5.Enter docker container

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. This refers to our system motherboard (jetson board or Raspberry PI,
etc.), and the host machine mentioned below refers to this.
```

2. What is GUI?

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

3. What is the docker container

```
when the docker image is open and running, it is the container
```

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

📒 1、Docker overview and docker installation
📒 2、Common commands for docker image containers
📒 3、Docker images deeply understand and publish images
📒 4、Docker hardware interaction and data processing

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 mirror version used on the trolley, and the user executes after the system image of the burned trolley is started:

```
pi@raspberrypi:~ $ docker images
REPOSITORY                      TAG       IMAGE ID       CREATED        SIZE
yahboomtechnology/ros2-foxy     2.0.1     4b8a5eee5110   7 days ago     5.61GB
yahboomtechnology/ros-melodic   1.2       1f3f7da97f2b   7 days ago     3.05GB
hello-world                     latest    b038788ddb22   7 months ago   9.14kB
```

You will see that there are multiple docker image versions, please select the name which is the highest tag version is the latest image version of the board. The actual situation is displayed based on the mirror you have queried.

2、Why can't you just put a docker image in the car system?

If you have read the tutorial in the chapter [Docker ------ 3, docker images to deeply understand and release images], you should know that docker images are layered mechanisms, 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, and the tags of these images will be updated incrementally.

In the future, we will update the new course, and we will also update the function by releasing a new docker image

5.3、Binding peripherals

The following uses Radar and serial device bonding as an example

This step is performed on the host::

1、Here is to view peripherals other than the camera

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


2、View camera device

```
ls /dev/video*
```

5.4、Edit scripts

Then edit the startup script according to the device found in the previous step

The following uses Radar and serial device bonding as an example

Edit the script that runs docker，This step is performed on the host:

1、Create a docker run script [docker_ros1.sh], It's usually in the home directory

```
chmod +x docker_ros1.sh              #Give the script executable permissions
```

【docker_ros1.sh】The contents of the script are as follows:

Without comments, you can copy it directly and modify it as needed

Note: When adding a host device to a container below, if the host does not have the device connected, you need to remove the corresponding add operation to open the container

```
#!/bin/bash
xhost +

docker run -it \
--net=host \
--env="DISPLAY" \
--env="QT_X11_NO_MITSHM=1" \
-v /tmp/.X11-unix:/tmp/.X11-unix \
-v /home/jetson/temp:/root/yahboomcar_ros2_ws/temp \
--device=/dev/video0 \
--device=/dev/myserial \
--device=/dev/rplidar \
-p 9090:9090 \
-p 8888:8888 \
yahboomtechnology/ros2-foxy:2.0.1 /bin/bash
```

Here is the annotated script description:

Note: When adding a host device to a container below, if the host does not have the device connected, you need to remove the corresponding add operation to open the container

```
#!/bin/bash
xhost +                      # xhost is used to support displaying GUIs in docker

docker run -it \                     # Run docker images interactively
--net=host \                         # The container network is set to host mode
--env="DISPLAY" \                    # Open the display GUI interface
--env="QT_X11_NO_MITSHM=1" \         # Port 1 of X11 is used for display
-v /tmp/.X11-unix:/tmp/.X11-unix \       # map shows the service node directory
-v /home/jetson/temp:/root/yahboomcar_ros2_ws/temp \    # As a directory for the
host and container to temporarily transfer files, you can use this directory if
you need to transfer files
--device=/dev/video0 \       # Add the video device to the container, No camera
connected, please remove this line
```

```
--device=/dev/myserial \      # Add the serial device to the container, No serial
device connected, please remove this line
--device=/dev/rplidar \       # Add the radar device to the container, No radar
device connected, please remove this line
-p 9090:9090 \                                    # Open port
-p 8888:8888 \
yahboomtechnology/ros2-foxy:2.0.1 /bin/bash       #  The name of the image to be
started, according to the modification queried in step 5.2; execute the /bin/bash
command inside the container
```

5.5、 Execute scripts

After the 5.4 step is completed, , open the terminal on the host of docker (can be on VNC or on the screen of the board)

Note: Here must be executed on the VNC of the trolley or on the trolley screen, not in the trolley terminal entered remotely through ssh (such as the trolley terminal entered through MobaXterm), otherwise the GUI image may not be displayed in the container，as follows in MobaXterm into the trolley terminal execution run_docker.sh after entering the container, rviz cannot be displayed

```
jetson@ubuntu:~$ ./run_docker.sh
access control disabled, clients can connect from any host
-------------------------------------------------
my_robot_type: x3 | my_lidar: a1 | my_camera: astrapro
-------------------------------------------------
root@ubuntu:~# rviz2  ←
MoTTY X11 proxy: Unsupported authorisation protocol
qt.qpa.xcb: could not connect to display localhost:12.0
qt.qpa.plugin: Could not load the Qt platform plugin "xcb" in "" even though it was found.
This application failed to start because no Qt platform plugin could be initialized. Reinstalling the application may fix this problem.

Available platform plugins are: eglfs, linuxfb, minimal, minimalegl, offscreen, vnc, xcb.

Aborted (core dumped)
```

Run the startup script you created earlier on the VNC interface or on the screen

(Note: Each time this script is executed, a new container is created from the image.)

```
./docker_ros1.sh
```

The container can be entered correctly, and the GUI screen can be displayed, and the rviz2 command test can be executed again.

```
rviz2
```

```
pi@raspberrypi:~ $ ./docker_ros1.sh
xhost:  unable to open display ""
WARNING: Published ports are discarded when using host network mode
```

If the GUI cannot be displayed after executing the rviz2 command, the following error is displayed: (Generally, it may appear in the Raspberry Pi)

```
root@ubuntu:~# rviz2
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
dbus[97]: The last reference on a connection was dropped without closing the con
nection. This is a bug in an application. See dbus_connection_unref() documentat
ion for details.
Most likely, the application was supposed to call dbus_connection_close(), since
 this is a private connection.
  D-Bus not built with -rdynamic so unable to print a backtrace
Aborted (core dumped)
```

You need to add another parameter to the startup script：

```
--security-opt apparmor:unconfined
```

such as:

```bash
#!/bin/bash
xhost +

docker run -it \
--net=host \
--env="DISPLAY" \
--env="QT_X11_NO_MITSHM=1" \
-v /tmp/.X11-unix:/tmp/.X11-unix \
--security-opt apparmor:unconfined \                      # Added this parameter
-v /home/jetson/temp:/root/yahboomcar_ros2_ws/temp \
--device=/dev/video0 \
--device=/dev/myserial \
--device=/dev/rplidar \
-p 9090:9090 \
-p 8888:8888 \
yahboomtechnology/ros2-foxy:2.0.1 /bin/bash
```

Then run the script again to enter the container and display the GUI screen.

5.6、 multiple terminals enter the same docker container

1、 In the above steps, a docker container has been opened, and another terminal can be opened on the host to view:

```
docker ps -a
```

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

```
docker exec -it [container id] /bin/bash
```

example：

Successfully entering the container, you can also open an unlimited number of terminals to enter the container.

3、 Note：

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

```
jetson@unbutu:~$ docker ps -a
CONTAINER ID   IMAGE                                COMMAND       CREATED             STATUS              PO
35c8de860c11   yahboomtechnology/ros2-base:2.0.2    "/bin/bash"   About an hour ago   Up 2 seconds
ebdc5cc1469f   ubuntu:latest                        "/bin/bash"   5 hours ago         Up 3 hours
```

(2) If the container is in the Exited shutdown state, see Step 5.7 below

```
jetson@unbutu:~$ docker ps -a
CONTAINER ID   IMAGE                                COMMAND       CREATED             STATUS
35c8de860c11   yahboomtechnology/ros2-base:2.0.2    "/bin/bash"   About an hour ago   Exited (0) 4 seconds ago
ebdc5cc1469f   ubuntu:latest                        "/bin/bash"   5 hours ago         Up 3 hours
jetson@unbutu:~$
```

5.7、How to open a container that is already in the [Exited] closed state

There are two cases here:: **Cameras and other related external equipment changes** and **Cameras and other related external equipment not changes** (See step 5.3 Checking Peripheral Connections)

1、Cameras and other related external equipment changes

If there are changes in the container that need to be preserved, can generate a new image by referring to the following command (See the previous tutorial for detailed steps to publish an image)

```
# Commit an image from the container:
docker commit [container id] [Target image name to be created:[tag]]
For example: docker commit 66c40ede8c68 yahboomtechnology/ros-foxy:1.1  # tag
names increment according to your own situation

Then run this new image into the container: see steps 5.2 to 5.5 in this section
```

2、Cameras and other related external equipment not changes, then directly refer to the [5.7.1、enter the [Exited] closed container again] step to execute.

5.7.1、enter the [Exited] closed container again

Open the terminal on the docker's host (can be on VNC or on the screen of the matherboard)

Note: Here must be executed on the VNC of the board or on the screen of the board, not in the board terminal entered remotely through ssh, otherwise the GUI image may not be displayed in the container, of course, how you do not need to display the GUI image, then you can.

1、First, check the status of the container

```
docker ps -a
```

2、Enable GUI access permissions

```
xhost +
```

3、Open the container [the ID of the container here can be abbreviated, as long as it can uniquely identify the container that currently exists]

```
docker start [container ID]
```

4、Enter the container again

```
docker exec -it [container ID] /bin/bash
```

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

```
rviz2
```