5. Affine transformation

Affine transformation (Affine Transformation or Affine Map) is a linear transformation from two-dimensional coordinates (x, y) to two-dimensional coordinates (u, v). Its mathematical expression is as follows:

$$\begin{cases} u = a_1 x + b_1 y + c_1 \\ v = a_2 x + b_2 y + c_2 \end{cases}$$

The corresponding homogeneous coordinate matrix representation is:

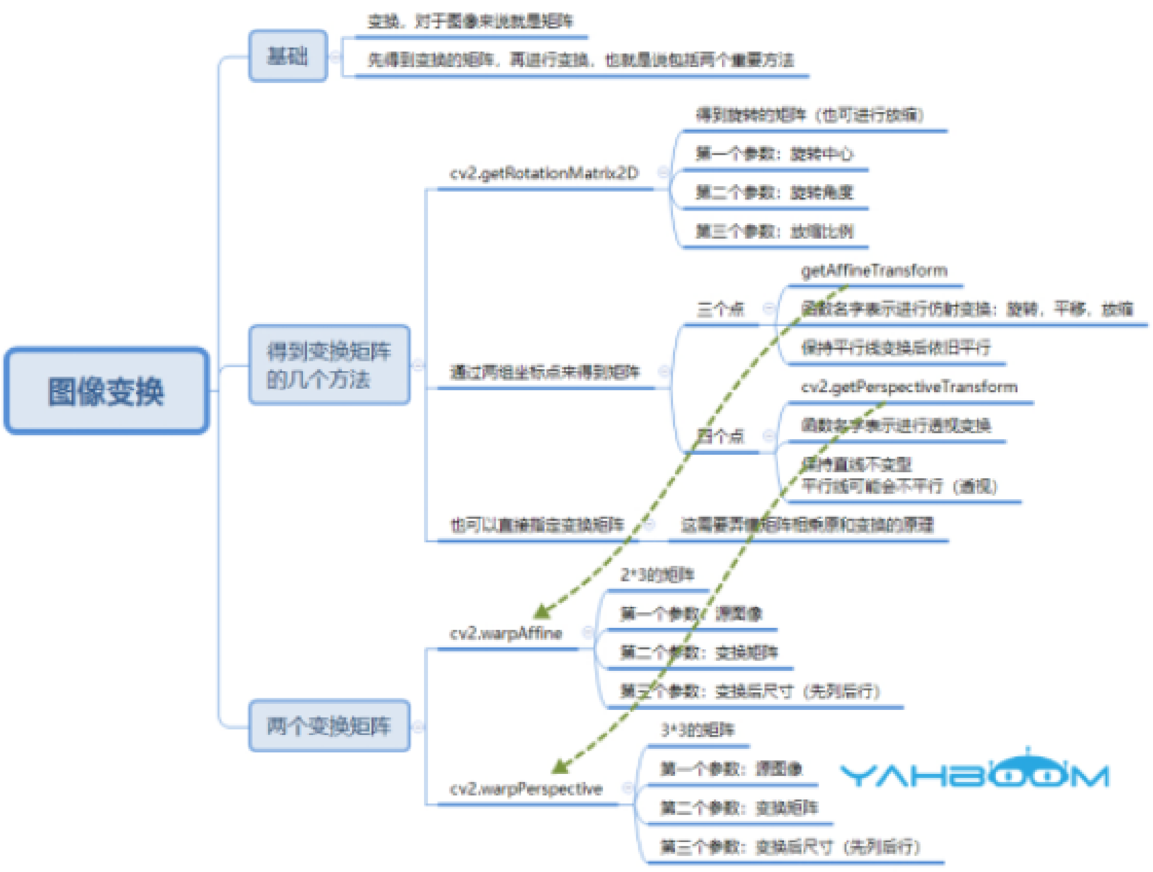
$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Affine transformation maintains the "flatness" of two-dimensional graphics (straight lines remain straight lines after affine transformation) and "parallelism" (the relative positional relationship between straight lines remains unchanged, and parallel lines remain straight after affine transformation) It is still a parallel line, and the position order of the points on the line will not change). Three non-collinear pairs of corresponding points determine a unique affine transformation.

Image rotation plus lifting is the image affine transformation. Affine change also requires an M matrix. However, due to the complexity of the affine transformation, it is generally difficult to find this matrix directly. OpenCV provides three parameters before and after the transformation. The corresponding relationship between points is used to automatically solve M. This function is

M=cv2.getAffineTransform(pos1,pos2), the two positions are the corresponding position relationships before and after the transformation. The output is the affine matrix M. Then use the function cv2.warpAffine().

Let’s look at the block diagram of the entire affine transformation and perspective transformation: two methods of image transformation, cv2.warpAffine and cv2.warpPerspective



The code is running on jupyter lab

```
import cv2

import numpy as np

import matplotlib.pyplot as plt

img = cv2.imread('yahboom.jpg',1)

img_bgr2rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

plt.imshow(img_bgr2rgb)

plt.show()

#cv2.waitKey(0)
```

```
imgInfo = img.shape

height = imgInfo[0]

width = imgInfo[1]

#src 3->dst 3 (upper left corner, lower left corner, upper right corner)

matSrc = np.float32([[0,0], [0,height-1], [width-1,0]])

matDst = np.float32([[50,50], [300,height-200], [width-300,100]])
```

```
#combination

matAffine = cv2.getAffineTransform(matSrc,matDst)# mat 1 src 2 dst

dst = cv2.warpAffine(img,matAffine,(width,height))

img_bgr2rgb = cv2.cvtColor(dst, cv2.COLOR_BGR2RGB)

plt.imshow(img_bgr2rgb)
```