Docker Startup Script

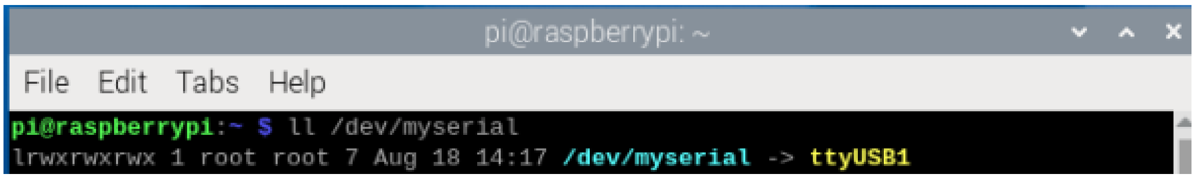
1. Check Bound Devices

The A1 robot has three bound devices. After booting successfully, enter the following command to check if the binding is successful.

- Checking the ROS expansion board binding status

```
ll /dev/myserial
```

If the following image appears, the binding is successful.

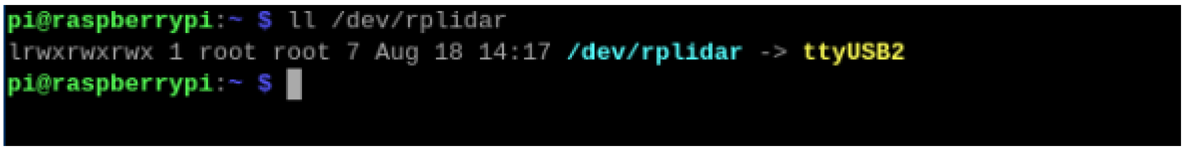


```
pi@raspberrypi:~ $ ll /dev/myserial
lrwxrwxrwx 1 root root 7 Aug 18 14:17 /dev/myserial -> ttyUSB1
```

- Checking the radar binding status

```
ll /dev/rplidar
```

If the following image appears, the binding is successful.

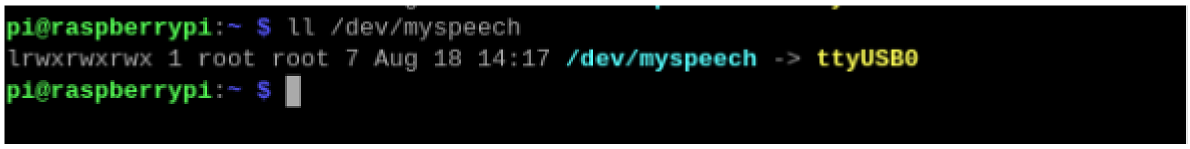


```
pi@raspberrypi:~ $ ll /dev/rplidar
lrwxrwxrwx 1 root root 7 Aug 18 14:17 /dev/rplidar -> ttyUSB2
pi@raspberrypi:~ $ 
```

- Checking the AI voice broadcast module binding status

```
ll /dev/myspeech
```

If the following image appears, the binding is successful.

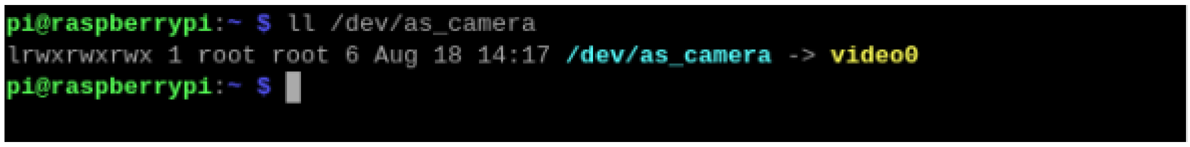


```
pi@raspberrypi:~ $ ll /dev/myspeech
lrwxrwxrwx 1 root root 7 Aug 18 14:17 /dev/myspeech -> ttyUSB0
pi@raspberrypi:~ $ 
```

If you purchased a depth camera, you can also enter the following command to check whether the depth camera is successfully bound.

```
ll /dev/as_camera
```

If the following image appears, the binding is successful.



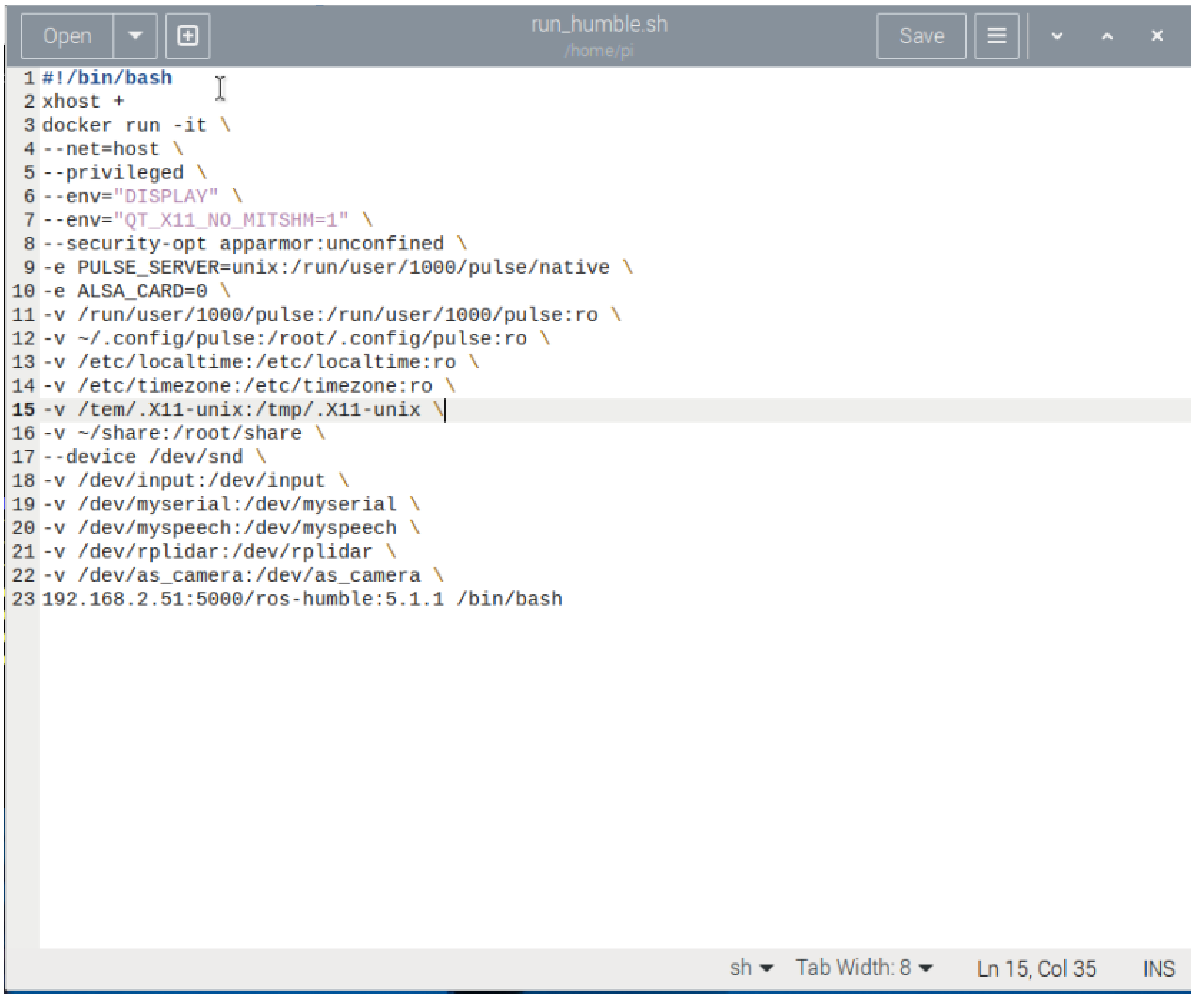
```
pi@raspberrypi:~ $ ll /dev/as_camera
lrwxrwxrwx 1 root root 6 Aug 18 14:17 /dev/as_camera -> video0
pi@raspberrypi:~ $ 
```

2. Check the Script Definition

On the Raspberry Pi or Jetson Nano host, enter the following command to open the Docker script.

```
sudo gedit run_humble.sh
```

Press Enter to see the script contents.



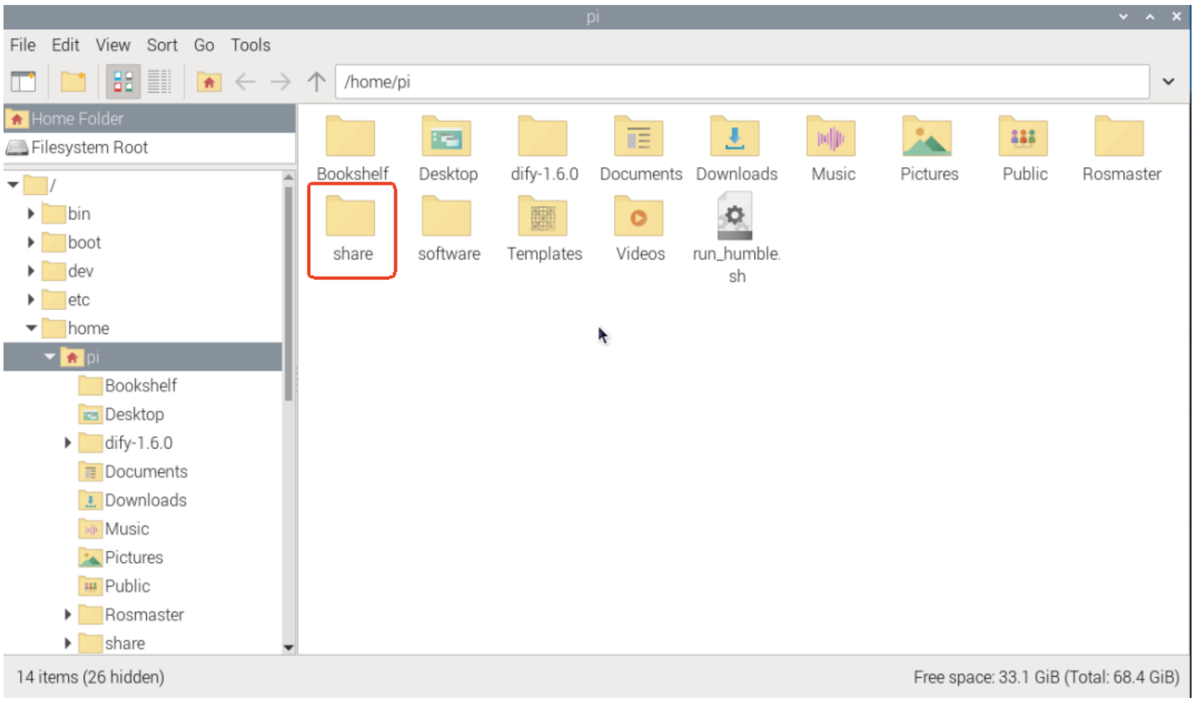
```
#!/bin/bash
xhost +
docker run -it \
--net=host \
--privileged \
--env="DISPLAY" \
--env="QT_X11_NO_MITSHM=1" \
--security-opt apparmor:unconfined \
-e PULSE_SERVER=unix:/run/user/1000/pulse/native \
-e ALSA_CARD=0 \
-v /run/user/1000/pulse:/run/user/1000/pulse:ro \
-v ~/.config/pulse:/root/.config/pulse:ro \
-v /etc/localtime:/etc/localtime:ro \
-v /etc/timezone:/etc/timezone:ro \
-v /tem/.X11-unix:/tmp/.X11-unix \
-v ~/share:/root/share \
--device /dev/snd \
-v /dev/input:/dev/input \
-v /dev/myserial:/dev/myserial \
-v /dev/myspeech:/dev/myspeech \
-v /dev/rplidar:/dev/rplidar \
-v /dev/as_camera:/dev/as_camera \
192.168.2.51:5000/ros-humble:5.1.1 /bin/bash
```

Below is a brief description of each line of the Docker script. Its main function is to open a Docker container and map the hardware devices connected to the Raspberry Pi to the Docker container. This allows Docker to freely access peripherals connected to the Raspberry Pi's interface.

```
#!/bin/bash
xhost + # Allow the container to access the host's X11 display service, thus
providing visualization capabilities for Docker.
docker run -it \
--net=host \ # Container uses the host's network stack directly.
--privileged \ # Access all hardware devices.
--env="DISPLAY" \ # Pass the visualization environment variable.
--env="QT_X11_NO_MITSHM=1" \
--security-opt apparmor:unconfined \ # Disable security policy when
visualization is disabled.
-e PULSE_SERVER=unix:/run/user/1000/pulse/native \ # PulseAudio service path
-e ALSA_CARD=0 \
-v /run/user/1000/pulse:/run/user/1000/pulse:ro \ # Mount the audio socket read-
only
-v ~/.config/pulse:/root/.config/pulse:ro \
-v /etc/localtime:/etc/localtime:ro \ # Synchronize local time
```

```
-v /etc/timezone:/etc/timezone:ro \
-v /tem/.X11-unix:/tmp/.X11-unix \
-v ~/share:/root/share \ # Shared folder between the host and Docker
--device /dev/snd \ # Access the host sound card
-v /dev/input:/dev/input \ #Mount all input devices (mouse/keyboard)
-v /dev/myserial:/dev/myserial \ #Mount the ROS expansion board
-v /dev/myspeech:/dev/myspeech \ #Mount the AI module serial port
-v /dev/rplidar:/dev/rplidar \ #Mount the radar
-v /dev/as_camera:/dev/as_camera \ #Mount the depth camera
192.168.2.51:5000/ros-humble-a1:1.0.6 /bin/bash #Select the Docker name to run
```

Note that the mounted peripherals are recognized properly and that the Docker container you want to run exists. A shared folder will be created on the host machine. You can store files or photos you want to transfer to the Docker container in this folder.

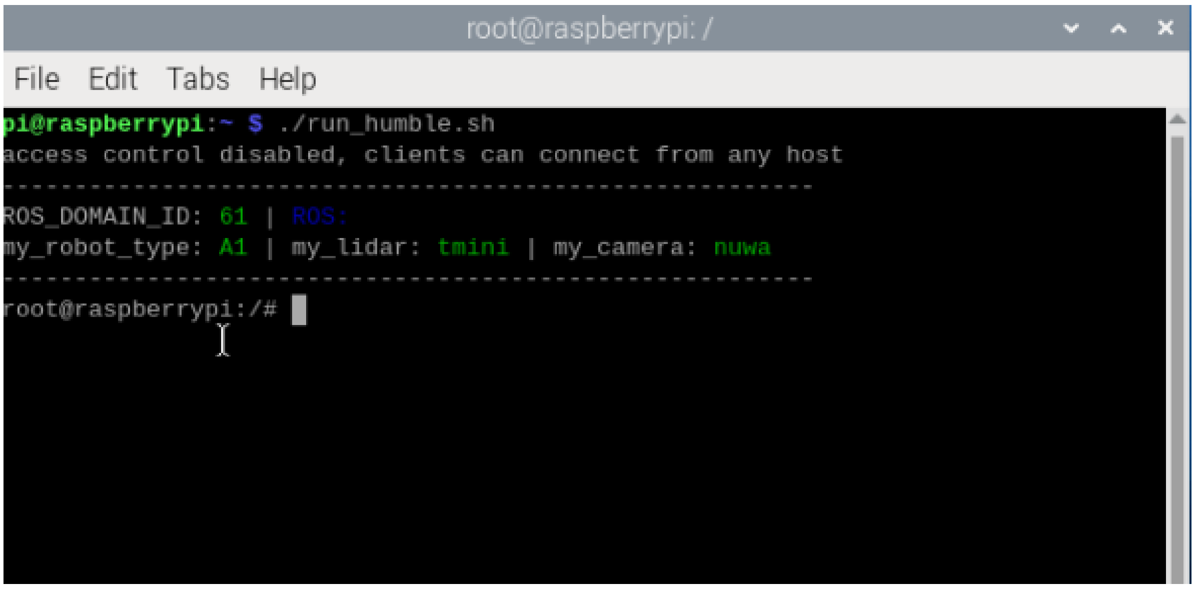


3. Run the Docker Script

When using the IP address on the OLED screen to access VNC remotely, you can open a terminal and run the following command to enter Docker:

```
./run_humble.sh
```

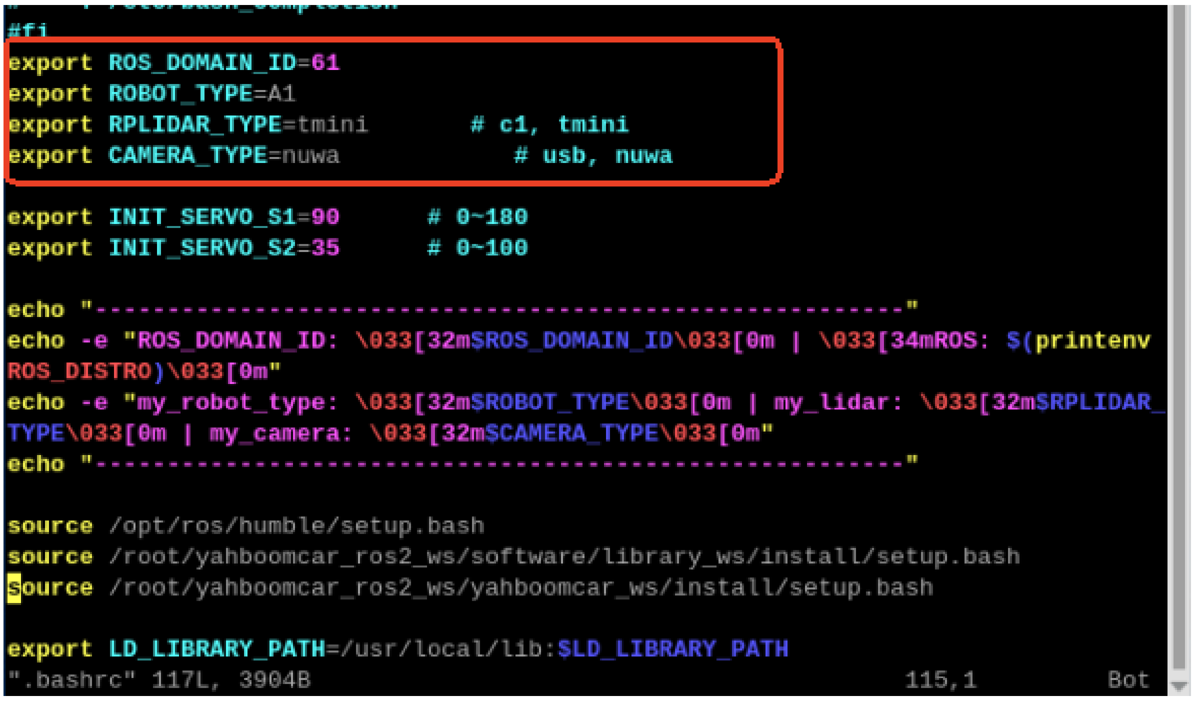
Successful:



```
pi@raspberrypi:~ $ ./run_humble.sh
access control disabled, clients can connect from any host
--------------------------------------------------------
ROS_DOMAIN_ID: 61 | ROS:
my_robot_type: A1 | my_lidar: tmini | my_camera: nuwa
--------------------------------------------------------
root@raspberrypi:/#
```

Since the robot has two cameras and two different radar models, the factory system has routines configured for multiple devices. However, since it cannot automatically identify the product, you need to manually set the machine type and radar model. Enter in the terminal:

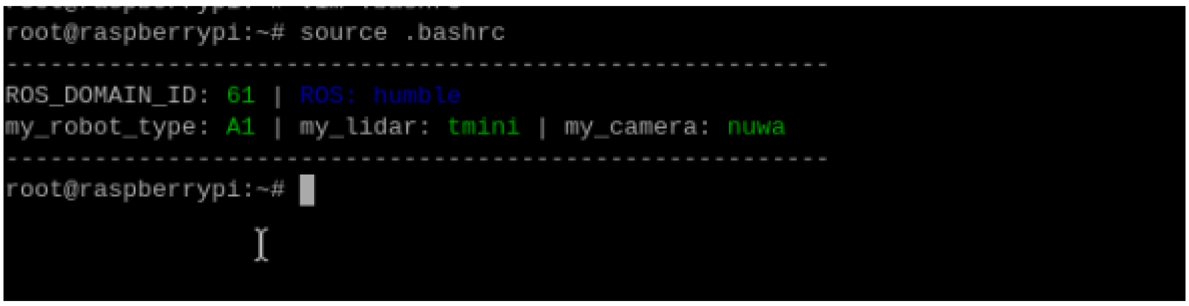
```
root@raspberrypi:/# cd
root@raspberrypi:~# vim .bashrc
```



```
#fi
export ROS_DOMAIN_ID=61
export ROBOT_TYPE=A1
export RPLIDAR_TYPE=tmini       # c1, tmini
export CAMERA_TYPE=nuwa            # usb, nuwa

export INIT_SERVO_S1=90      # 0~180
export INIT_SERVO_S2=35      # 0~100

echo "--------------------------------------------------------"
echo -e "ROS_DOMAIN_ID: \033[32m$ROS_DOMAIN_ID\033[0m | \033[34mROS: $(printenv ROS_DISTRO)\033[0m"
echo -e "my_robot_type: \033[32m$ROBOT_TYPE\033[0m | my_lidar: \033[32m$RPLIDAR_TYPE\033[0m | my_camera: \033[32m$CAMERA_TYPE\033[0m"
echo "--------------------------------------------------------"

source /opt/ros/humble/setup.bash
source /root/yahboomcar_ros2_ws/software/library_ws/install/setup.bash
source /root/yahboomcar_ros2_ws/yahboomcar_ws/install/setup.bash

export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
".bashrc" 117L, 3904B                                     115,1         Bot
```

After modifying your peripherals, save and exit vim, then execute:

```
source .bashrc
```

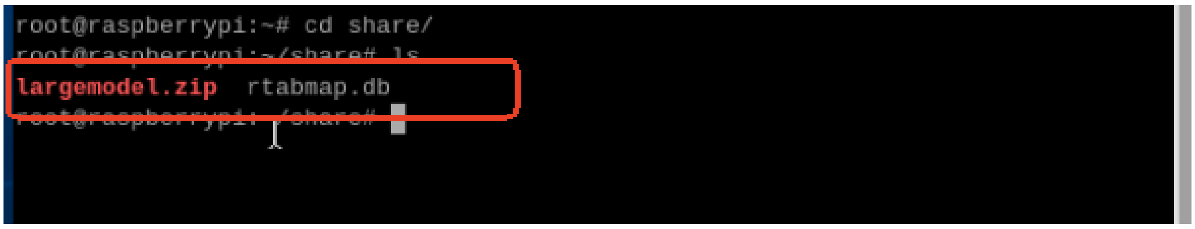


```
root@raspberrypi:~# source .bashrc
--------------------------------------------------------
ROS_DOMAIN_ID: 61 | ROS: humble
my_robot_type: A1 | my_lidar: tmini | my_camera: nuwa
--------------------------------------------------------
root@raspberrypi:~#
```

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

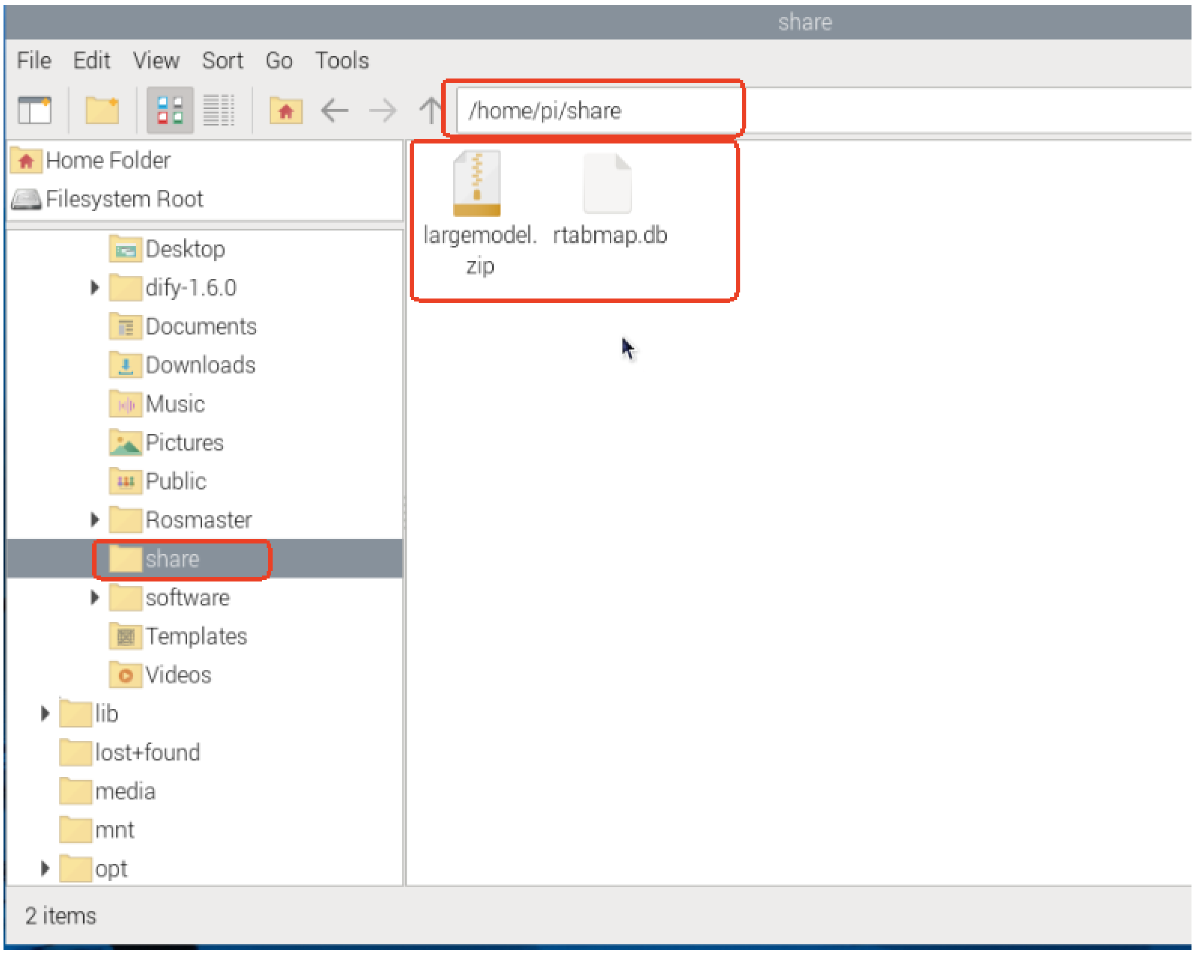
To view the shared folder, enter the following command:

```
root@raspberrypi:~# cd share/
root@raspberrypi:~/share# ls
```



```
root@raspberrypi:~# cd share/
root@raspberrypi:~/share# ls
largemodel.zip  rtabmap.db
root@raspberrypi:~/share#
```

We can see a .db file and a compressed archive. The same files are also present on the Raspberry Pi.



This principle can be used to transfer files between the host machine and Docker.