



Digital Signal Processing Laboratory





Lab 5 Pixel Array Manipulation

1. Show the results in each part with two images. One is given by TA and the other is your own image.

(1) Grayscale


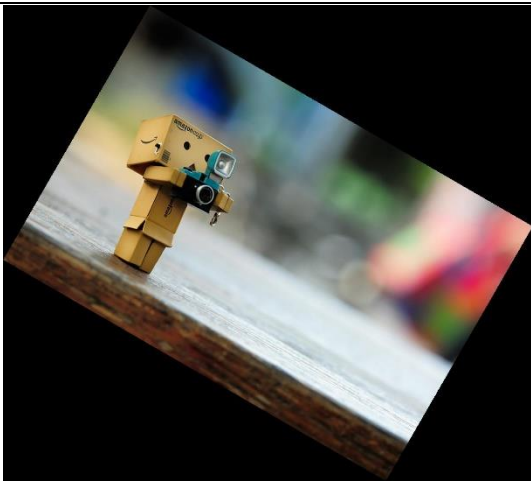

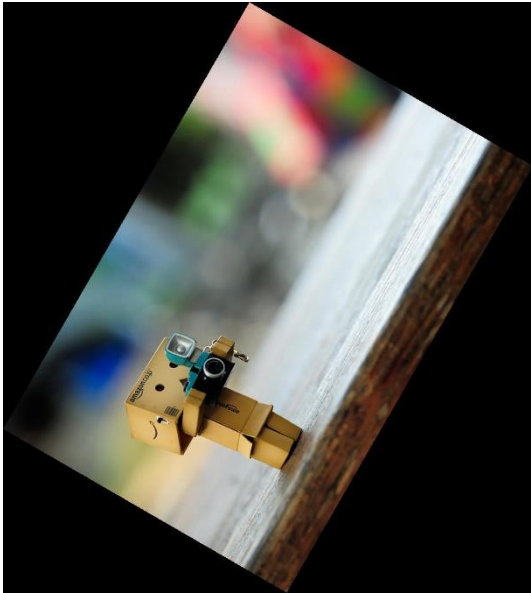
Original image	Processed image
	

(2) Image flipping

Original image	Horizontal flipping
	
Original image	Vertical flipping
	
Original image	Horizontal + Vertical flipping







(3) Image rotation



Original image	Radius = $\pi/6$
	
Original image	Radius = $-\pi/3$
	

(4) Shear transformation

Original image	shear_x = -0.8, shear_y = 0.2
----------------	-------------------------------

	
Original image	$\text{shear}_x = 0.3, \text{shear}_y = -0.6$
	

(5) Image scaling

Original image	x0.6
	

2. There are two ways for image warping. One is forward warping and the other is backward warping. What are the differences between them? Try to describe the differences subjectively and objectively.

Forward warping 是指在知道新舊影像之間轉換關係的情況下，直接把原影像的每個座標點變換到新的座標上。此方法的好處在於想法直接、容易實現，但問題在於許多時候轉換後的座標並不是整數，通常會取距離最近的點，但是就會造成有

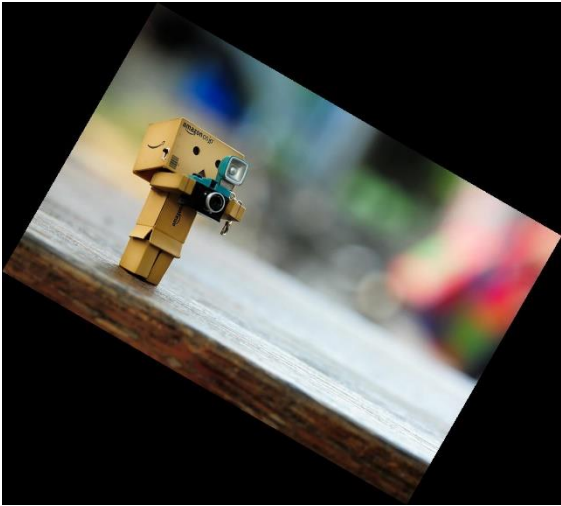
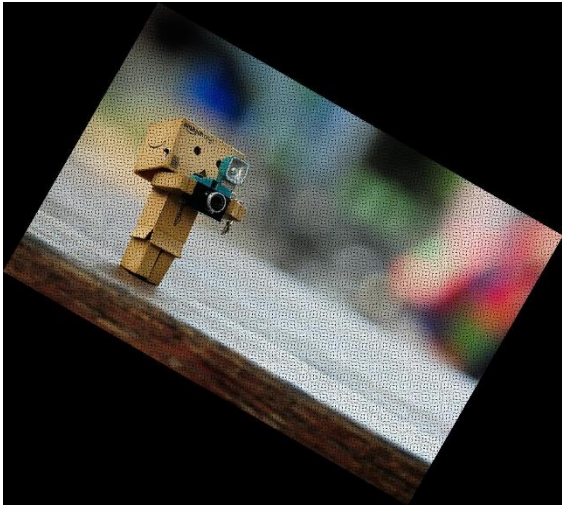
些新的座標點沒有值，或是有些座標點會被重複賦值。

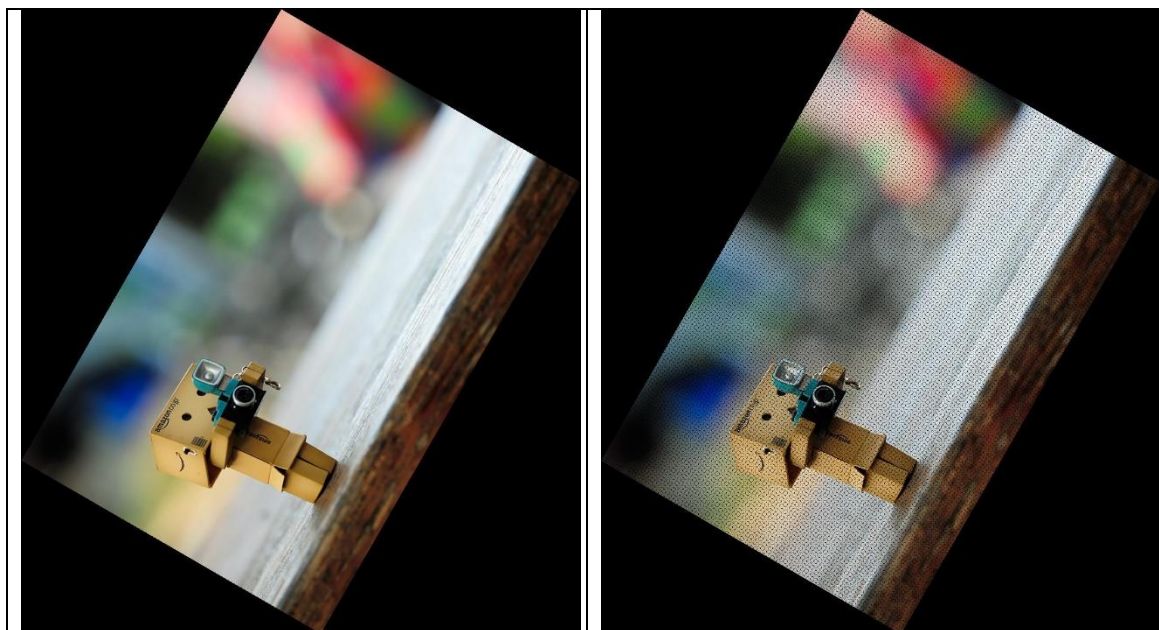
Backward warping 是對於每個新影像的座標點，用反向的轉換函數(反矩陣)，找到它在原影像中的位置，然後讓新影像的值等於原影像的；若在原影像中對應的位置不是整數時，就利用插值法獲得。此方法雖然較不直觀，在實作上也較困難，但是它解決了 forward warping 中遇到的問題，它可以讓新影像中的每個座標點都有對應的值，且這些相鄰的象素點也會砍起來連續。

3. Try to use the forward warping method mentioned above to implement image rotation. Specify the flow of your implementation and show your results.

Forward warping 在前面的步驟和 backward warping 類似，會先利用旋轉矩陣求出新影像的各頂點座標，並利用這些頂點的大小關係求出新影像所需的 height 和 width，以及 x 軸與 y 軸要 shift 的大小。此目的在於經過旋轉後新影像可能會超出原影像的範圍，導致影像會有一部分被截掉，因此需要先算出新影像的大小與平移的距離。接著有別於 backward warping，這邊直接將舊座標經過旋轉後平移的結果等於新座標，然後將新座標賦予舊座標的 RGB 數值，重複此步驟直到所有的舊座標點都做過一遍。

結果顯示於下表的右側欄，而左側為原本 backward warping 的結果。經過比較後可發現，雖然兩者都能正確旋轉影像，但 forward warping 會讓部分象素點沒有正確賦予到值，使得圖片看起來有點模糊，放大後會清楚看到許多原本沒有的黑點，原因就在於 forward warping 無法處理轉換後非整數的點，這些點剛好就是 backward warping 中需要做 bilinear interpolation 的那些點。

Backward warping (radius = $\pi/6$)	Forward warping (radius = $\pi/6$)
	
Backward warping (radius = $-\pi/3$)	Forward warping (radius = $-\pi/3$)



4. Try to implement at least one other method for grey scale, image flipping, image rotation, or image resize. Specify your method and show the results. Compare the differences between the original method in this lab and your proposed method.

在 image flipping 這部分，原本的方法是當 $\text{type}=0$ 時(水平翻轉)，對於每個象素點，將其舊座標的 x 軸減去 width 再減一，取絕對值後成為其新的座標；當 $\text{type}=1$ 時(垂直翻轉)，對於每個象素點，將舊座標的 y 軸減去 height 再減一，取絕對值後成為其新的座標。

而我所使用的新的方法，是利用矩陣來達成。當水平翻轉時，設定座標轉換矩陣為 $\begin{bmatrix} -1 & 0 & x_shift \\ 0 & 1 & y_shift \end{bmatrix}$ ，其中 $\begin{cases} x_shift = \text{width} + 1 \\ y_shift = 0 \end{cases}$ ，而輸入的矩陣為 $\begin{bmatrix} x_old \\ y_old \\ 1 \end{bmatrix}$ ，因此新







的坐標軸為 $\begin{bmatrix} x_new \\ y_new \end{bmatrix} = \begin{bmatrix} -1 & 0 & x_shift \\ 0 & 1 & y_shift \end{bmatrix} \cdot \begin{bmatrix} x_old \\ y_old \\ 1 \end{bmatrix} = \begin{bmatrix} -x_old + x_shift \\ y_old \end{bmatrix}$ 。當垂直翻轉時，

則座標轉換矩陣為 $\begin{bmatrix} 1 & 0 & x_shift \\ 0 & -1 & y_shift \end{bmatrix}$ ，其中 $\begin{cases} x_shift = 0 \\ y_shift = \text{height} + 1 \end{cases}$ 。當水平加垂直翻轉

時，則座標轉換矩陣為 $\begin{bmatrix} -1 & 0 & x_shift \\ 0 & -1 & y_shift \end{bmatrix}$ ，其中 $\begin{cases} x_shift = \text{width} + 1 \\ y_shift = \text{height} + 1 \end{cases}$ 。利用矩陣做運算，可以簡化程式碼的長度，也容易針對新的功能進行擴充。

下表是原本方法與新方法的結果，兩者皆做出相同的影像。

Original method (horizontal)	Proposed method (horizontal)
------------------------------	------------------------------

	
Original method (vertical)	Proposed method (vertical)
	
Original method (horizontal + vertical)	Proposed method (horizontal + vertical)
	

5. Conclusion.

經過此次實驗，我對於 pixel array manipulation 有了更深的認識，也了解了該如何在 MATLAB 中輸入、輸出影像，以及 grayscale、image flipping、rotation、shear 和 resize 這些平常只要一行程式碼就可以幫我們做出來的這些影像處理方法，他們是如何做出來的。經過實際操作後，我了解到影像陣列的結構、RGB 數值代表的意義、以及在實作時需要考慮到的資料型態的轉換。另外，我也了解到在做影像的座標轉換時，常常會遇到轉換後的座標非整數的情況，造成新影像的某些象素沒有值，因此需要用到 backward warping 和 bilinear interpolation 等方式進行處理，以降低新影像的失真。