4.Hardware interaction and data processing

4.1. Hardware mounting (port binding)

1. Establish udev rules (/etc/udev/rules.d/) in the host machine, see chapter [6. Linux operating system ---- 6. Bind device ID]
2. Then when opening the container, mount the devices with the rules set into the docker container through parameters such as --device=/dev/myserial --device=/dev/rplidar and so on.

```
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   # There are already myserial and rplidar in
docker
console  fd  full  mqueue  myserial  null  ptmx  pts  random  rplidar  shm
stderr  stdin  stdout  tty  urandom  zero
```

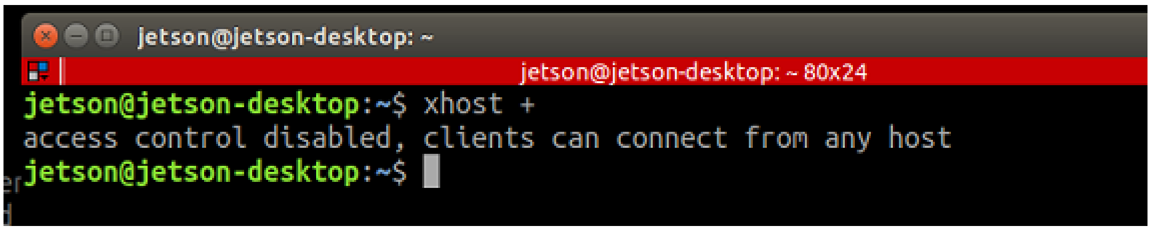
4.2. Display of GUI in docker

1. Install on the host machine:

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

2. Execute in the host machine: xhost +

After the following picture is displayed normally, perform 3 steps:



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

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

```
docker run -it \                                    # Run the docker image
interactively
--env="DISPLAY" \                                   # Turn on the display GUI
interface
--env="QT_X11_NO_MITSHM=1" \                        # Use X11 port 1 for display
-v /tmp/.X11-unix:/tmp/.X11-unix \                  # Map the service node
directory
yahboomtechnology/ros-foxy:3.3.9                    # Image name to be started
/bin/bash                                           # Execute the /bin/bash
command within the container
```

4. Test

```
Execute in container: rviz2
```

4.3. Transfer files between docker container and host machine

4.3.1. Use cp naming

4.3.1.1. Copy files from the container to the host

```
# command
docker cp container id: path within the container destination host path

# test
# Execute within the container and 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    #Press ctrl+P+Q to exit the
container without stopping.
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 container id: path within the container

#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 file has been copied in
11.txt  test.txt
```

4.3.2. Use data volume

4.3.2.1.Overview of data volumes

The application and running environment are packaged to form a container for operation. The operation can be accompanied by the container, but our requirements for data are that we hope to be persistent! It's like if you install a mysql and delete the container, it's equivalent to deleting the database and running away. This is definitely not okay! So we hope that it is possible to share data between containers. If the data generated by the docker container does not generate a new image 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 won't work!

In order to save data in docker we can use volumes! Let the data be mounted locally! This way the data will not be lost due to deletion of the container!

Features:

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

4.3.2.2. Data volume usage

```
# command
docker run -it -v Host absolute path directory: directory in 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, or they can be changed to other directories you want to
share.
```