4、Docker hardware interaction and data processing

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

- REFERENCE MODEL: ROSMASTER X3
- Robot hardware configuration: Arm series main control, Silan A1 lidar, AstraPro Plus depth camera
- Robot system: Ubuntu (version not required) + docker (version 20.10.21 and above)
- PC Virtual Machine: Ubuntu (20.04) + ROS2 (Foxy)
- Usage scenario: Use on a relatively clean 2D plane

4.1、Hardware mounting (port binding)

1. Create udev rules in the host (/etc/udev/rules.d/), see [6. Linux operating system ---- 6. Binding Device ID section
2. Then, when opening the container, mount the devices with rules set on the rules into the docker container through parameters such as --device=/dev/myserial --device=/dev/rplidar

```
docker run -it --device=/dev/myserial --device=/dev/rplidar ubuntu:latest /bin/bash
```

3. The device can be found in the docker container

```
jetson@ubuntu:~$ docker images
REPOSITORY                    TAG       IMAGE ID       CREATED             SIZE
ubuntu                        1.0       78ca7be949b6   About an hour ago   69.2MB
pengan88/ubuntu               1.0       78ca7be949b6   About an hour ago   69.2MB
yahboomtechnology/ros-foxy    3.4.0     49581aa78b6b   6 hours ago         24.3GB
yahboomtechnology/ros-foxy    3.3.9     cefb5ac2ca02   4 days ago          20.5GB
yahboomtechnology/ros-foxy    3.3.8     49996806c64a   4 days ago          20.5GB
yahboomtechnology/ros-foxy    3.3.7     8989b8860d17   5 days ago          17.1GB
yahboomtechnology/ros-foxy    3.3.6     326531363d6e   5 days ago          16.1GB
```

```
mysql                         latest    5371f8c3b63e   6 days ago          592MB
ubuntu                        latest    bab8ce5c00ca   6 weeks ago         69.2MB
hello-world                   latest    46331d942d63   13 months ago       9.14kB
jetson@ubuntu:~$ ll /dev | grep ttyUSB*
lrwxrwxrwx   1 root   root           7 Apr 23 18:07 myserial -> ttyUSB0
lrwxrwxrwx   1 root   root           7 Apr 23 18:07 rplidar -> ttyUSB1
crwxrwxrwx   1 root   dialout 188,   0 Apr 23 18:07 ttyUSB0
crwxrwxrwx   1 root   dialout 188,   1 Apr 23 18:07 ttyUSB1
jetson@ubuntu:~$ docker run -it   --device=/dev/myserial --device=/dev/rplidar
ubuntu:latest  /bin/bash
root@03522257ba30:/# ls /dev   # docker中已经有myserial和rplidar
console  fd  full  mqueue  myserial  null  ptmx  pts  random  rplidar  shm  stderr
stdin  stdout  tty  urandom  zero
```

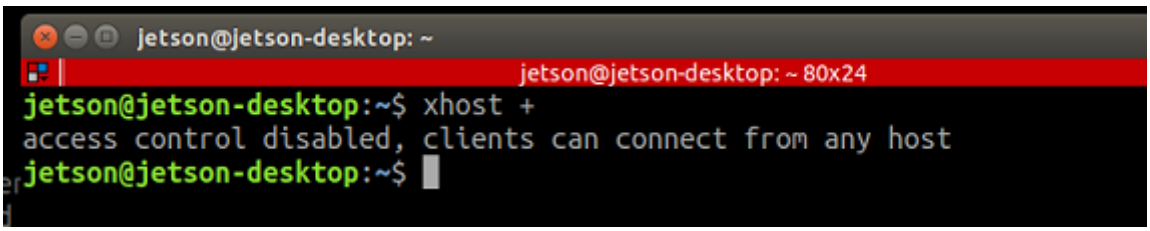
4.2、GUI display in docker

1. Install in the host:

```
sudo apt-get install tigervnc-standalone-server tigervnc-viewer
sudo apt-get install x11-xserver-utils
```

2. Execute in the host: xhost +

After the following image is displayed, perform 3 steps:



3. Execute the command in the host to enter the container:

```
docker run -it  # Run docker images interactively
--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/. The X11-UNIX# map shows the service node directory
yahboomtechnology/ros-foxy:3.3.9 # The name of the image to start
/bin/bash # Executes the /bin/bash command inside the container
```

4. Testing

```
Execute in container: rviz2
```

4.3、docker containers and hosts transfer files to each other

4.3.1、use cp naming

4.3.1.1、Copy files from the container to the host

```
# command
Docker cp container ID: Intra-container path Destination host path

# Test
# Execute inside the container, create a file test
jetson@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE           COMMAND       CREATED       STATUS
PORTS     NAMES
c54bf9efae47   ubuntu:latest   "/bin/bash"   2 hours ago   Up 9 minutes
          funny_hugle
3b9c01839579   hello-world     "/hello"      3 hours ago   Exited (0) 3 hours ago
          jovial_brown
jetson@ubuntu:~$ docker attach c5
root@c54bf9efae47:/# ls
bin  boot  dev  etc  home  lib  media  mnt  opt  proc  root  run  sbin  srv  sys
tmp  usr  var
root@c54bf9efae47:/# cd
root@c54bf9efae47:~# ls
root@c54bf9efae47:~# touch test.txt
root@c54bf9efae47:~# ls
test.txt
root@c54bf9efae47:~# pwd
/root
root@c54bf9efae47:/# read escape sequence    #ctrl+P+Q The container does not stop
exiting
jetson@ubuntu:~$ docker cp c54bf9efae47:/root/test.txt ~/
jetson@ubuntu:~$ ls      # The test .txt file has been copied in
Desktop  Documents  Downloads  fishros  Music  openvino  Pictures  Public
rootOnNVMe  run_docker.sh  sensors  snap  temp  Templates  test.txt  Videos
```

4.3.1.2、Copy files from the host to the container

```
# command
docker cp host_file_path docker_id:the path within the docker

#test
jetson@ubuntu:~$ docker ps -a
CONTAINER ID   IMAGE           COMMAND       CREATED       STATUS
PORTS     NAMES
c54bf9efae47   ubuntu:latest   "/bin/bash"   2 hours ago   Up 5 minutes
          funny_hugle
3b9c01839579   hello-world     "/hello"      3 hours ago   Exited (0) 3 hours ago
          jovial_brown
jetson@ubuntu:~$ ls
Desktop  Documents  Downloads  fishros  Music  openvino  Pictures  Public
rootOnNVMe  run_docker.sh  sensors  snap  temp  Templates  test.txt  Videos
jetson@ubuntu:~$ touch 11.txt
```

```
jetson@ubuntu:~$ ls
11.txt  Desktop  Documents  Downloads  fishros  Music  openvino  Pictures  Public
rootOnNVMe  run_docker.sh  sensors  snap  temp  Templates  test.txt  Videos
jetson@ubuntu:~$ docker cp 11.txt c54bf9efae47:/root/
jetson@ubuntu:~$ docker attach c5
root@c54bf9efae47:/# ls
bin  boot  dev  etc  home  lib  media  mnt  opt  proc  root  run  sbin  srv  sys
tmp  usr  var
root@c54bf9efae47:/# cd /root/
root@c54bf9efae47:~# ls     # 11.txt The file has been copied in
11.txt  test.txt
```

4.3.2、Using Data Volumes

4.3.2.1、Data Volume Overview

Package the application and the running environment to form a container to run, and the run can be accompanied by the container, but our data requirements are expected to be persistent! For example, if you install a mysql and you delete the container, it is equivalent to deleting the library and running away, which is definitely not okay! So we hope that it is possible to share data between containers, and the data generated by docker containers, if a new image is not generated through docker commit, so that the data is saved as part of the image, then when the container is deleted, the data will naturally be gone! This will not work!

In order to save data, we can use volumes in Docker! Let the data be mounted locally to us! So that the data is not lost due to container deletion!

Peculiarity:

1. Data volumes can share or reuse data between containers
2. Changes in the volume can take effect directly
3. Changes in the data volume will not be included in the update of the image
4. The lifecycle of a data volume lasts until no container uses it

4.3.2.2、Data volume use

```
# command
docker run -it -v host absolute path directory: directory inside the container image
name

# Test
docker run -it -v /home/jetson/temp:/root/temp yahboomtechnology/ros-foxy:3.4.0
/bin/bash

The /home/jetson/temp directory in the host and the /root/temp directory in the
container can share data
```