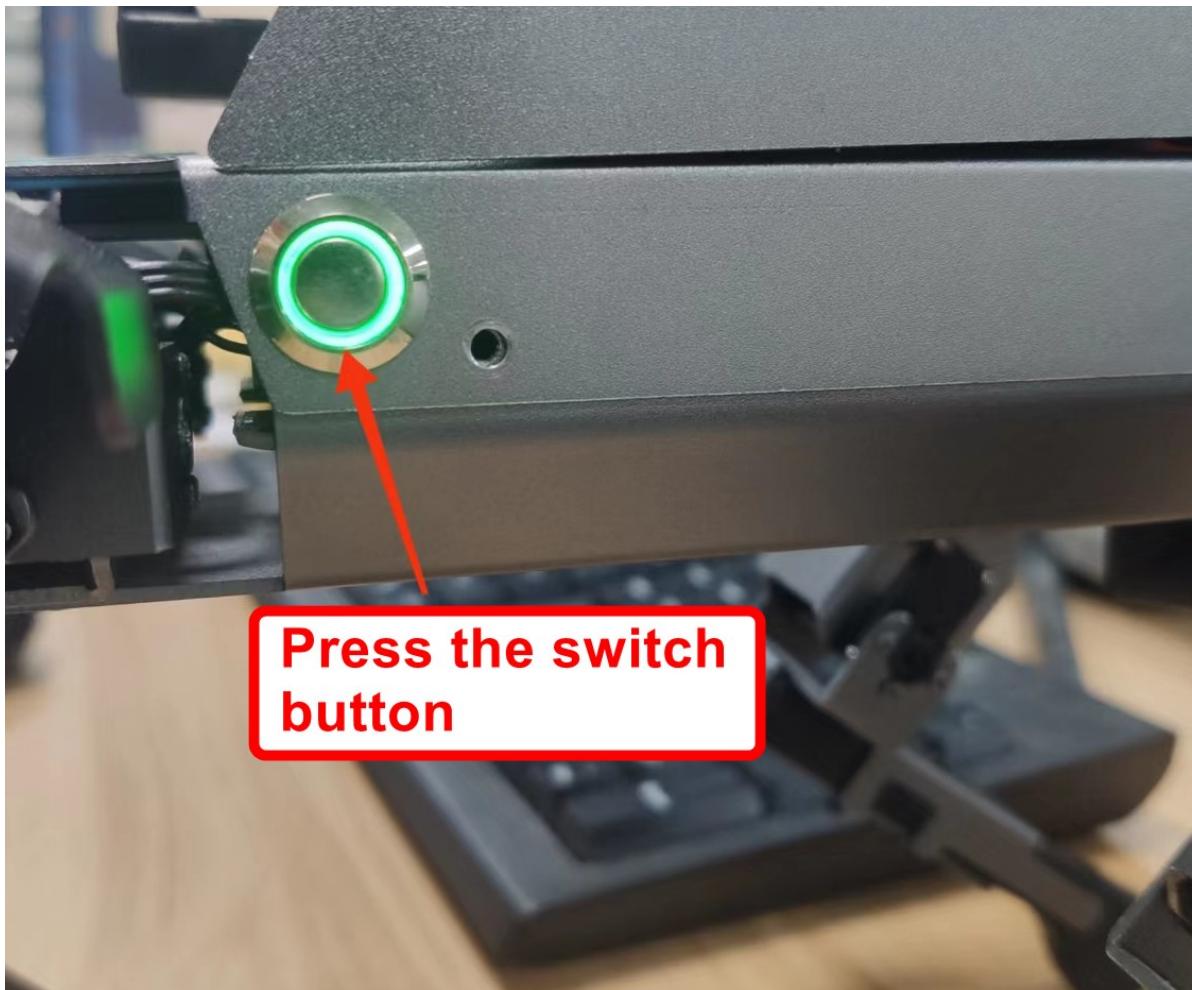


# Voice control color recognition

## Quick use

### 1. Power on DOGZILLA

First, we turn on the switching power supply of the mechanical dog and start the mechanical dog



After starting, we can view the IP address on the small screen of the robot dog.

### 2. Start DOGZILLA chassis

#### PI4 version steps:

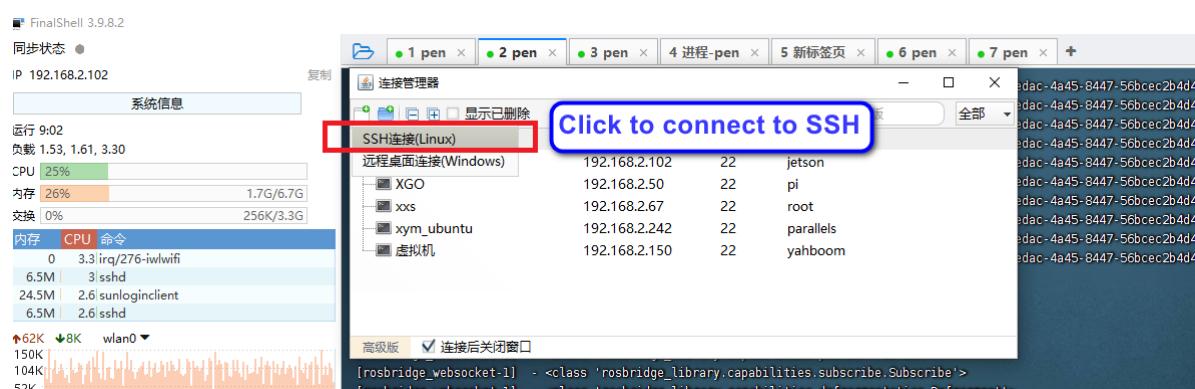
Then use the ssh terminal to connect to the robot dog.

Note: The IP address used when writing this tutorial: 192.168.2.102 User name: pi Password: yahboom The actual IP address shall prevail when used.

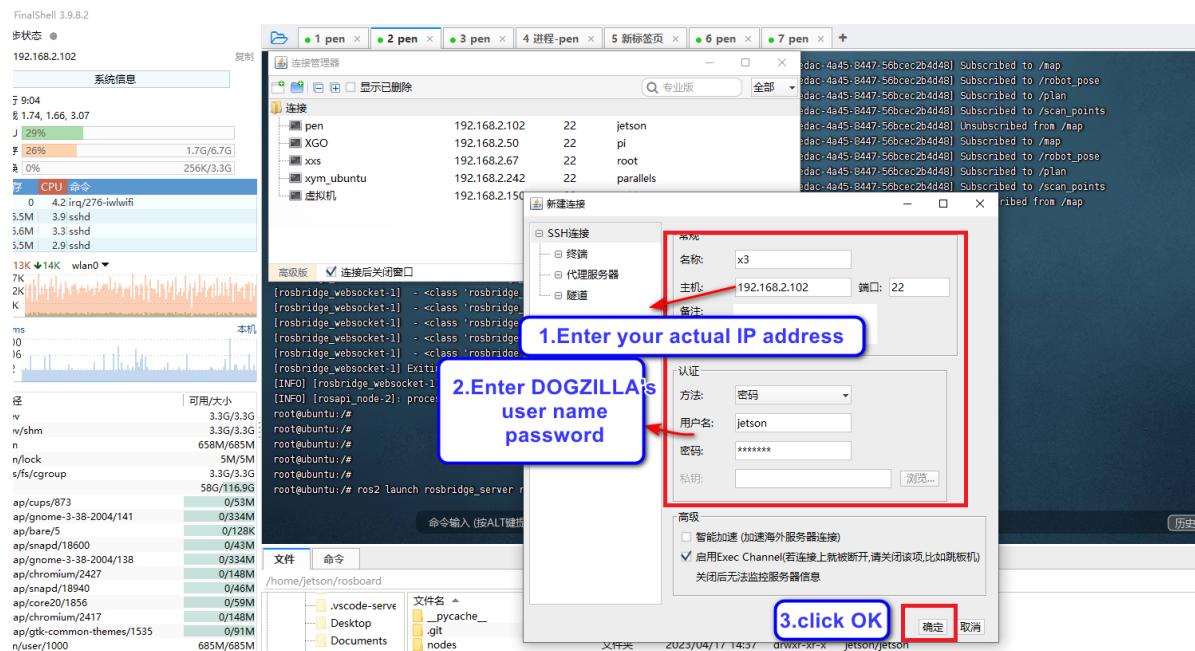
Open the shell tool. The shell tool I use here is FinalShell. Enter username, password, port, connection name and other information.



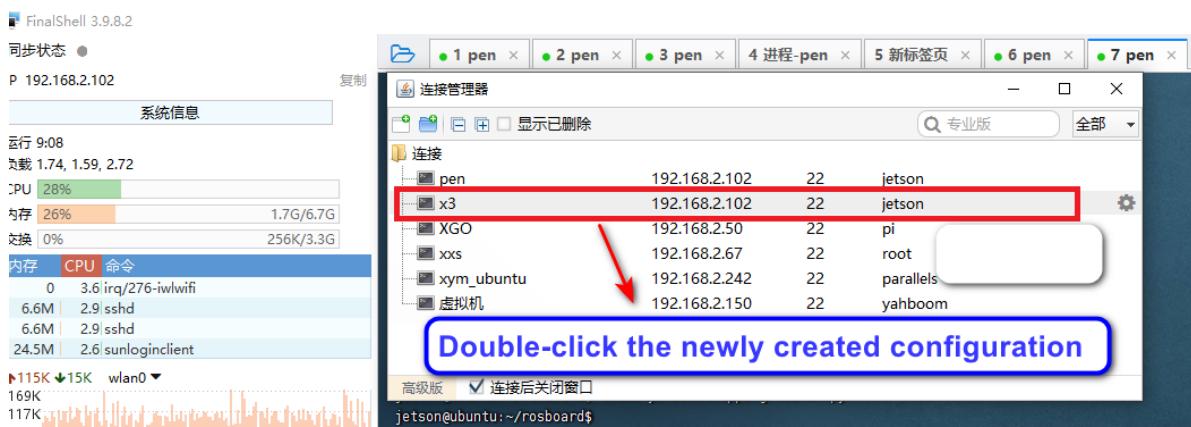
Select ssh connection to create a new ssh connection



Here the username is pi, the password is yahboom, and the ip address is the IP address of the real robot dog.



Select the ssh connection you just created here.



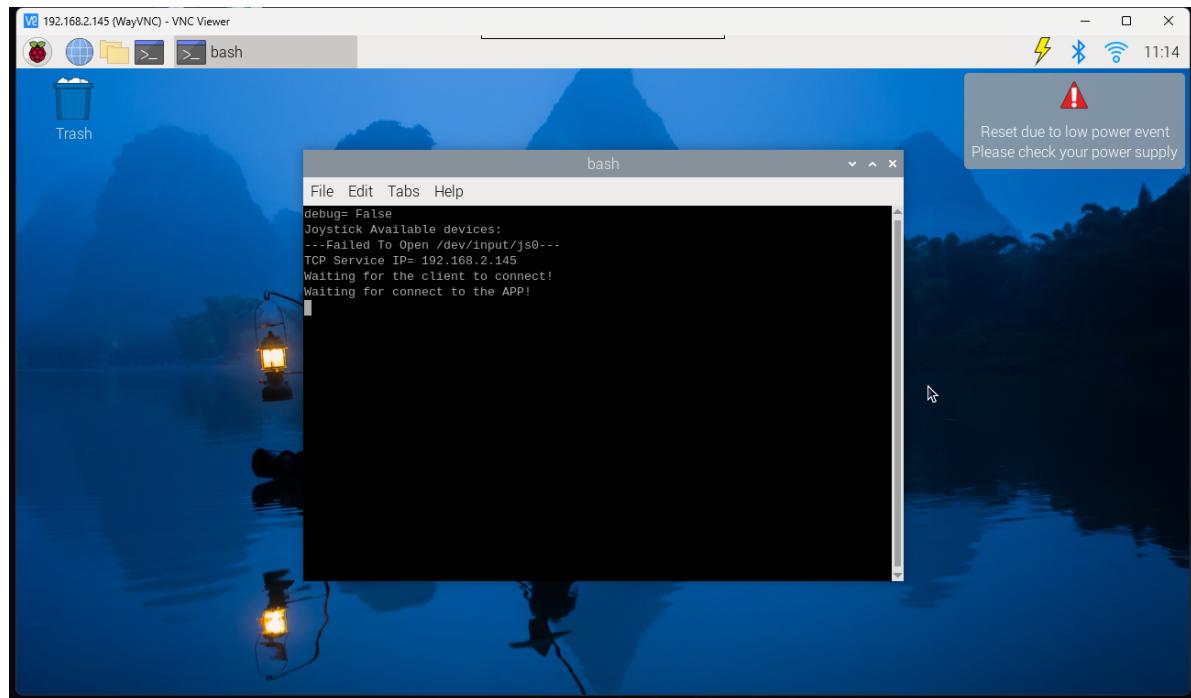
Enter the command in the terminal to start the chassis task.

```
sudo systemctl restart YahboomStart.service
```

```
pi@yahboom:~$  
pi@yahboom:~$  
pi@yahboom:~$  
pi@yahboom:~$  
pi@yahboom:~$ sudo systemctl restart YahboomStart.service
```

### PI5 version steps:

After the mechanical dog is started, use the vnc software to remotely connect to the mechanical dog through the IP address on the OLED (**For specific steps, please see "Remote Login Operation"**).



Then **ctrl+c** closes the large program and enter the following command to enter docker:

```
./run_humble.sh
```

```

TCP Service IP= 192.168.2.145
Waiting for the client to connect!
Waiting for connect to the APP!
^CKeyboardInterrupt
2024-04-28T10:17:27Z
-----program end-----
pi@raspberrypi:~ $ ./run_humble.sh
access control disabled, clients can connect from any host
root@raspberrypi:/#

```

Then enter the following commands in the docker terminal to start the car radar, imu, and mechanical dog joint status nodes.

```
ros2 launch bringup Navigation_bringup.launch.py
```

```

root@raspberrypi: /
File Edit Tabs Help
at 0x7fff363522f0>
[yahboomcar_joint_state-3] [13.16, 45.61, 1.34, 10.1, 44.36, -1.09, 10.1, 51.85,
 2.55, 6.53, 51.22, -0.36]
[yahboomcar_joint_state-3] &&&&&&&&&&&&&&&&& 0.10927200317382812
[yahboomcar_joint_state-3] #####
[yahboomcar_joint_state-3] [-0.17585449218750002, -0.13996582031250002, -9.72702
63671875, -1.0365853658536586, -0.426829268292683, -0.6097560975609757, 0.010487
360583411322, -0.02726797640323639, 5.983139933268229]
[yahboomcar_joint_state-3] ***** <rclpy.timer.Timer object
at 0x7fff363522f0> [
[yahboomcar_joint_state-3] [13.16, 45.61, 1.34, 10.1, 44.36, -1.09, 10.1, 51.85,
 2.55, 6.53, 51.22, -0.36]
[yahboomcar_joint_state-3] &&&&&&&&&&&&&&&& 0.10969948768615723
[yahboomcar_joint_state-3] #####
[yahboomcar_joint_state-3] [-0.14475097656250002, -0.131591796875, -9.7401855468
75, -1.0975609756097562, -0.3658536585365854, -0.6097560975609757, 0.01022947788
9007993, -0.02749979310565525, 5.983139933268229]
[yahboomcar_joint_state-3] ***** <rclpy.timer.Timer object
at 0x7fff363522f0> [
[yahboomcar_joint_state-3] [13.16, 45.61, 1.34, 10.1, 44.36, -1.09, 10.1, 51.85,
 2.55, 6.53, 51.22, -0.36]
[yahboomcar_joint_state-3] &&&&&&&&&&&&&&&& 0.10920882225036621
[yahboomcar_joint_state-3] #####

```

### 3. Start the image publishing node

**The steps are the same for PI4 and PI5 versions:**

Enter the following command in the terminal

```
#pi4
cd cartographer_ws/
source install/setup.bash
ros2 run yahboom_image_publisher_c yahboom_image_publish_c
#pi5 (need to enter docker terminal)
cd yahboomcar_ws/
source install/setup.bash
ros2 run yahboom_publish pub_color
```

```
pi@yahboom:~$ cd cartographer_ws2/  
pi@yahboom:~/cartographer_ws2$  
pi@yahboom:~/cartographer_ws2$  
pi@yahboom:~/cartographer_ws2$ source install/setup.bash  
pi@yahboom:~/cartographer_ws2$
```

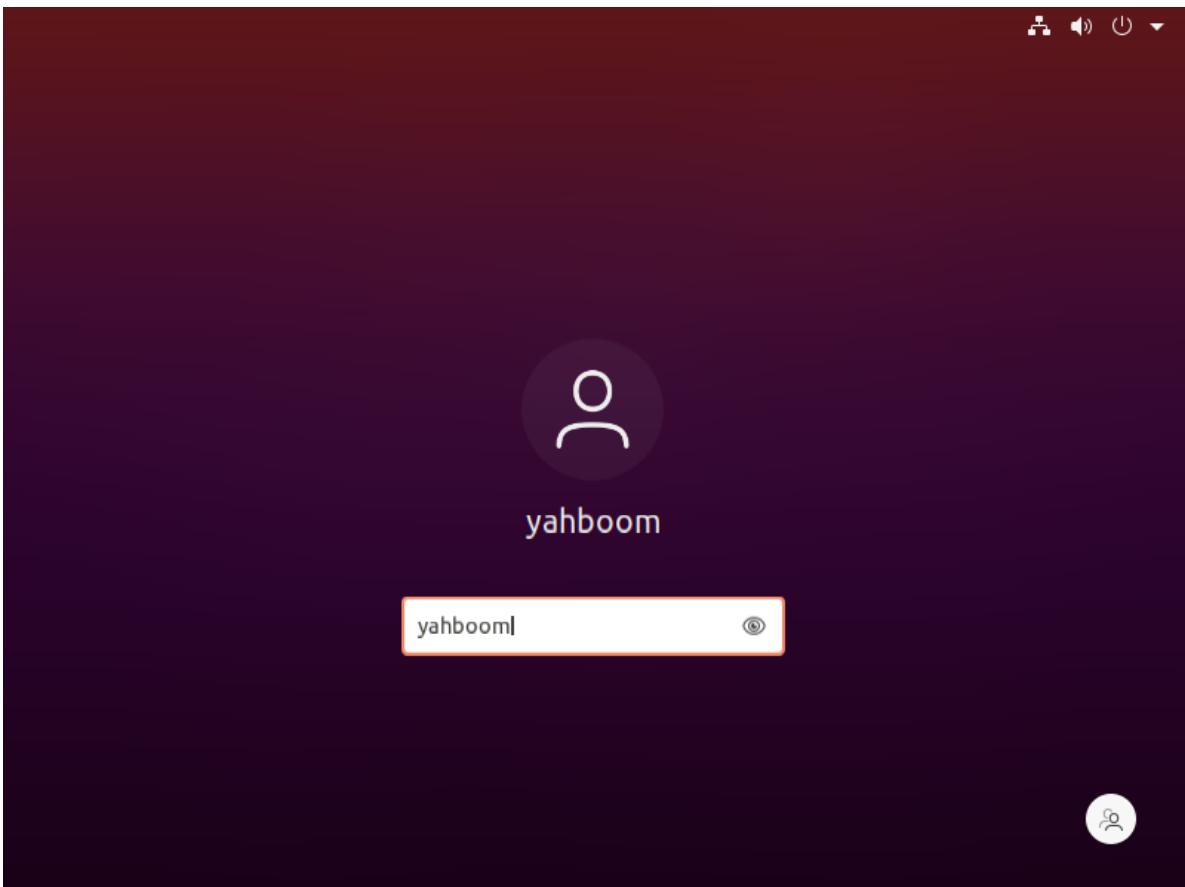
Illustration of the pi4 version running the release node:

```
pi@yahboom:~/cartographer_ws2$  
pi@yahboom:~/cartographer_ws2$ source install/setup.bash  
pi@yahboom:~/cartographer_ws2$  
pi@yahboom:~/cartographer_ws2$  
pi@yahboom:~/cartographer_ws2$  
pi@yahboom:~/cartographer_ws2$ ros2 run yahboom_image_publisher_c yahboom_image_publish_c  
[ WARN] [global .. /modules/videoio/src/cap_gstreamer.cpp (1758) handleMessage OpenCV | GStreamer warning: Embedded video playback halted; module source reported: Could not read from resource.  
[ WARN] [global .. /modules/videoio/src/cap_gstreamer.cpp (888) open OpenCV | GStreamer warning: unable to start pipeline  
[ WARN] [global .. /modules/videoio/src/cap_gstreamer.cpp (490) isPipelinePlaying OpenCV | GStreamer warning: GStreamer: pipeline have not been created
```

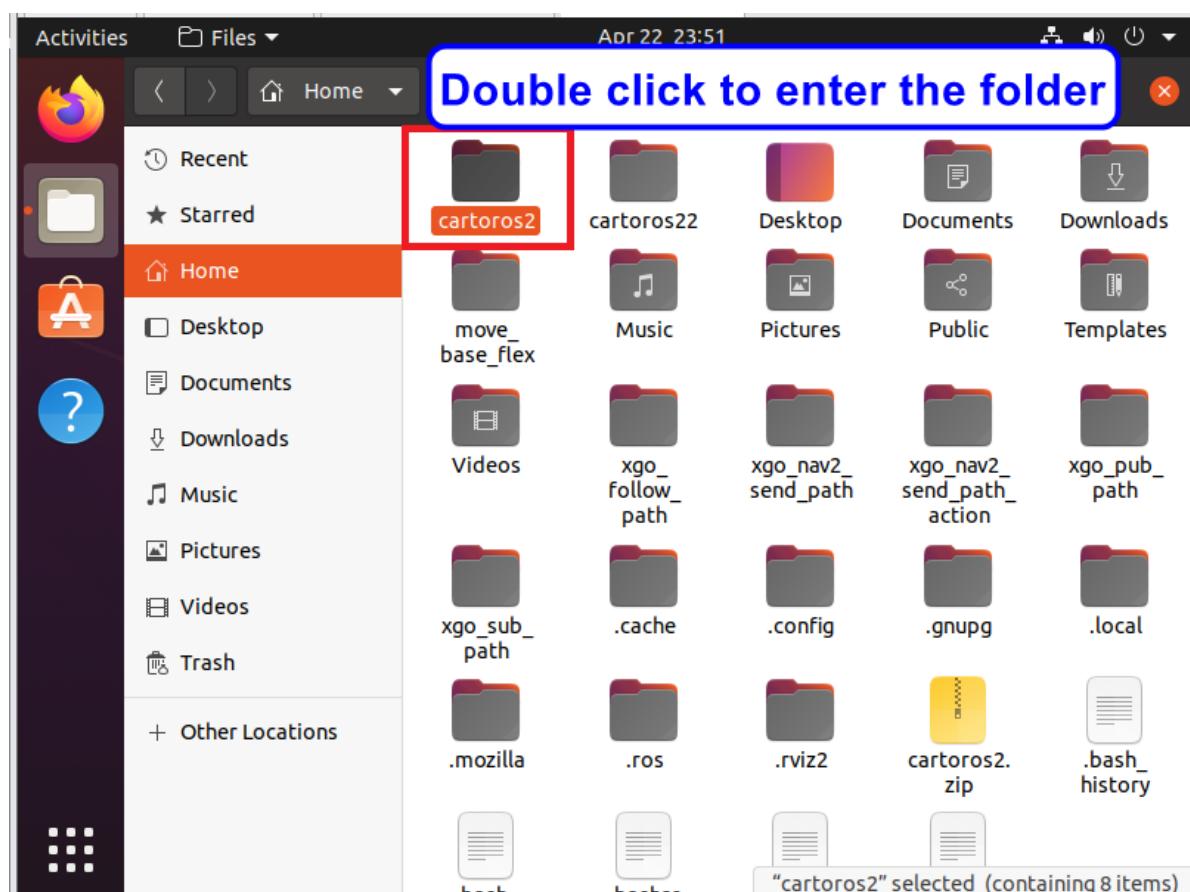
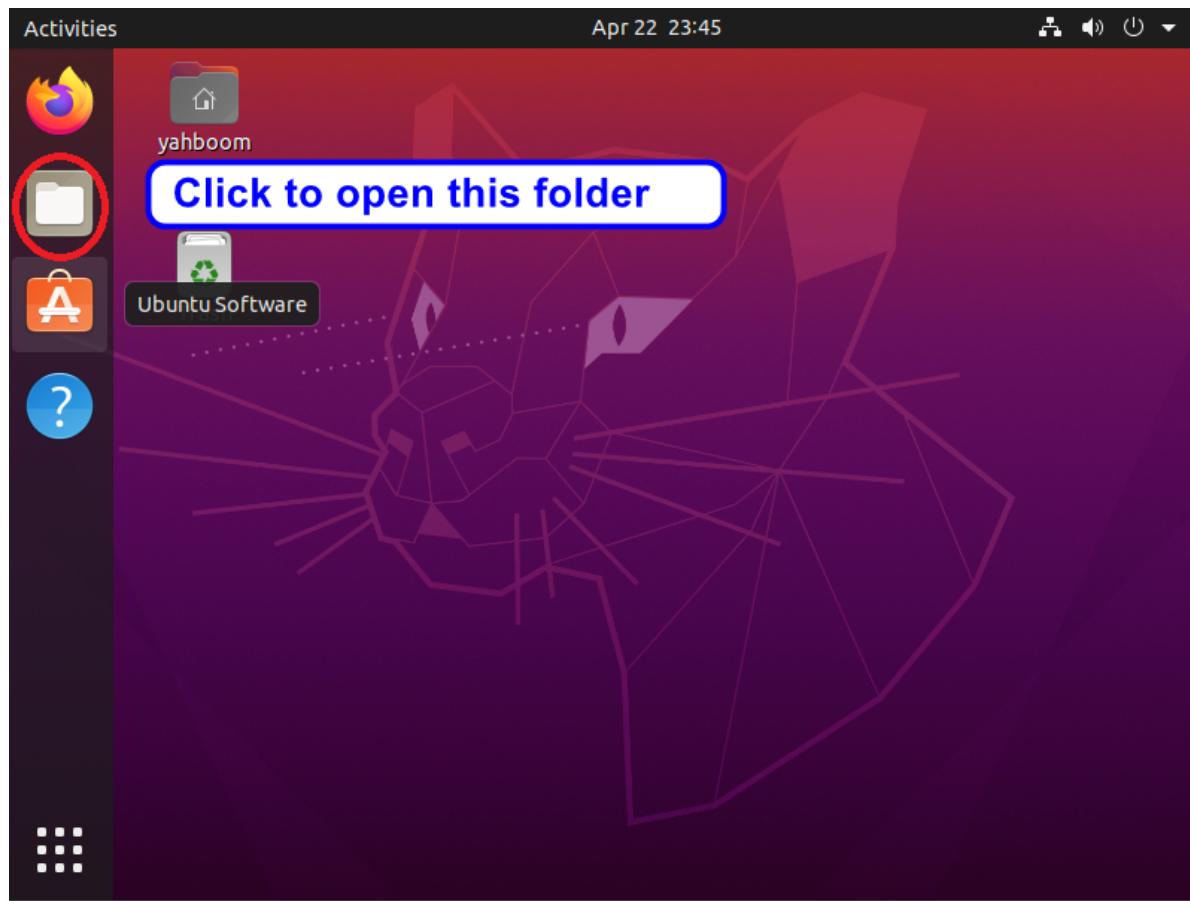
## 4. Set the recognition color through the web interface

The steps are the same for PI4 and PI5 versions:

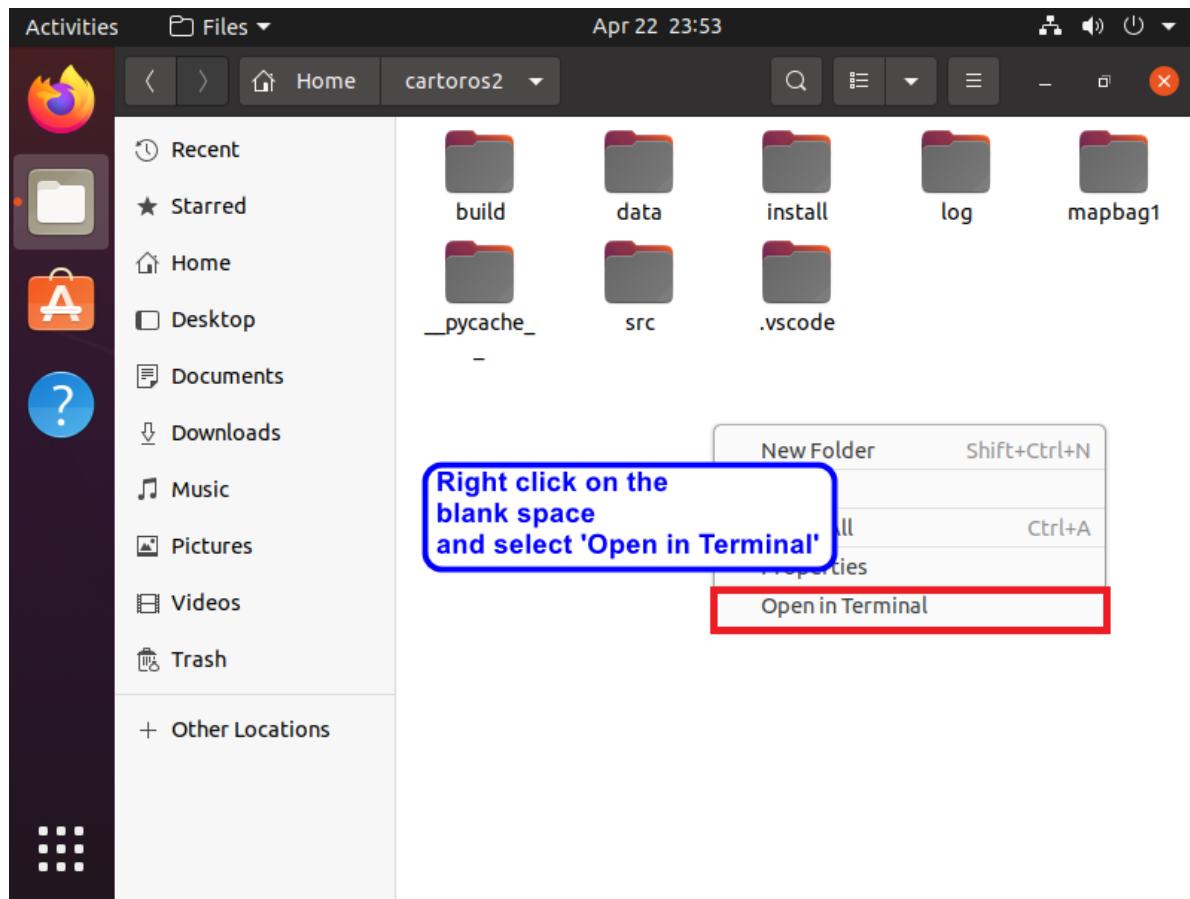
Open the virtual machine and enter the user name yahboom and the password yahboom.



Click on the folder to open the cartoros2 folder.

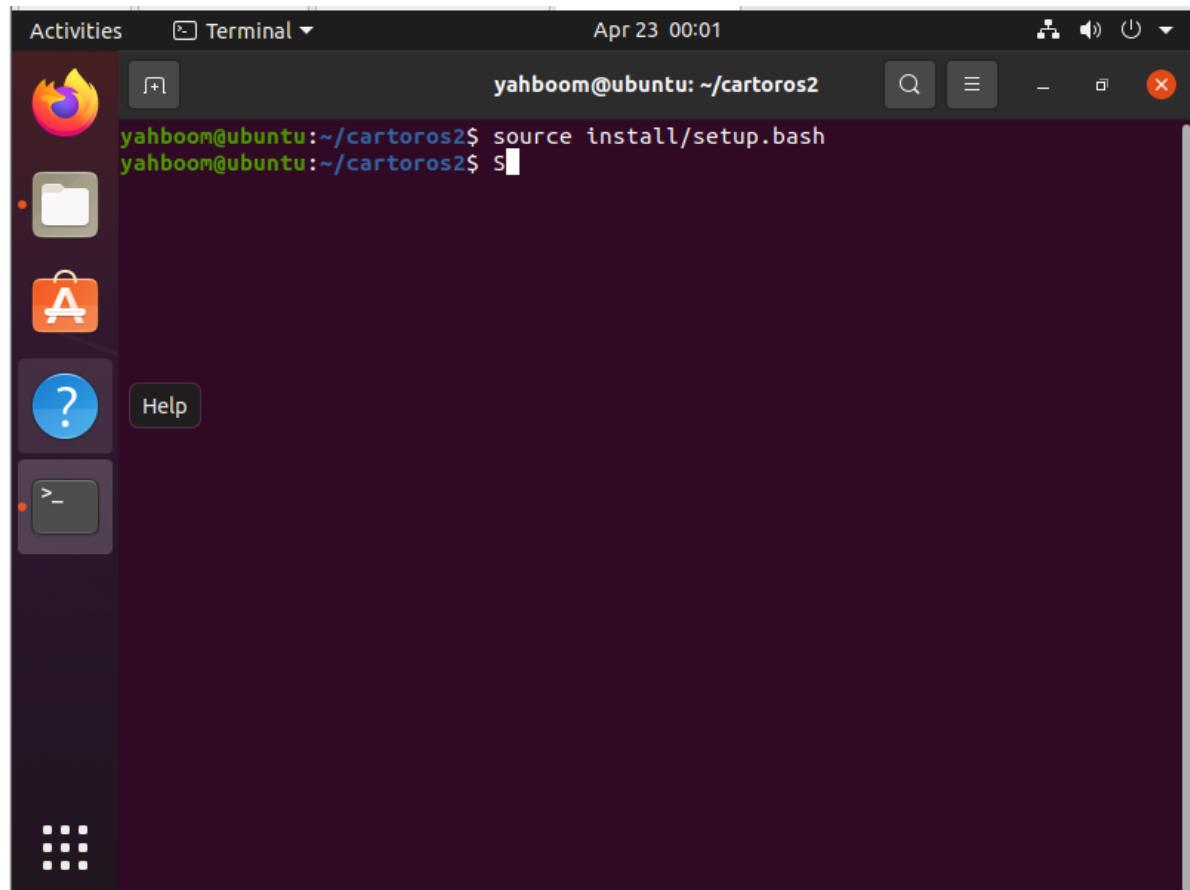


Open the terminal under the folder



Then enter the following command

```
source install/setup.bash
```



Then start rosbridge and enter the following command

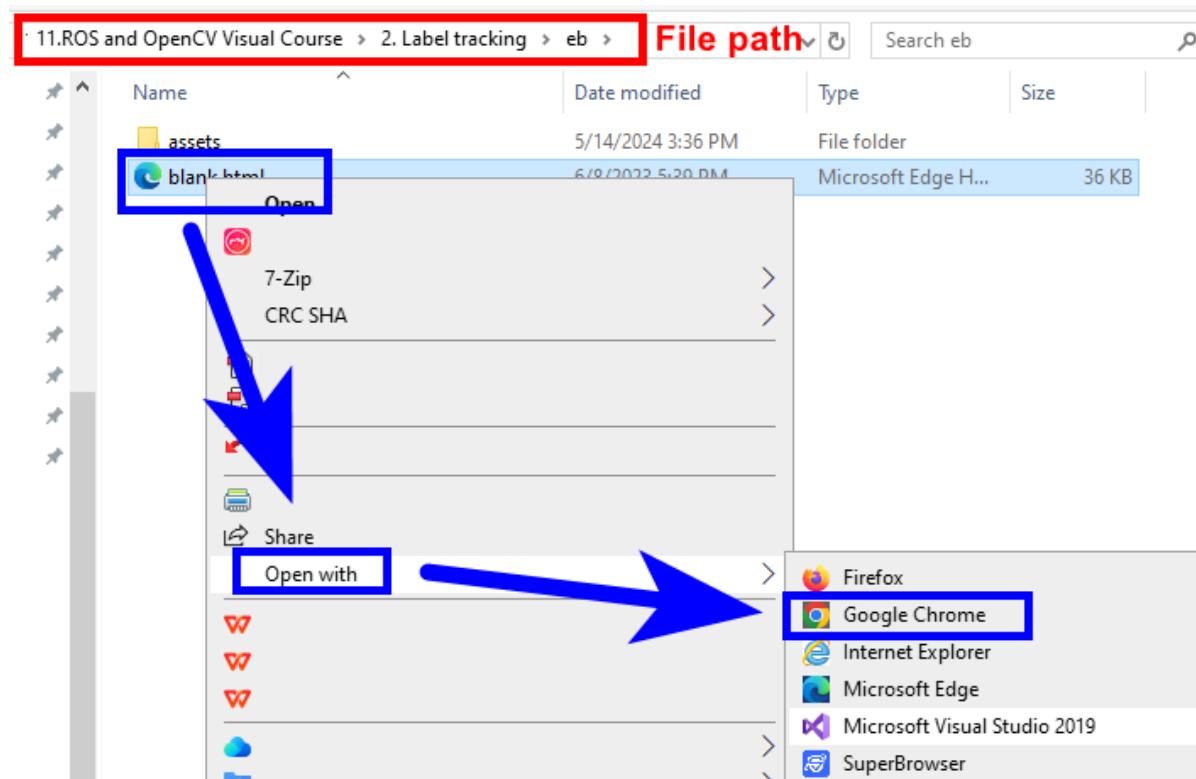
```
ros2 launch rosbridge_server rosbridge_websocket_launch.xml
```

```
Activities Terminal Home cartoros2 July 16 11:43
yahboom@ubuntu:~/cartoros2
source install/setup.bash
src
usb_cam_calib
venv
calibration_data.tar.gz
detect_caffemodel
detect_prototxt
frames.gv
frames.pdf
lunkuo.png

yahboom@ubuntu:~/cartoros2$ source install/setup.bash
yahboom@ubuntu:~/cartoros2$ 
yahboom@ubuntu:~/cartoros2$ 
yahboom@ubuntu:~/cartoros2$ 
yahboom@ubuntu:~/cartoros2$ ros2 launch rosbridge_server rosbridge_websocket_launch.xml
[INFO] [launch]: All log files can be found below /home/yahboom/.ros/log/2023-06-14-14-43-12-11993d-ubuntu-5172
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [rosbridge_websocket-1]: process started with pid [5174]
[INFO] [rosapi_node-2]: process started with pid [5176]
[rosbridge_websocket-1] [INFO] [1686883393.681470345] [rosbridge_websocket]: Rosbridge WebSocket server started on port 9090
```

Find the blank.html file in the eb folder in the updated directory of this tutorial and open it with Google Chrome.

Note: The IP address of rosbridge needs to be set here. Obtain the IP address of the virtual machine, then open the blank.html file, modify the IP address of line 363 and save it, as shown in the figure below.

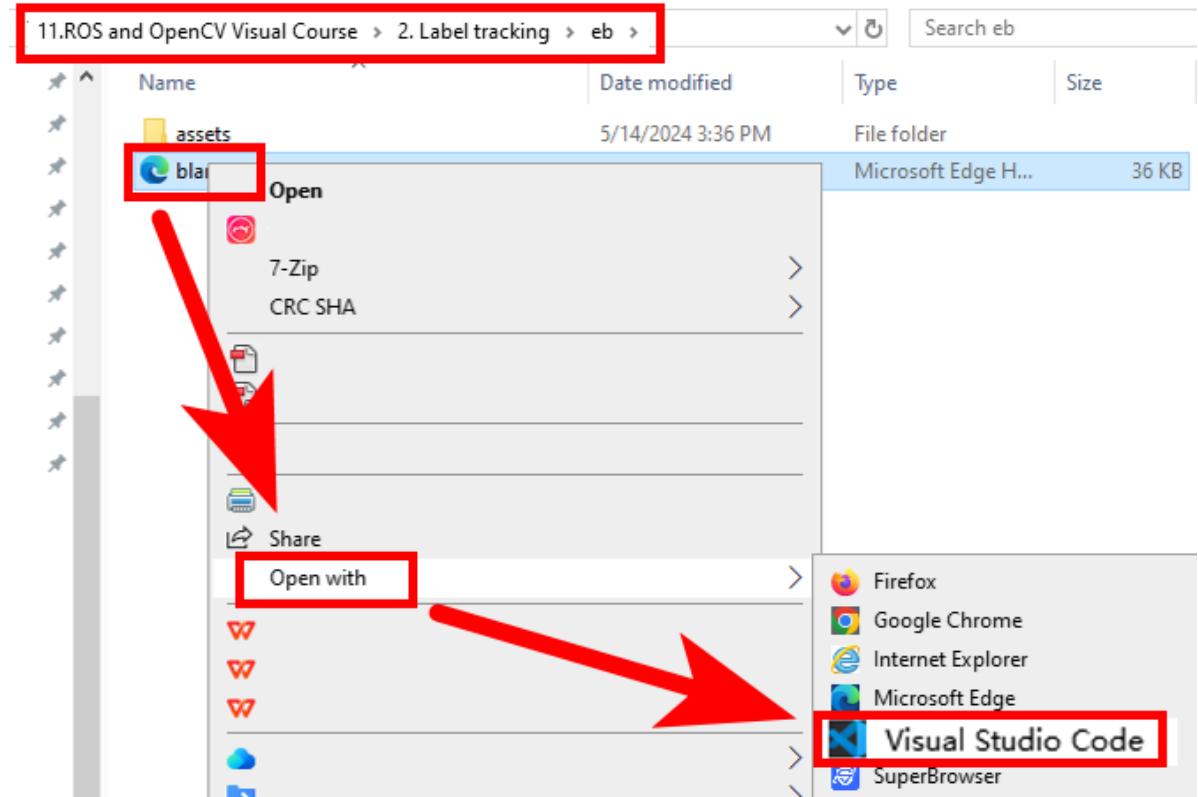


```

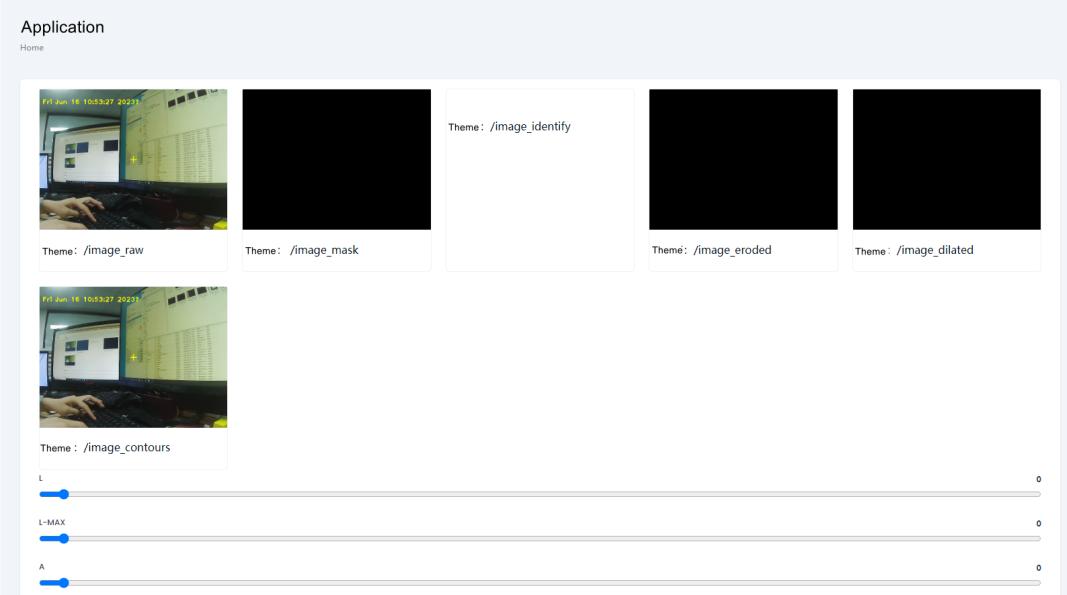
345
346
347
348     </div>
349
350     </div>
351     <script src="assets/js/roslib.js"></script>
352     <script src="assets/js/alpine.js"></script>
353     <script src="assets/js/perfect-scrollbar.js"></script>
354     <script src="assets/js/choices.js"></script>
355     <script src="assets/js/chart.js"></script>
356     <script src="assets/js/apexchart.js"></script>
357     <script src="assets/js/quill.js"></script>
358     <script src="assets/js/rangeslider.min.js"></script>
359     <script src="assets/js/main.js"></script>
360
361
362
363     var ros = new ROSLIB.Ros({
364       url : 'ws://192.168.2.117:9090'
365     });
366
367     ros.on('connection', function() {
368       console.log('Connected to websocket server.');
369     });
370
371     ros.on('error', function(error) {
372       console.log('Error connecting to websocket server: ', error);
373     });
374
375     ros.on('close', function() {
376       console.log('Connection to websocket server closed.');
377     });
378
379     var imageListener_compressed= new ROSLIB.Topic({
380       ros : ros,
381       name : 'image_raw/compressed',
382       messageType : 'sensor_msgs/msg/CompressedImage',
383       throttle_rate : 100
384     });
385
386     var that = this;
387     imageListener_compressed.subscribe(function(data) {
388       // update the robots position on the path

```

Change the IP address here to the actual IP address of the Rosb Ridge



As shown in the picture below, you can see the pictures transmitted by the camera.



Then we set the LAB value of the color through the slider.

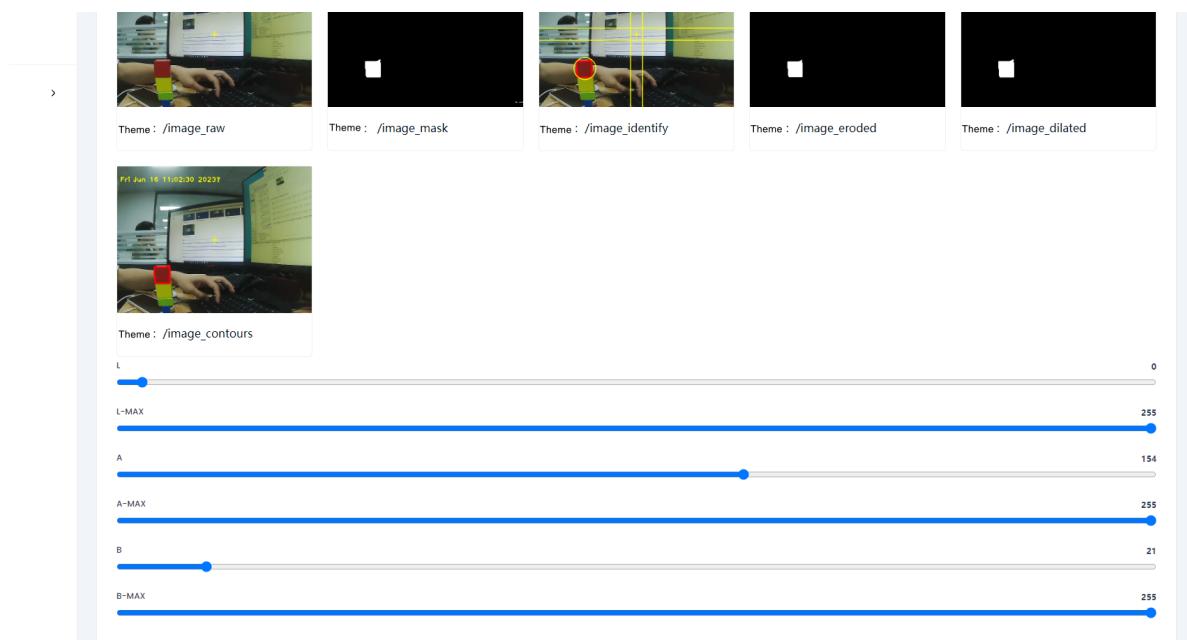
```
Yellow: {"l":96, "a": 55, "b":188, "l_max": 252 , "a_max": 141, "b_max": 255}
```

```
Red: {"l":0, "a": 155, "b":21, "l_max": 255 , "a_max": 255, "b_max": 255}
```

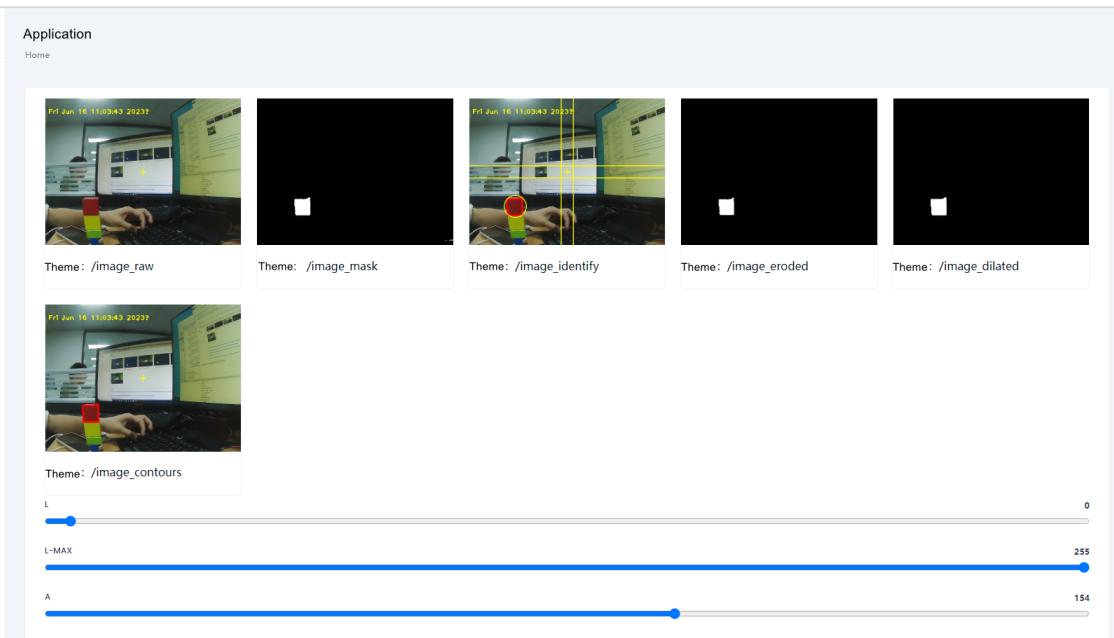
```
Green: {"l":26, "a": 7, "b":170, "l_max": 143 , "a_max": 110, "b_max": 255}
```

```
Blue: {"l":0, "a": 0, "b":0, "l_max": 255 , "a_max": 255, "b_max": 102}
```

Above are the LAB values of several colors. We can choose one to set. For example, we set red, as shown in the figure below, and move the slider.



In the picture we can see that the red square has been identified.



In the shell terminal, we open a new terminal and enter the command in the terminal

Note: The terminal here is the terminal connected to the mechanical dog, not the terminal of the virtual machine.

```
#pi4
cd cartographer_ws2/
source install/setup.bash
ros2 run voice_xgo_ctrl_run voice_xgo_ctrl_color_identify

#pi5 (need to enter docker terminal)
cd yahboomcar_ws/
source install/setup.bash
ros2 run voice_xgo_ctrl_run voice_xgo_ctrl_color_identify
```

PI4 version running command icon:

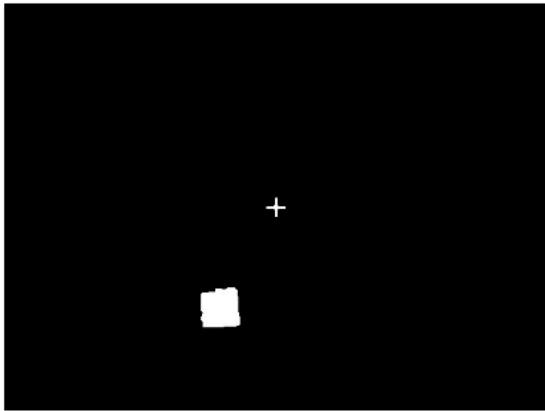
```
pi@yahboom: ~$ cd cartographer_ws2/
pi@yahboom:~/cartographer_ws2$ 
pi@yahboom:~/cartographer_ws2$ source install/setup.bash
pi@yahboom:~/cartographer_ws2$ 
pi@yahboom:~/cartographer_ws2$ 
pi@yahboom:~/cartographer_ws2$ ros2 run voice_xgo_ctrl_run voice_xgo_ctrl_color_identify
Speech Serial Opened! Baudrate=115200
```

Then he said to the mechanical dog: "Hello, Yahboom"

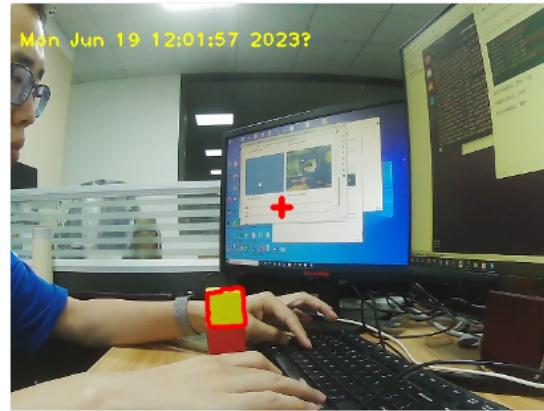
The robot dog will respond: "Yes."

Then say to the mechanical dog: "Yellow"

Yellow is recognized as shown in the picture below



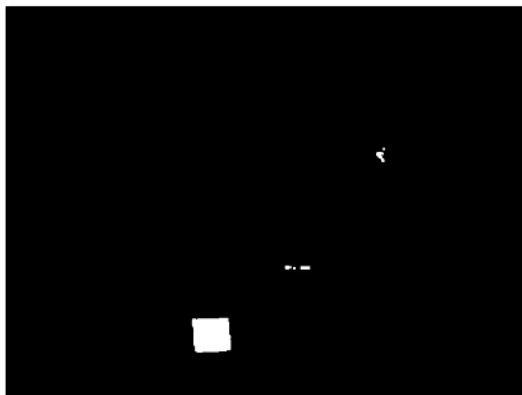
Theme: /image\_dilated



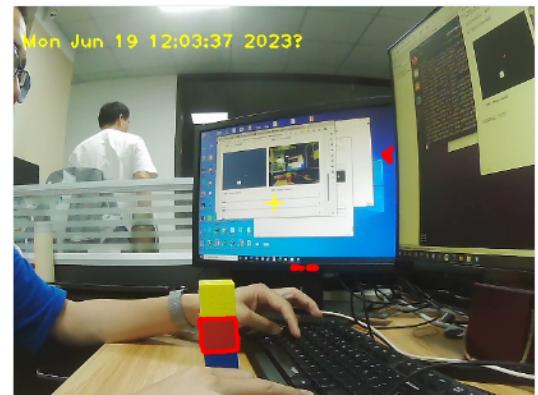
Theme: /image\_contours

Say "red" to the mechanical dog

As shown in the figure below, red is recognized



Theme: /image\_dilated



Theme: /image\_contours