Chapter3: HSV color space conversion（RGB-HSV）

！Note: When running the program of this course, there must be a desktop for displaying pictures. It is recommended that you use VNC to log in to the system so that the pictures can be displayed.

1.1 Introduction to color space

RGB is also known as the three primary color space. Any color can be a mixture of these three colors. However, the effective processing of the image in the color space is generally performed in the HSV space. HSV (Humidity, Saturation, Brightness Value) is a color space created according to the intuitive characteristics of the color, also called the hexagonal cone model.

Detail: https://blog.csdn.net/taily_duan/article/details/51506776

HSV space is wider and more convenient than RGB space recognition.

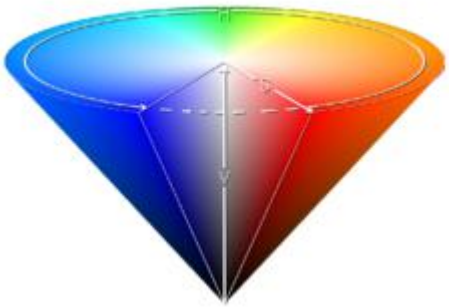


Figure 1-1 HSV color space model

1.2 Three color spaces conversion (gray BGR HSV)

The common two color conversion methods: BGR->Gray and BGR->HSV.

!!!Note: Gray and HSV cannot be converted to each other.

Color space conversion function:

cv2.cvtColor(input_image, flag)

BGR->Gray: flag is cv2.COLOR_BGR2GRAY

BGR->HSV: flag is cv2.COLOR_BGR2HSV

The value range of the HSV color space in OpenCV: H [0, 179] S [0, 255] V [0, 255]

	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

Figure 1-2 Range of commonly used colors

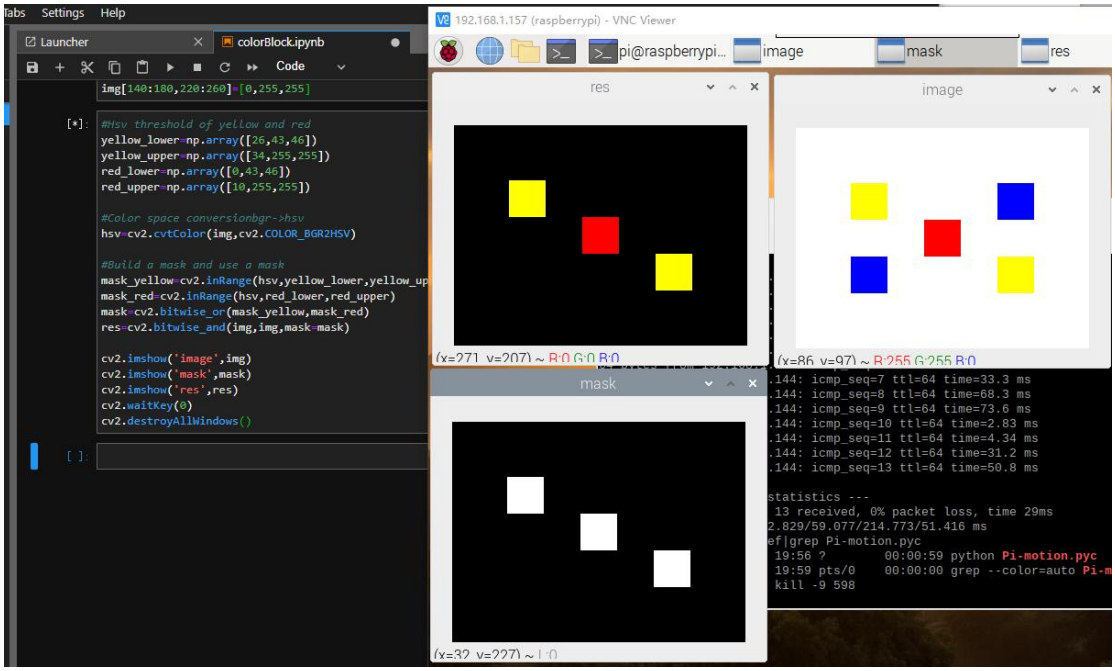
The source code of the program is located

/home/pi/yahboom/colorBlock/colorBlock.py

The program is shown below:

```
* @par Copyright (C): 2010-2019, Shenzhen Yahboom Tech
* @file         colorBlock
* @version      V1.0
* @details
* @par History
* @author       LongfuSun
"""

import cv2
import numpy as np

#Create picture and Color block
img=np.ones((240,320,3),dtype=np.uint8)*255
img[100:140,140:180]=[0,0,255]
img[60:100,60:100]=[0,255,255]
img[60:100,220:260]=[255,0,0]
img[140:180,60:100]=[255,0,0]
img[140:180,220:260]=[0,255,255]

#Hsv threshold of yellow and red
yellow_lower=np.array([26,43,46])
yellow_upper=np.array([34,255,255])
red_lower=np.array([0,43,46])
red_upper=np.array([10,255,255])

#Color space conversionbgr->hsv
hsv=cv2.cvtColor(img,cv2.COLOR_BGR2HSV)

#Build a mask and use a mask
mask_yellow=cv2.inRange(hsv,yellow_lower,yellow_upper)
mask_red=cv2.inRange(hsv,red_lower,red_upper)
mask=cv2.bitwise_or(mask_yellow,mask_red)
res=cv2.bitwise_and(img,img,mask=mask)

cv2.imshow('image',img)
cv2.imshow('mask',mask)
cv2.imshow('res',res)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

After the program is running, we can see the picture displayed on the Raspberry Pi system desktop, as shown below.



Then, we press the "S" key or "ECS" on the keyboard to exit the process.