Enter docker container

5.1, related concepts

1. What is the host of docker

```
The host is the server where we call commands to create containers using images.
Here it refers to the main control Raspberry Pi on our car, and the host
mentioned below refers to this.
```

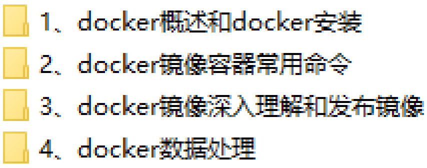
2. What is GUI

```
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 raspbot v2 car, that is, the raspbot v2 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, and you will feel very relaxed after mastering it. Come on, you are the best!

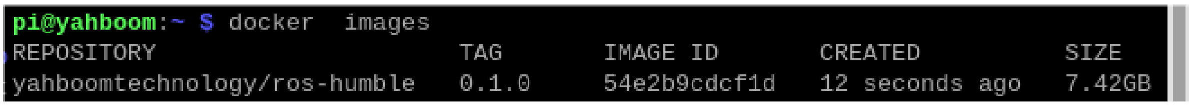


1、docker概述和docker安装
2、docker镜像容器常用命令
3、docker镜像深入理解和发布镜像
4、docker数据处理

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:

```
docker images
```



```
pi@yahboom:~ $ docker  images
REPOSITORY                        TAG       IMAGE ID       CREATED          SIZE
yahboomtechnology/ros-humble      0.1.0     54e2b9cdcf1d   12 seconds ago   7.42GB
```

Please select the name [yahboomtechnologyros-humble]. The version with the highest tag is the latest image version of the robot

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

If you have read the tutorial in [13, 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.

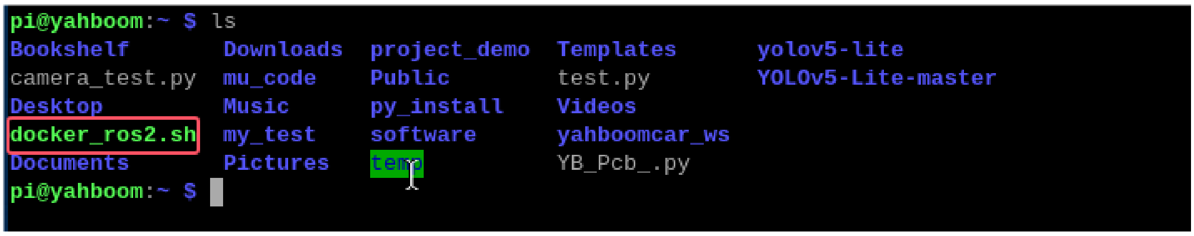
Later, when we update new courses, we will also update functions by releasing new docker images.

5.3. Edit scripts

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

1. The script to run docker [docker_ros2.sh] is generally placed in the root directory of the car's home 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 docker_ros2.sh #Give the script executable permissions
```



```
pi@yahboom:~ $ ls
Bookshelf       Downloads   project_demo   Templates       yolov5-lite
camera_test.py  mu_code     Public         test.py         YOLOv5-Lite-master
Desktop         Music       py_install     Videos
docker_ros2.sh  my_test     software       yahboomcar_ws
Documents       Pictures    temp           YB_Pcb_.py
pi@yahboom:~ $
```

The contents of the [docker_ros2.sh] script are as follows:

You can copy it directly without comments and modify it as needed

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

```
#!/bin/bash
xhost +
docker run -it \
--privileged=true \
--net=host \
--env="DISPLAY" \
--env="QT_X11_NO_MITSHM=1" \
-v /tmp/.X11-unix:/tmp/.X11-unix \
--security-opt apparmor:unconfined \
-v /home/pi/temp:/root/temp \
-v /dev/i2c-1:/dev/i2c-1 \
-v /dev/i2c-0:/dev/i2c-0 \
--device=/dev/video0 \
--device=/dev/video1 \
--device=/dev/gpiomem \
yahboomtechnology/ros-humble:0.1.0 /bin/bash
```

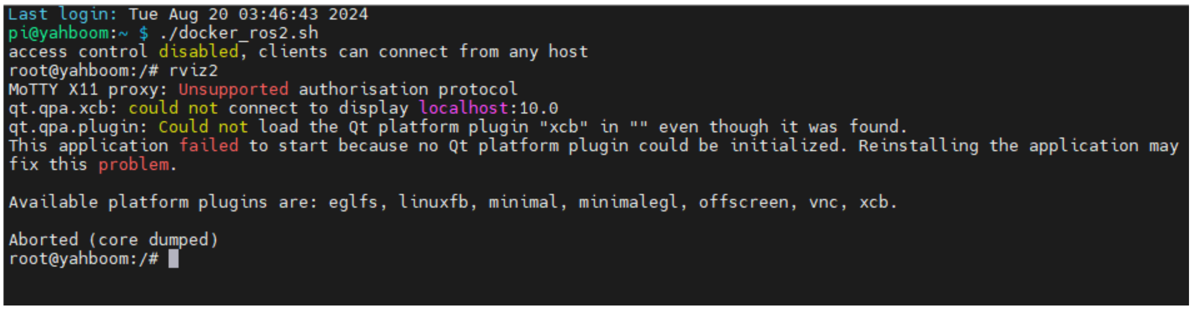
Annotated script description:

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
--net=host \                                                              # set the
container network to host mode
--env="DISPLAY" \                                                         # open 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
display service node directory
--security-opt apparmor:unconfined \
-v /home/pi/temp:/root/temp \                                             # as a
temporary file transfer directory between the host and the container, if you need
to transfer files, you can use this directory
-v /dev/i2c-1:/dev/i2c-1 \                                                # add the
host device to the container, i2c device port, if the car is not connected,
please remove this line
-v /dev/i2c-0:/dev/i2c-0 \                                                #Add host
device to container, i2c device port, if the car is not connected, please remove
this line
--device=/dev/video0 \                                                    #Add host
device to container, camera device port, if the car is not connected to the
camera, please remove this line
--device=/dev/video1 \                                                    #Add host
device to container, camera device port, if the car is not connected to the
camera, please remove this line
--device=/dev/gpiomem \                                                   #Add host
device to container, Raspberry Pi gpio device port
yahboomtechnology/ros-humble:0.1.0 /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
```

5.4, Execute the script

Note: This must be executed on the VNC of the car or on the car screen. It 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

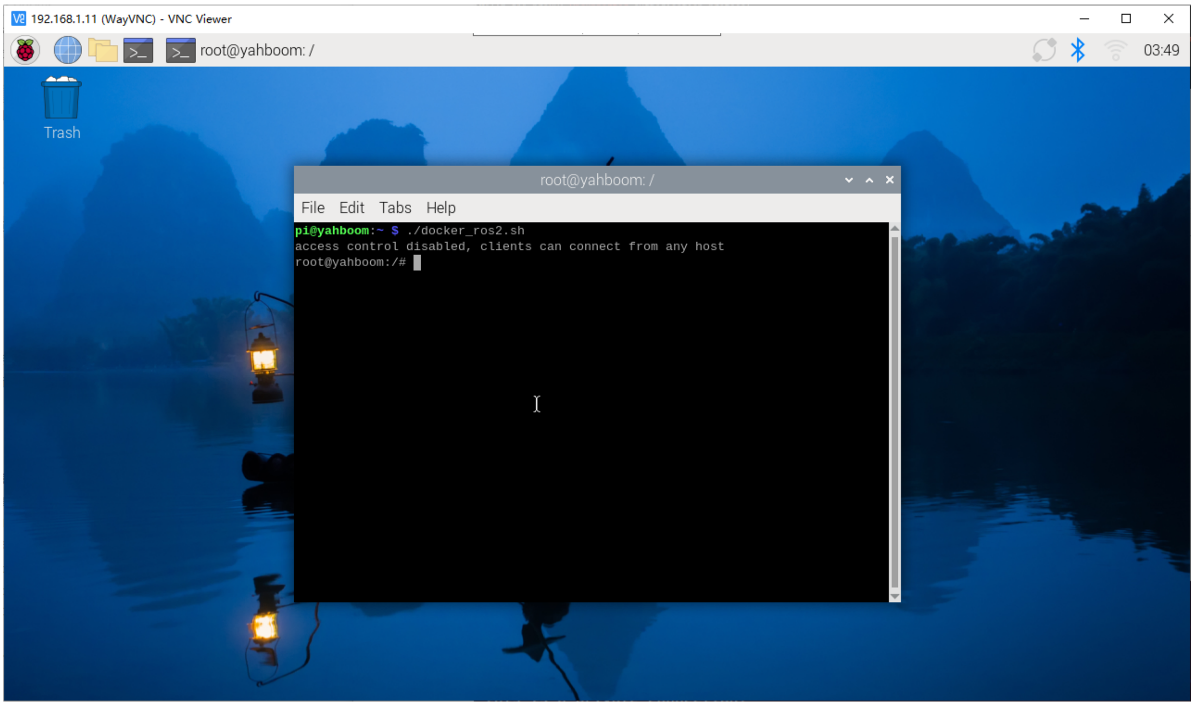


```
Last login: Tue Aug 20 03:46:43 2024
pi@yahboom:~ $ ./docker_ros2.sh
access control disabled, clients can connect from any host
root@yahboom:/# rviz2
MoTTY X11 proxy: Unsupported authorisation protocol
qt.qpa.xcb: could not connect to display localhost:10.0
qt.qpa.plugin: Could not load the Qt platform 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)
root@yahboom:/# 
```

Execute on the VNC interface of the car or on the car screen:

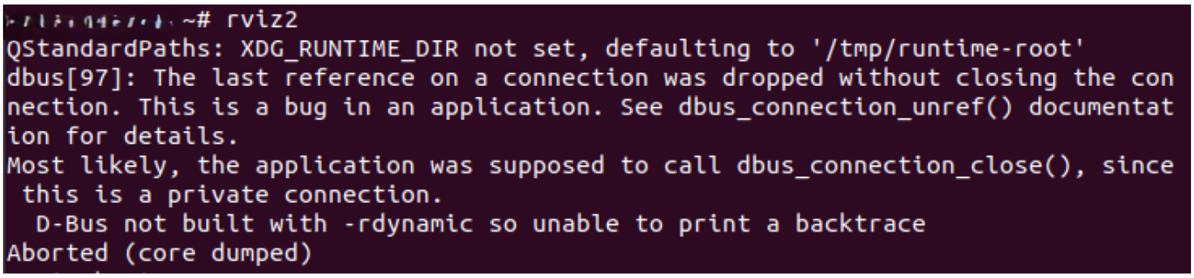
```
./docker_ros2.sh
```

You can enter the container correctly and display the GUI screen. You can execute the rviz2 command again for testing.



```
pi@yahboom:~ $ ./docker_ros2.sh
access control disabled, clients can connect from any host
root@yahboom:/# 
```

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



```
[illegible]~# 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
```

That is:

```
#!/bin/bash
xhost +
docker run -it \
--privileged=true \
--net=host \
--env="DISPLAY" \
--env="QT_X11_NO_MITSHM=1" \
-v /tmp/.X11-unix:/tmp/.X11-unix \
--security-opt apparmor:unconfined \ #Parameters for this sentence
-v /home/pi/temp:/root/temp \
-v /dev/i2c-1:/dev/i2c-1 \
-v /dev/i2c-0:/dev/i2c-0 \
--device=/dev/video0 \
--device=/dev/video1 \
--device=/dev/gpiomem \
yahboomtechnology/ros-humble:0.1.0 /bin/bash
```

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

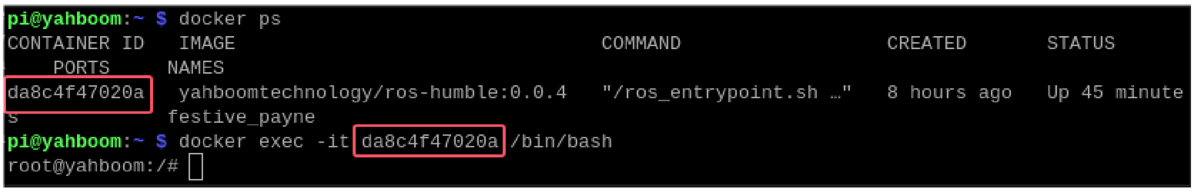
5.5. multiple terminals enter the same docker container

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
```

2. Now open a new terminal and enter the same docker. Change the following da8c4f47020a to the ID displayed in the actual terminal

```
docker exec -it da8c4f47020a /bin/bash
```



```
pi@yahboom:~ $ docker ps
CONTAINER ID   IMAGE                                COMMAND                   CREATED         STATUS
    PORTS     NAMES
da8c4f47020a   yahboomtechnology/ros-humble:0.0.4   "/ros_entrypoint.sh …"   8 hours ago     Up 45 minute
s             festive_payne
pi@yahboom:~ $ docker exec -it da8c4f47020a /bin/bash
root@yahboom:/# 
```

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
```

5.6, how to open a container that is already in the [Exited] closed state

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

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. 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 check whether the GUI screen can be opened

```
rviz2
```

6. The specific execution is as follows:

```
pi@yahboom:~$ docker ps -a
CONTAINER ID   IMAGE                              COMMAND       CREATED
STATUS                     PORTS     NAMES
5b698ea10535   yahboomtechnology/ros-foxy:3.3.9   "/bin/bash"   3 days ago
Exited (0) 8 seconds ago             ecstatic_lewin
pi@yahboom:~$ xhost +
access control disabled, clients can connect from any host
pi@yahboom:~$ docker start 5b
5b
pi@yahboom:~$ docker exec -it 5b /bin/bash
root@yahbooom:/# rviz2
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
[INFO] [1682298616.634096279] [rviz2]: Stereo is NOT SUPPORTED
[INFO] [1682298616.634576375] [rviz2]: OpenGl version: 3.1 (GLSL 1.4)
[INFO] [1682298617.959654036] [rviz2]: Stereo is NOT SUPPORTED
```