

Viewing camera on ROS2

Viewing camera on ROS2

0. Install Docker on a Linux system
1. Start ROS Wifi image transmission module ROS2 agent
 - Publishing nodes
 - Topic information
 - Message type
 - Topic posting frequency
2. ROS2 obtains camera image
3. View images in RVIZ
4. matters needing attention

0. Install Docker on a Linux system

```
# **If using a system provided by the data, skip this step**  
sudo apt update  
sudo apt install docker.io docker-compose  
docker -v
```

Download and install Docker as shown in the picture

```
yahboom@yahboom-VM:~$ docker -v  
Docker version 24.0.7, build afdd53b
```

1. Start ROS Wifi image transmission module ROS2 agent

1. First, start the STA mode of the ROS Wifi image transmission module (default start) and allow ESP32 to connect to an internet enabled WiFi in the local environment.
Note: When using AP mode, you cannot enter the proxy because entering the proxy requires a network, and AP mode does not have a network
2. The proxy host also needs to be connected to the same local area network as the ROS Wifi image transmission module
Proxy host: It is a Linux system with ROS2 (Humble version) installed
3. Query the IP address of the proxy host
Enter the command ifconfig

```

yahboom@yahboom-VM:~$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:93:f5:44:96 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.100 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::eeb4:aaad:2fb9:8c7 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:df:42:3c txqueuelen 1000 (Ethernet)
    RX packets 1160 bytes 839024 (839.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 651 bytes 263777 (263.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

4. Then use a serial port tool to configure the ROS Wifi image transmission module's ROS2 proxy host address

- For example, in the above figure, my IP address is **192.168.2.100**, and the configuration of the serial port is shown in the figure



- After clicking send, there are two types of information to print, **both of which are configured**
 - If the original proxy IP of the ROS Wifi image transmission module is different from the current one, the current proxy IP will be updated and restarted

```
[2024-03-29 14:45:45.751]# RECV ASCII>
ESP-ROM: esp32s3-20210327
Build:Mar 27 2021
rst:0x3 (RTC_SW_SYS_RST),boot:0x8 (SPI_FAST_FLASH_BOOT)
Saved PC:0x40375a0c
SPIWP:0xee
mode:DIO, clock div:1
load:0x3fce3818,len:0x514
load:0x403c9700,len:0x4
load:0x403c9704,len:0xb58
load:0x403cc700,len:0x2a8c
entry 0x403c98fc
```

```
[2024-03-29 14:45:45.937]# RECV ASCII>
[0:32mI (196) cpu_start: Multicore app[0m
[0:32mI (197) esp_psram: Found 2MB PSRAM device[0m
[0:32mI (197) esp_psram: Speed: 80MHz[0m
[0:32mI (198) cpu_start: Pro cpu up. [0m
[0:32mI (201) cpu_start: Starting app cpu, €
[2024-03-29 14:45:49.631]# RECV ASCII>
YAHBOOM VerSion:V2.0.5
```

数据发送 | 1. DCD 2. RXD 3. TXD 4. DTR 5. GND 6. 清除

ros2_ip:192.168.2.100

2. If the original proxy IP of the ROS Wifi image transmission module is the same as now, the ROS Wifi image transmission module will return an OK message



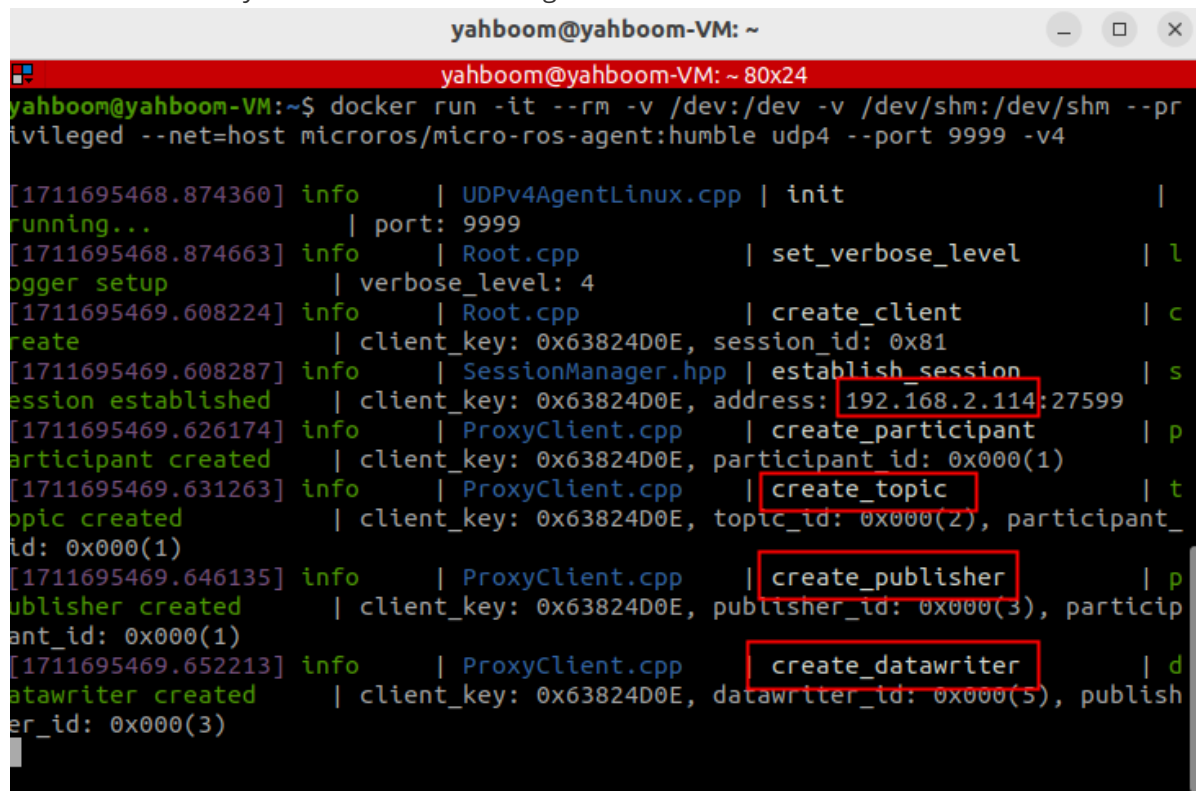
5. After completing the proxy configuration of the ROS Wifi image transmission module, execute the following command on the proxy host

```
# If the virtual machine using the data runs the command directly
sh start_Camera_computer.sh
```

```
# Not a virtual machine provided by the data
docker run -it --rm -v /dev:/dev -v /dev/shm:/dev/shm --privileged --net=host
microros/micro-ros-agent:humble udp4 --port 9999 -v4
```

- This 9999 port will be used to enter the proxy, so I need to avoid using this port in my future development

The successful entry result is shown in the figure:



```
yahboom@yahboom-VM: ~
yahboom@yahboom-VM: ~ 80x24
yahboom@yahboom-VM:~$ docker run -it --rm -v /dev:/dev -v /dev/shm:/dev/shm --pr
ivileged --net=host microros/micro-ros-agent:humble udp4 --port 9999 -v4

[1711695468.874360] info      | UDPv4AgentLinux.cpp | init      |
running...          | port: 9999          |
[1711695468.874663] info      | Root.cpp            | set_verbose_level | l
logger setup        | verbose_level: 4    |
[1711695469.608224] info      | Root.cpp            | create_client    | c
create              | client_key: 0x63824D0E, session_id: 0x81 |
[1711695469.608287] info      | SessionManager.hpp  | establish_session | s
session established | client_key: 0x63824D0E, address: 192.168.2.114:27599 |
[1711695469.626174] info      | ProxyClient.cpp     | create_participant | p
participant created | client_key: 0x63824D0E, participant_id: 0x000(1) |
[1711695469.631263] info      | ProxyClient.cpp     | create_topic      | t
topic created       | client_key: 0x63824D0E, topic_id: 0x000(2), participant_
id: 0x000(1)
[1711695469.646135] info      | ProxyClient.cpp     | create_publisher   | p
publisher created   | client_key: 0x63824D0E, publisher_id: 0x000(3), particip
ant_id: 0x000(1)
[1711695469.652213] info      | ProxyClient.cpp     | create_datawriter  | d
datawriter created  | client_key: 0x63824D0E, datawriter_id: 0x000(5), publish
er_id: 0x000(3)
```

There must be so much information to correctly activate the agent. Missing one may require manually powering off and resetting the ROS Wifi image transmission module. According to the proxy results in the above figure, the IP of the ROS Wifi image transmission module can be obtained as 192.168.2.114

6. Query ROS2 nodes, topic information, topic publishing frequency, and topic message types published by the ROS Wifi image transmission module

Publishing nodes

```
yahboom@yahboom-VM:~$ ros2 node list
/espRos/Esp32Node
```

Topic information

```
yahboom@yahboom-VM:~$ ros2 topic list
/espRos/esp32camera
/parameter_events
/rosout
yahboom@yahboom-VM:~$
```

```
yahboom@yahboom-VM:~$ ros2 topic echo /espRos/esp32camera
header:
  stamp:
    sec: 1410
    nanosec: 165194
  frame_id: esp32_Camera
format: jpeg
data:
- 255
- 216
- 255
- 224
- 0
- 16
- 74
- 70
- 73
```

Message type

```
yahboom@yahboom-VM:~$ ros2 topic info /espRos/esp32camera
Type: sensor_msgs/msg/CompressedImage
Publisher count: 1
Subscription count: 0
```

Topic posting frequency

```

yahboom@yahboom-VM:~$ ros2 topic hz /espRos/esp32camera
average rate: 5.173
  min: 0.128s max: 0.377s std dev: 0.08695s window: 6
average rate: 5.896
  min: 0.117s max: 0.377s std dev: 0.06701s window: 13
average rate: 6.457
  min: 0.111s max: 0.377s std dev: 0.05652s window: 21
average rate: 6.582
  min: 0.111s max: 0.377s std dev: 0.05274s window: 28
average rate: 6.780
  min: 0.111s max: 0.377s std dev: 0.04775s window: 36
average rate: 6.918
  min: 0.111s max: 0.377s std dev: 0.04374s window: 44
average rate: 6.963
  min: 0.111s max: 0.377s std dev: 0.04227s window: 52
average rate: 7.070
  min: 0.111s max: 0.377s std dev: 0.03980s window: 60
average rate: 7.160
  min: 0.108s max: 0.377s std dev: 0.03802s window: 68
average rate: 7.213
  min: 0.108s max: 0.377s std dev: 0.03628s window: 76
average rate: 7.268
  min: 0.108s max: 0.377s std dev: 0.03495s window: 84

```

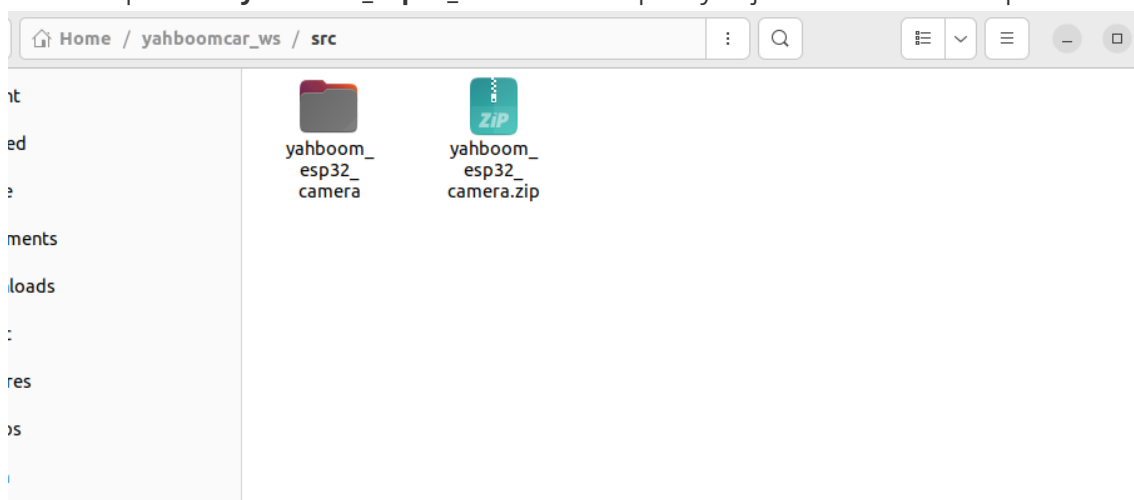
2.ROS2 obtains camera image

Note: If using a virtual machine provided in the data, only refer to step 4

1. First, create a folder and execute the following command

```
mkdir -p ~/yahboomcar_ws/src
```

2. Place the provided **yahboom_esp32_camera** in the path you just created and unzip it



3. Then compile the feature pack and add the environment

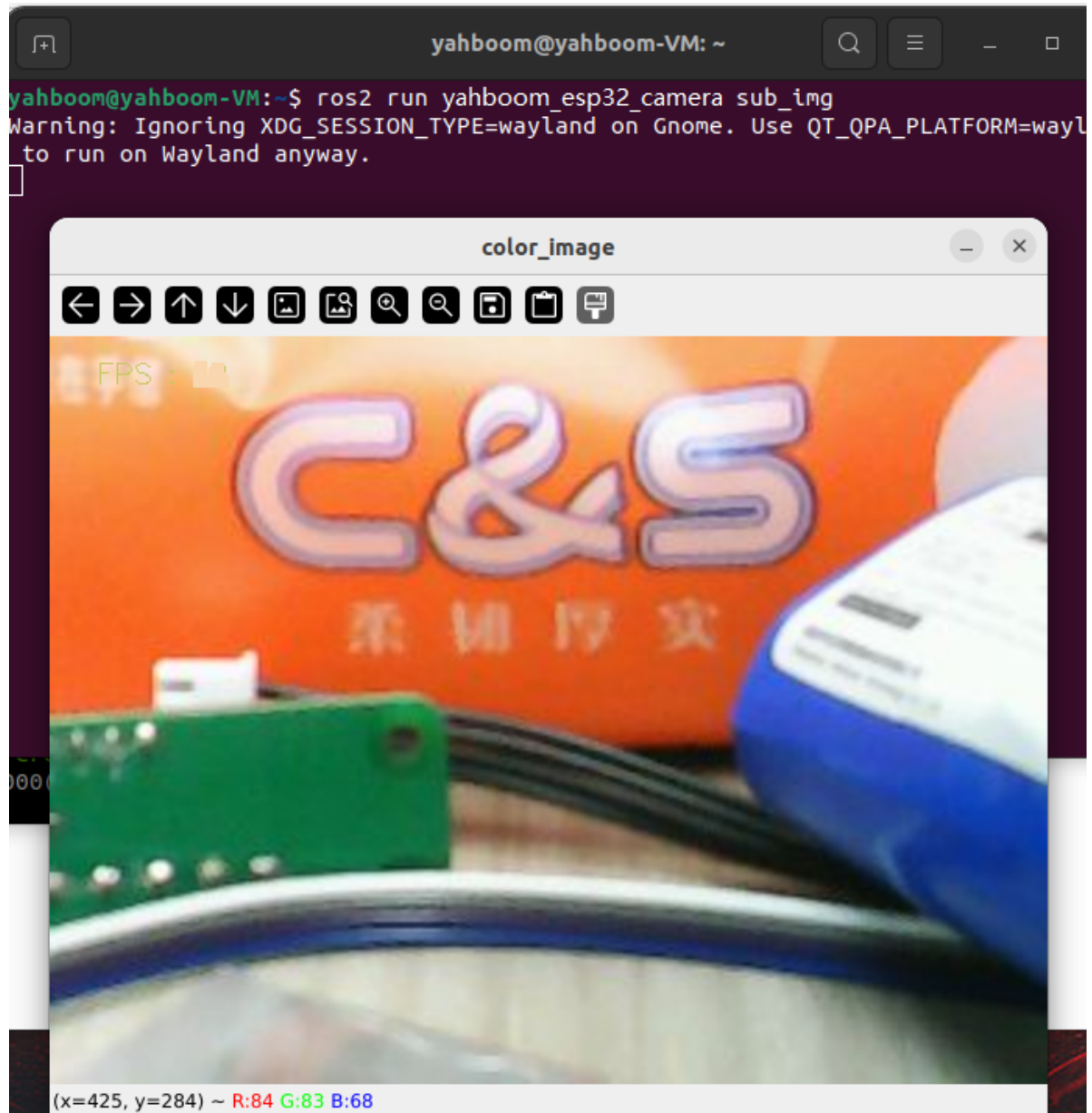
```

colcon build
echo "source ~/yahboomcar_ws/install/setup.bash " >> ~/.bashrc
source ~/.bashrc

```

4. Run the command to open the camera screen

```
ros2 run yahboom_esp32_camera sub_img
```

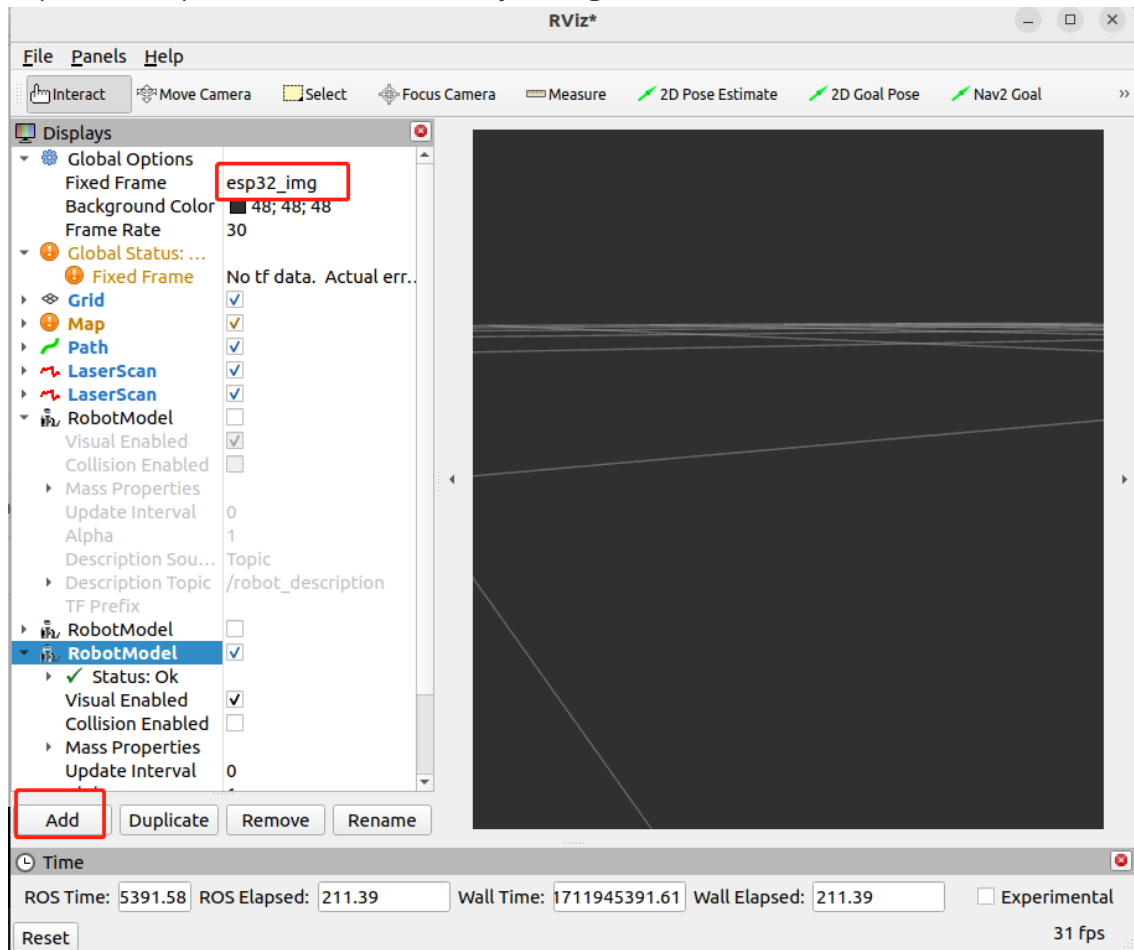


3.View images in RVIZ

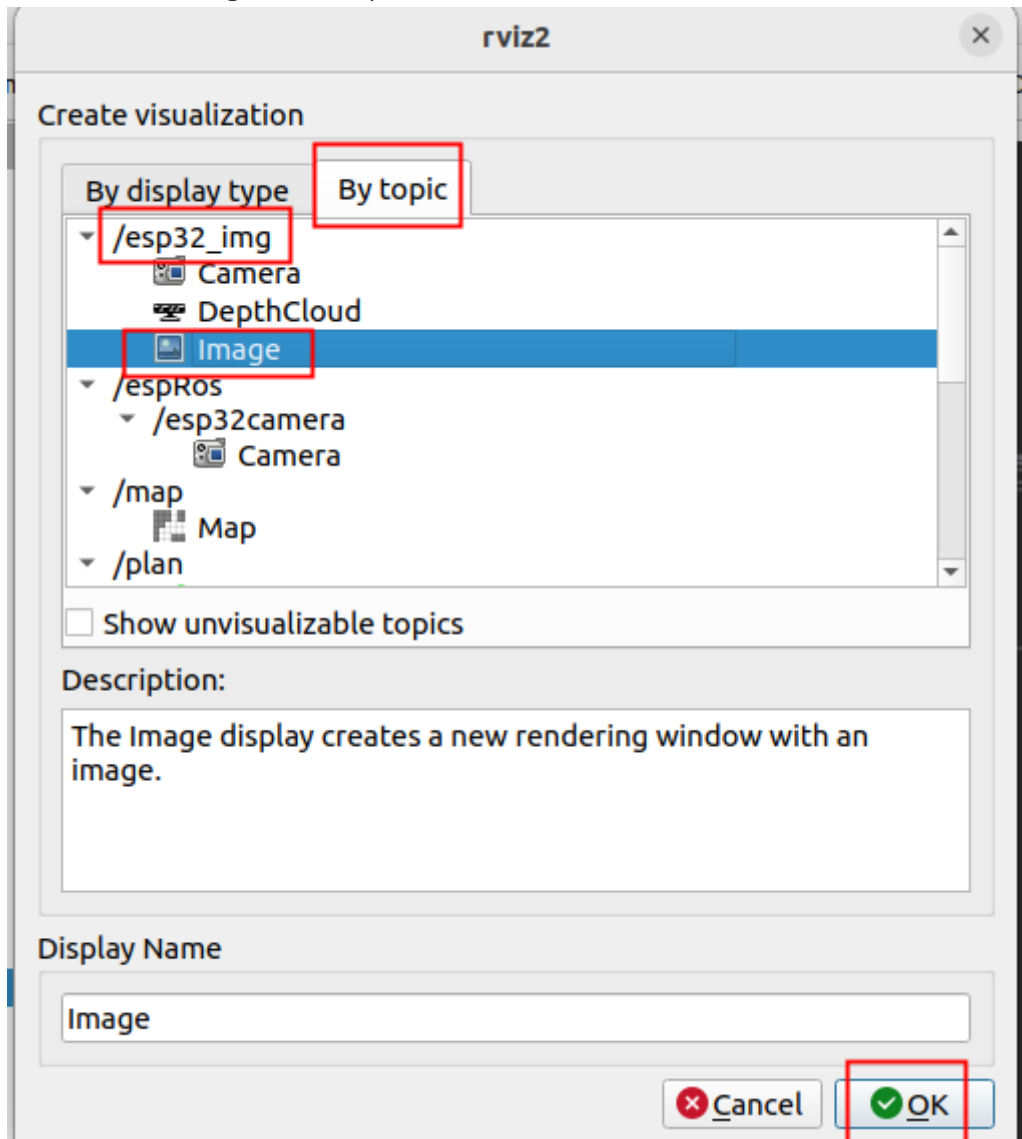
1. Under the premise that ROS2 obtains the camera image, open the installed rviz software

```
rviz2 #rviz/rviz2
```

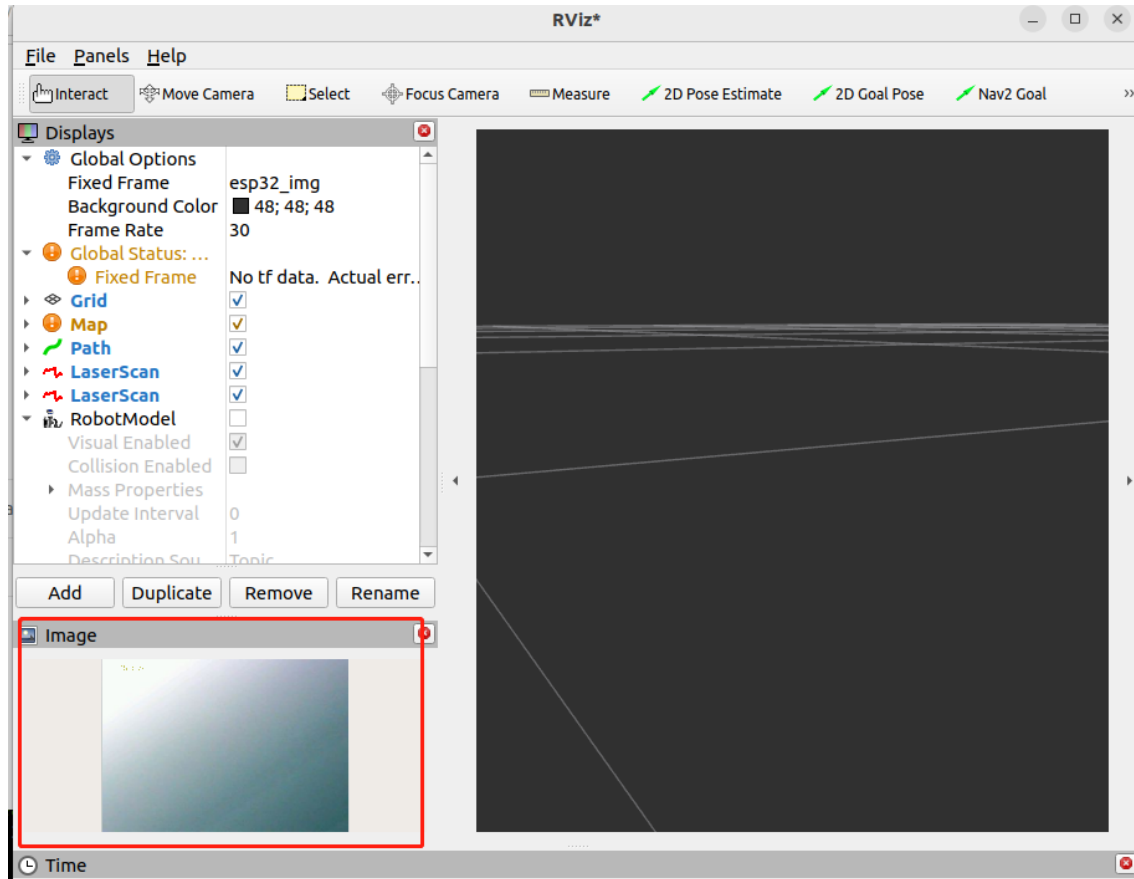
2. Replace the topic of fixed frame with `/esp32_img`



3. Then load the image of this topic



4. Finally, rviz displays the image of the camera



4.matters needing attention

1. If the host actively disconnects the agent, the ROS Wifi image transmission module needs to be manually powered off and restarted before it can be restored and connected to the agent
2. This port of 9999 will be used for proxy entry, so I should avoid using this port in my future development
3. The use of proxies can only be configured through STA (Local Area Network) mode.