5. Enter the robot's docker container

***ORIN's ROS1 course is all in the docker container. ***

5. Enter the robot's docker container

- 5.1. Related concepts
- 5.2, How to query the docker image version used by the robot
- 5.3. Bind peripherals
- 5.4. Check the connection of peripherals
- 5.5, Edit the script
- 5.6, execute the script
- 5.7, Switching models, radars and cameras
- 5.8. Multiple terminals enter the same docker container
- 5.9. How to open a container that is already in the [Exited] closed state
  - 5.9.1. Need to use the camera
  - 5.9.2, no need to use the camera
  - 5.9.3, re-enter the container in the [Exited] closed state

The operating environment and software and hardware reference configuration are as follows:

- Reference vehicle: ROSMASTER X3PLUS
- Robot hardware configuration: Arm series master, EAI 4ROS LiDAR, AstraPro Plus depth camera
- Robot system: Ubuntu (no version requirement) + docker (20.10.21 and above)
- PC virtual machine: Ubuntu (20.04) + ROS (Noetic)
- Usage scenario: Use on a relatively clean 2D plane

5.1. Related concepts

1. What is the host of docker

```
The host is the server where we call the command to create a container using the
image. Here it refers to the master on our car (jetson or Raspberry Pi, etc.),
and the host mentioned below refers to this.
```

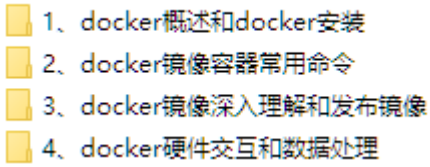
2. What is GUI

```
GUI is a graphical user interface, which mainly refers to: image window displayed
by opencv, rviz interface, rqt interface, etc.
```

3. What is the docker container of the robot

```
The robot here is the Rosmaster car, that is, the Rosmaster car container with
various development dependent environments configured
```

4. Before operating this chapter tutorial, please make sure that you have mastered the knowledge of the following chapters, otherwise you may find it difficult to learn. In this case, please check the following pre-knowledge content repeatedly. You will feel very relaxed after mastering it. Come on, you are the best!



5.2, How to query the docker image version used by the robot

1. The docker image version used by the robot is also the image version used on the car. After the user burns the system image of the car and starts it, execute:

```
jetson@yahboom:~$ docker images
REPOSITORY                      TAG       IMAGE ID       CREATED
SIZE
yahboomtechnology/ros-noetic    3.0.1     2bb046f7e0a3   Less than a second ago
14.9GB
192.168.2.51:5000/ros-noetic    3.0.0     38494195a8e0   Less than a second ago
14.9GB
ghcr.io/open-webui/open-webui   main      8f1fbf4363d3   Less than a second ago
3.72GB
heartexlabs/label-studio        latest    0ee0e4188523   Less than a second ago
662MB
```

You will see multiple docker image versions. Please select the name [yahboomtechnology/ros-noetic]. The version with the highest tag is the latest image version of the robot. If you find it here, use the version [yahboomtechnology/ros-noetic:3.0.1]

2. Why can't we just put one docker image in the car system?

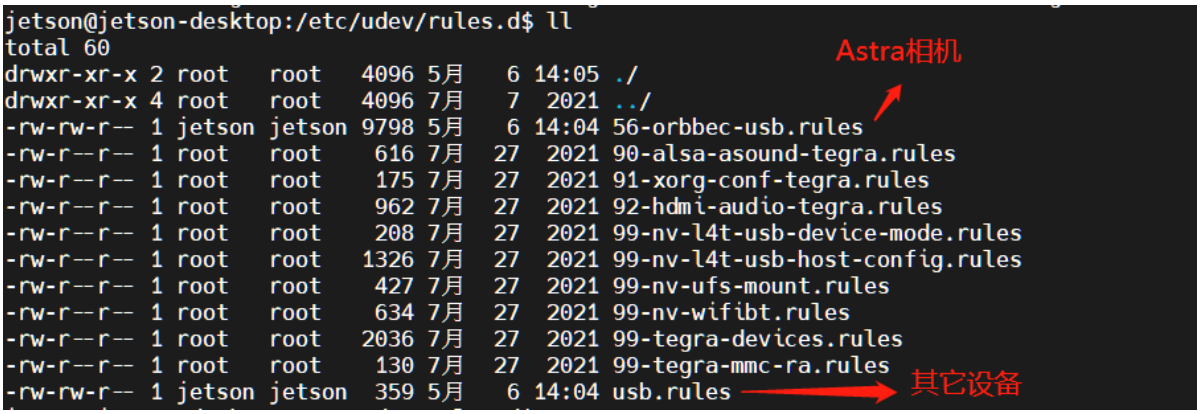
If you have read the tutorial of [07, Docker ------ 3, docker image in-depth understanding and release image], you should know that docker image is a layered mechanism, that is, the image of the latter tag depends on the image of the previous tag. Therefore, there may be multiple versions of docker images in the host machine, and the tags of these images will be updated in an incremental manner.

Later, when we update new courses, we will also update functions by releasing new docker images.

5.3. Bind peripherals

- First, make sure that the car has been connected to various peripherals and that port binding has been done for the peripherals. Port binding is handled on the docker host (car)
- Common peripherals include: serial devices, lidar, RGBD camera, voice control module, handle remote control, etc.
- **By default, the car has bound Astra camera, lidar and serial device**. If you need to bind other devices, please refer to the port binding tutorial
- For the steps of port binding, please refer to the tutorial section [Six, Linux operating system ----- 06. Bind device ID]

Port binding has been configured in the host. If you need to modify it, you can view the content inside and modify it:

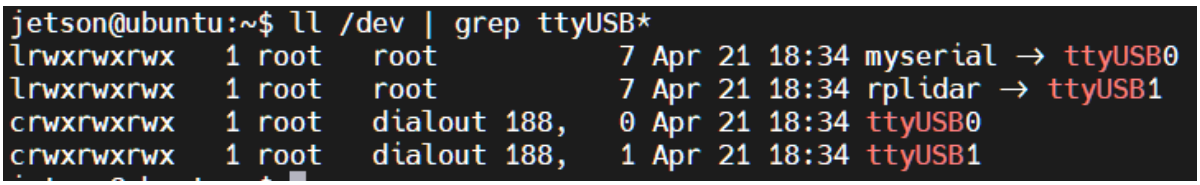


5.4. Check the connection of peripherals

This step is performed on the host machine:

1. Here is to check the peripherals other than the camera. There is no voice control module connected here. If it is connected, the [myspeech] device will be displayed

```
ll /dev | grep ttyUSB*
```



2. Check the port of the AstraPro Plus camera as follows:

```
jetson@yahboom:~$ ll /dev/as*
lrwxrwxrwx 1 root root 15 Jan 1 08:00 /dev/astro_pro_plus -> bus/usb/001/012
lrwxrwxrwx 1 root root 15 Jan 1 08:00 /dev/astro_pro_plus_rgb -> bus/usb/001/014
```

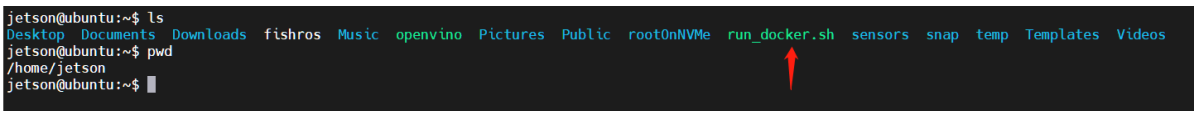
5.5, Edit the script

Since the port number often changes after the AstraPro Plus camera is plugged in and out, you need to re-edit the script to configure the port of the AstraPro Plus camera.

Edit the script to run docker. This step is performed on the host machine:

1. The script to run docker [run_docker.sh] is generally placed in the root directory of the car's main directory. I am in the following path. If you don't have it, you can create the file yourself. Remember to give the script executable permissions after creation.

```
chmod +x run_docker.sh #Give the script executable permissions
```



[run_docker.sh] The contents of the script are as follows:

The uncommented ones can be copied directly and modified as needed

Note: When adding host devices to the container below, if the host is not connected to the device, you need to remove the corresponding addition operation to start the container

```
#!/bin/bash
xhost +
docker run -it \
--gpus all \
--runtime=nvidia \
--net=host \
--env="DISPLAY" \
--env="QT_X11_NO_MITSHM=1" \
--env="NO_AT_BRIDGE=1" \
-v /tmp/.X11-unix:/tmp/.X11-unix \
-v /home/jetson/temp:/root/temp \
-v /dev/bus/usb/001/012:/dev/bus/usb/001/012 \
-v /dev/bus/usb/001/010:/dev/bus/usb/001/010 \
--device=/dev/myserial \
--device=/dev/myspeech \
--device=/dev/ydlidar \
--device=/dev/astro_pro_plus \
--device=/dev/astro_pro_plus_rgb \
--device=/dev/input \
--device=/dev/camera_usb \
--device=/dev/camera_depth \
-p 9090:9090 \
-p 8888:8888 \
yahboomtechnology/ros-noetic:3.0.1 /bin/bash
```

Commented script description:

Note: When adding a host device to a container, if the host is not connected to the device, you need to remove the corresponding addition operation to start the container

```
#!/bin/bash
xhost +                                                    # xhost is used to
support the display of GUI in docker
docker run -it \                                           # Interactively run
docker image
--gpus all \                                               # Allow docker to use
all gpus
--runtime=nvidia \                                         # Allow docker to use
nvidia components
--net=host \                                               # Set the container
network to host mode
--env="DISPLAY" \                                          # Enable the display
of the GUI interface
--env="QT_X11_NO_MITSHM=1" \                               # Use X11 port 1 for
display
-v /tmp/.X11-unix:/tmp/.X11-unix \                         # Mapping shows the
service node directory
-v /home/jetson/temp:/root/temp \                          # As a temporary file
transfer directory between the host and the container, you can use this directory
if you need to transfer files
-v /dev/bus/usb/001/010:/dev/bus/usb/001/010 \             # Add the host device
to the container, here is the astrpro plus device port, if the car is not
connected to the camera, please remove this line
-v /dev/bus/usb/001/011:/dev/bus/usb/001/011 \
```

```
                                                    # Add the host device
to the container, here is the astrpro plus device port, if the car is not
connected to the camera, please remove this line
--device=/dev/astro_pro_plus \                      # Add the host device
to the container, here is the astrpro plus device port, if the car is not
connected to the camera, please remove this line
--device=/dev/astro_pro_plus_rgb \                  # Add the host device
to the container, here is the astrpro plus device port, if the car is not
connected to the camera, please remove this line
--device=/dev/myserial \                            # Add host device to
container, here is the serial device port, if the car is not connected to the
serial port, please remove this line
--device=/dev/rplidar \                             # Add host device to
container, here is the radar device port, if the car is not connected to the
radar, please remove this line
--device=/dev/myspeech \                            # Add host device to
container, here is the voice control device port, if the car is not connected to
the voice control device, please remove this line
--device=/dev/input \                               # Add host device to
container, here is the handle device port, if the car is not connected to the
handle, please remove this line
--device=/dev/camera_usb \                          # Add host device to
container, here is the usb camera, if the car is not connected to the camera,
please remove this line
--device=/dev/camera_depth \                        # Add host device to
container, here is astrpro plus color camera, if the car is not connected to the
camera, please remove this line
-p 9090:9090 \                                      # Open port
-p 8888:8888 \
yahboomtechnology/ros-noetic:3.0.1 /bin/bash        # The image name to
be started, according to the modification found in step 5.2; execute the
/bin/bash command in the container

#Note: When adding the host device to the container above, if the host is not
connected to the device, you need to remove the corresponding addition operation
to start the container
```

2. Modify the above script. These two lines are the port numbers of the AstraPro Plus camera. Since the port number will change after the camera is plugged in and out, you need to reconfigure the camera port

```
-v /dev/bus/usb/001/010:/dev/bus/usb/001/010 \      # Mount the storage
volume to the container, mount it to a directory in the container, here are the
camera's rgb and depth ports mounted
-v /dev/bus/usb/001/011:/dev/bus/usb/001/011 \
```

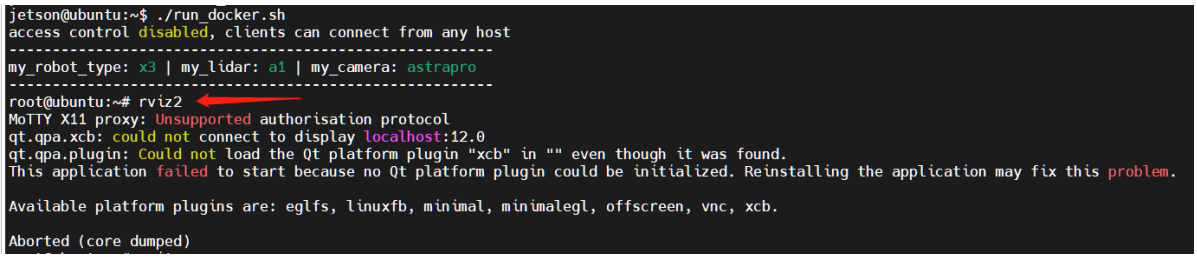
This is the camera port queried in step 5.4 2. This port may change after the camera is plugged in and out, so everyone's port is different and needs to be configured by yourself.

```
-v /dev/bus/usb/001/007:/dev/bus/usb/001/007 \      # Mount the storage
volume to the container, mount it to a directory in the container, here are the
camera's rgb and depth ports mounted
-v /dev/bus/usb/001/009:/dev/bus/usb/001/009 \
```

5.6, execute the script

After step 5.5 is executed, Open the terminal on the docker host [i.e., the car, which can be executed on VNC or on the car screen]

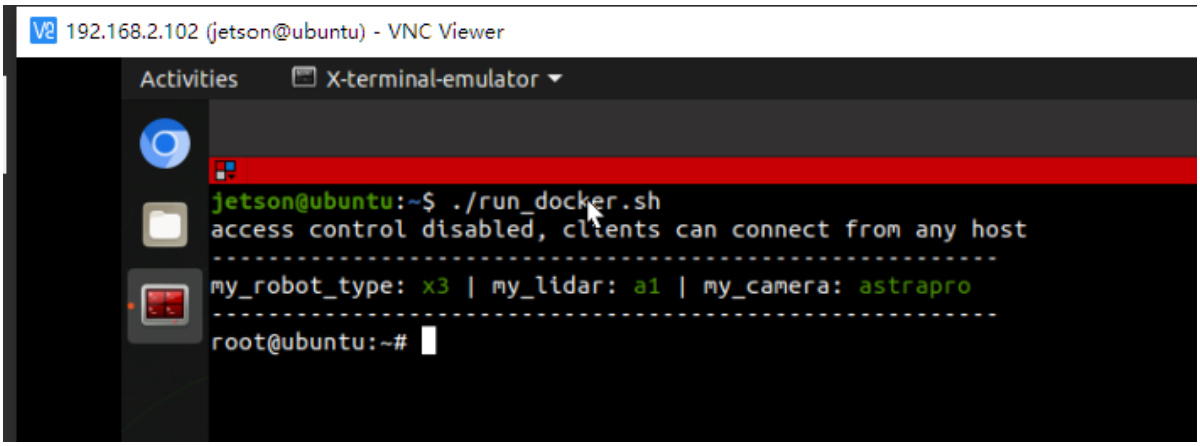
Note: This must be executed on the car's VNC or on the car screen, and cannot be executed in the car terminal remotely entered through ssh (such as the car terminal entered through MobaXterm), otherwise the GUI image may not be displayed in the container. As shown below, after entering the car terminal in MobaXterm and executing run_docker.sh to enter the container, rviz cannot be displayed



Execute on the car's VNC interface or on the car screen:

```
./run_docker.sh
```

You can enter the container correctly and display the GUI screen. You can execute the rviz command again for testing.



5.7, Switching models, radars and cameras

Note: Since the ROSMASTER series robots are divided into multiple robots and multiple devices, the factory system has been configured with routines for multiple devices, but since the product cannot be automatically identified, the machine type and radar model need to be manually set.

After entering the container: Make the following changes according to the car model, radar type and camera type:

```
root@ubuntu:/# cd
root@ubuntu:~# vim .bashrc
```

```
# env
alias python=python3
export ROS_DOMAIN_ID=112

export ROBOT_TYPE=r2            # r2, x1, x3
export RPLIDAR_TYPE=a1          # a1, s2, 4ROS
export CAMERA_TYPE=astraplus    # astrapro, astraplus
echo "------------------------------------------------------"
echo -e "ROS_DOMAIN_ID: \033[32m$ROS_DOMAIN_ID\033[0m"
echo -e "my_robot_type: \033[32m$ROBOT_TYPE\033[0m | my_lidar: \033[32m$RPLIDAR_TYPE\033[0m | my_camera: \033[32m$CAMERA_TYPE\033[0m"
echo "------------------------------------------------------"

#colcon_cd
source /usr/share/colcon_cd/function/colcon_cd.sh
export _colcon_cd_root=/root/yahboomcar_ros2_ws/yahboomcar_ws
source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash

#ros2
source /opt/ros/foxy/setup.bash
source /root/yahboomcar_ros2_ws/yahboomcar_ws/install/setup.bash
source /root/yahboomcar_ros2_ws/software/library_ws/install/setup.bash
```

After the modification is completed, save and exit vim, and then execute:

```
root@ubuntu:~# source .bashrc
------------------------------------------------------
ROS_DOMAIN_ID: 12
my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus
------------------------------------------------------
root@ubuntu:~#
```

You can see the current modified car model, radar type and camera type

The robot project files are stored in the following directory:

```
/root/yahboomcar_ws
```

5.8. Multiple terminals enter the same docker container

1. In the above steps, a docker container has been opened. You can open another terminal on the host (car) to view it:

```
docker ps -a
```

```
jetson@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE                                  COMMAND       CREATED       STATUS       PORTS     NAMES
5b698ea10535   yahboomtechnology/ros-foxy:3.3.9   "/bin/bash"   3 days ago   Up 9 hours             ecstatic_lewin
jetson@ubuntu:~$
```

2. Now enter the docker container in this newly opened terminal:

```
docker exec -it 5b698ea10535 /bin/bash
```

```
jetson@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE                                  COMMAND       CREATED       STATUS       PORTS     NAMES
5b698ea10535   yahboomtechnology/ros-foxy:3.3.9   "/bin/bash"   3 days ago   Up 9 hours             ecstatic_lewin
jetson@ubuntu:~$ docker exec -it 5b698ea10535 /bin/bash
------------------------------------------------------
my_robot_type: x3 | my_lidar: a1 | my_camera: astrapro
------------------------------------------------------
root@ubuntu:/#
```

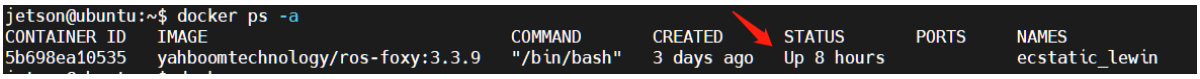
After successfully entering the container, you can open countless terminals to enter the container.

The robot project files are stored in the following directory:

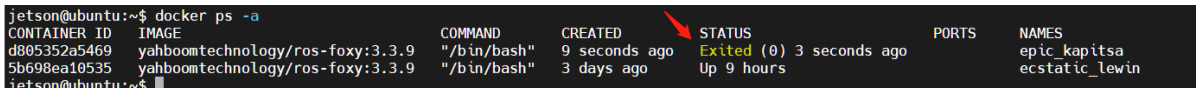
```
/root/yahboomcar_ws
```

3. Note:

(1) When executing the command in step 2, make sure the container is in the [UP] state



(2) If the container is in the [Exited] closed state, please refer to the following step 1.6



5.9. How to open a container that is already in the [Exited] closed state

There are two situations here: still need to use the camera and no longer need to use the camera

5.9.1. Need to use the camera

First, follow the instructions in the above [5.3, Check the peripheral connection status] step to check whether the port of the AstraPro Plus camera has changed.

1. If the port of the Astra Pro camera has changed, you cannot enter the container again.

(1) If there are some changes in the container that need to be retained, you can refer to the following command to generate a new image,

```
Submit an image from the container:
docker commit container id target image name to be created: [label name]
For example: docker commit 66c40ede8c68 yahboomtechnology/ros-noetic:3.0.1 #
Label name increments according to your own situation

Then run this new image to enter the container: refer to this section [5.2 to
5.5] steps to execute
```

(2) If there are no changes that need to be retained, directly refer to this section [5.2 to 5.5] steps to execute to enter the container.

2. If the port of the AstraPro Plus camera has not changed, refer to [5.7.3, re-enter the container in the [Exited] closed state] step.

5.9.2, no need to use the camera

Refer to [5.7.3, re-enter the container in the [Exited] closed state] step.

5.9.3, re-enter the container in the [Exited] closed state

Open the terminal on the docker host [i.e. the cart, which can be executed on VNC or on the cart screen]

Note: This must be executed on the cart's VNC or on the cart screen, and cannot be executed in the cart terminal remotely accessed through ssh (such as the cart terminal accessed through MobaXterm), otherwise the GUI image may not be displayed in the container. Of course, if you do not need to display the GUI image, then it is OK.

1. First check the status of the container

```
docker ps -a
```

2. Enable GUI access

```
xhost +
```

3. Enable the container [the container ID here can be abbreviated as long as it can uniquely identify the existing container]

```
docker start 5b
```

4. Enter the container again

```
docker exec -it 5b /bin/bash
```

5. Open rviz to see if the GUI screen can be opened

```
rviz
```

6. Execute as follows:

```
jetson@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE                                COMMAND       CREATED
STATUS                      PORTS     NAMES
5b698ea10535   yahboomtechnology/ros-melodic:1.4.1   "/bin/bash"   3 days ago
Exited (0) 8 seconds ago              ecstatic_lewin

jetson@ubuntu:~$ xhost +
access control disabled, clients can connect from any host

jetson@ubuntu:~$ docker start 5b
5b

jetson@ubuntu:~$ docker exec -it 5b /bin/bash
------------------------------------------------------------
my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus
------------------------------------------------------------
root@ubuntu:/# rviz
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
[INFO] [1682298616.634096279] [rviz]: Stereo is NOT SUPPORTED
[INFO] [1682298616.634576375] [rviz]: OpenGl version: 3.1 (GLSL 1.4)
[INFO] [1682298617.959654036] [rviz]: Stereo is NOT SUPPORTED
```