

3. Image translation

3. Image translation

3.1. Image translation

3.2. Actual effect display

3.1. Image translation

`cv2.warpAffine(src, M, dsize[,dst[, flags[, borderMode[, borderValue]]]])`

Parameter meaning:

src - input image. M - transformation matrix. dsize - size of output image. flags - combination of interpolation methods (int type!) borderMode - border pixel mode (int type!) borderValue - (important!) border fill value; by default, it is 0.

Among the above parameters: M is an affine transformation matrix, which generally reflects the relationship of translation or rotation and is a 2×3 transformation matrix of InputArray type. In daily affine transformation, only the first three parameters are set, such as `cv2.warpAffine(img,M, (rows,cols))` to achieve basic affine transformation effects.

How to get the transformation matrix M? The following example illustrates that:

The original image src is converted to the target image dst through the conversion matrix M:

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

The original image src is moved 200 pixels to the right and 100 pixels downward, and the corresponding relationship is:

$dst(x, y) = src(x + 200, y + 100)$

Complete the above expression, that is:

$dst(x, y) = src(1 \cdot x + 0 \cdot y + 200, 0 \cdot x + 1 \cdot y + 100)$

According to the above expression, the values of each element in the corresponding transformation matrix M can be determined as follows:

$M_{11}=1$

$M_{12}=0$

$M_{13}=200$

$M_{21}=0$

$M_{22}=1$

$M_{23}=100$

Substituting the above values into the transformation matrix M, we get:

$$\begin{bmatrix} 1 & 0 & 200 \\ 0 & 1 & 100 \end{bmatrix}$$

3.2. Actual effect display

Code path:

/home/pi/Rider-pi_class/4.Open Source

CV/B.Geometric_Transformations/03_Image_Translation.ipynb

```
import cv2
import numpy as np
img = cv2.imread('yahboom.jpg',1)
#cv2.imshow('src',img)
imgInfo = img.shape
height = imgInfo[0]
width = imgInfo[1]
####
matShift = np.float32([[1,0,200],[0,1,100]])# 2*3
dst = cv2.warpAffine(img,matShift,(height,width))#1 data 2 mat 3 info
# 移位 矩阵 Shift Matrix
# cv2.imshow('dst',dst)
# cv2.waitKey(0)
```

```
#bgr8转jpeg格式 bgr8 to jpeg format
import enum
import cv2

def bgr8_to_jpeg(value, quality=75):
    return bytes(cv2.imencode('.jpg', value)[1])
```

```
import ipywidgets.widgets as widgets

image_widget1 = widgets.Image(format='jpg', )
image_widget2 = widgets.Image(format='jpg', )
# create a horizontal box container to place the image widget next to eachother
image_container = widgets.HBox([image_widget1, image_widget2])

# display the container in this cell's output
display(image_container)
#display(image_widget2)

img1 = cv2.imread('yahboom.jpg',1)

image_widget1.value = bgr8_to_jpeg(img1)
image_widget2.value = bgr8_to_jpeg(dst)
```

