# Color Block Tracking

Orin board users can directly open a web page and enter IP address:8888 to access jupyter-lab and run directly. Jetson-Nano board users need to first enter the docker container, then enter the following command in docker:

```
cd
jupyter-lab --allow-root
```

Then open a web page and enter IP address:9999 to access jupyter-lab and run the following program.

This tutorial is an advanced version of the color block positioning experiment, adding robotic arm tracking functionality.

Overview: Target tracking includes two modes: color tracking and color picking tracking (learning tracking). The principle is to process the image through the camera in a certain way, identify the target through specific methods, obtain the coordinate position of the target under the camera, calculate the deviation value between the target center point and the image center point, adjust through PID algorithm, and drive the robotic arm to move. This makes the target center point coincide with the image center point.

Code path:

```
#Jetson-Nano users need to enter the docker container to view
~/dofbot_pro/dofbot_color_follow/scripts/Color_follow.ipynb
```

## 1. Widget Design

- Import header files

```
import cv2 as cv
import threading
import random
from time import sleep
import ipywidgets as widgets
from IPython.display import display
from color_follow import color_follow
from dofbot_utils.fps import FPS
from dofbot_utils.robot_controller import Robot_Controller
from dofbot_utils.dofbot_config import *
```

- Create instances, initialize parameters

```
robot = Robot_Controller()
robot.move_init_pose()
fps = FPS()
follow = color_follow()
model = 'General'
HSV_learning = ((0, 240, 54), (8, 255, 255))
# HSV_learning = ()
color_hsv = {"red": ((0, 25, 90), (10, 255, 255)),
```

```
            "green": ((53, 36, 40), (80, 255, 255)),
            "blue": ((110, 80, 90), (120, 255, 255)),
            "yellow": ((25, 20, 55), (50, 255, 255))}
color = [[random.randint(0, 255) for _ in range(3)] for _ in range(255)]
HSV_path="/home/jetson/dofbot_pro/dofbot_color_follow/scripts/HSV_config.txt"
try: read_HSV(HSV_path,color_hsv)
except Exception: print("Read HSV_config Error !!!")
```

- Create widgets

```
button_layout = widgets.Layout(width='200px', height='100px',
align_self='center')
# 输出控件 Output widget
output = widgets.Output()
# 颜色追踪 Color tracking
color_follow = widgets.Button(description='color_follow', button_style='success',
layout=button_layout)
# 选择颜色 Select color
choose_color = widgets.ToggleButtons(options=['red', 'green', 'blue', 'yellow'],
button_style='success',
            tooltips=['Description of slow', 'Description of regular',
'Description of fast'])
# 取消追踪 Cancel tracking
follow_cancel = widgets.Button(description='follow_cancel',
button_style='danger', layout=button_layout)
# 学习颜色 Learn color
learning_color = widgets.Button(description='learning_color',
button_style='primary', layout=button_layout)
# 学习颜色追踪 Learn color tracking
learning_follow = widgets.Button(description='learning_follow',
button_style='success', layout=button_layout)
# 退出 exit
exit_button = widgets.Button(description='Exit', button_style='danger',
layout=button_layout)
# 图像控件 Image widget
imgbox = widgets.Image(format='jpg', height=480, width=640,
layout=widgets.Layout(align_self='auto'))
# 垂直布局 Vertical layout
img_box = widgets.VBox([imgbox, choose_color],
layout=widgets.Layout(align_self='auto'))
# 垂直布局 Vertical layout
Slider_box = widgets.VBox([color_follow, learning_color,
learning_follow,follow_cancel,exit_button],
                          layout=widgets.Layout(align_self='auto'))
# 水平布局 Horizontal layout
controls_box = widgets.HBox([img_box, Slider_box],
layout=widgets.Layout(align_self='auto'))
# ['auto', 'flex-start', 'flex-end', 'center', 'baseline', 'stretch', 'inherit',
'initial', 'unset']
```

- Mode switching

```
def color_follow_Callback(value):
    global model
    model = 'color_follow'
def learning_color_Callback(value):
```

```
    global model
    model = 'learning_color'
def learning_follow_Callback(value):
    global model
    model = 'learning_follow'
def follow_cancel_Callback(value):
    global model
    model = 'General'
    robot.move_init_pose()
def exit_button_Callback(value):
    global model
    model = 'Exit'
color_follow.on_click(color_follow_Callback)
learning_color.on_click(learning_color_Callback)
learning_follow.on_click(learning_follow_Callback)
follow_cancel.on_click(follow_cancel_Callback)
exit_button.on_click(exit_button_Callback)
```

- Main program

```
def camera():
    global HSV_learning,model
    # 打开摄像头 Open camera
    capture = cv.VideoCapture(0, cv.CAP_V4L2)
    capture.set(3, 640)
    capture.set(4, 480)
    capture.set(5, 30)
    # Be executed in loop when the camera is opened normally
    while capture.isOpened():
        try:
            _, img = capture.read()
            fps.update_fps()
            if model == 'color_follow':
                img = follow.follow_function(img, color_hsv[choose_color.value])
                cv.putText(img, choose_color.value, (int(img.shape[0] / 2), 50),
cv.FONT_HERSHEY_SIMPLEX, 2, color[random.randint(0, 254)], 2)
            if model == 'learning_color':
                img,HSV_learning = follow.get_hsv(img)
            if model == 'learning_follow' :
                if len(HSV_learning)!=0:
                    print("HSV_learning", HSV_learning)
                    img = follow.learning_follow(img, HSV_learning)
                    cv.putText(img,'LeColor', (240, 50), cv.FONT_HERSHEY_SIMPLEX,
1, color[random.randint(0, 254)], 1)
            if model == 'Exit':
                cv.destroyAllWindows()
                capture.release()
                break
            fps.show_fps(img)
            imgbox.value = cv.imencode('.jpg', img)[1].tobytes()
        except KeyboardInterrupt:capture.release()
```

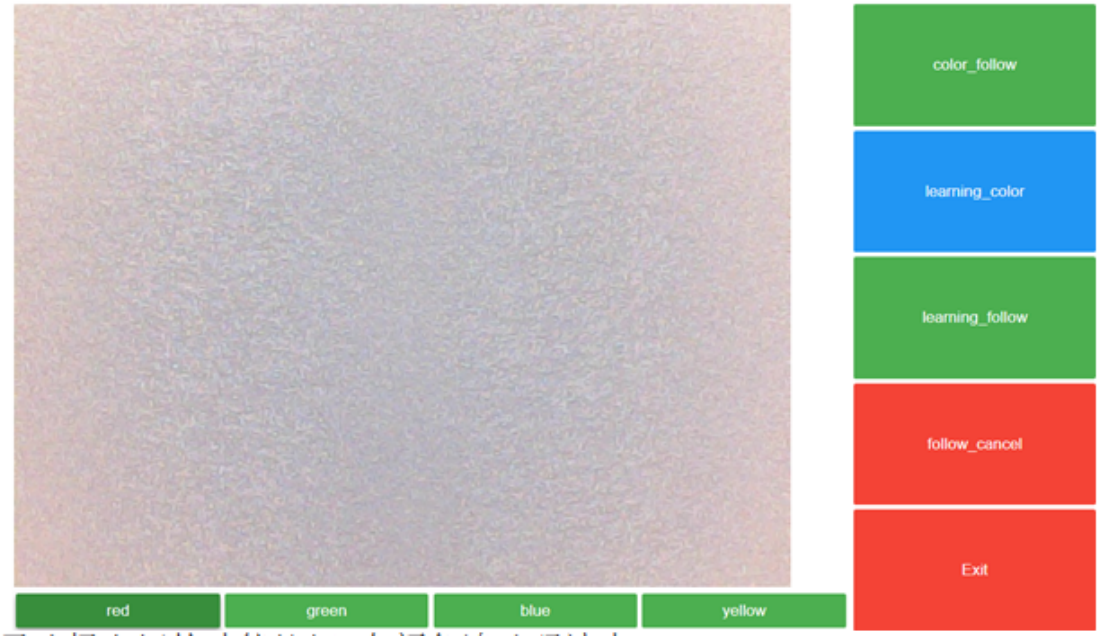
- Startup

```
display(controls_box,output)
threading.Thread(target=camera, ).start()
```

## 2. Run the Program

Click the run entire program button on the jupyterlab toolbar, then scroll to the bottom.


- **Color Tracking**



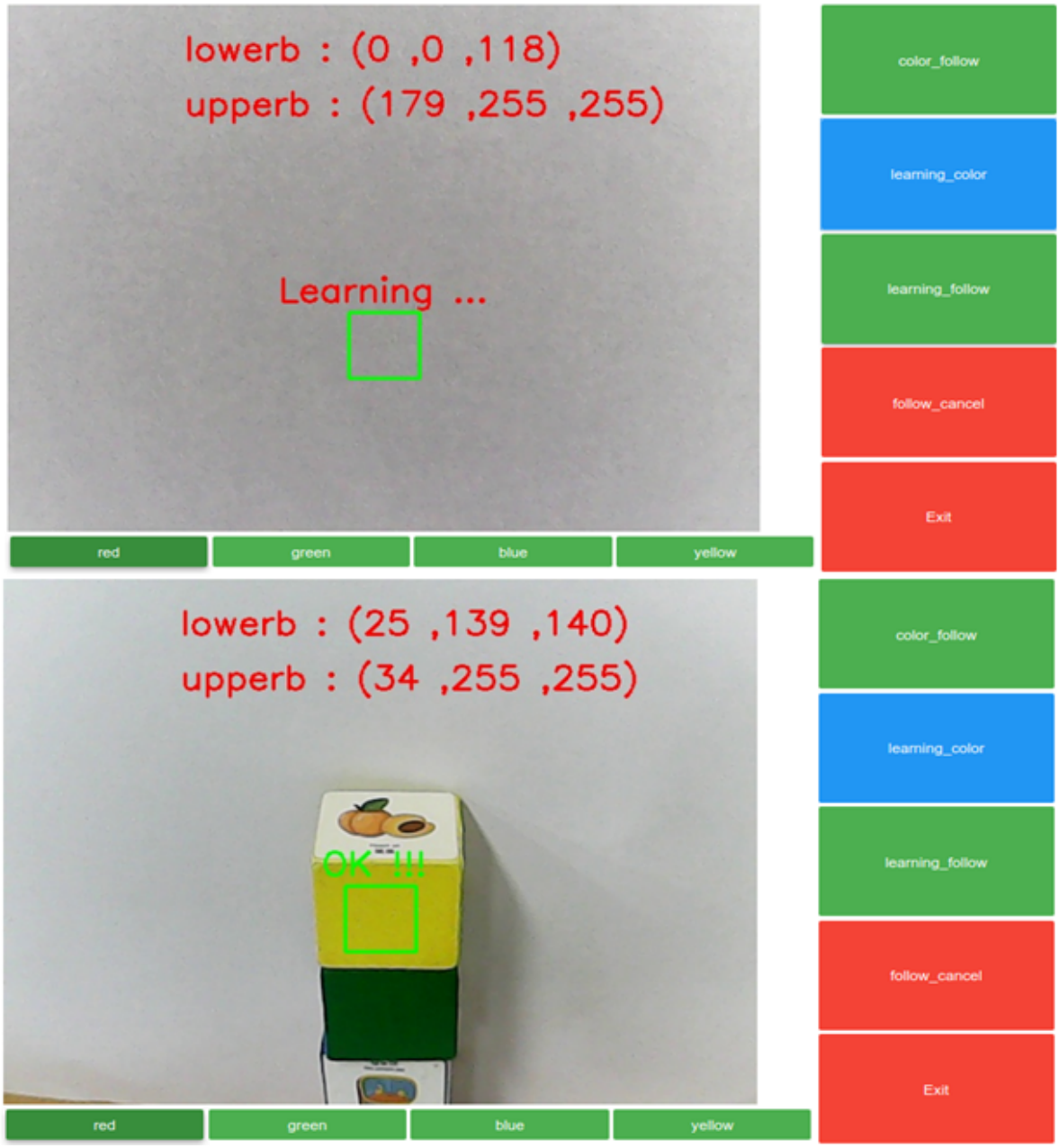

The default mode is no tracking, you need to select the corresponding function button.

Start the code block, click the [color_follow] button to start color tracking. You can switch colors below the image. Place the block in the field of view, wait for the camera to recognize it, and the robotic arm will track the block movement, facing the center of the block.

Click the [follow_cancel] button to cancel tracking.

Click the [Exit] button to exit the program.

- **Color Picking Tracking**




(1) Start the code block, click the [learning_color] button, a box will appear in the center of the image, as shown in the left image.

(2) Place an object within the box, the HSV high and low thresholds will be printed in real time. When [OK !!!] appears below the box, it indicates successful recognition.

(3) Click the [learning_follow] button, as long as the contour of the learned color can be detected, it can track in real time.

(4) Click the [follow_cancel] button to cancel tracking.

Click the [Exit] button to exit the program.