2. Image Binarization

2.1 Core Idea of Binarization

Set a threshold value, and the value greater than the threshold value is 0 (black) or 255 (white), so that the image is called a black and white image. The threshold value can be fixed or adaptive. The adaptive threshold value is generally a comparison between a pixel at a point and the average value of the pixels in the region with this point as the middle order or the weighted sum of the Gaussian distribution, in which a difference value can be set or not.

OpenCV provides a threshold function: cv2.threshold (src, threshold, maxValue, thresholdType)

Parameter meaning:

src: original image

threshold: current threshold

maxVal: maximum threshold, usually 255

thresholdType: threshold type, usually has the following values

enum ThresholdTypes { THRESH_BINARY = 0, #The grayscale value of pixels greater than the threshold is set to maxValue (such as the maximum grayscale value of 8-bit is 255), and the grayscale value of pixels less than the threshold is set to 0. THRESH_BINARY_INV = 1, #The grayscale value of pixels greater than the threshold is set to 0, and that less than the threshold is set to maxValue. THRESH_TRUNC = 2, #The grayscale value of pixels greater than the threshold is set to 0, and that less than the threshold is set to maxValue. THRESH_TOZERO = 3, #No change is made to the grayscale value of the pixel point that is less than the threshold, and the grayscale value of the pixel point that is greater than the threshold is all changed to 0. THRESH_TOZERO_INV = 4 #No change is made to the grayscale value of the pixel point that is greater than the threshold, and the grayscale value of the pixel point that is less than the threshold is all changed to 0. }

Return value:

retval: consistent with the parameter thresh

dst: result image

Note: Before binarization, we need to grayscale the color image to obtain a grayscale image.

1.2 Actual effect display

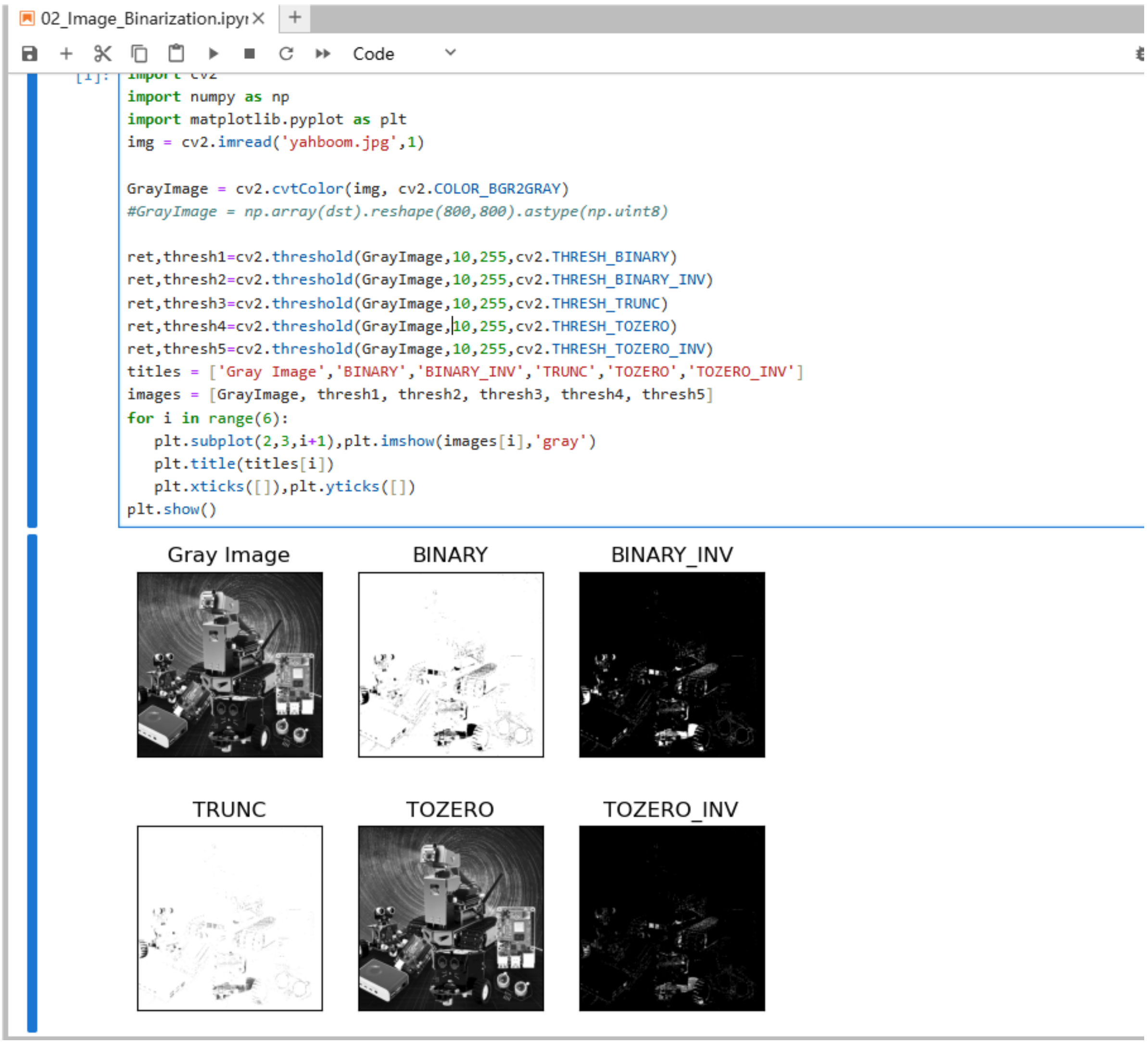
Source code path:

/home/pi/Rider-pi_class/4.Open Source CV/C.Image_Processing_Text_Drawing/02_Image_Binarization.ipynb

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

GrayImage = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#GrayImage = np.array(dst).reshape(800,800).astype(np.uint8)

ret,thresh1=cv2.threshold(GrayImage,10,255,cv2.THRESH_BINARY)
ret,thresh2=cv2.threshold(GrayImage,10,255,cv2.THRESH_BINARY_INV)
ret,thresh3=cv2.threshold(GrayImage,10,255,cv2.THRESH_TRUNC)
ret,thresh4=cv2.threshold(GrayImage,10,255,cv2.THRESH_TOZERO)
ret,thresh5=cv2.threshold(GrayImage,10,255,cv2.THRESH_TOZERO_INV)
titles = ['Gray Image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
images = [GrayImage, thresh1, thresh2, thresh3, thresh4, thresh5]
for i in range(6):
   plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')
   plt.title(titles[i])
   plt.xticks([]),plt.yticks([])
plt.show()
```



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

GrayImage = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#GrayImage = np.array(dst).reshape(800,800).astype(np.uint8)

ret,thresh1=cv2.threshold(GrayImage,10,255,cv2.THRESH_BINARY)
ret,thresh2=cv2.threshold(GrayImage,10,255,cv2.THRESH_BINARY_INV)
ret,thresh3=cv2.threshold(GrayImage,10,255,cv2.THRESH_TRUNC)
ret,thresh4=cv2.threshold(GrayImage,10,255,cv2.THRESH_TOZERO)
ret,thresh5=cv2.threshold(GrayImage,10,255,cv2.THRESH_TOZERO_INV)
titles = ['Gray Image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
images = [GrayImage, thresh1, thresh2, thresh3, thresh4, thresh5]
for i in range(6):
   plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')
   plt.title(titles[i])
   plt.xticks([]),plt.yticks([])
plt.show()
```