- # Color threshold adjustment color block calibration

By adjusting the high and low thresholds of HSV, interfering colors are filtered out, so that the blocks can be ideally identified in complex environments. It is best to calibrate when using color-based gameplay for the first time.

Code path:

```
~/jetcobot_ws/src/jetcobot_color_identify/scripts/HSV_calibration.ipynb
```

2.1, HSV introduction

HSV (Hue, Saturation, Value) is a color space created by A. R. Smith in 1978 based on the intuitive characteristics of color, also known as the Hexcone Model. The color parameters in this model are: hue (H), saturation (S), and brightness (V).

H: 0 — 180

S: 0 — 255

V: 0 — 255

- HSV parameter table:

	Black	Gray	White	Red	Orange	Yellow	Green	Cyan	Blue	Purple		
H_min	0	0	0	0	156	11	26	35	78	100	125	
H_max	180	180	180	10	180	25	34	77	99	124	155	
S_min	0	0	0	43	43	43	43	43	43	43		
S_max	255	43	30	255	255	255	255	255	255	255		
V_min	0	0	0	46	46	46	46	46	46			
V_max	46	220	255	255	255	255	255	255	255			

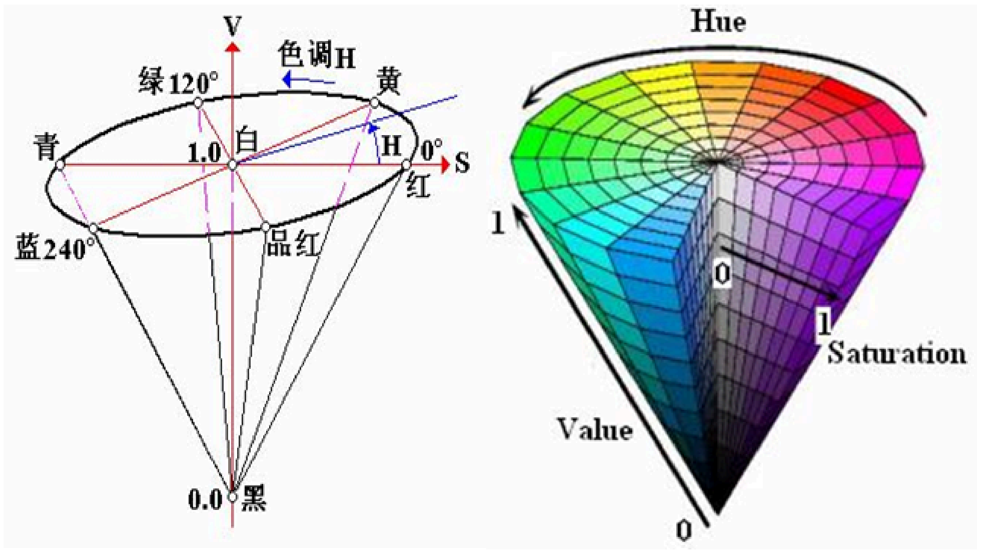
- HSV hexagonal pyramid

(1) Hue H. Represents color information, that is, the position of the spectral color. This parameter is expressed as an angle, ranging from 0° to 360°, starting from red and counting counterclockwise, with red being 0°, green being 120°, and blue being 240°. Their complementary colors are: yellow is 60°, cyan is 180°, and purple is 300°.

(2) Saturation S. Saturation S: It is expressed as the ratio between the purity of the selected color and the maximum purity of the color. When S=0, there is only grayscale. They are 120 degrees apart. The complementary colors differ by 180 degrees. A color can be regarded as the result of mixing a certain spectral color with white. The greater the proportion of the spectral color, the closer the color is to the spectral color, and the higher the saturation of the color. The higher the saturation, the deeper and brighter the color. The white light component of the spectral color is 0, and the saturation reaches the highest. The value range is usually 0% to 100%. The larger the value, the more saturated the color.

(3) Brightness V. Brightness indicates the brightness of the color. For light source color, the brightness value is related to the brightness of the light source; for object color, this value is related to the transmittance or reflectance of the object. The value range is usually 0% (black) to 100% (white). One thing to note: there is no direct connection between it and light intensity. The three-dimensional representation of the HSV model evolved from the RGB

cube. Imagine looking from the white vertex along the diagonal of the RGB cube to the black vertex, and you can see the hexagonal shape of the cube. The hexagonal border represents the color, the horizontal axis represents the purity, and the brightness is measured along the vertical axis.



2.2, Code design

- Import header files

```python
import threading
import cv2 as cv
from time import sleep
from jetcobot_config import *
import ipywidgets as widgets
from IPython.display import display
```

- Create an instance and initialize parameters

```python
# Create and update HSV instance
update_hsv = update_hsv()
# Initialize num parameter
num=0
# Initialize mode
model = "General"
# Initialize HSV name
HSV_name=None
# Initialize HSV value
color_hsv = {"red" : ((0, 143, 163), (11, 255, 255)),

"green" : ((55, 80, 66), (78, 255, 255)),

"blue" : ((110, 100, 121), (117, 255, 255)),

"yellow": ((26, 100, 91), (32, 255, 255))}

# HSV parameter path (jupyter lab and ros function packages need to use absolute paths)
HSV_path="/home/jetson/jetcobot_ws/src/jetcobot_color_identify/scripts/HSV_config.txt"
```

```
# Read HSV configuration file and update HSV value
try: read_HSV(HSV_path,color_hsv)
except Exception: print("No HSV_config file!!!")
```

- Create widgets

```
button_layout = widgets.Layout(width='320px', height='55px', align_self='center')
output = widgets.Output()
# Enter color update mode
HSV_update_red= widgets.Button(description='HSV_update_red',
button_style='success',layout=button_layout)
...
# Adjust the slider

H_min_slider=widgets.IntSlider(description='H_min
:',value=0,min=0,max=255,step=1, orientation='horizontal')
...
# Exit
exit_button=widgets.Button(description='Exit',button_style='danger',layout=button
_layout)
```

- Color update callback

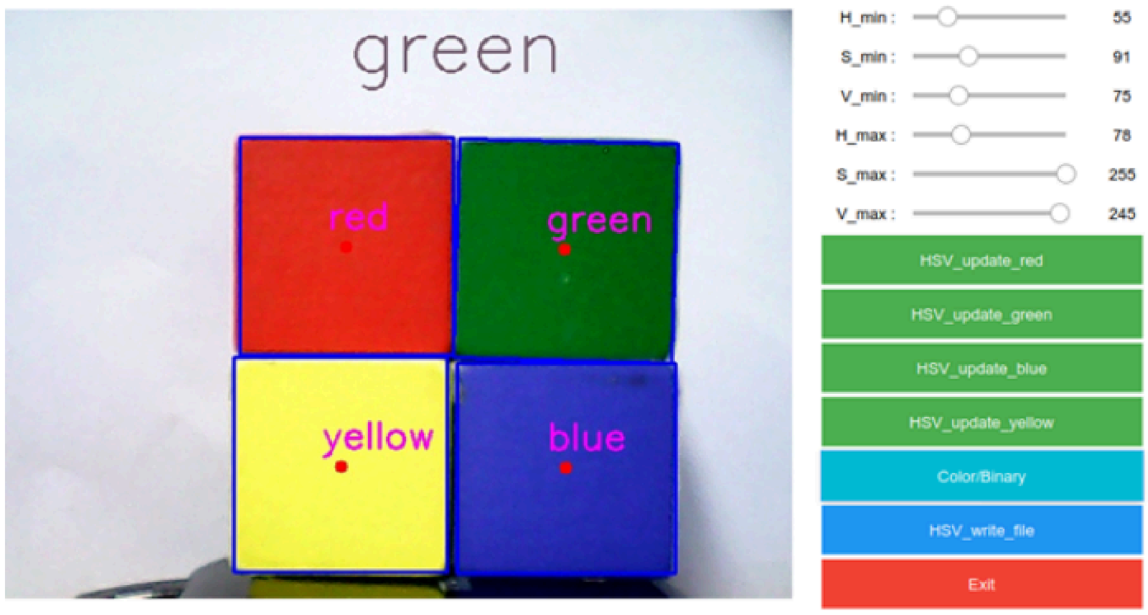
```
def update_red_Callback(value):
pass

def update_green_Callback(value):
pass

def update_blue_Callback(value):
pass

def update_yellow_Callback(value):
pass

HSV_update_red.on_click(update_red_Callback)
HSV_update_green.on_click(update_green_Callback)
HSV_update_blue.on_click(update_blue_Callback)
HSV_update_yellow.on_click(update_yellow_Callback)
```

- Mode switch control

```
def write_file_Callback(value):
pass

def Color_Binary_Callback(value):
pass

def exit_button_Callback(value): pass

HSV_write_file.on_click(write_file_Callback)
Color_Binary.on_click(Color_Binary_Callback)
exit_button.on_click(exit_button_Callback)
```

```
For detailed code, see jetcobot_ws/src/jetcobot_color_identify/scripts/HSV校
正.ipynb
```

- Interface example



H_min : 55
S_min : 91
V_min : 75
H_max : 78
S_max : 255
V_max : 245

The upper part of the screen shows which color is selected. The six sliders on the upper right correspond to the six values of HSV. Slide the slider to adjust the HSV threshold of each color in real time.

2.3. Operation process

(1). After starting all code blocks, the interface shown in the figure is displayed at the bottom of the code. The default color selection is empty, so no color is recognized.

(2). Click the [HSV_update_green] button to start recognizing green objects (Note: This color recognition detects the contour, so the object can only be recognized normally when it is completely within the camera range). Slide the slider on the upper right to adjust the green HSV threshold. When adjusting, pay attention to multi-directional adjustment and adjust in different field of view environments until the object can be clearly recognized in complex environments without being blocked by other objects. Until the object interferes [other colors are similar].

(3). The [Color/Binary] button switches between color and binary images, and is only valid after the color is selected. After switching, only the selected color binary image is displayed, which is more convenient for us to debug.

(4). [HSV_write_file] button, after debugging the HSV values of all colors, click this button. Save all the debugged parameters in the [.txt] format in the same path as the code, and automatically read the file parameters next time it is started.

(5). [Exit] button, turn off the camera and exit the program.