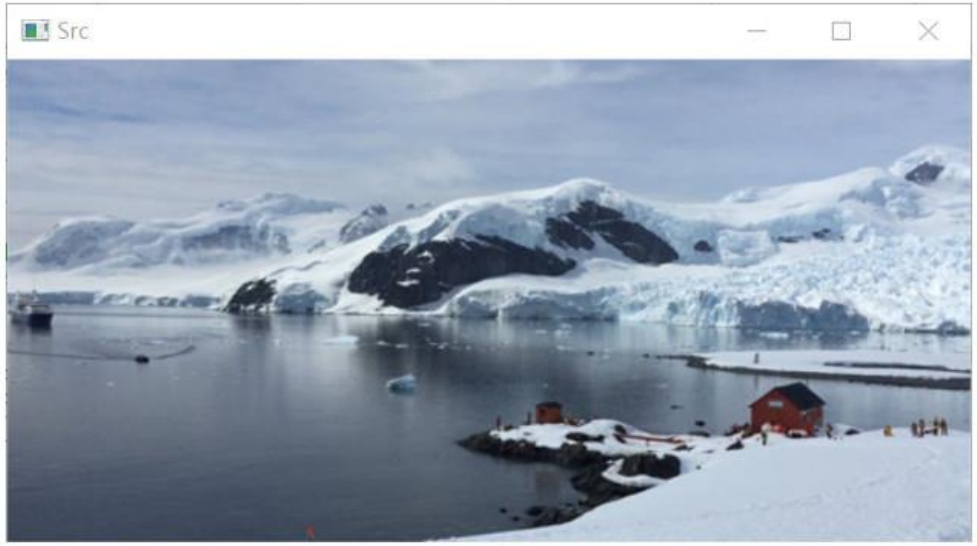
原始影像



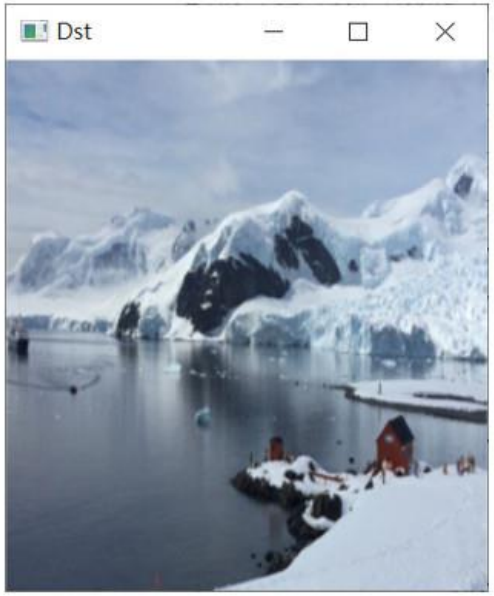
新影像

10-12：使用fx和fy執行影像的縮放

- 程式實例ch10_2.py：使用fx = 0.5，fy = 1.1更改影像大小，這個程式在執行時同時會列出原始影像和新影像的大小。



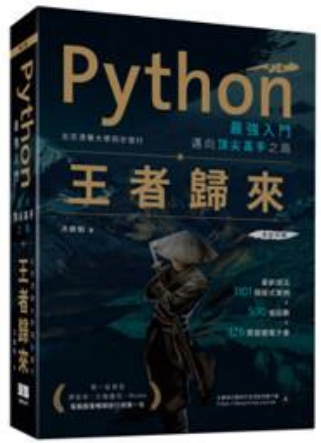
原始影像



新影像

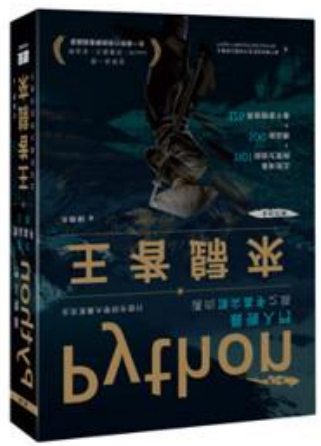
10-2：影像翻轉

水平翻轉或稱 y 軸翻轉



x

垂直翻轉或稱 x 軸翻轉



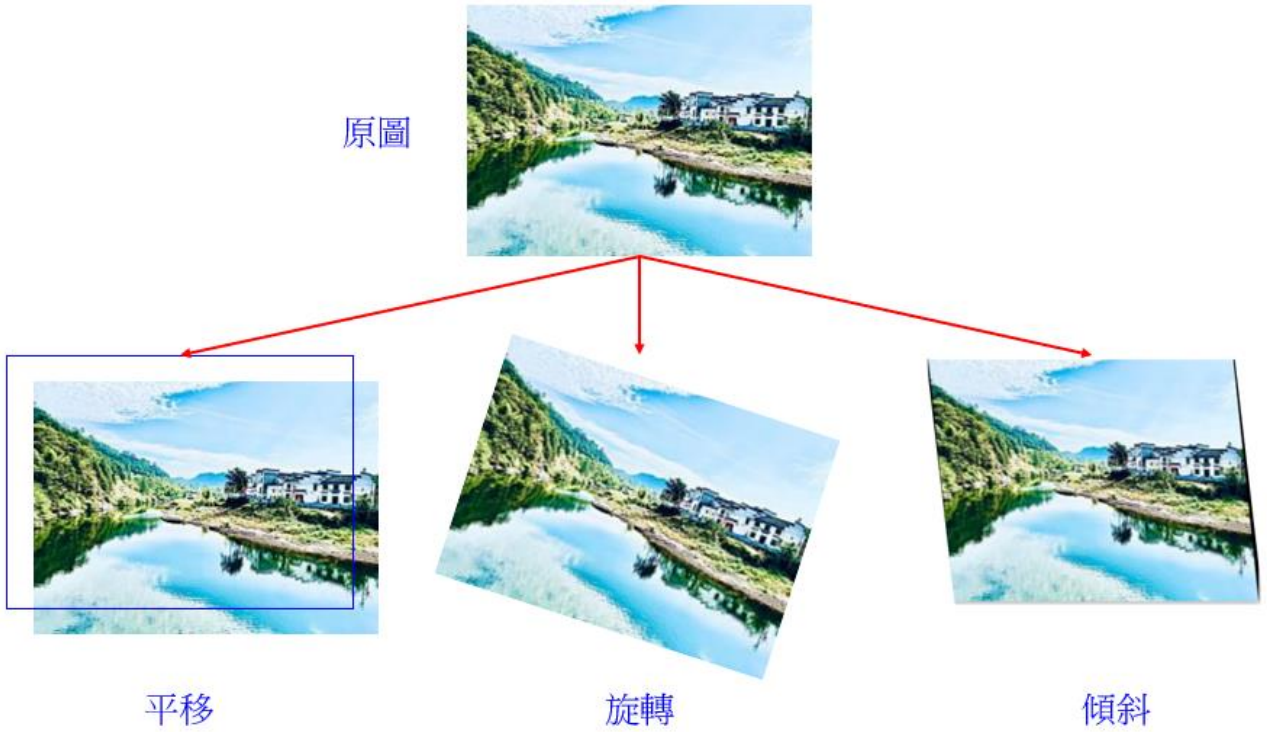
同時水平與垂直翻轉

- dst = cv2.flip(src, flipCode)
- 程式實例ch10_3.py：將一個影像同時做垂直翻轉、水平翻轉、水平與垂直翻轉。

原始影像　垂直翻轉　水平翻轉　與垂直翻轉水平

10-3：影像仿射

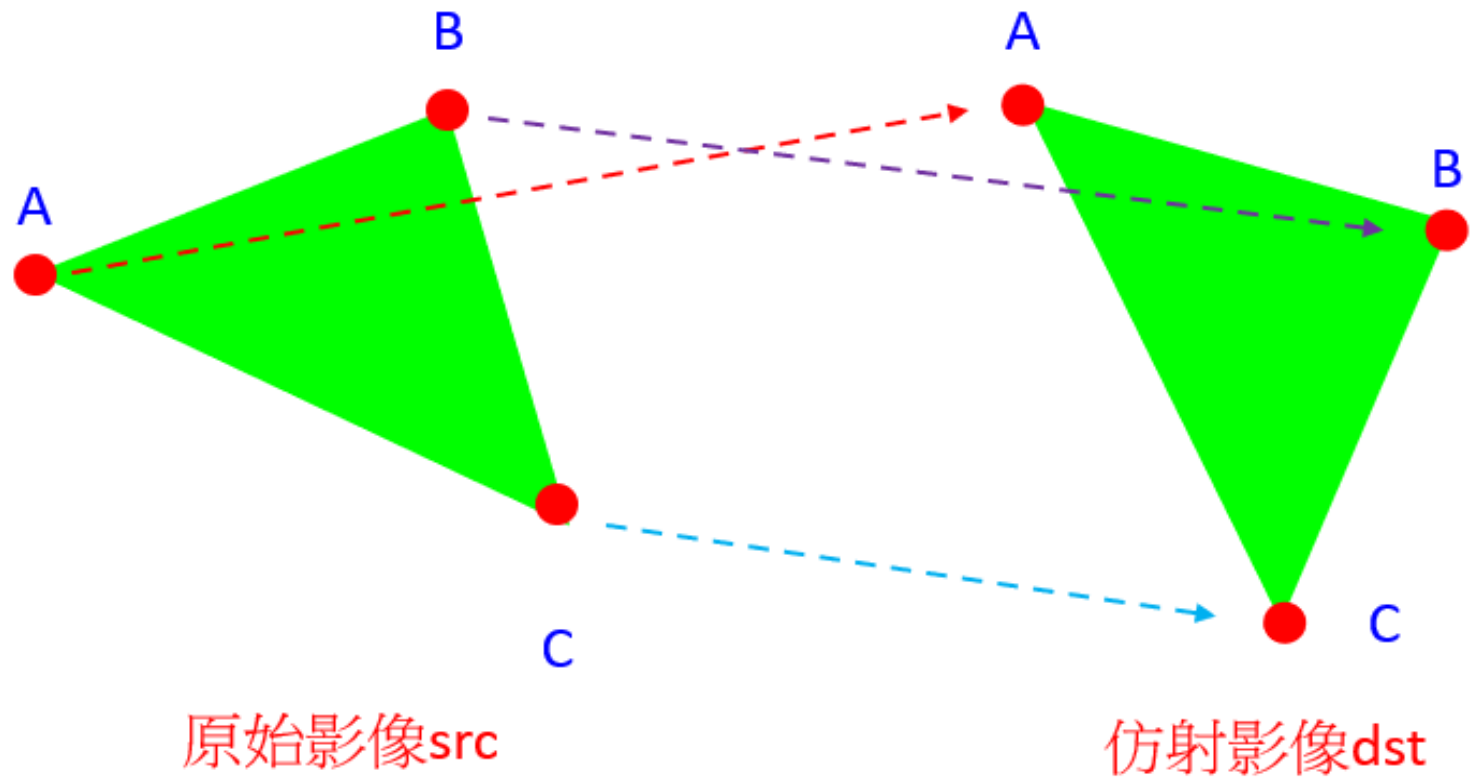
- 影像仿射是指影像在二維空間的幾何變換，變換後的影像仍可以保持平行性與平直性。



10.3.1：仿射的數學基礎

$$M = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \end{bmatrix}$$

$$dst(x, y) = src(M_{11}x + M_{12}y + M_{13}, M_{21}x + M_{22}y + M_{23})$$


B
A
C
A
B
C
原始影像src
仿射影像dst

10_3_2：仿射的函數語法

- dst = cv2.warAffine(src, M, dsize, flags, boderMode, boderValue)

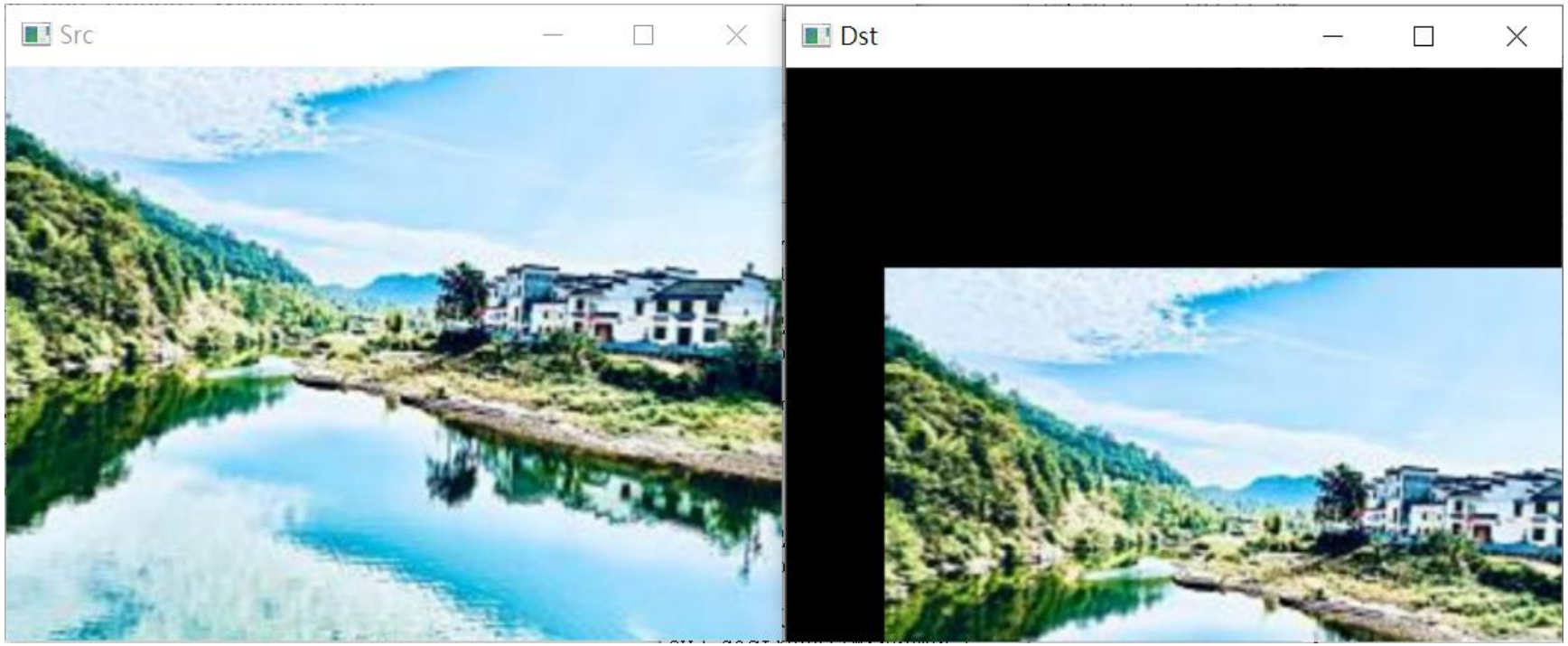
10_3_3：影像平移

$$dst(x,y) = src(x+50, y+100)$$

$$dst(x,y) = src(1*x+0*y+50, 0*x+1*y+100)$$

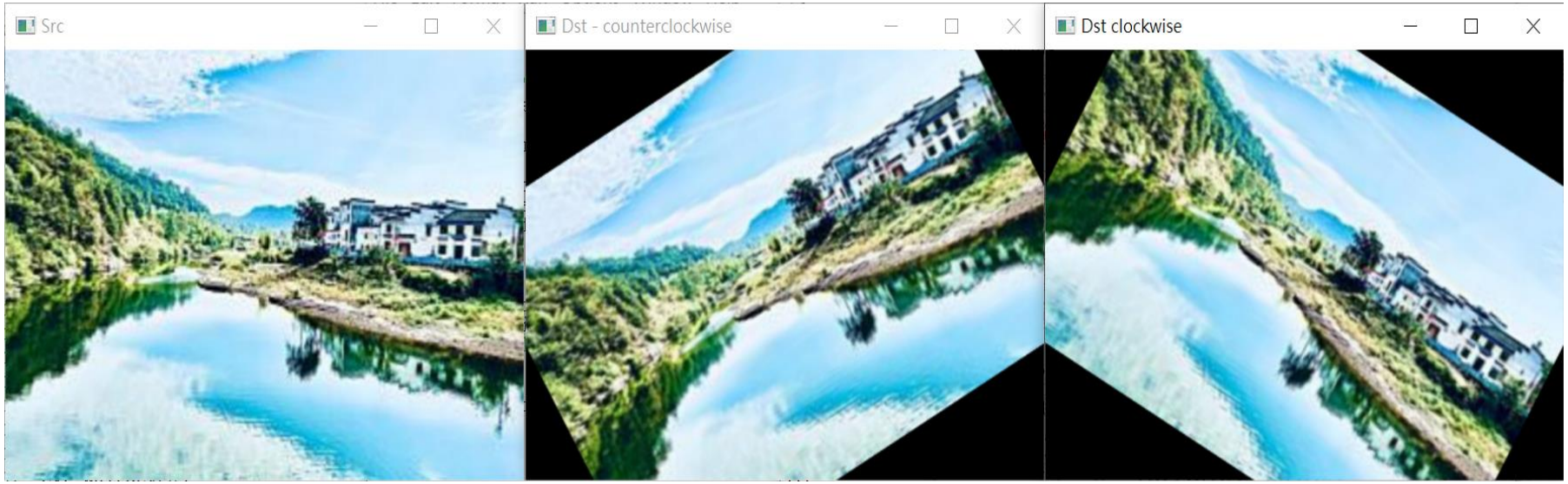
$$M = \begin{bmatrix} 1 & 0 & 50 \\ 0 & 1 & 100 \end{bmatrix}$$

- 程式實例ch10_4.py：影像平移x = 50, y = 100的應用。

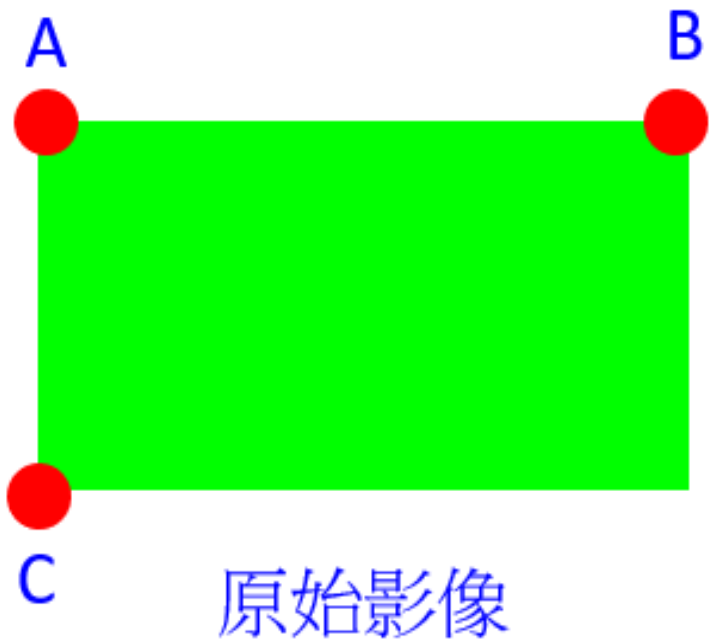


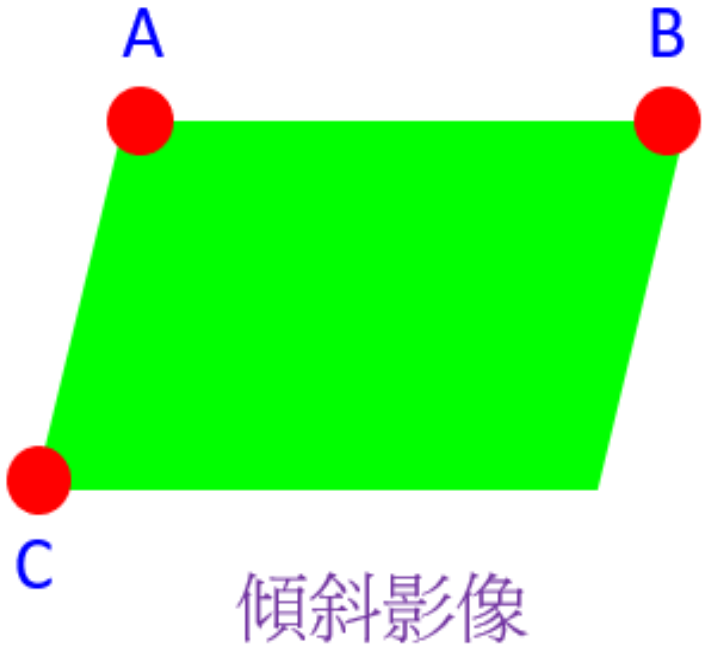
10-3-4：影像旋轉

- M = cv2.getRotationMatrix2D(center, angle, scale)
- 程式實例ch10_5.py：逆時鐘30度與順時鐘30度的影像旋轉。



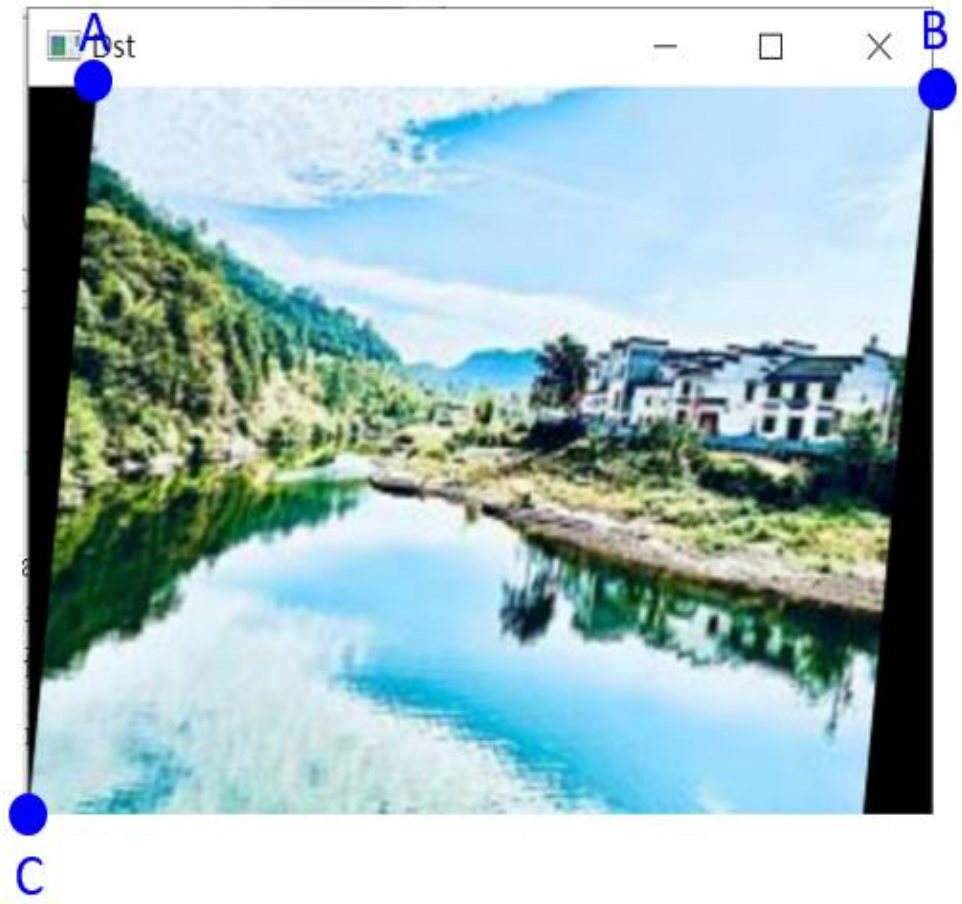
10_3_5：影像傾斜





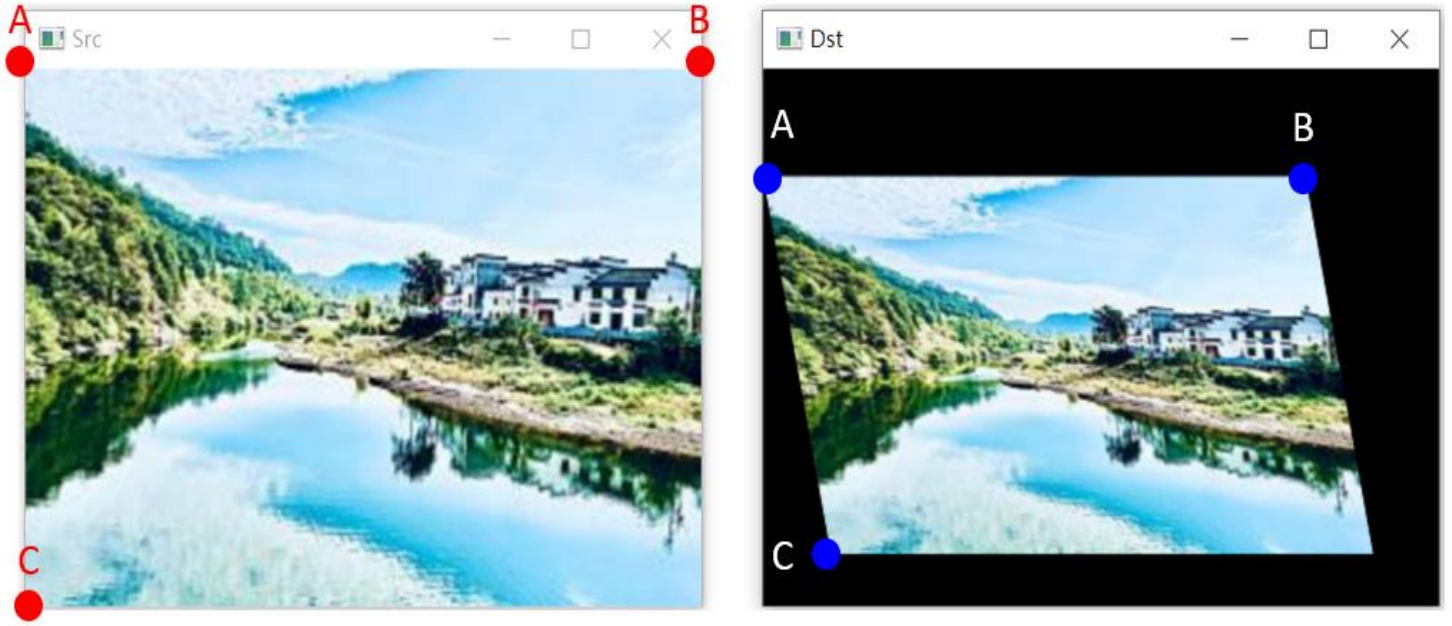
- M = cv2.getAffineTransform(src, dst)
- 程式實例ch10_6.py：影像向右上方傾斜的設計，src影像的3個座標分別如下：
- 左上方：[0, 0]
- 右上方：[width-1, 0]
- 左下方：[0, height – 1]
- dst影像的3個座標分別如下：
- 左上方：[30, 0]
- 右上方：[width-1, 0]
- 左下方：[0, height – 1]


A
Src
B
C

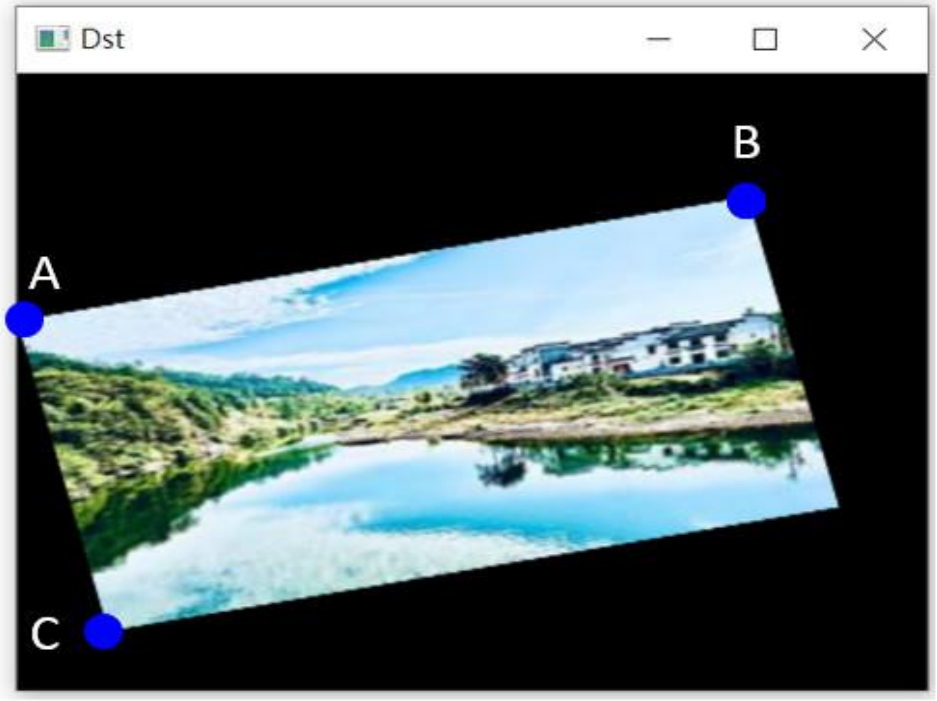

A
Dst
B
C

- 程式實例ch10_7.py：下列是改為dst影像向左上方傾斜，下列是dst影像A、B、C三個點座標以及程式碼。
- 左上方A：[0, 0]
- 右上方B：[width-1-30, 0]
- 左下方C：[30, height – 1]

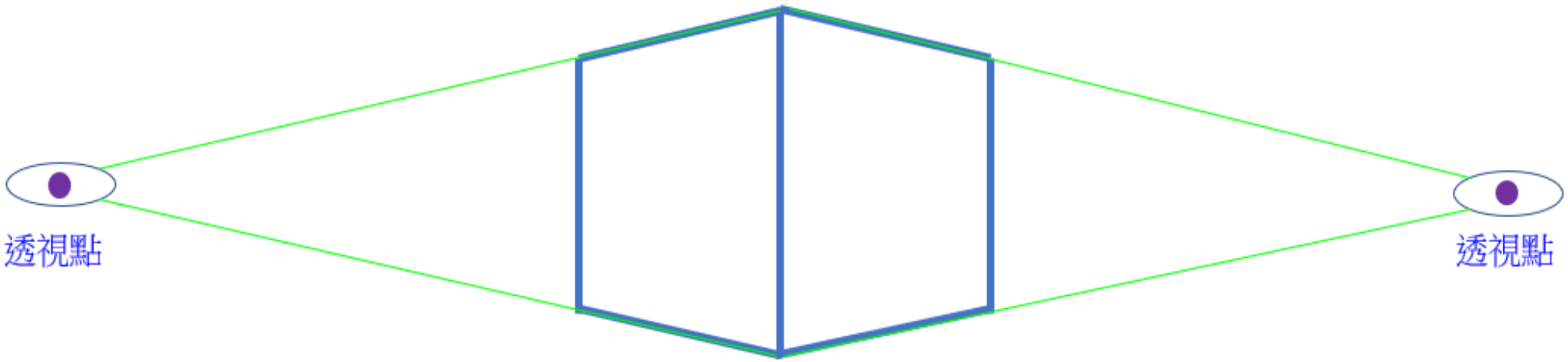
- 程式實例ch10_8.py：傾斜時更改寬度的設計，下列是dst影像A、B、C三個點座標。
- 左上方A：[0, height*0.2]
- 右上方B：[width*0.8, height*0.2]
- 左下方C：[width*0.1, height*0.9]
- 由於A、B、C三個點座標比較複雜，所以筆者分成3列程式碼。



- 程式實例ch10_9.py：使用ab點是傾斜線重新設計ch10_8.py，筆者只是修改dst的A點座標。



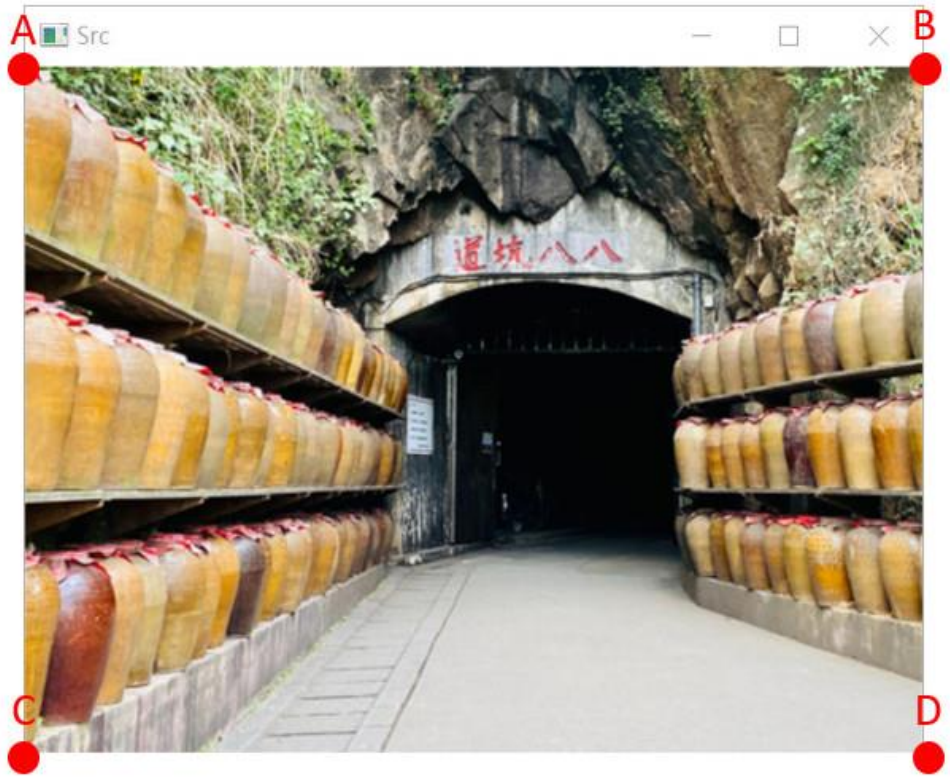
10-4：影像透視



- M = cv2. getPerspectiveTransform(src, dst)

- dst = cv2.warpPerspective(src, M, dsize, flags, borderMode, borderValue)

- 程式實例ch10_10.py：透視圖的應用，透視點是在正前方下方，可以得到影像上方變窄的透視效果。


A
Src
B
C
D

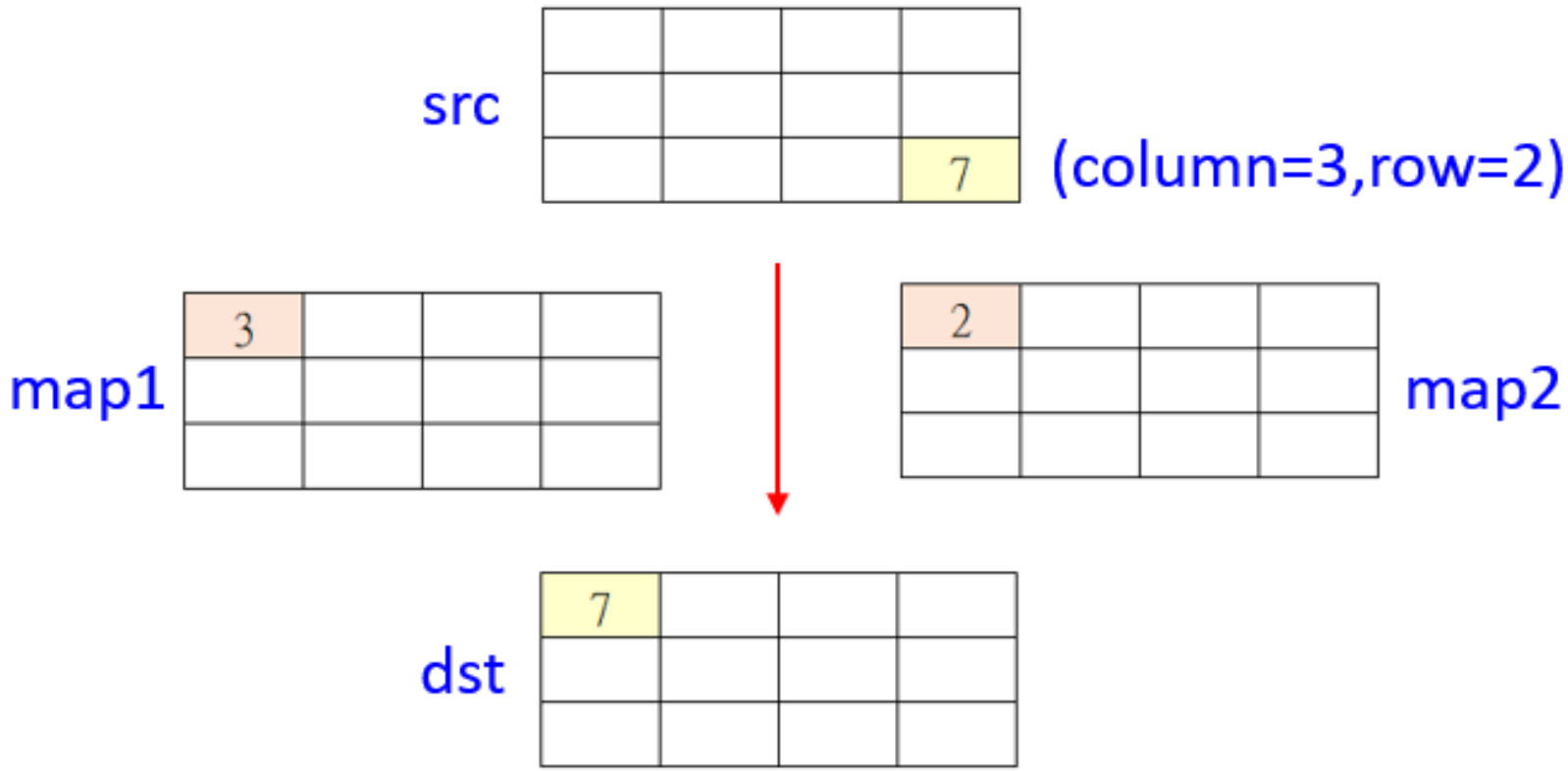

Dst
A
B
C
D

10.5：重映射

- dst = cv2.remap(src, map1, map2, interpolation, borderMode, borderValue)

10.5.1：解說map1和map2

- map1：dst影像的每一個像素內容都是由src影像的某個像素對應而得到，map1則是存放src影像的x座標(columns)，因此程式設計時又喜歡用mapx代替map1。
- map2：dst影像的每一個像素內容都是由src影像的某個像素對應而得到，map2則是存放src影像的y座標(rows)，因此程式設計時又喜歡用mapy代替map2。


src
7
(column=3,row=2)
map1
3
2
map2
dst
7

- 程式實例ch10_10_1.py：用陣列了解映射的基礎操作，將所有目的影像(dst)的像素值皆是來自原始影像(src)座標(3,2)(相當於column=3, row=2)。

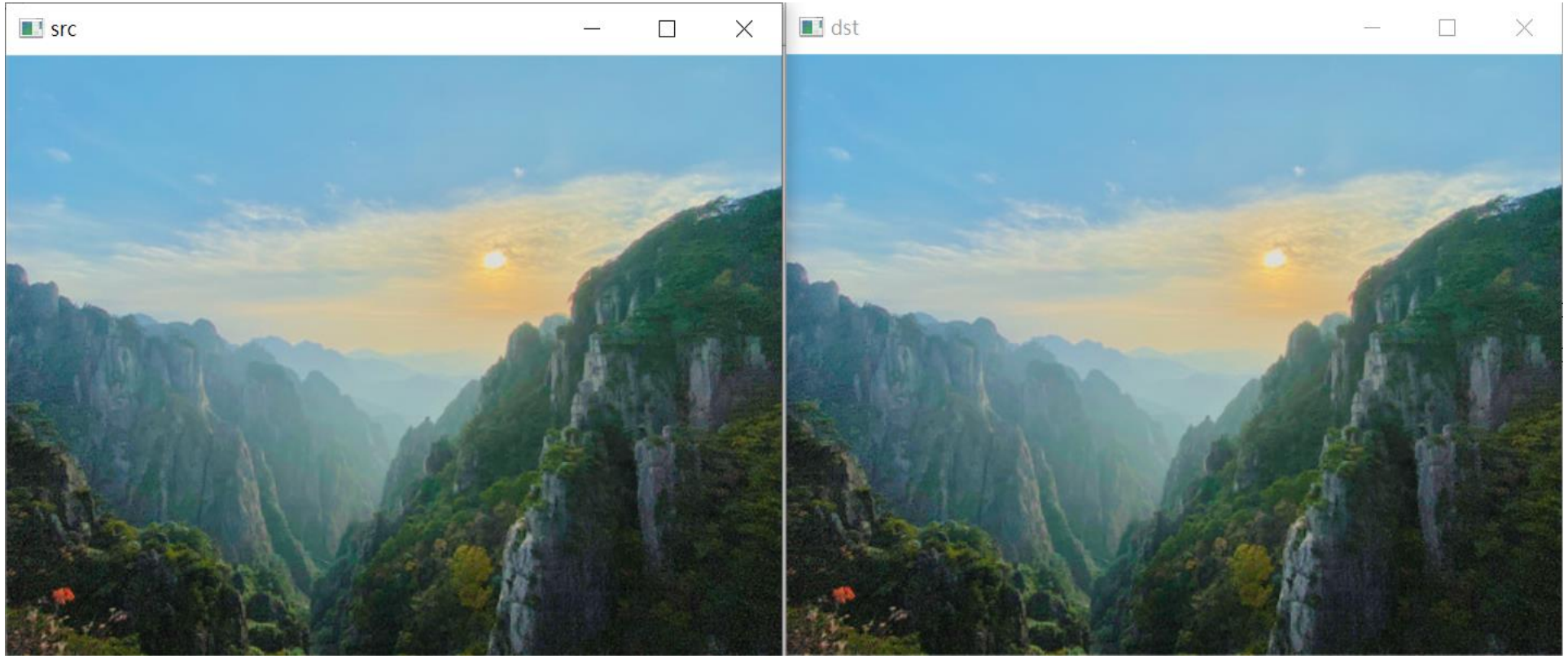
```
================== RESTART: D:/OpenCV_Python/ch10/ch10_10_1.py ==================
src =
 [[ 41 198 132 209]
 [ 81 123 118 225]
 [118 107 223  62]]
mapx =
 [[3. 3. 3. 3.]
 [3. 3. 3. 3.]
 [3. 3. 3. 3.]]
mapy =
 [[2. 2. 2. 2.]
 [2. 2. 2. 2.]
 [2. 2. 2. 2.]]
dst =
 [[62 62 62 62]
 [62 62 62 62]
 [62 62 62 62]]
```

10.5.2：影像複製

- map1：設定為對應位置的x軸座標。
- map2：設定為對應位置的y軸座標。
- 程式時例ch10_11.py：使用映射remap( )函數執行矩陣複製的實例。

```
================== RESTART: D:/OpenCV_Python/ch10/ch10_11.py ==================
src =
 [[136 195   4 165  93]
 [253   0 121  98 105]
 [179 220 156  98  56]]
mapx =
 [[0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]]
mapy =
 [[0. 0. 0. 0. 0.]
 [1. 1. 1. 1. 1.]
 [2. 2. 2. 2. 2.]]
dst =
 [[136 195   4 165  93]
 [253   0 121  98 105]
 [179 220 156  98  56]]
```

- 程式實例ch10_12.py：使用映射remap( )函數執行影像複製的實例。

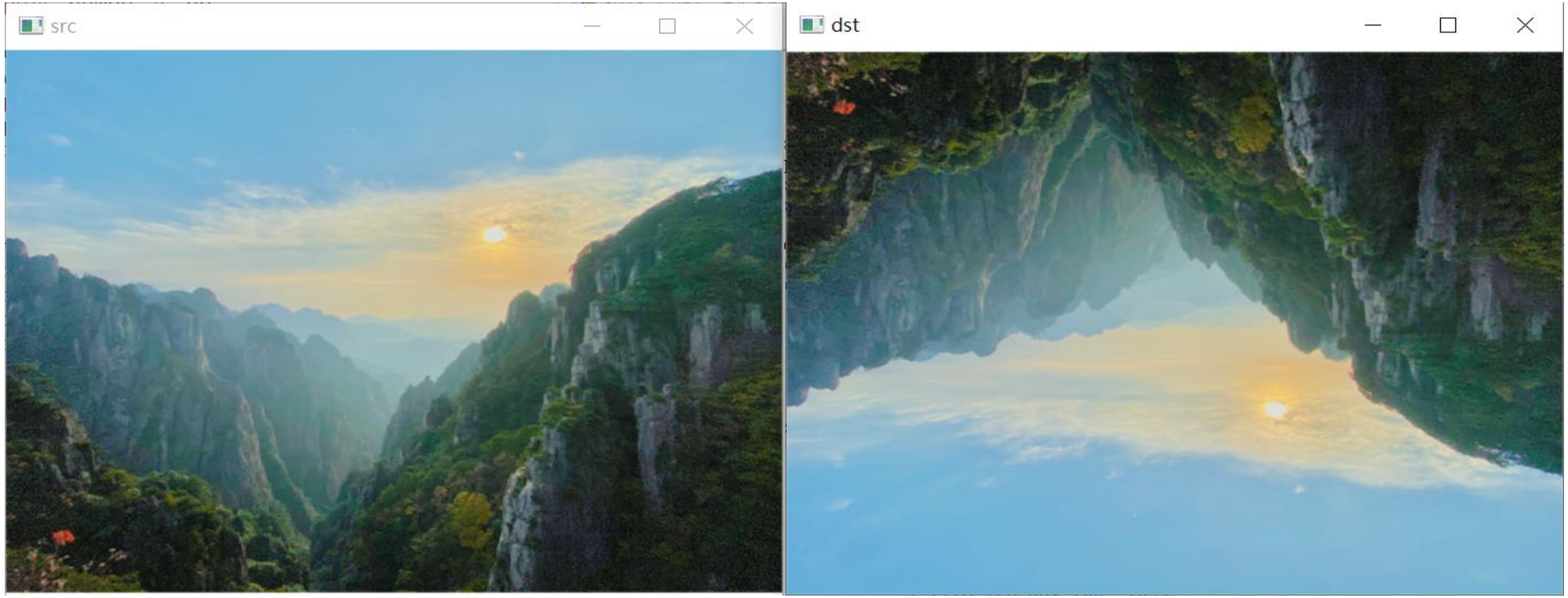


10.5.3：垂直翻轉

- 所謂的垂直翻轉就是影像沿著x軸做翻轉，這時mapx與mapy的設定觀念如下：
- mapx：x軸的座標不更改。
- mapy：假設y軸的列數是rows，則可用公式”rows – 1 – x”。
-
- 程式實例ch10_13.py：使用映射remap( )函數執行垂直翻轉的矩陣實例。

```
================== RESTART: D:/OpenCV_Python/ch10/ch10_13.py ==================
src =
 [[  2   1  43 133 237]
 [193 107 211 131  96]
 [133  43 102 218 252]]
mapx =
 [[0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]]
mapy =
 [[2. 2. 2. 2. 2.]
 [1. 1. 1. 1. 1.]
 [0. 0. 0. 0. 0.]]
dst =
 [[133  43 102 218 252]
 [193 107 211 131  96]
 [  2   1  43 133 237]]
```

- 程式實例ch10_14.py：使用映射remap( )函數執行影像垂直翻轉的實例。

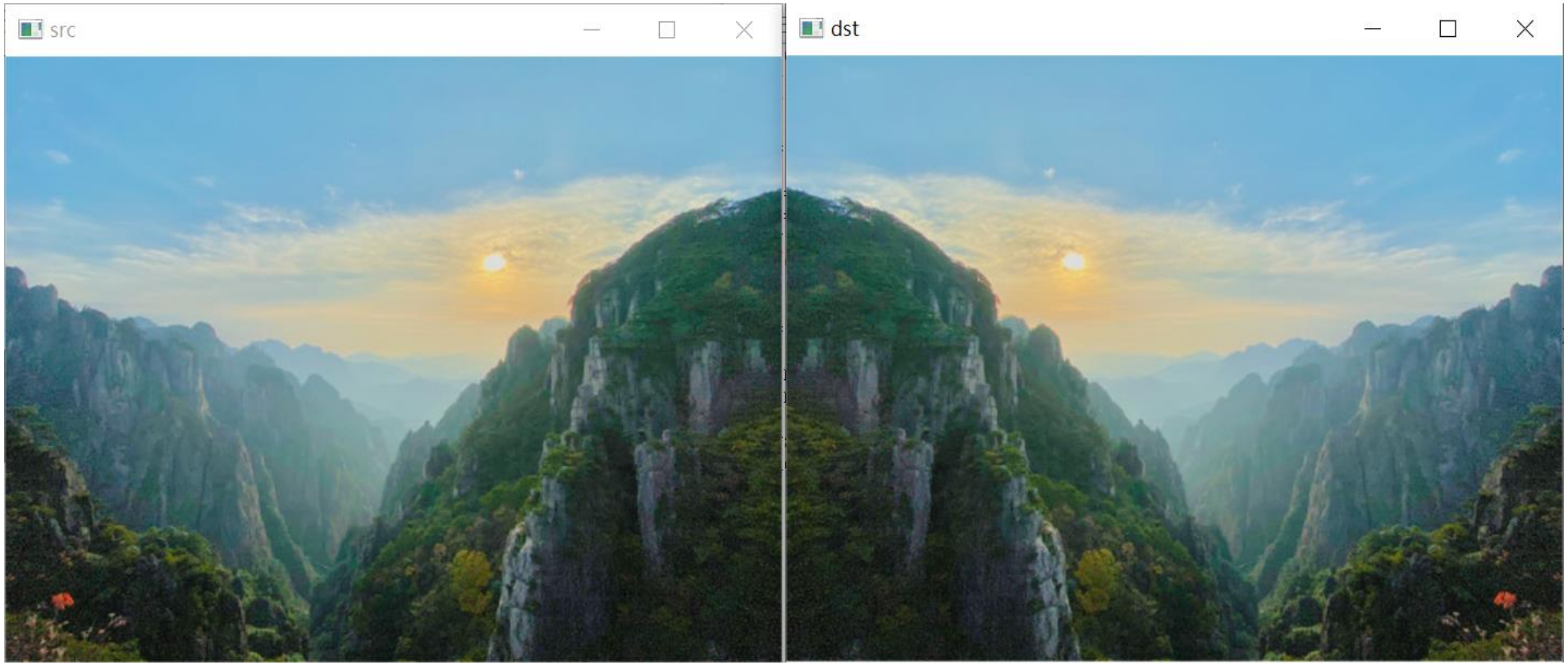


10-5-4：水平翻轉的實例

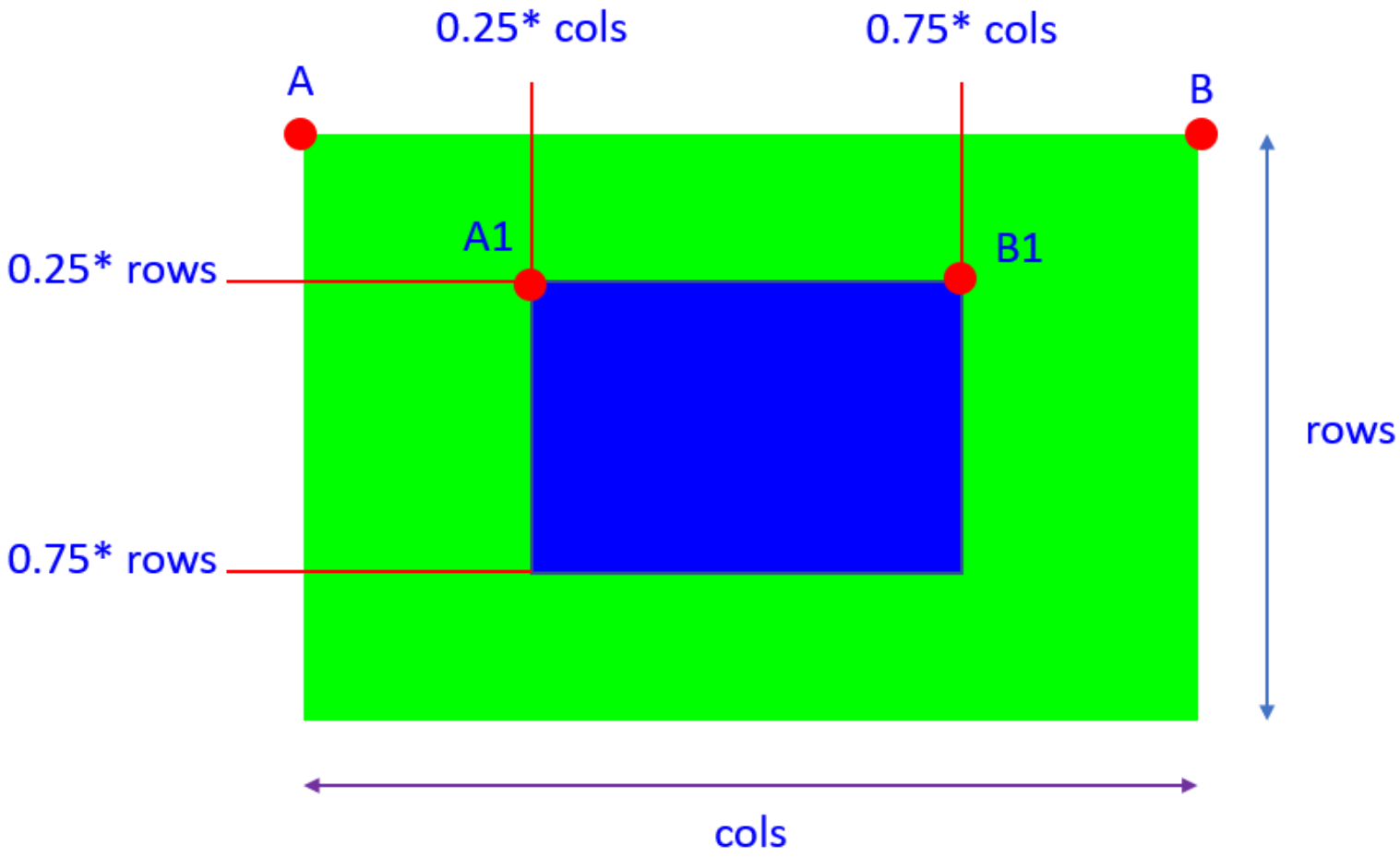
- 所謂的水平翻轉就是影像沿著y軸做翻轉，這時mapx與mapy的設定觀念如下：
- mapx：假設x軸的列數是cols，則可用公式”cols – 1 – x”。
- mapy：y軸的座標不更改。
- 程式實例ch10_15.py：使用映射remap( )函數執行水平翻轉的矩陣實例。

```
================== RESTART: D:/OpenCV_Python/ch10/ch10_15.py ==================
src =
 [[226  74 131 101 188]
 [208 141  43   5 241]
 [109  34 228   0 165]]
mapx =
 [[4. 3. 2. 1. 0.]
 [4. 3. 2. 1. 0.]
 [4. 3. 2. 1. 0.]]
mapy =
 [[0. 0. 0. 0. 0.]
 [1. 1. 1. 1. 1.]
 [2. 2. 2. 2. 2.]]
dst =
 [[188 101 131  74 226]
 [241   5  43 141 208]
 [165   0 228  34 109]]
```

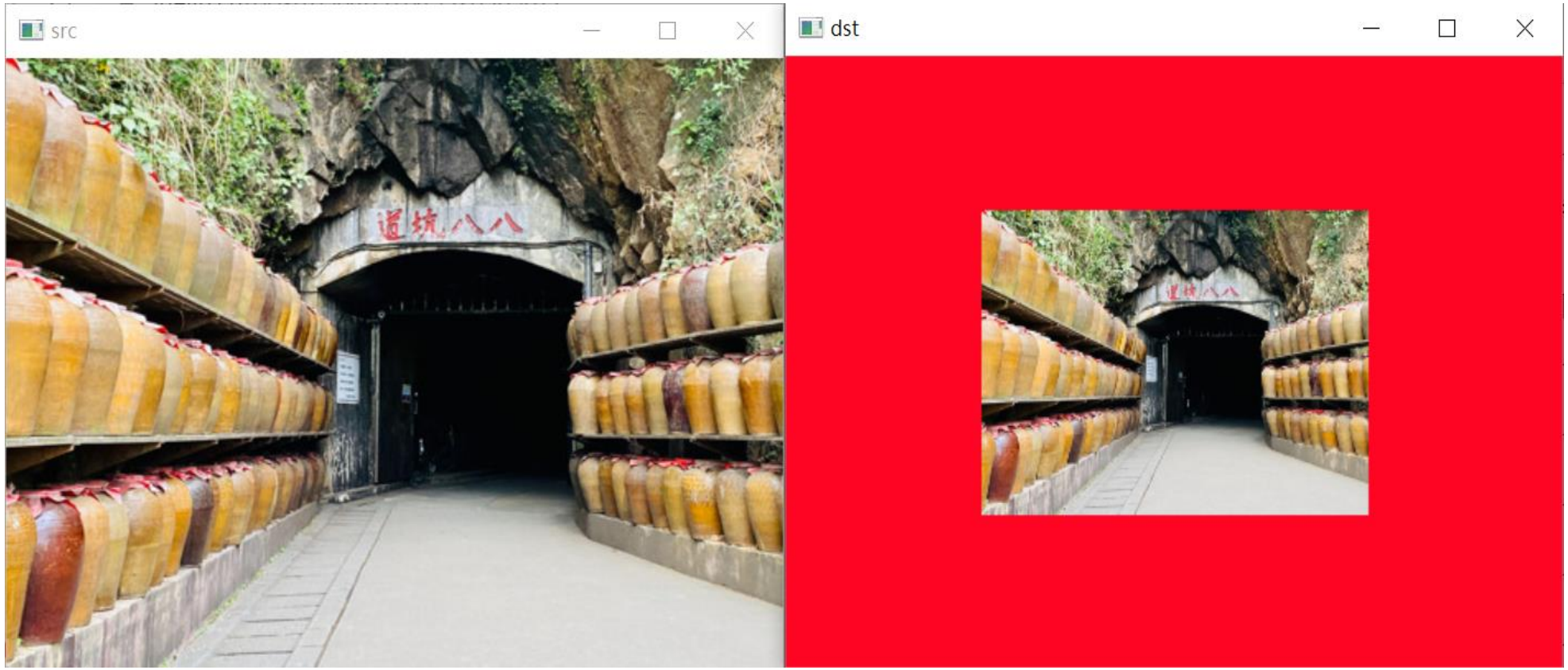
- 程式實例ch10_16.py：使用映射remap( )函數執行影像水平翻轉的實例。



10.5.5：影像縮放



- 程式實例ch10_17.py：影像縮小的實例，由於設定目的影像外圍使用(0, 0)座標的值，剛好這是紅色，所以目的影像外圍是紅色。



10.5.6：影像垂直壓縮

- 程式實例ch10_18.py：將影像垂直壓縮一半。