Color recognition

1.Introduction

OpenCV is a cross-platform computer vision and machine learning software library based on BSD license (open source), which can run on Linux, on Windows, Android and MacOS operating systems.

[1] It is lightweight and efficient - structured from a set of C functions and a handful of C++ classes.
At the same time, it provides the interface of Python, Ruby, MATLAB and other languages, and realizes many common algorithm in image processing and computer vision.

2. Color recognition

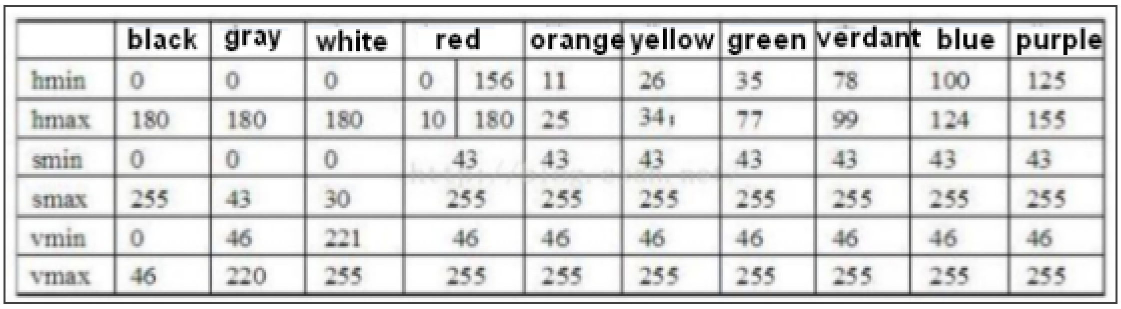
- **HSV**

H: 0 — 180

S: 0 — 255

V: 0 — 255

Part of the red is classified as the purple range here:



	black	gray	white	red		orange	yellow	green	verdant	blue	purple
hmin	0	0	0	0	156	11	26	35	78	100	125
hmax	180	180	180	10	180	25	34	77	99	124	155
smin	0	0	0	43		43	43	43	43	43	43
smax	255	43	30	255		255	255	255	255	255	255
vmin	0	46	221	46		46	46	46	46	46	46
vmax	46	220	255	255		255	255	255	255	255	255

3. Code

Code path: ~/astra_ws/src/astra_visual/color

3.1 Code analysis to identify a picture

First, open a picture to identify the color of the picture, and analyze the important code to open a picture below.

The code to open a picture is the color_text.py file.

```
while 1:
    frame = cv2.imread("./test.png")
```

The function of this line of code is to open the picture of test.png in the same directory as the color_text.py file.

Then invert and grayscale the image, the purpose is to identify the color in the image more accurately in the future, the code is as follows.

```
frame=cv2.flip(src=frame,flipCode=2)
hsv_frame=cv2.cvtColor(src=frame,code=cv2.COLOR_BGR2HSV)
```

If it is a colorful picture, the machine may be disturbed and cannot get the correct result. Therefore, in order to ensure the accuracy, we can only select a certain area of a picture for detection, so as to obtain the color.

Code as show below.

```
#获取检测点位置
cx = width // 2 + change_X
cy=height//2 + change_Y

#绘制检测区域
cv2.rectangle(img=frame,pt1=(cx-10,cy-10),pt2=(cx+10,cy+10),color=(0,255,0),thickness=2)
```

The next step is to convert the selected area to HSV color.

```
pixel_center=hsv_frame[cy,cx]
```

Then, compare the HSV table to get the color of the area.

```
for i in range(len(plate_type)):
    color = color_shibei(pixel_center,color_dict,plate_type[i])
    if color == plate_type[i]:
        break
```

```
#(数组，字典，字符串)
def color_shibei(color_hsv,dictpp,color_str):
    if color_hsv[0] >= dictpp[color_str][0][0] and color_hsv[0] <= dictpp[color_str][1][0] :
        if color_hsv[1] >= dictpp[color_str][0][1] and color_hsv[1] <= dictpp[color_str][1][1] :
            if color_hsv[2] >= dictpp[color_str][0][2] and color_hsv[2] <= dictpp[color_str][1][2] :
                return color_str
    return "Unkown"
```

3.2 Code analysis to open camera

After startup, the system defaults to [Target Detection Mode], as shown below.

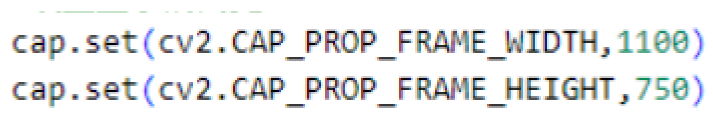
In the previous step, we got the color of a certain area of a picture, and the next step is to open the code analysis of the camera.
First, copy the code that gets the color from a picture and rename it to color_main.py, and then regard opening the camera as opening a video

video, and the function call of opencv to open a video is cap=cv2.VideoCapture("video path")

The setting path to open the camera of Rising Sun X3 is: /dev/video0-8

So the code to open the camera is: cap=cv2.VideoCapture(8)

After turning on the camera, opencv can set the size of the screen, the code is as shown below.



```
cap.set(cv2.CAP_PROP_FRAME_WIDTH,1100)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT,750)
```

4.Color recognition result

After writing the code, click Save and enter in the terminal: python3 color_main.py

You can clearly see the picture displayed by the camera, and the color of a certain area is displayed in the picture.