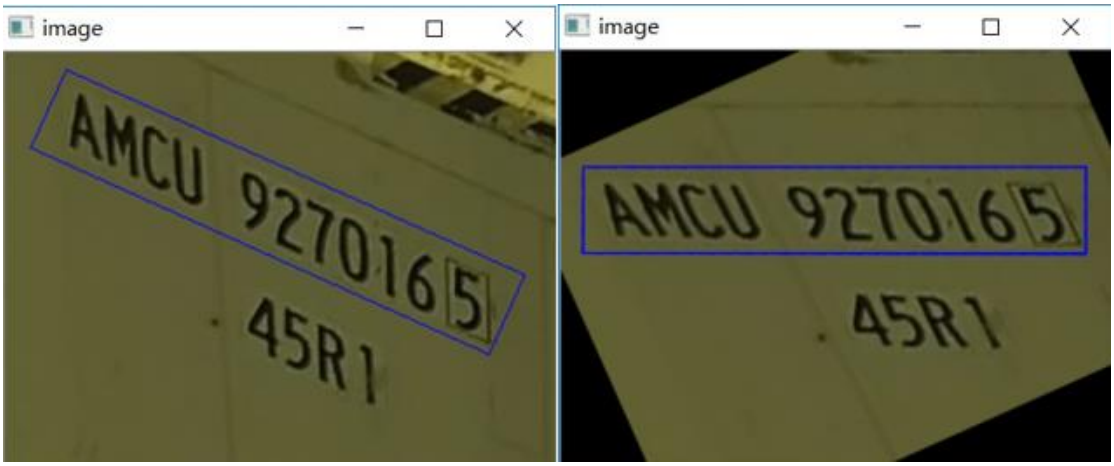
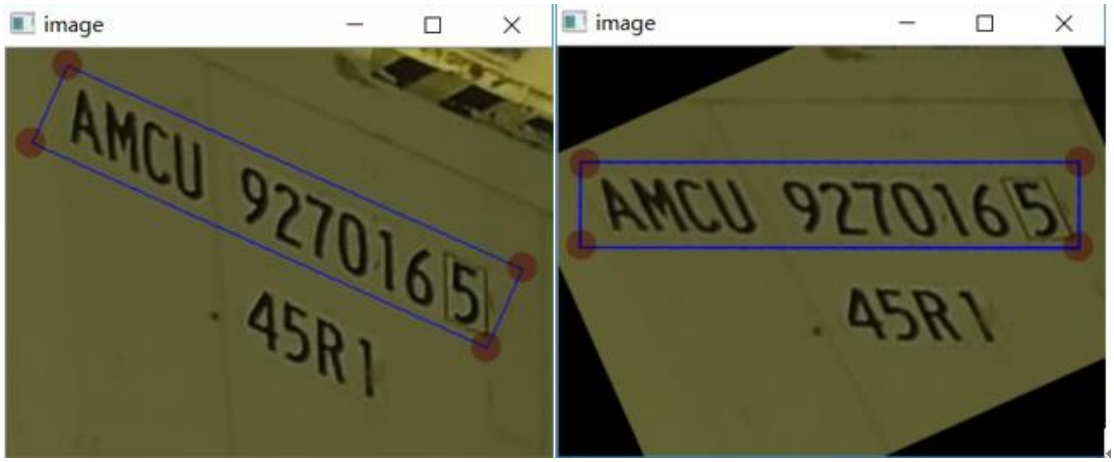
矫正字符块遇到问题

在使用 RRPN 找到精确的字符块区域之后，要使用透视变换将倾斜的矩形图像块矫正为水平方向。之前一直没有注意到，以为很简单，但今天做简单测试之后发现行不通。

假设下图是使用 RRPN 网络找到的倾斜图像块（左）和进行水平矫正后的效果：

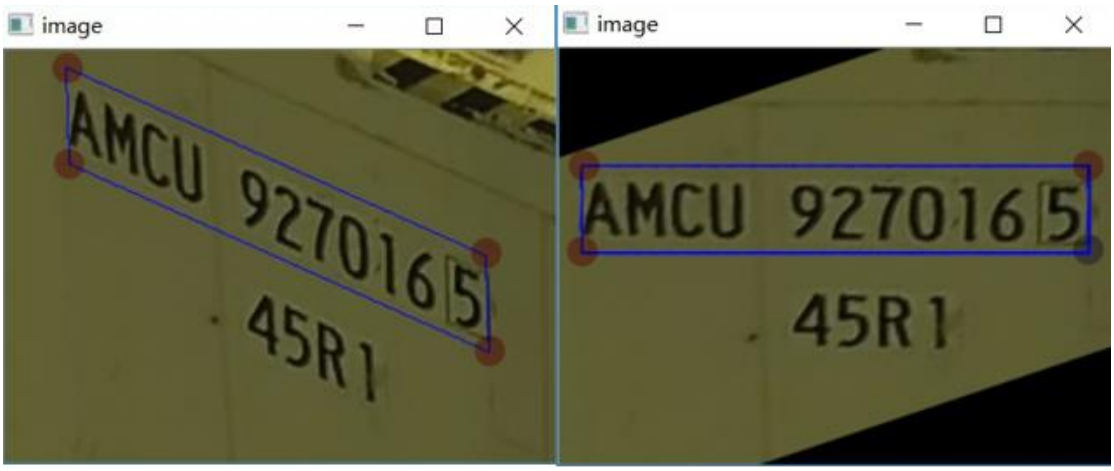


计算透视变换矩阵需要源图像的目标图像的对应的 4 个坐标点，我采用的是经过旋转和没经过旋转的矩形框的 4 个顶点坐标，如下图所示：



但从结果看，旋转后的图像块显然不是理想的效果。因为水平矫正后的字符的形状仍然是倾斜的，其实这就是一个只做了旋转处理的仿射变换。这样的倾斜字符，后面难以进行最后一步的字符识别。

真正要想将字符形状给矫正，上图左边源图像找到的 4 个对应顶点围成的应该是一个平行四边形，对应着右边图像的水平矩形。理想的矫正效果如下图所示（只是手动将左图中左上角和右下角的顶点位置缩短了）：



调整的代码如下：

```
k = 25      人为使用了一个变量k，这里只是为了画出理想效果，实际操作中的k值如何确定？
for j in range(num_objs):
    M = cv2.getRotationMatrix2D((boxes[j][0], boxes[j][1]), -boxes[j][-1], 1)
    ix = boxes[j][0]
    iy = boxes[j][1]
    iw = boxes[j][2]
    ih = boxes[j][3]
    left_top = np.array([ix-iw/2.0, iy-ih/2.0, 1])
    right_top = np.array([ix+iw/2.0-k, iy-ih/2.0, 1])
    left_bottom = np.array([ix-iw/2.0+k, iy+ih/2.0, 1])
    right_bottom = np.array([ix+iw/2.0, iy+ih/2.0, 1])
    rp1 = np.dot(M, left_top)
    rp2 = np.dot(M, right_top)
    rp3 = np.dot(M, left_bottom)
    rp4 = np.dot(M, right_bottom)
    cv2.line(img, (int(rp1[0]),int(rp1[1])), (int(rp2[0]),int(rp2[1])), (255,0,0), 1)
    cv2.line(img, (int(rp2[0]),int(rp2[1])), (int(rp4[0]),int(rp4[1])), (255,0,0), 1)
    cv2.line(img, (int(rp4[0]),int(rp4[1])), (int(rp3[0]),int(rp3[1])), (255,0,0), 1)
    cv2.line(img, (int(rp3[0]),int(rp3[1])), (int(rp1[0]),int(rp1[1])), (255,0,0), 1)
    cv2.imshow('image', img)
    cv2.waitKey()
```