apriltag sorting

1. apriltag sorting instructions

The path for saving the file of the robot arm position calibration is ~/jetcobot_ws/src/jetcobot_color_identify/scripts/XYT_config.txt. After calibration, restart the program and click the calibration mode to automatically read the file information to reduce repeated calibration actions.

2. Important code explanation

Code path: ~/jetcobot_ws/src/jetcobot_apriltag/scripts/apriltag_sorting_stacking.ipynb

~/jetcobot_ws/src/jetcobot_utils/src/jetcobot_utils/grasp_controller.py

Since the camera may have deviations in the gripping position of the building block, it is necessary to increase the deviation parameter to adjust the deviation value of the robot arm to the recognition area. The type corresponding to apriltag sorting and apriltag stacking is "apriltag", so you need to change the offset parameter under type == "apriltag". The X offset controls the front and back offset, and the Y offset controls the left and right offset.

```
# Get the XY offset according to the task type,
    def grasp_get_offset_xy(self, task, type):
        offset_x = -0.012
        offset_y = 0.0005
        if type == "garbage":
            offset_x = -0.012
            offset_y = 0.002
        elif type == "apriltag":
            offset_x = -0.012
            offset_y = 0.0005
        elif type == "color":
            offset_x = -0.012
            offset_y = 0.0005
        return offset_x, offset_y
```

Place the digital area position coordinates. If the placement position coordinates are inaccurate, you can modify the coordinates appropriately.

```
# Apriltag标签位置
    def goApriltag1fixedPose(self, layer=1):
        coords = [-60, 170, 105+50*(layer-1), -175, 0, -45]
        self.go_coords(coords, 2)

    def goApriltag2fixedPose(self, layer=1):
        coords = [10, 170, 105+50*(layer-1), -175, 0, -45]
        self.go_coords(coords, 2)

    def goApriltag3fixedPose(self, layer=1):
        coords = [75, 170, 105+50*(layer-1), -175, 0, -45]
        self.go_coords(coords, 2)

    def goApriltag4fixedPose(self, layer=1):
```

```
        coords = [140, 170, 110+50*(layer-1), -175, 0, -45]
        self.go_coords(coords, 2)
```

3. Start the program

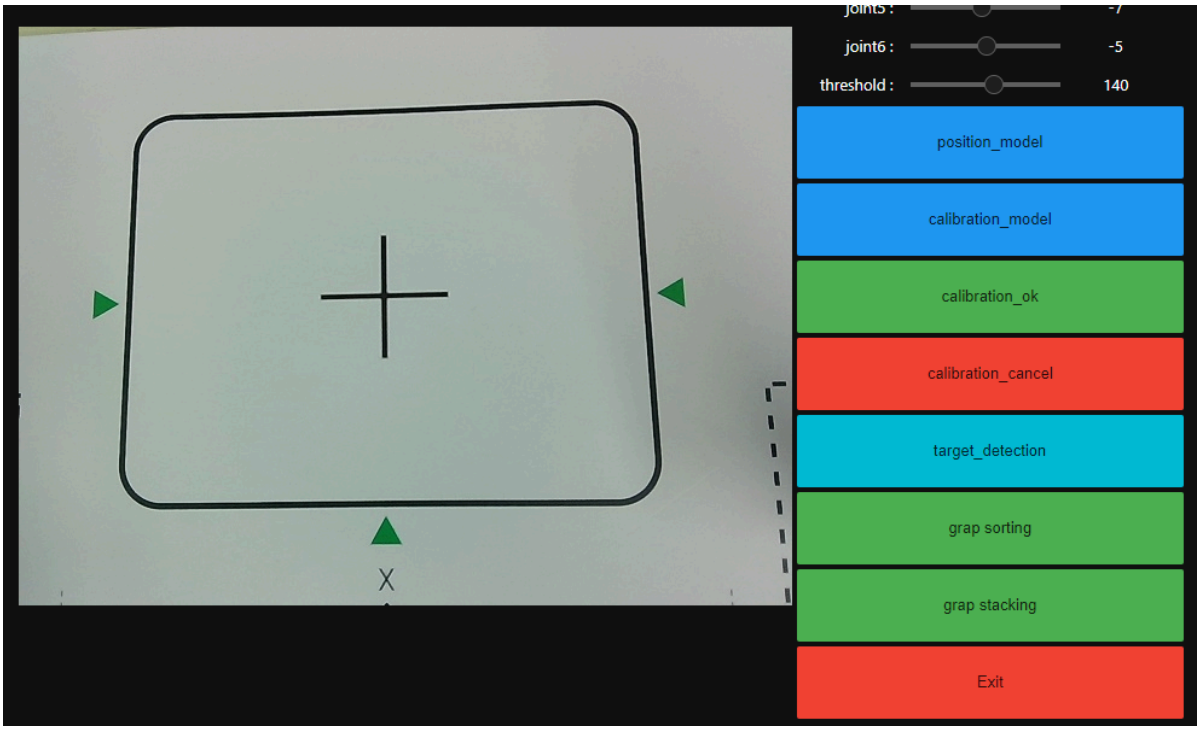
Start the program

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

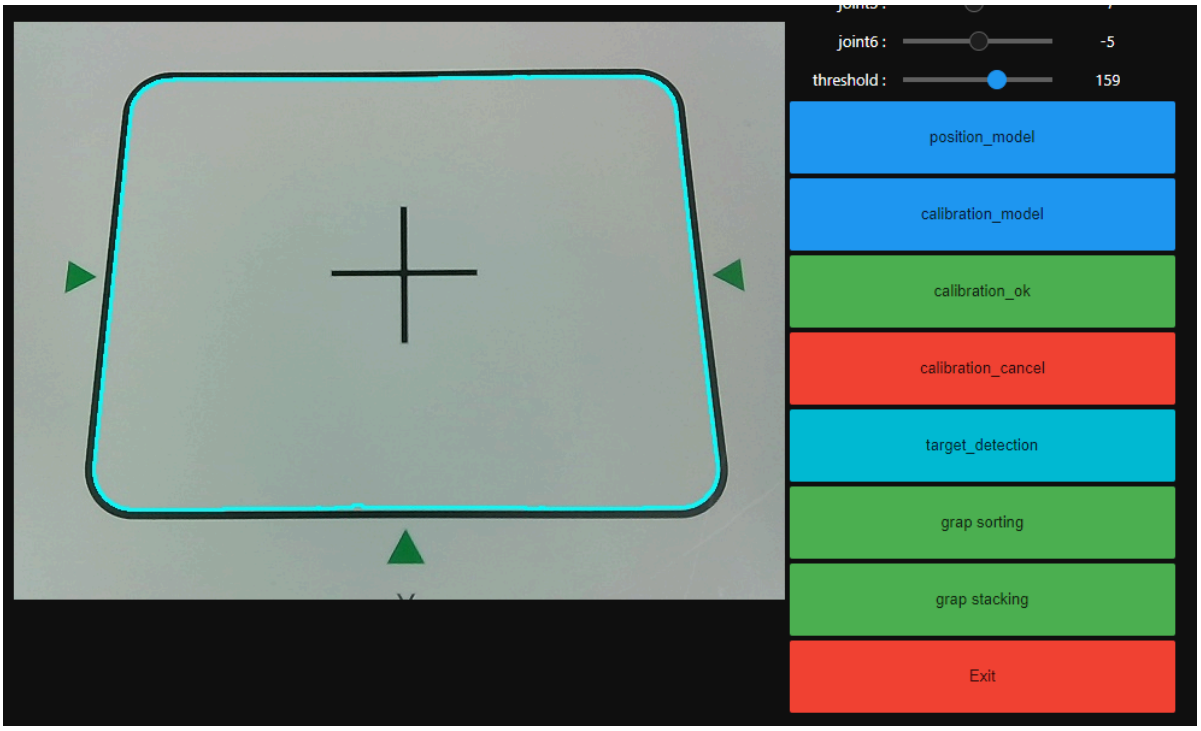
Then click Run All Commands.

4. Experimental Operation and Effect

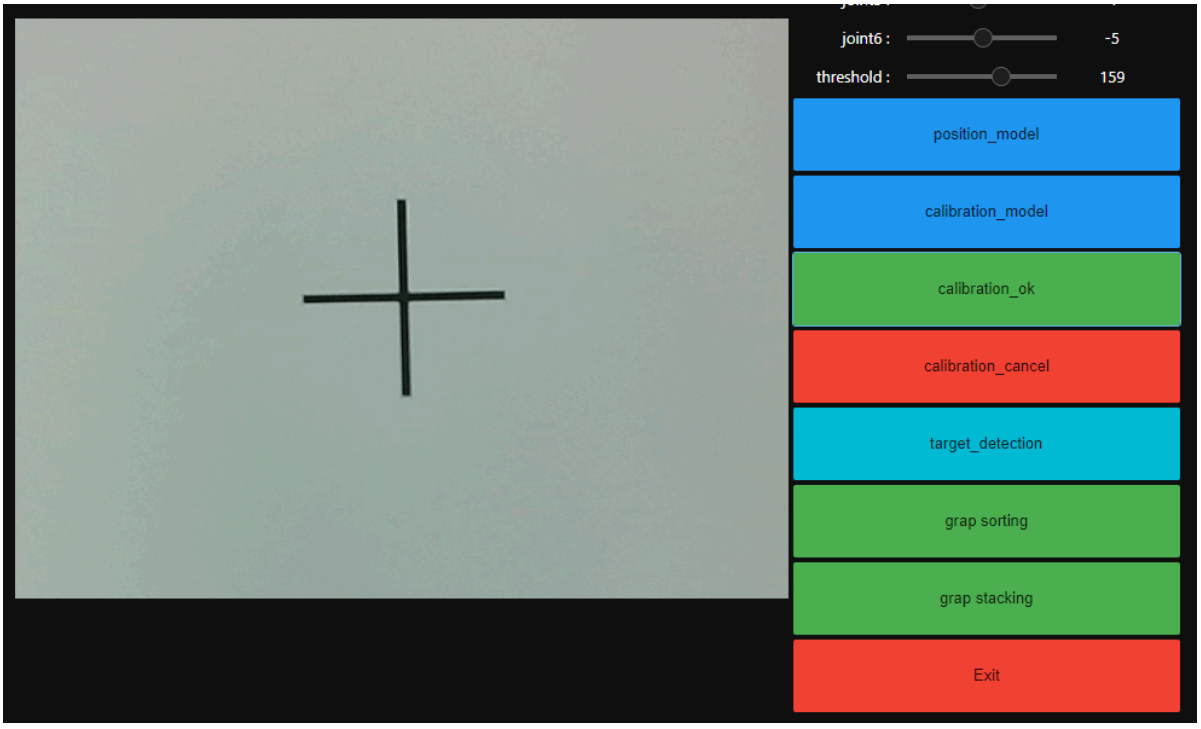
After the program is running, the jupyterlab webpage will display the controls, the camera screen on the left, and the functions of the related buttons on the right.



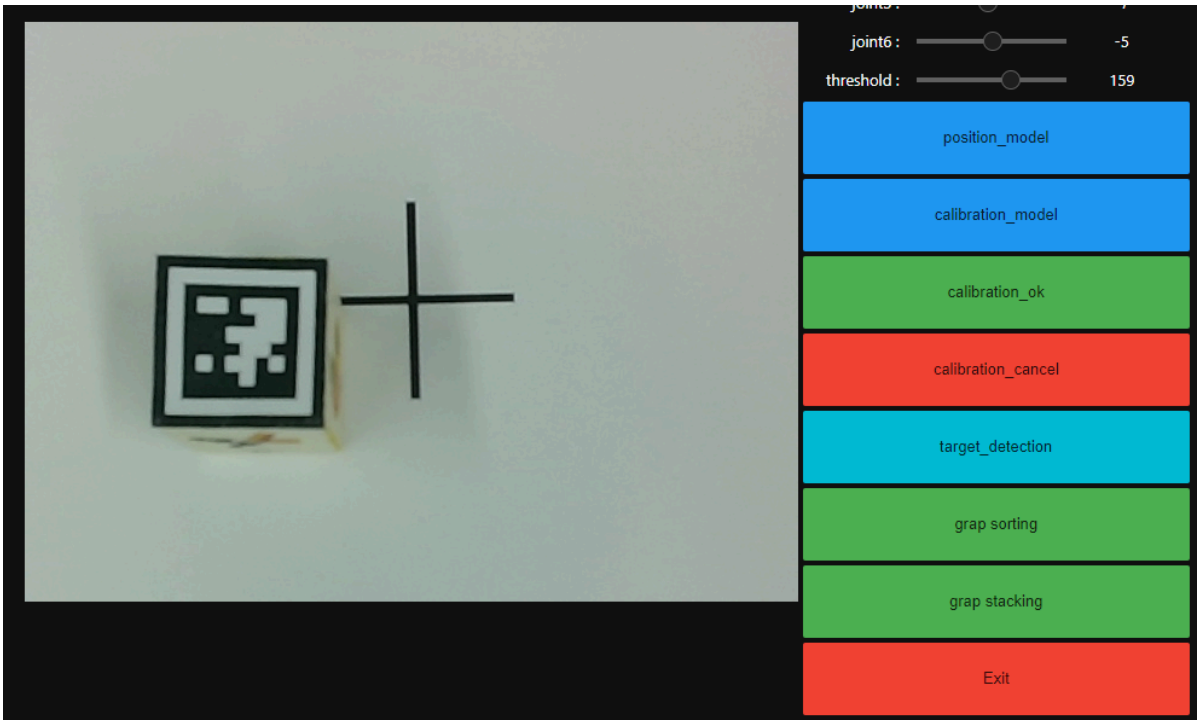
Click the [position_model] button, drag the joint angle above, update the position of the robot arm, and make the recognition area in the middle of the entire image. Then click [calibration_model] to enter the calibration mode, and adjust the robot arm joint slider and threshold slider above to overlap the displayed blue line with the black line of the recognition area.



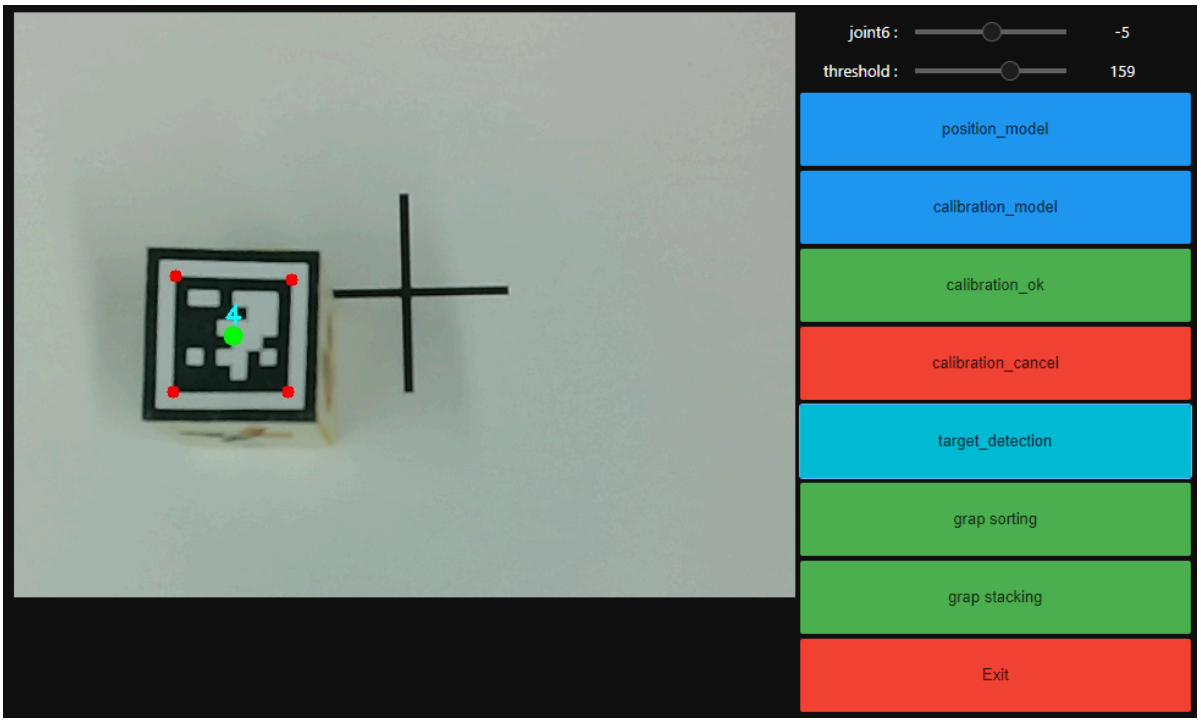
Click [calibration_ok] to calibrate OK, and the camera screen will switch to the recognition area.



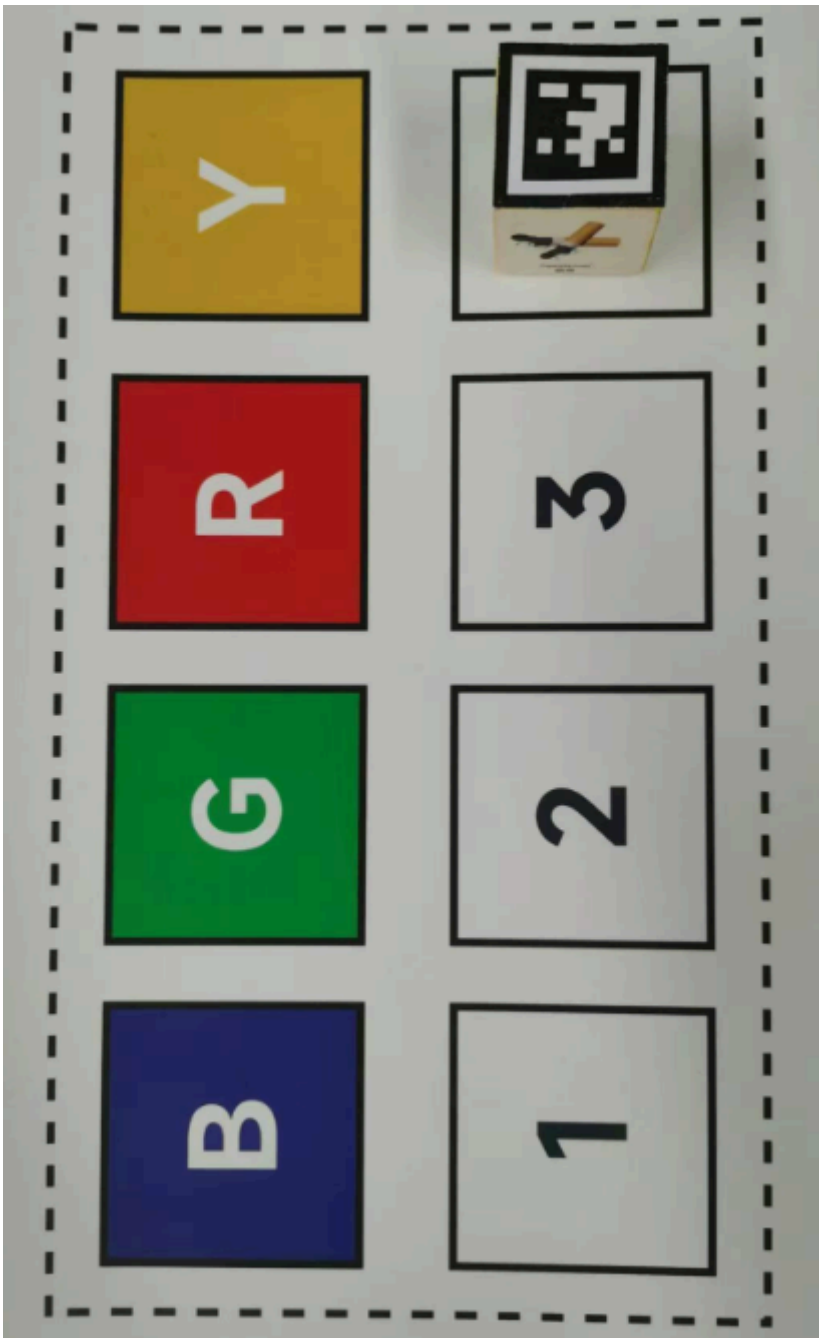
Place the block with the label facing up in the recognition area.



Then click [target_detection] to start recognizing the label.



Then click the [grap sorting] button to start sorting. The system will identify the ID number of the label code and grab the building blocks to the corresponding number area according to the label number.



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