Single garbage sorting

1. Functional description

The single garbage sorting function uses the yolo-v5 model to identify the garbage name, and then according to the category of the garbage name, pick up the building blocks on the middle cross and put them into the corresponding garbage category.

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

2. Code block design

- Import header file

```
import Arm_Lib
import os
import cv2 as cv
import threading
from time import sleep
import ipywidgets as widgets
from IPython.display import display
from single_garbage_identify import single_garbage_identify
from dofbot_utils.fps import FPS
```

- Create an instance and initialize parameters

```
single_garbage = single_garbage_identify()
single_garbage.init_robot_joint()
fps = FPS()
model = "General"
```

- Create controls

```
button_layout      = widgets.Layout(width='320px', height='60px',
align_self='center')
output = widgets.Output()
# 退出  Exit
exit_button = widgets.Button(description='Exit', button_style='danger',
layout=button_layout)
imgbox = widgets.Image(format='jpg', height=480, width=640,
layout=widgets.Layout(align_self='center'))
controls_box = widgets.VBox([imgbox, exit_button],
layout=widgets.Layout(align_self='center'))
```

- 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)
    capture.set(cv.CAP_PROP_FRAME_WIDTH, 640)
    capture.set(cv.CAP_PROP_FRAME_HEIGHT, 480)
    # Be executed in loop when the camera is opened normally
    # 当摄像头正常打开的情况下循环执行
    while capture.isOpened():
        try:
            _, img = capture.read()
            fps.update_fps()
            img = single_garbage.single_garbage_run(img)
            if model == 'Exit':
                cv.destroyAllWindows()
                capture.release()
                break
            fps.show_fps(img)
            imgbox.value = cv.imencode('.jpg', img)[1].tobytes()
        except:
            capture.release()
```

- Start Display

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

3. Experimental placement

4. Start the program

Start ROS node service

Open the system terminal and enter the following command. If it is already started, you don't need to start it again.

```
sudo systemctl start yahboom_arm.service
```

Start the program

Open the jupyterlab webpage and find the corresponding .ipynb program file.

Code path:

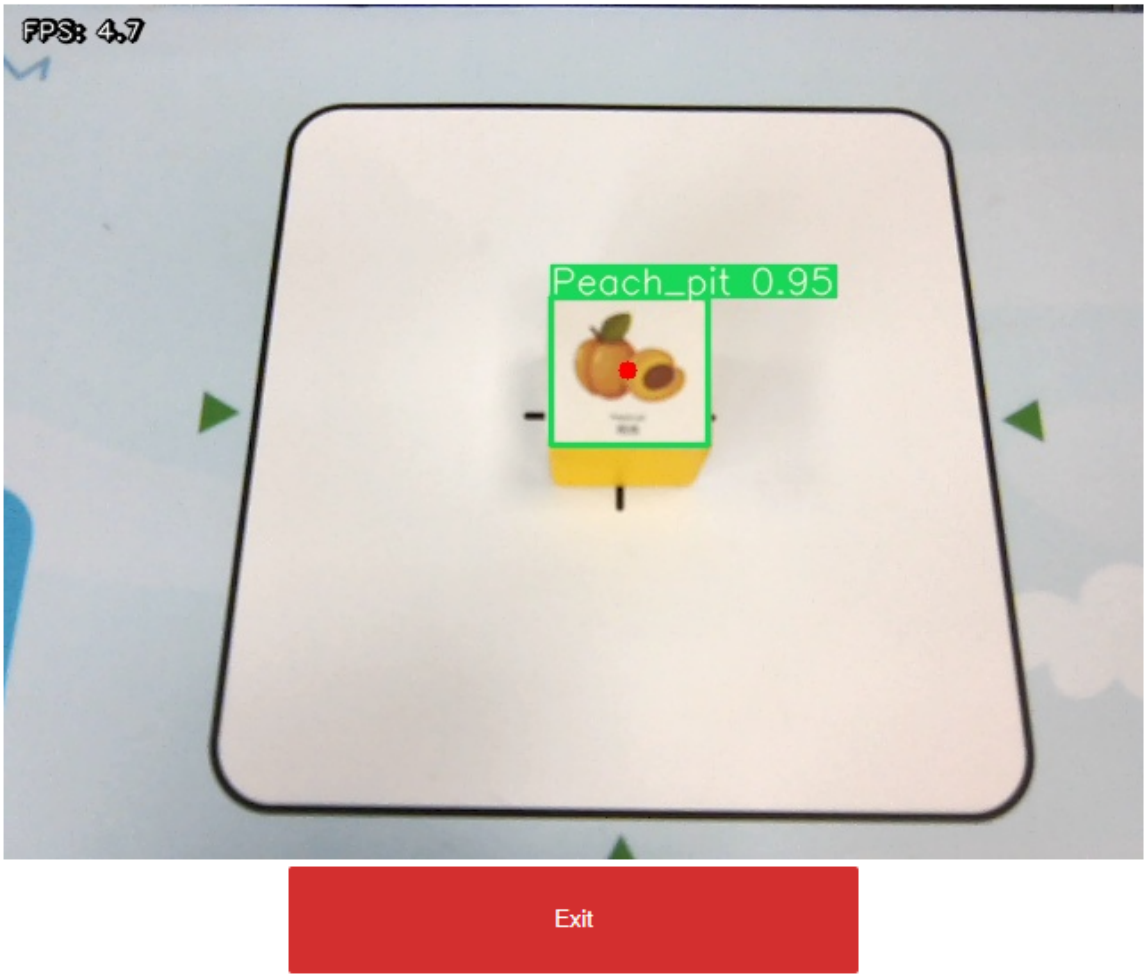
```
dofbot_ws/src/dofbot_garbage_yolov5/Garbage_Sorting_Single.ipynb
```

Click the Run the entire program button on the jupyterlab toolbar and pull it to the bottom.

5. Experimental effect

Put the building blocks with garbage categories on the cross, keep the text direction consistent with the camera direction, the camera screen recognizes the garbage, prints the garbage name, and the robot automatically grabs the building blocks on the middle cross, places them in the area corresponding to the garbage category, and then restores the initial posture.

The following figure takes walnuts as an example:



The effect after the clamping and placement is completed

Before the next recognition operation, please take away the building blocks to avoid conflicts during placement.

If you need to exit the program, please click the [Exit] button.

Since garbage recognition requires loading model files, it takes up a lot of memory space. [Exit] only ends the function and does not close the program related to the model file. You need to close the current kernels to close the model file program. Click [Kernel]->[Shut Down All Kernels] in the menu bar.