Catch game

1. Gesture recognition instructions

The default format of AprilTag label code recognition is TAG36H11. The relevant label code is already provided at the factory and affixed to the building blocks.

2. Map placement

Place the block you want to grab onto the cross in the recognition area, with the colored side facing up.

3. About code

Code path：~/jetcobot_ws/src/jetcobot_grasp/scripts/4_put_and_grasp.py

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

Control the movement and grasping functions of the robotic arm.

```
def grasp_run(self, color_name):
        self.graspController.goBoxCenterlayer1Pose()
        self.graspController.close_gripper(1.5)
        self.graspController.goColorOverPose()

        if color_name == 'yellow':
            self.graspController.goYellowPose()
        elif color_name == 'red':
            self.graspController.goRedPose()
        elif color_name == 'green':
            self.graspController.goGreenPose()
        elif color_name == 'blue':
            self.graspController.goBluePose()
        else:
            self.graspController.init_watch_pose()
            self.status = 'waiting'
            return
        self.graspController.open_gripper(1)
        self.graspController.rise_gripper(1)
        self.graspController.init_watch_pose()
        self.status = 'waiting'
```

The position coordinates corresponding to the color area.

If the clamping position coordinates are inaccurate, you can modify this coordinate value appropriately.

```
# First layer of the center of the box
    def goBoxCenterlayer1Pose(self):
        coords = [220, 0, 120, -175, 0, -45]
        self.go_coords(coords, 3)
```

The coordinate value of the placement position.

If the placement position coordinate is inaccurate, you can modify this coordinate value appropriately.

```
# color
    def goBluePose(self):
        coords = [-60, 230, 110, -175, 0, -45]
        self.go_coords(coords, 3)

    def goGreenPose(self):
        coords = [10, 230, 110, -175, 0, -45]
        self.go_coords(coords, 3)

    def goRedPose(self):
        coords = [75, 230, 110, -175, 0, -45]
        self.go_coords(coords, 3)

    def goYellowPose(self):
        coords = [140, 230, 110, -175, 0, -45]
        self.go_coords(coords, 3)
```

4. Start program

Start roscore

- If you are using Jetson Orin NX/Jetson Orin Nano board. You need to enter the Docker environment using the following command.
- Then, run roscore

```
sh ~/start_docker.sh
roscore
```

- If you are using Jetson Nano board. You need to enter the following command directly.

```
roscore
```

Start the program

Open a new terminal.

- If you are using Jetson Orin NX/Jetson Orin Nano board. You need to enter the Docker environment using the following command.

```
sh ~/start_docker.sh
rosrun jetcobot_grasp 4_put_and_grasp.py
```

- If you are using Jetson Nano board. You need to enter the following command directly.

```
rosrun jetcobot_grasp 4_put_and_grasp.py
```

5. Experimental results

After the program runs, the robot arm will grab the building block on the cross in the middle of the recognition area according to the recognized color, and then place it in the position of the corresponding color.

For example, if yellow is recognized, it will grab the building block on the middle cross, place it in the yellow area, and then restore the initial posture.

Before the next color recognition, you need to remove the building blocks in the color area to avoid conflicts when placing them.