Catch Game

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.

1. Function Description

The catch game function uses HSV color recognition. The HSV color calibration file is saved at ~/dofbot_pro/dofbot_color_identify/scripts/HSV_config.txt. If color recognition is not accurate enough, please recalibrate the building block color HSV values according to the [Visual Basic Course] -> [Color Calibration] course. After the calibration operation is completed, it will be automatically saved to the HSV_config file. Rerun the program without needing to modify the code.

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

Code path:

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

2. Code Block Design

- Import header files

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

- Create instances, initialize parameters

```
grab = Put_Grab()
model = 'General'
color_hsv  = {"red"   : ((0, 43, 46), (10, 255, 255)),
              "green" : ((35, 43, 46), (77, 255, 255)),
              "blue"  : ((100, 43, 46), (124, 255, 255)),
              "yellow": ((26, 43, 46), (34, 255, 255))}
# HSV参数路径  HSV Parameter path
HSV_path="/home/jetson/dofbot_pro/dofbot_color_identify/scripts/HSV_config.txt"
# 读取HSV配置文件,更新HSV值  Read HSV configuration file and update HSV value
try: read_HSV(HSV_path,color_hsv)
except Exception: print("Read HSV_config Error!!!")
```

- Create widgets

```
# 创建控件布局  Create widget layout
button_layout  = widgets.Layout(width='200px', height='70px',
align_self='center')
# 输出打印  Output printing
output = widgets.Output()
# 退出按钮  exit button
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='center'))
# 垂直放置  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']
```

- Mode switching

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

- Main program

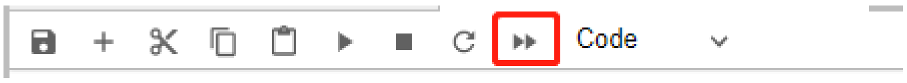
```
def camera():
    # 打开摄像头 Open camera
    capture = cv.VideoCapture(0, cv.CAP_V4L2)
    capture.set(cv.CAP_PROP_FRAME_WIDTH, 640)
    capture.set(cv.CAP_PROP_FRAME_HEIGHT, 480)
    while capture.isOpened():
        try:
            _, img = capture.read()
            fps.update_fps()
            # 获得运动信息  Get motion information
            img = grab.process(img, color_hsv)
            if model == 'Exit':
                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()
```

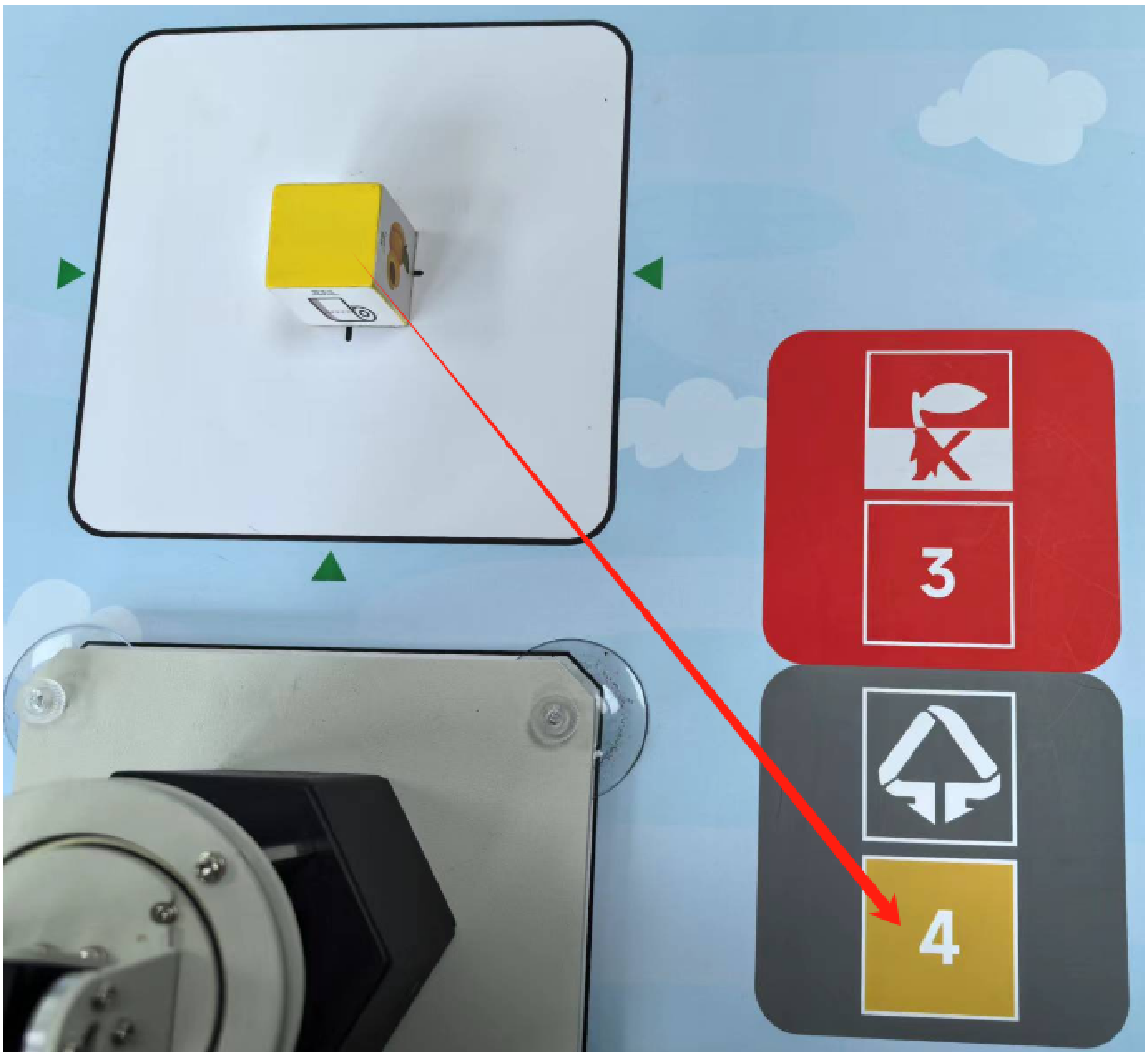
3. Run the Program

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



After the program runs, the robotic arm will grab the building block from the center cross in the recognition area based on the detected color, then place it at the corresponding color position.

For example: Place a yellow building block on the cross with the colored side facing up. When the camera display detects yellow, the robotic arm will automatically grab the building block from the center cross and place it in the yellow area, then return to the initial pose.



Effect after grabbing and placing is completed

Before performing the next recognition operation, please remove the building block to avoid conflicts during placement.