OpenCV introduction

1.OpenCV introduce

What is opencv? Its full name is open source computer vision library. As shown in the figure above, what we see is the logo of OpenCV. We can see that it is composed of three small rings of distinct R, G and B primary colors. In other words, it is an open source API function library for computer vision.This means:

(1) Whether it is scientific research or commercial application, it can be used for development;

(2) The source code of all API functions is public, and you can see the program steps of its internal implementation;

(3) You can modify the source code of OpenCV and compile the specific API functions you need.

The image processing on raspberry pie uses some functions of OpenCV's function library, or it can be said that it can not be separated from its existence in most image processing design fields. As early as many years ago, opencv has shown its talents in intrusion detection, specific target tracking, target detection, face detection, human face recognition, face tracking and other fields, and these are only the tip of its application. Since we realize that opencv is so universal, in this chapter, we will introduce some basic image processing functions and common functions used in our course. If you are interested in the development of OpenCV computer vision library and want to know more about it, the following websites are provided for your reference and study:

Opencv official home page: https://www.opencv.org

Opencv Chinese Forum: http://www.opencv.org.cn

Opencv CSDN Forum : https://bbs.csdn.net/forums/OpenCV

2.Opencv image reading and display

1. Image reading:

img = cv2.imread('yahboom.jpg', 0)   The first parameter is the path of the picture, and the second parameter is how to read the picture.

cv2.IMREAD_UNCHANGED：Keep the original format unchanged , -1;

cv2.IMREAD_COLOR：Read the picture in grayscale mode, which can be represented by 0;

cv2.IMREAD_GRAYSCALE,1： Read in a color picture, which can be represented by 1; Default
cv2.IMREAD_UNCHANGED： Read in a picture and include its alpha channel, which can be represented by 2.

2. Display of images

```
import cv2

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

# cv2.imshow('image', img)    #This line can only be executed by the PY file at the command line, and a video window will pop up

# cv2.waitKey (0)
```

```
#Bgr8 to JPEG format

import enum

import cv2



def bgr8_to_jpeg(value, quality=75):

return bytes(cv2.imencode('.jpg', value)[1])
```

```
# The image component in jupyterlab displays the read image

import ipywidgets.widgets as widgets

image_widget = widgets.Image(format='jpg', width=800, height=800)

display(image_widget)



image_widget.value = bgr8_to_jpeg(img)
```

3.Opencv picture writing

Function method: cv2.imwrite('yahboom1.jpg',   img)

The first parameter is the saved file name, and the second parameter is the saved image.

The first parameter is the saved file name, and the second parameter is the saved image. Next, we will demonstrate the method of image writing. First, read an image yahboom.jpg, and then write it to yahboom.jpg.

```
import cv2



# (1) file reading (2) package format analysis (3) data decoding (4) data loading

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

# cv2.imshow('yahboom, img)    #See below and pay attention to the explanation

cv2.imwrite('yahboom1.jpg', img) # 1 name 2 data
```

Note: Here we cannot execute the cv2.imshow ('yahboom, IMG) function in jupylab. If you need to use this sentence to display the read image, you need to execute the python file in the raspberry pie image interface through the command: python3 xx.py.

```
#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_widget = widgets.Image(format='jpg', width=320, height=240)

display(image_widget)

img = cv2.imread('yahboom1.jpg',1)

image_widget.value = bgr8_to_jpeg(img)
```

4. Opencv image quality

1.Compression mode:

cv2.imwrite('yahboomTest.jpg',img,[cv2.IMWRITE_JPEG_QUALITY,   50])

cv2.CV_IMWRITE_JPEG_QUALITY： Set the image quality with the image format of. JPEG or. Jpg. The value is 0 --- 100 (the higher the value, the higher the quality). The default is 95.

cv2.CV_IMWRITE_WEBP_QUALITY： Set the picture quality in the format of. Webp, and the value is 0 -- 100.

cv2.CV_IMWRITE_PNG_COMPRESSION： Set the compression ratio of. PNG format. The value is 0-9 (the larger the value, the larger the compression ratio). The default value is 3.

```
import cv2

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

cv2.imwrite('yahboomTest.jpg',   img, [cv2.IMWRITE_JPEG_QUALITY, 50])

#1M 100k 10k 0-100 Lossy compression
```

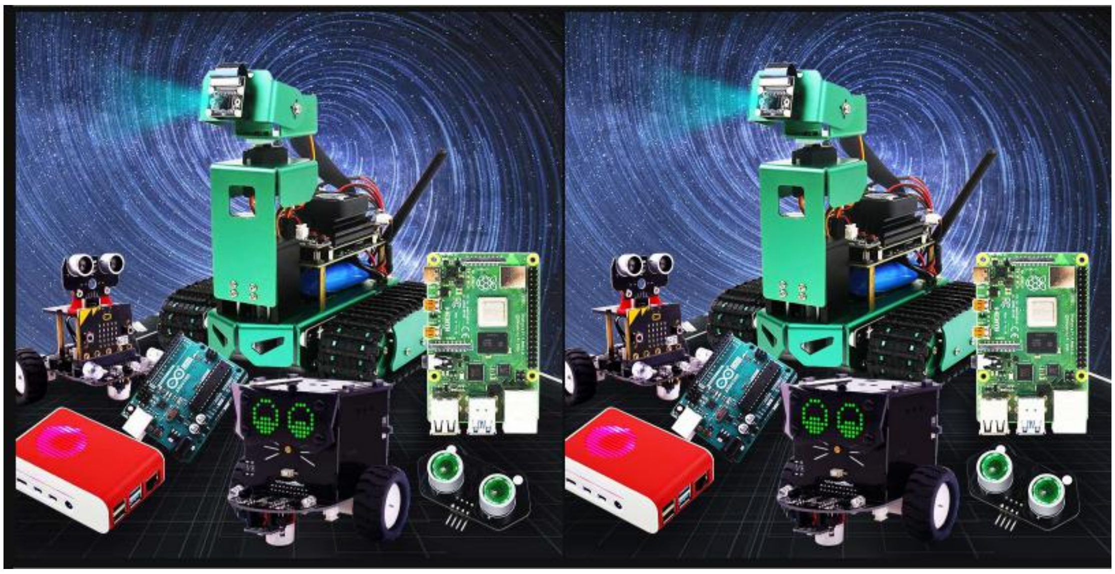
```
# 1:lossless     2:transparency attribute
```

```
import cv2

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

cv2.imwrite('yahboomTest.png', img, [cv2.IMWRITE_PNG_COMPRESSION,0])

# jpg 0 High compression ratio 0-100          png 0 Low compression ratio 0-9
```

```
#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)



img1 = cv2.imread('yahboomTest.jpg',1)
```

```
img2 = cv2.imread('yahboomTest.png',1)

image_widget1.value = bgr8_to_jpeg(img1)

image_widget2.value = bgr8_to_jpeg(img2)
```



5.Opencv pixel operation

Pixel operation, we can change any position to a new pixel color. Here, we first read the image, and then assign a region as white.

```
import cv2



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

(b,g,r) = img[100,100]

print(b,g,r)# bgr

#10 100 --- 110 100

i=j=0

for j in range(1,500):

    img[i,j] = (255,255,255)

    for i in range(1,500):
            img[i,j] = (255,255,255)



# cv2.imshow('image',img)

# cv2.waitKey(0) #1000 ms
```

```
#Bgr8 to JPEG format

import enum

import cv2



def bgr8_to_jpeg(value, quality=75):

      return bytes(cv2.imencode('.jpg', value)[1])
```

Use jupyterlab to display the comparison of images before and after:

```
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)



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

```
image_widget1.value = bgr8_to_jpeg(img1)    #original

image_widget2.value = bgr8_to_jpeg(img)     #Pixel operated
```