

# Perspective transformation

Perspective transformation is also called projection transformation. The affine transformation we often talk about is a special case of perspective transformation. The purpose of perspective transformation is to transform objects that are straight lines in reality into straight lines in the picture.

Perspective transformation can map a rectangle to an arbitrary quadrilateral.

This technology is used when the robot is driving automatically. Perspective transformation is done through the function:

```
dst = cv2.warpPerspective(src, M, dsize[,flag, [,borderMode[,borderValue]]])
```

dst : The output image after perspective transformation, dsize determines the actual size of the output.

src: Source Image

M: 3X3 transformation matrix

dsize: Output image size.

flags: Interpolation method, the default is INTER\_LINEAR (bilinear interpolation). When it is WARP\_INVERSE\_MAP, it means that M is an inverse transformation, which can achieve the inverse transformation from the target dst to src.

borderMode: The edge type. The default is BORDER\_CONSTANT. When the value is BORDER\_TRANSPARENT, the values in the target image are not changed, and these values correspond to the outliers in the original image.

borderValue: Boundary value, the default is 0. Like affine transformation, OpenCV still provides a function cv2.getPerspectiveTransform() to provide the transformation matrix above.

The function is as follows.

```
matAffine = cv2.getPerspectiveTransform(matSrc, matDst)
```

matSrc: The coordinates of the four vertices of the input image.

matDst: Output the coordinates of the four vertices of the image.

Code path:

```
~/dofbot_pro/dofbot_opencv/scripts/2.Transform/07_perspectivity.ipynb
```

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
img = cv2.imread('yahboom.jpg',1)

imgInfo = img.shape
height = imgInfo[0]
width = imgInfo[1]
#src 4->dst 4 (Upper left corner, Lower left corner, Upper right corner, Lower
right corner)
matSrc = np.float32([[200,100],[200,400],[600,100],[width-1,height-1]])
```

```

matDst = np.float32([[200,200],[200,300],[500,200],[500,400]])
#Combination

matAffine = cv2.getPerspectiveTransform(matSrc,matDst)# mat 1 src 2 dst
dst = cv2.warpPerspective(img,matAffine,(width,height))
img_bgr2rgb = cv2.cvtColor(dst, cv2.COLOR_BGR2RGB)
plt.imshow(img_bgr2rgb)
plt.show()

```

