Color recognition grab blocks

1. Functional introduction

The color recognition grab block function mainly detects the HSV value of the color in the image box, distinguishes the color of the current block according to the HSV value, and then the robot arm rotates to the position of the corresponding color on the map to grab the block and put it in the middle cross position.

Note: Before starting the program, please follow the [Assembly and Assembly Tutorial] -> [Install Map] tutorial and install the map correctly before operating.

Code path:

```
~/dofbot_pro/dofbot_color_grab/scripts/Color_Grab.ipynb
```

2. Code block design

- Import header files

```
import cv2 as cv
import threading
import ipywidgets as widgets
from IPython.display import display
from color_grab import Color_Grab
from dofbot_utils.fps import FPS
from dofbot_utils.robot_controller import Robot_Controller
```

- Create an instance and initialize parameters

```
grab = Color_Grab()
robot = Robot_Controller()
robot.move_look_front()

fps = FPS()
model = 'Start'
```

- Create controls

```
# Create control layout Create widget layout
button_layout = widgets.Layout(width='200px', height='70px', align_self='center')
# Output printing Output printing
output = widgets.Output()
# exit button exit button
exit_button = widgets.Button(description='Exit', button_style='danger',
layout=button_layout)
# Image widget Image widget
imgbox = widgets.Image(format='jpg', height=480, width=640,
layout=widgets.Layout(align_self='center'))
# Vertical placement Vertical placement
controls_box = widgets.VBox([imgbox, exit_button],
layout=widgets.Layout(align_self='center'))
# ['auto', 'flex-start', 'flex-end', 'center', 'baseline', 'stretch', 'inherit',
'initial', 'unset']
```

- Switch mode

```
def exit_button_Callback(value):
    global model
    model = 'Exit'
#     with output: print(model)
exit_button.on_click(exit_button_Callback)
```

- Main program

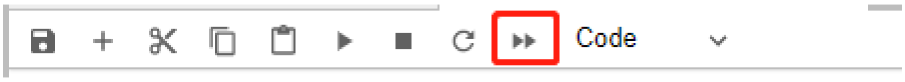
```
def camera():
    # 打开摄像头 Open camera
    capture = cv.VideoCapture(0, cv.CAP_V4L2)
    capture.set(cv.CAP_PROP_FRAME_WIDTH, 640)
    capture.set(cv.CAP_PROP_FRAME_HEIGHT, 480)
    while capture.isOpened():
        try:
            if model == 'Exit':
                capture.release()
                break
            _, img = capture.read()
            fps.update_fps()
            # 获得运动信息  Get motion information
            img = grab.start_grab(img)
            fps.show_fps(img)
            imgbox.value = cv.imencode('.jpg', img)[1].tobytes()
        except KeyboardInterrupt:capture.release()
        except Exception as e:
            print(e)
```

- Start

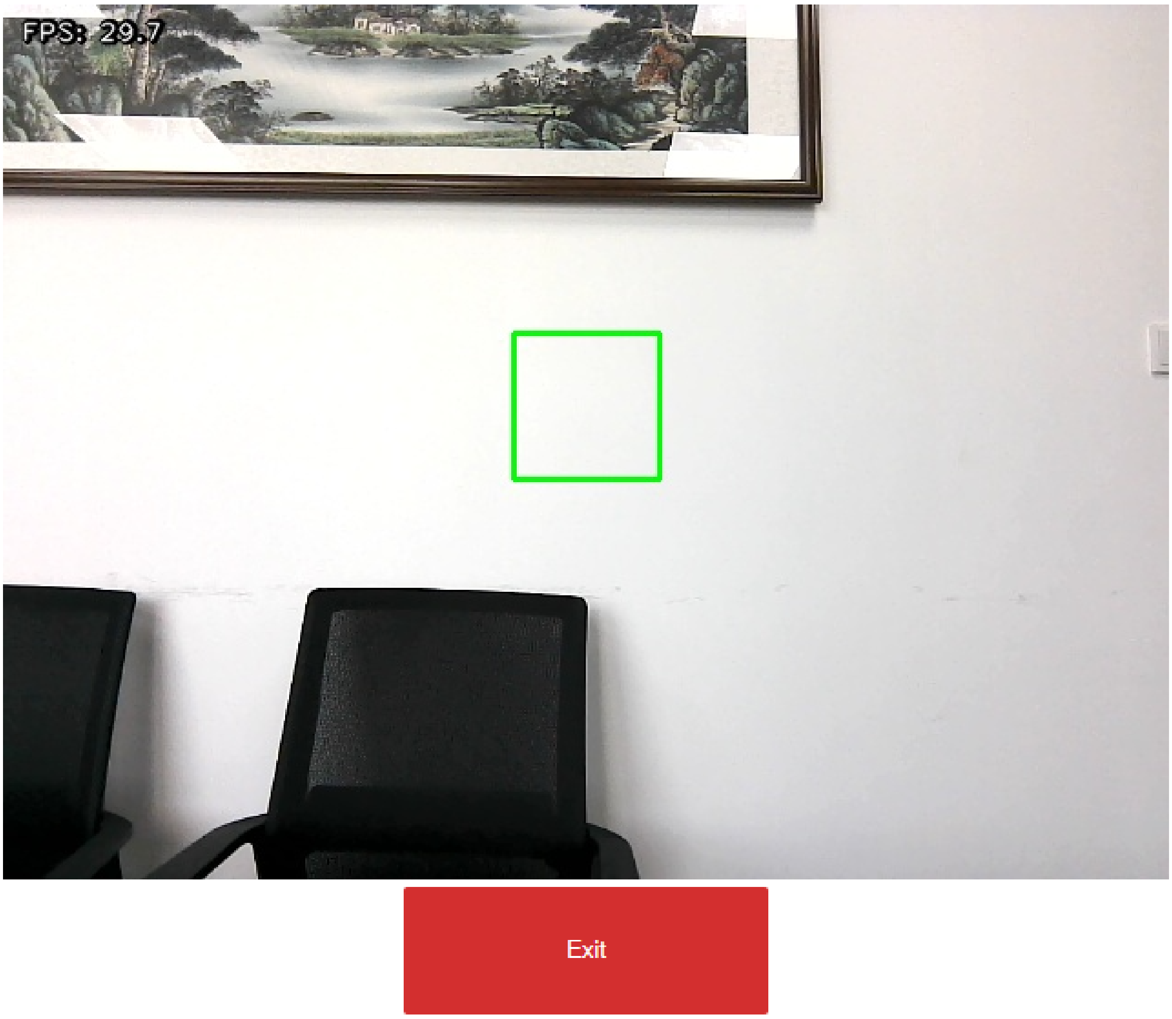
```
display(controls_box,output)
threading.Thread(target=camera, ).start()
```

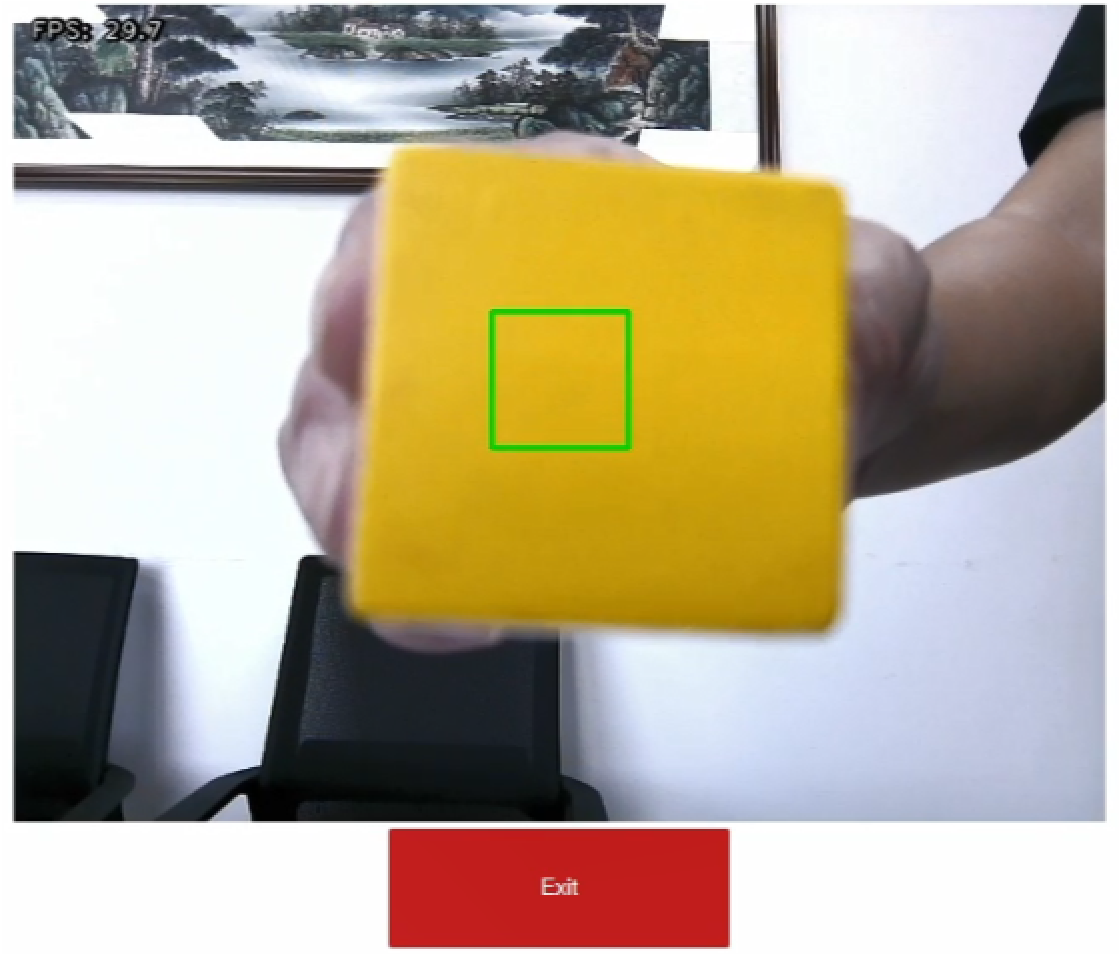
3. Run the program

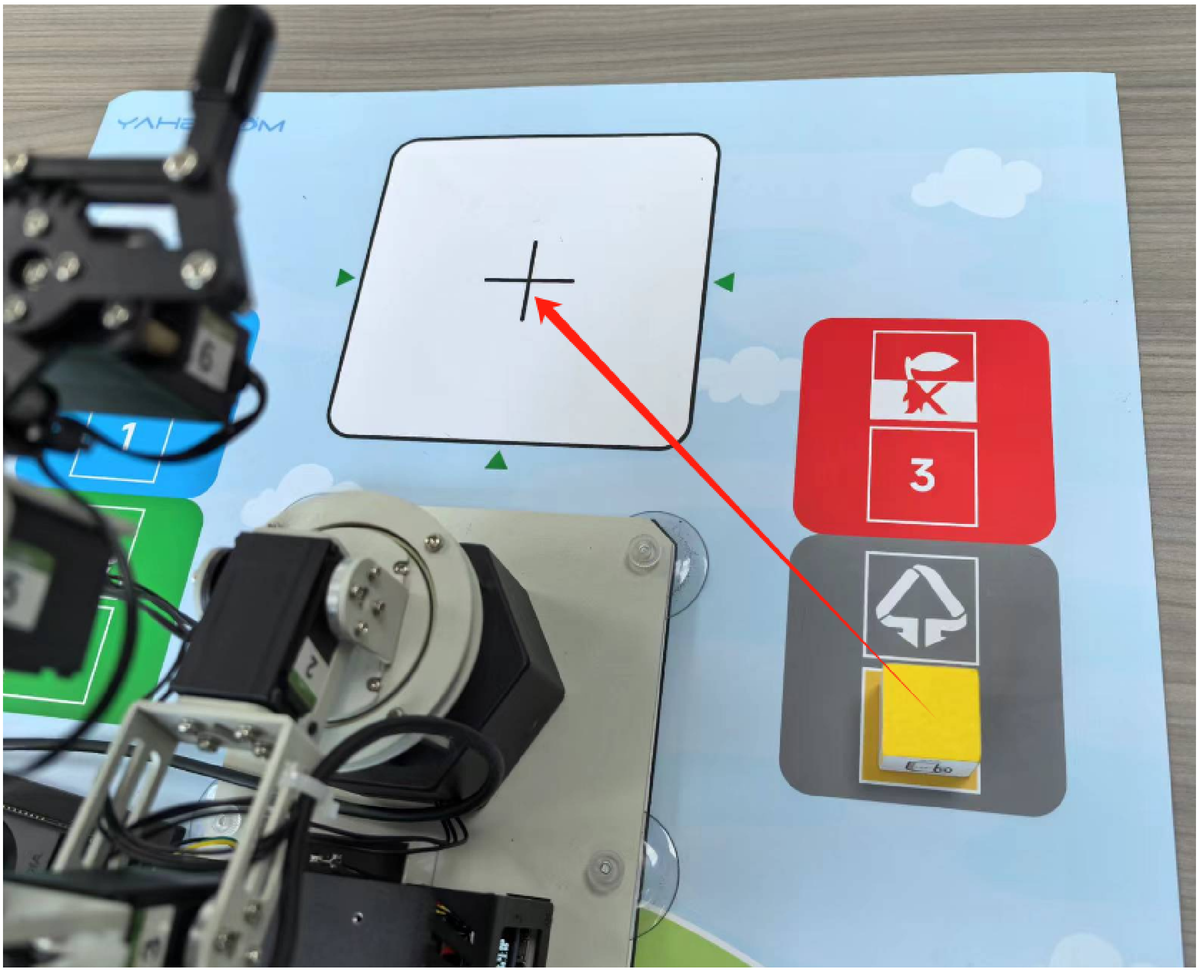
Click the Run the entire program button on the jupyterlab toolbar and then scroll to the bottom.

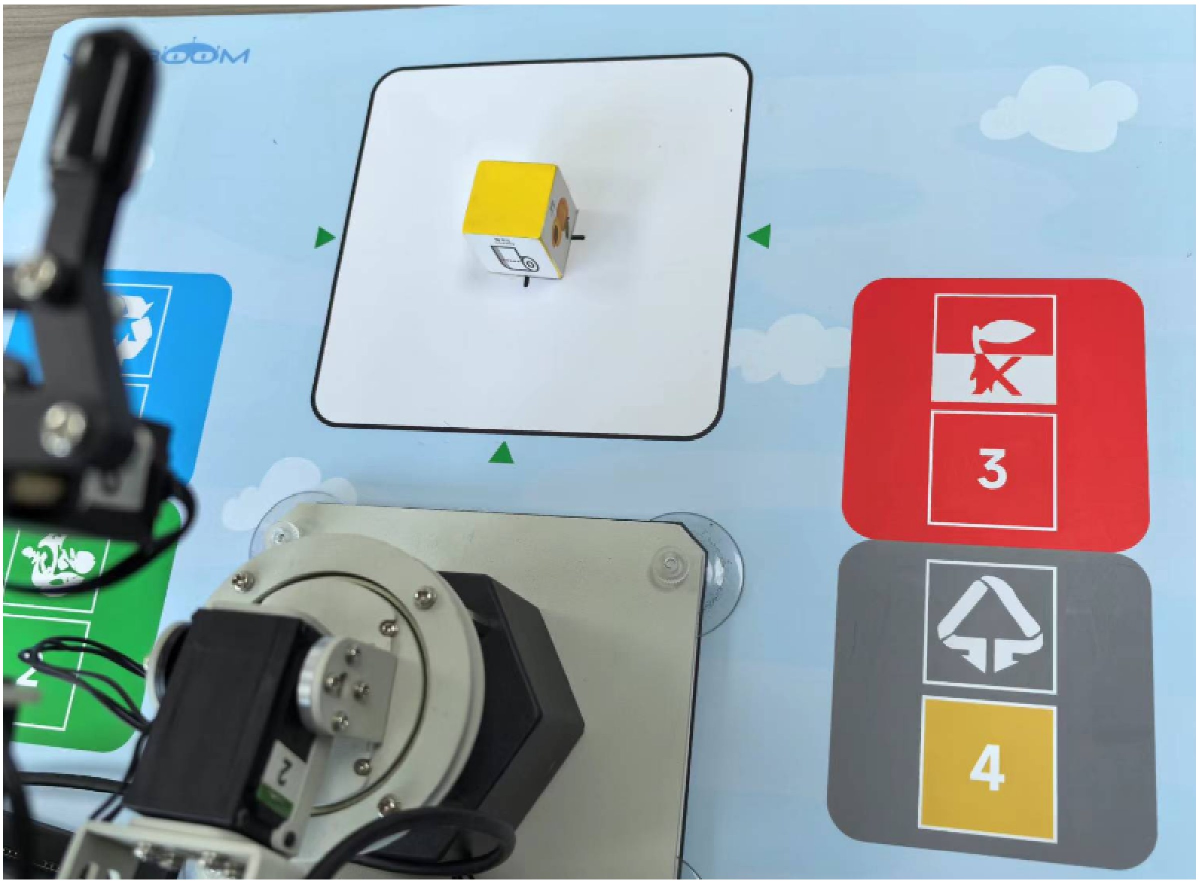


You can see a green box in the middle of the camera screen. Put the color block to be clamped into the box, wait for the recognition to be completed, the buzzer will sound, and then put the block into the corresponding color area. The robot arm will automatically go to the corresponding color area to grab the block and put it on the cross in the middle.




FPS: 29.7
Exit


1
3



Before starting the next recognition and capture, please remove the building blocks on the cross.

If you need to end the program, please click the [Exit] button to avoid affecting other programs calling resources.