

# Dabai\_DCW2 Camera

Before starting this function, you need to close the large program and APP processes. If you need to restart the large program and APP later, start them from the terminal:

```
bash ~/dofbot_pro/APP_DOFBOT_PRO/start_app.sh
```

## 1. Function Description

After starting the camera, you can obtain RGB color images, Depth images, and IR infrared images.

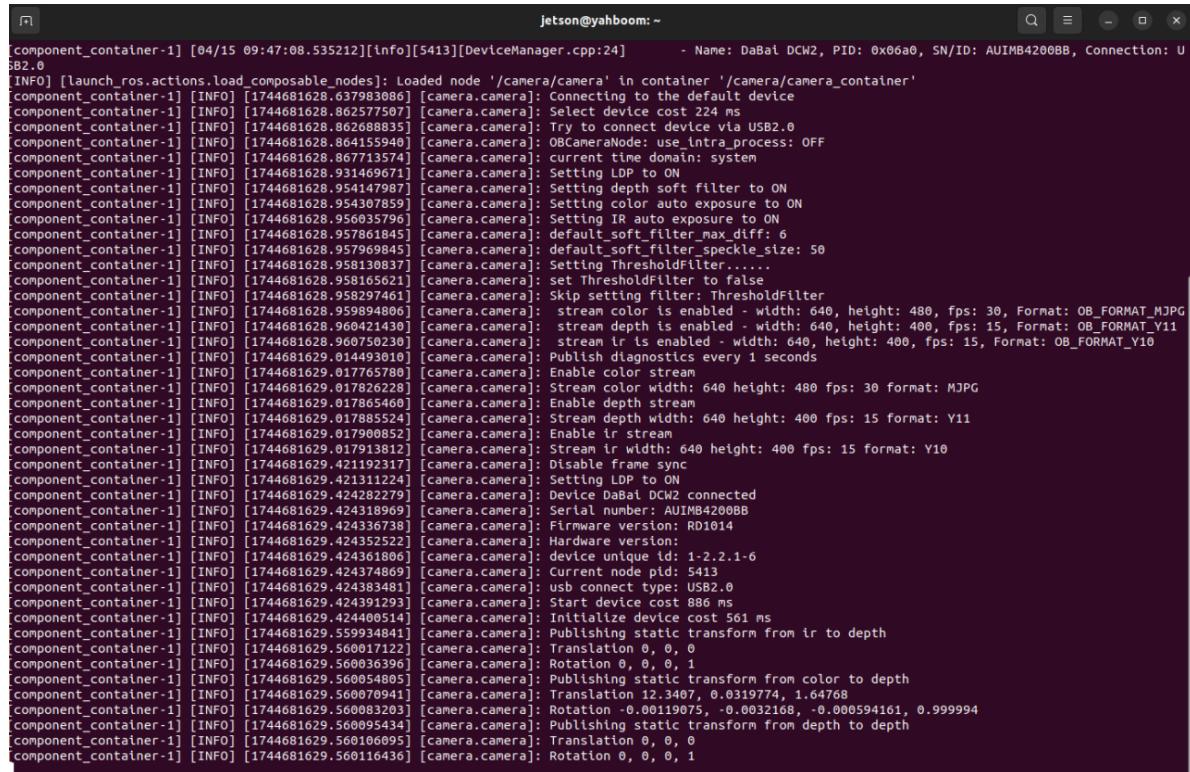
## 2. Startup and Operation

### 2.1. Startup

After connecting the camera to the main board via data cable, open the terminal and enter:

```
ros2 launch orbbec_camera dabai_dcw2.launch.py
```

As shown in the figure below, if the terminal prints the above content, it means the camera is running normally and started successfully. You can use the following command to view the topics published by the camera node. Enter in the terminal:



```
jetson@yahboom: ~
[component_container-1] [04/15 09:47:08.535212][INFO][5413][DeviceManager.cpp:24] - Name: DaBal DCW2, PID: 0x06a0, SN/ID: AUIMB4200BB, Connection: U
[component_container-1] [INFO] [1744681628.637983086] [camera.camera]: Connecting to the default device
[component_container-1] [INFO] [1744681628.862577507] [camera.camera]: Select device cost 224 ms
[component_container-1] [INFO] [1744681628.86268835] [camera.camera]: Try to connect device via USB2.0
[component_container-1] [INFO] [1744681628.864155940] [camera.camera]: OB Camera Node: use_intra_process: OFF
[component_container-1] [INFO] [1744681628.867713574] [camera.camera]: current time domain: system
[component_container-1] [INFO] [1744681628.931469671] [camera.camera]: Setting LDP to ON
[component_container-1] [INFO] [1744681628.954147987] [camera.camera]: Setting depth soft filter to ON
[component_container-1] [INFO] [1744681628.954307859] [camera.camera]: Setting color auto exposure to ON
[component_container-1] [INFO] [1744681628.956835796] [camera.camera]: Setting IR auto exposure to ON
[component_container-1] [INFO] [1744681628.957861845] [camera.camera]: default_soft_filter_max_diff: 6
[component_container-1] [INFO] [1744681628.957969845] [camera.camera]: default_soft_filter_speckle_size: 50
[component_container-1] [INFO] [1744681628.958130837] [camera.camera]: Setting ThresholdFilter.....
[component_container-1] [INFO] [1744681628.958165621] [camera.camera]: set ThresholdFilter to false
[component_container-1] [INFO] [1744681628.958297461] [camera.camera]: Skt setting filter: ThresholdFilter
[component_container-1] [INFO] [1744681628.959894806] [camera.camera]: stream color is enabled - width: 640, height: 480, fps: 30, Format: OB_FORMAT_MJPG
[component_container-1] [INFO] [1744681628.960421430] [camera.camera]: stream depth is enabled - width: 640, height: 400, fps: 15, Format: OB_FORMAT_Y11
[component_container-1] [INFO] [1744681628.960750230] [camera.camera]: stream ir is enabled - width: 640, height: 400, fps: 15, Format: OB_FORMAT_Y10
[component_container-1] [INFO] [1744681629.014493010] [camera.camera]: Publish diagnostics every 1 seconds
[component_container-1] [INFO] [1744681629.017765780] [camera.camera]: Enable color stream
[component_container-1] [INFO] [1744681629.017826228] [camera.camera]: Stream color width: 640 height: 480 fps: 30 format: MJPG
[component_container-1] [INFO] [1744681629.017865460] [camera.camera]: Enable depth stream
[component_container-1] [INFO] [1744681629.017885524] [camera.camera]: Stream depth width: 640 height: 400 fps: 15 format: Y11
[component_container-1] [INFO] [1744681629.017900852] [camera.camera]: Enable ir stream
[component_container-1] [INFO] [1744681629.017913812] [camera.camera]: Stream ir width: 640 height: 400 fps: 15 format: Y10
[component_container-1] [INFO] [1744681629.421129317] [camera.camera]: Disable frame sync
[component_container-1] [INFO] [1744681629.421311224] [camera.camera]: Setting LDP to ON
[component_container-1] [INFO] [1744681629.424282279] [camera.camera]: Device DaBal DCW2 connected
[component_container-1] [INFO] [1744681629.424318969] [camera.camera]: Serial number: AUIMB4200BB
[component_container-1] [INFO] [1744681629.424336738] [camera.camera]: Firmware version: RD1014
[component_container-1] [INFO] [1744681629.424352522] [camera.camera]: Hardware version:
[component_container-1] [INFO] [1744681629.424361886] [camera.camera]: device unique id: 1-2.2.1-6
[component_container-1] [INFO] [1744681629.424374869] [camera.camera]: Current node pid: 5413
[component_container-1] [INFO] [1744681629.424383481] [camera.camera]: usb connect type: USB2.0
[component_container-1] [INFO] [1744681629.424391293] [camera.camera]: Start device cost 886 ms
[component_container-1] [INFO] [1744681629.424406514] [camera.camera]: Initialize device cost 561 ms
[component_container-1] [INFO] [1744681629.559934841] [camera.camera]: Publishing static transform from ir to depth
[component_container-1] [INFO] [1744681629.560017122] [camera.camera]: Translation 0, 0, 0
[component_container-1] [INFO] [1744681629.560036396] [camera.camera]: Rotation 0, 0, 0, 1
[component_container-1] [INFO] [1744681629.560054805] [camera.camera]: Publishing static transform from color to depth
[component_container-1] [INFO] [1744681629.560070941] [camera.camera]: Translation 12.3407, 0.0319774, 1.64768
[component_container-1] [INFO] [1744681629.560083203] [camera.camera]: Rotation -0.00119075, -0.0032168, -0.000594161, 0.999994
[component_container-1] [INFO] [1744681629.560095434] [camera.camera]: Publishing static transform from depth to depth
[component_container-1] [INFO] [1744681629.560106095] [camera.camera]: Translation 0, 0, 0
[component_container-1] [INFO] [1744681629.560116436] [camera.camera]: Rotation 0, 0, 0, 1
```

```
ros2 node info /camera/camera
```

As shown in the figure below, the published color image topic is **/camera/color/image\_raw**, the published depth image topic is **/camera/depth/image\_raw**, the published infrared image topic is **/camera/ir/image\_raw**, and the published point cloud data topic is **/camera/depth/points**.

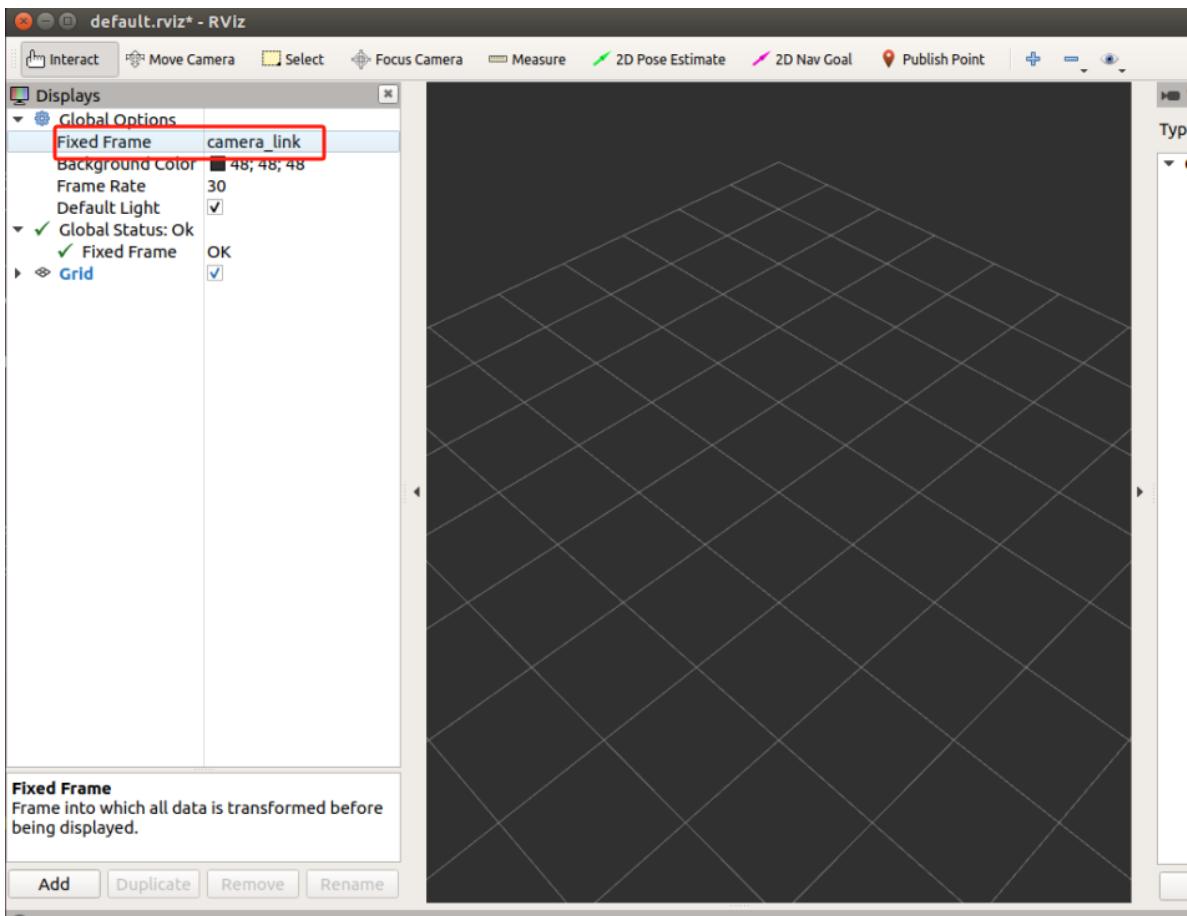
```
jetson@yahboom:~$ ros2 node info /camera/camera
/camera/camera
Subscribers:
  /parameter_events: rcl_interfaces/msg/ParameterEvent
Publishers:
  /camera/color/camera_info: sensor_msgs/msg/CameraInfo
  /camera/color/image_raw: sensor_msgs/msg/Image
  /camera/color/image_raw/compressed: sensor_msgs/msg/CompressedImage
  /camera/color/image_raw/compressedDepth: sensor_msgs/msg/CompressedImage
  /camera/color/image_raw/theora: theora_image_transport/msg/Packet
  /camera/depth/camera_info: sensor_msgs/msg/CameraInfo
  /camera/depth/image_raw: sensor_msgs/msg/Image
  /camera/depth/image_raw/compressed: sensor_msgs/msg/CompressedImage
  /camera/depth/image_raw/compressedDepth: sensor_msgs/msg/CompressedImage
  /camera/depth/image_raw/theora: theora_image_transport/msg/Packet
  /camera/depth/points: sensor_msgs/msg/PointCloud2
  /camera/depth_filter_status: std_msgs/msg/String
  /camera/depth_to_color: orbbec_camera_msgs/msg/Extrinsics
  /camera/depth_to_ir: orbbec_camera_msgs/msg/Extrinsics
  /camera/ir/camera_info: sensor_msgs/msg/CameraInfo
  /camera/ir/image_raw: sensor_msgs/msg/Image
  /camera/ir/image_raw/compressed: sensor_msgs/msg/CompressedImage
  /camera/ir/image_raw/compressedDepth: sensor_msgs/msg/CompressedImage
  /camera/ir/image_raw/theora: theora_image_transport/msg/Packet
  /diagnostics: diagnostic_msgs/msg/DiagnosticArray
  /parameter_events: rcl_interfaces/msg/ParameterEvent
  /rosout: rcl_interfaces/msg/Log
  /tf: tf2_msgs/msg/TFMessage
  /tf_static: tf2_msgs/msg/TFMessage
```

## 2.2. Operation

Use rviz to view the three images and point cloud provided by the camera. Restart a terminal and enter:

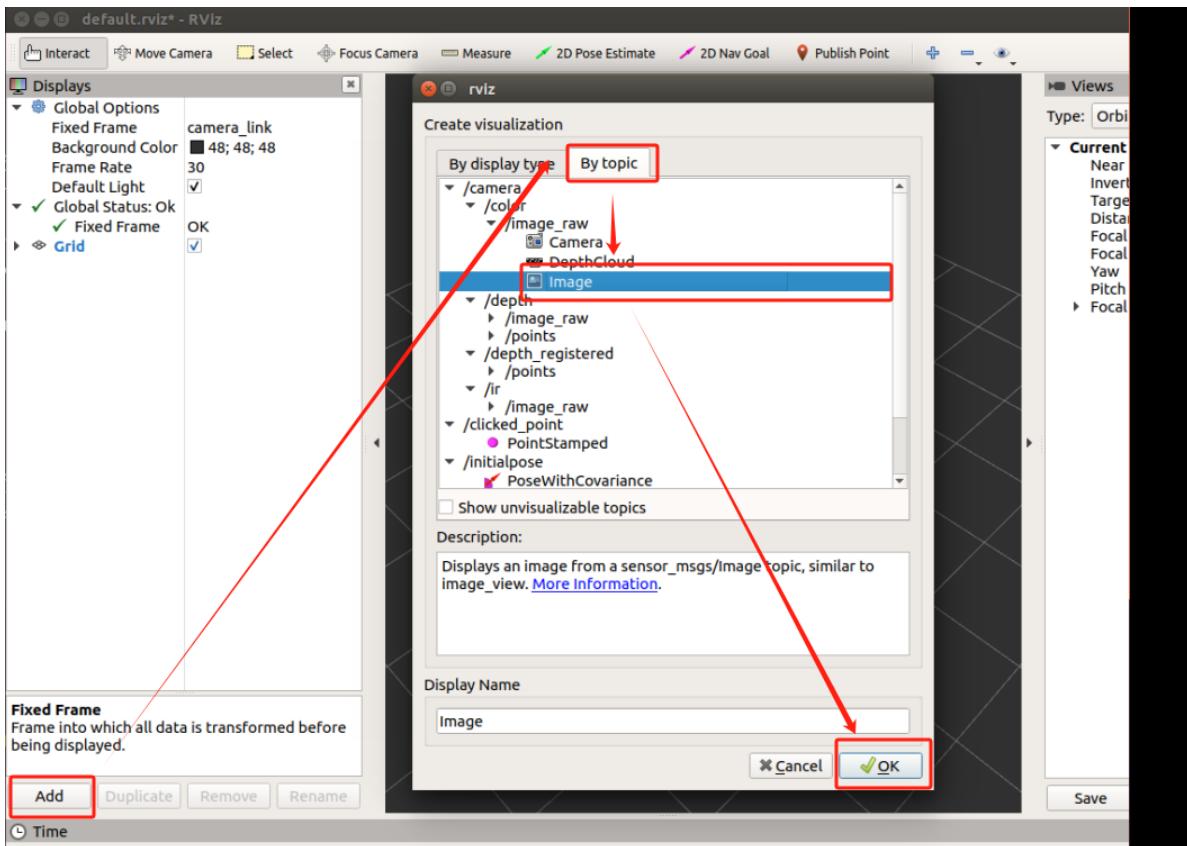
```
rviz2
```

After rviz starts, configure rviz according to the following diagram:

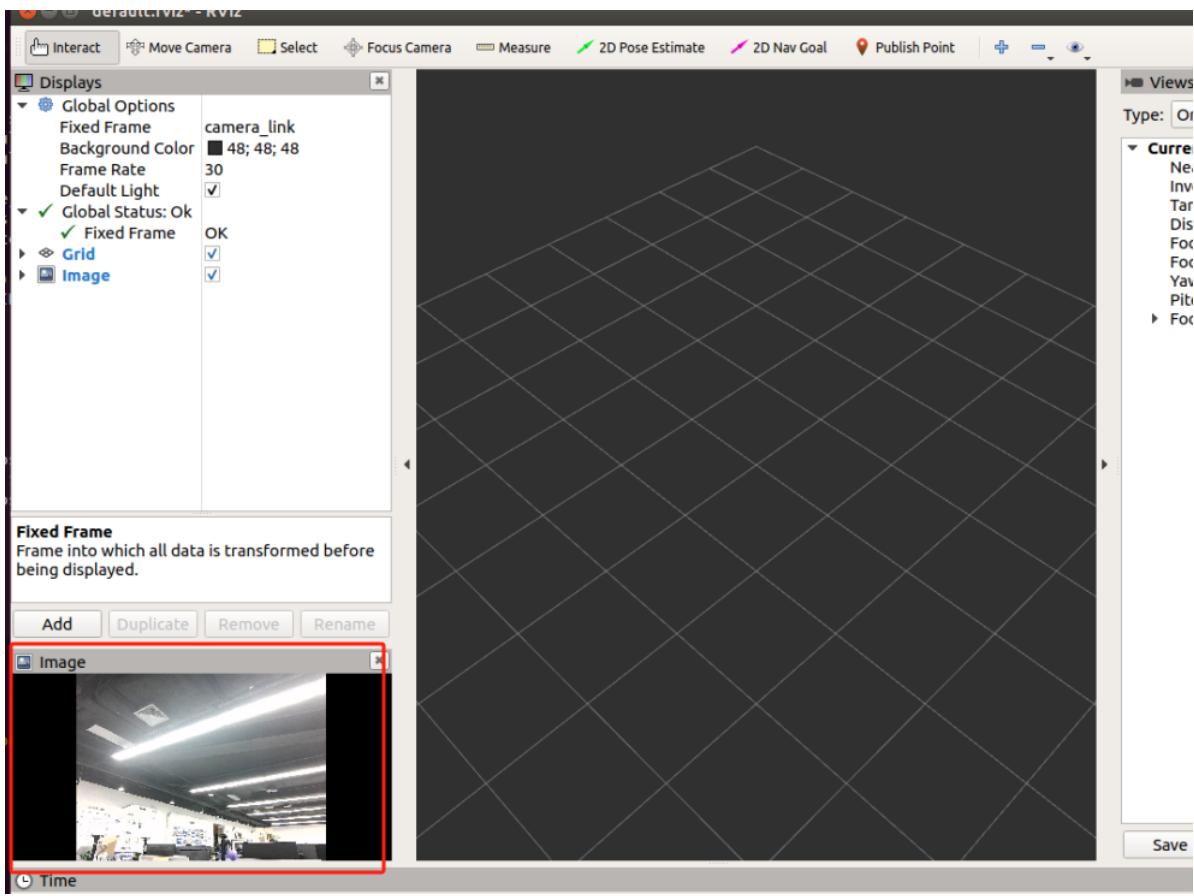


Change [Fixed Frame] here to **camera\_link**

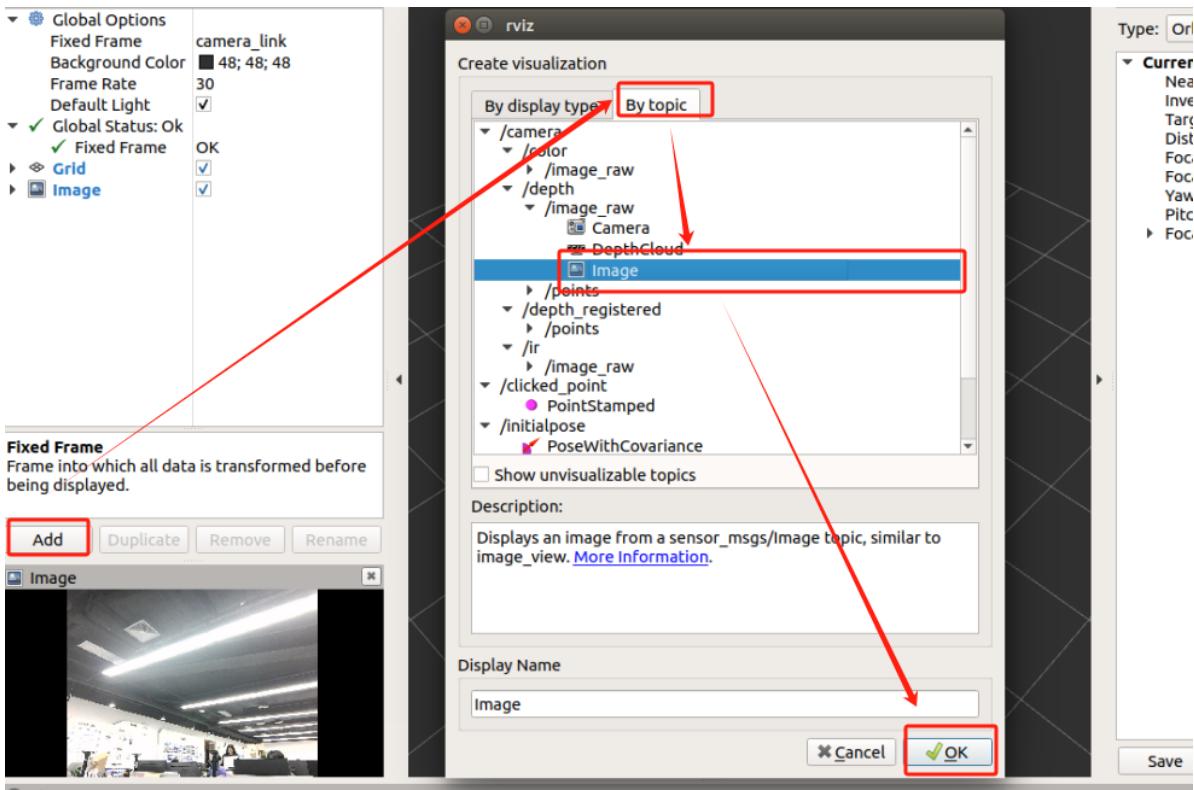
Add color image display



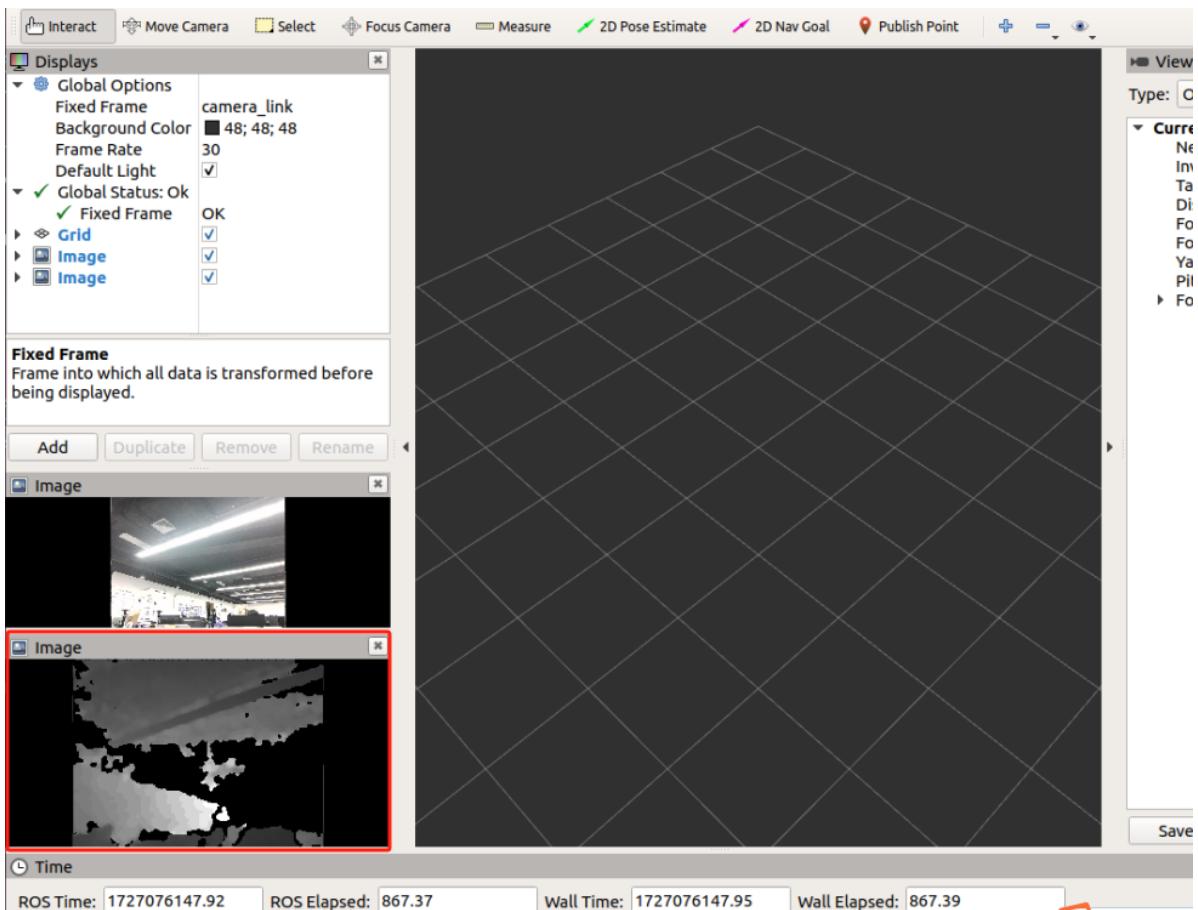
Click [Add], then click [By topic], select [/color/image\_raw/Image], and finally click [ok] to confirm.  
The resulting color image is as follows:



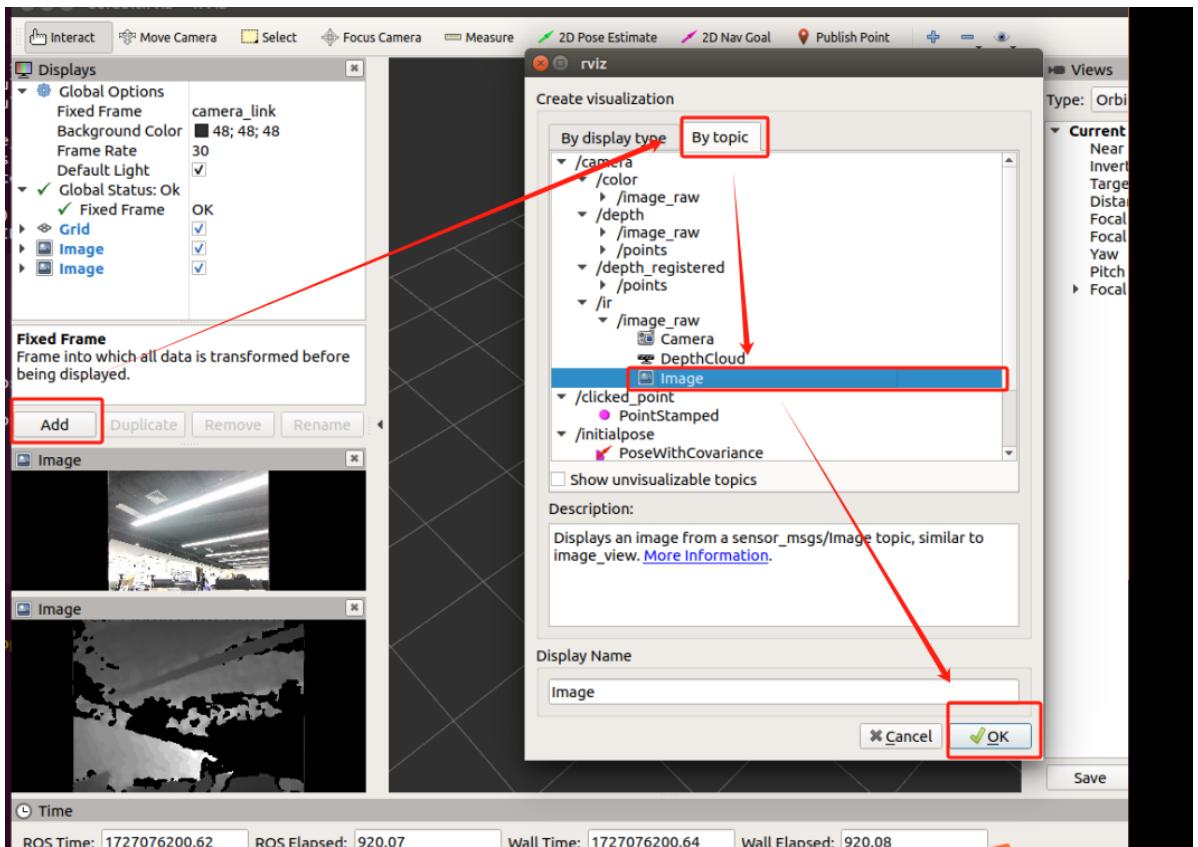
Add depth image display



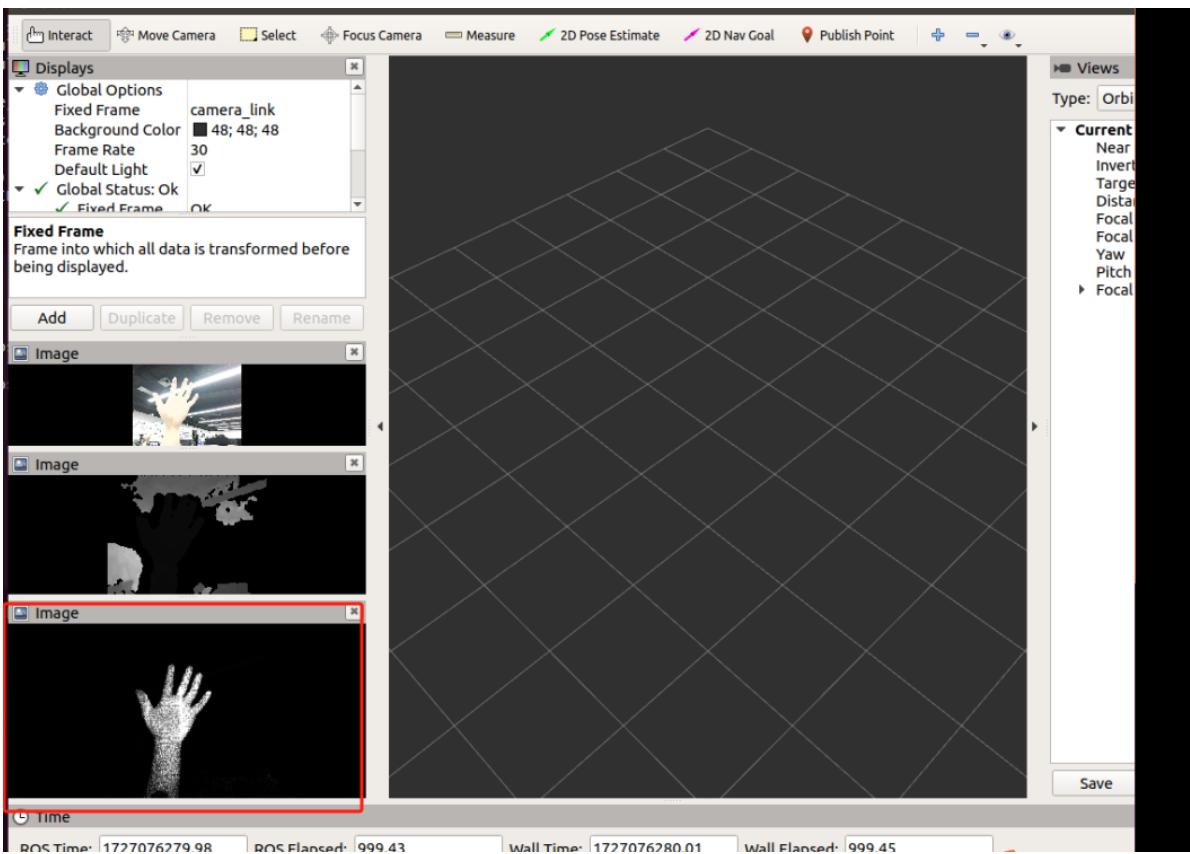
Click [Add], then click [By topic], select `[/depth/image_raw/Image]`, and finally click [ok] to confirm. The resulting depth color image is as follows:



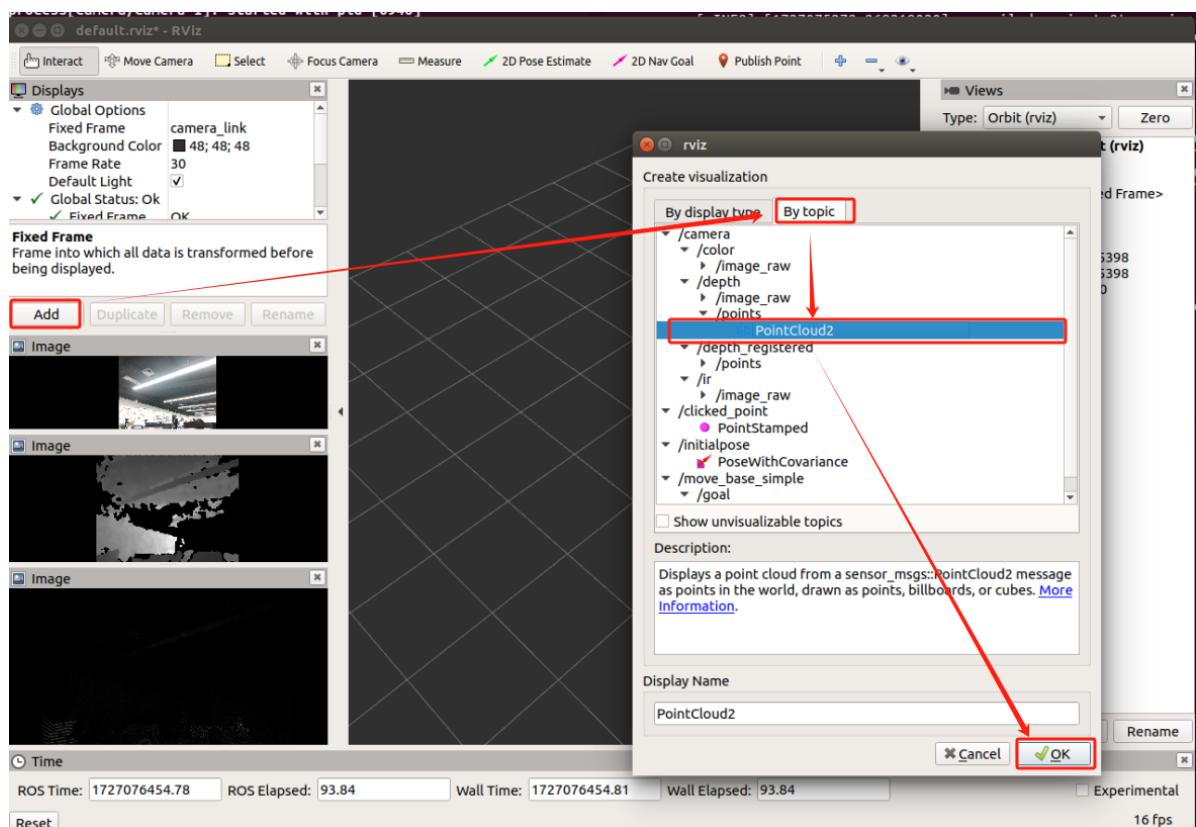
Add infrared image display



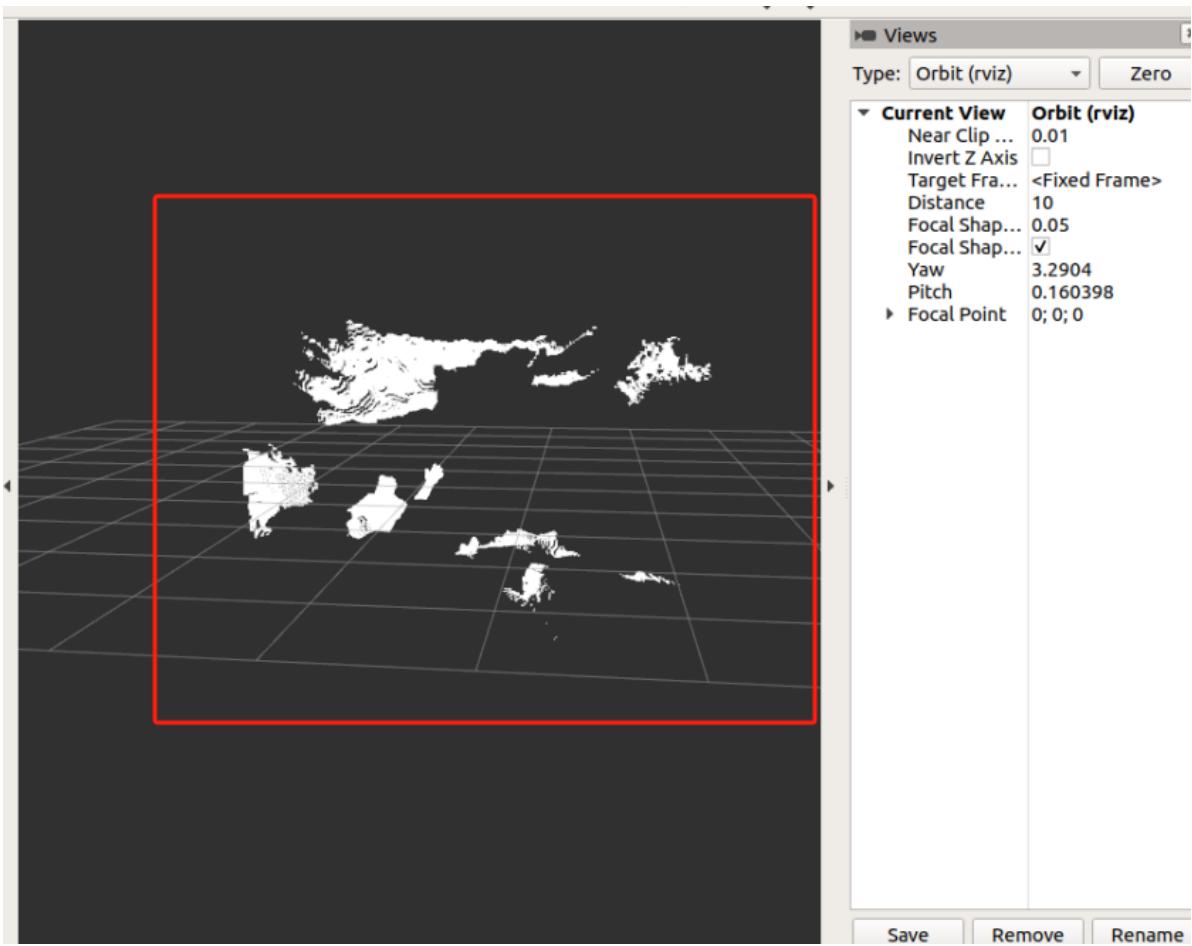
Click [Add], then click [By topic], select [/ir/image\_raw/Image], and finally click [ok] to confirm. The resulting infrared image is as follows:



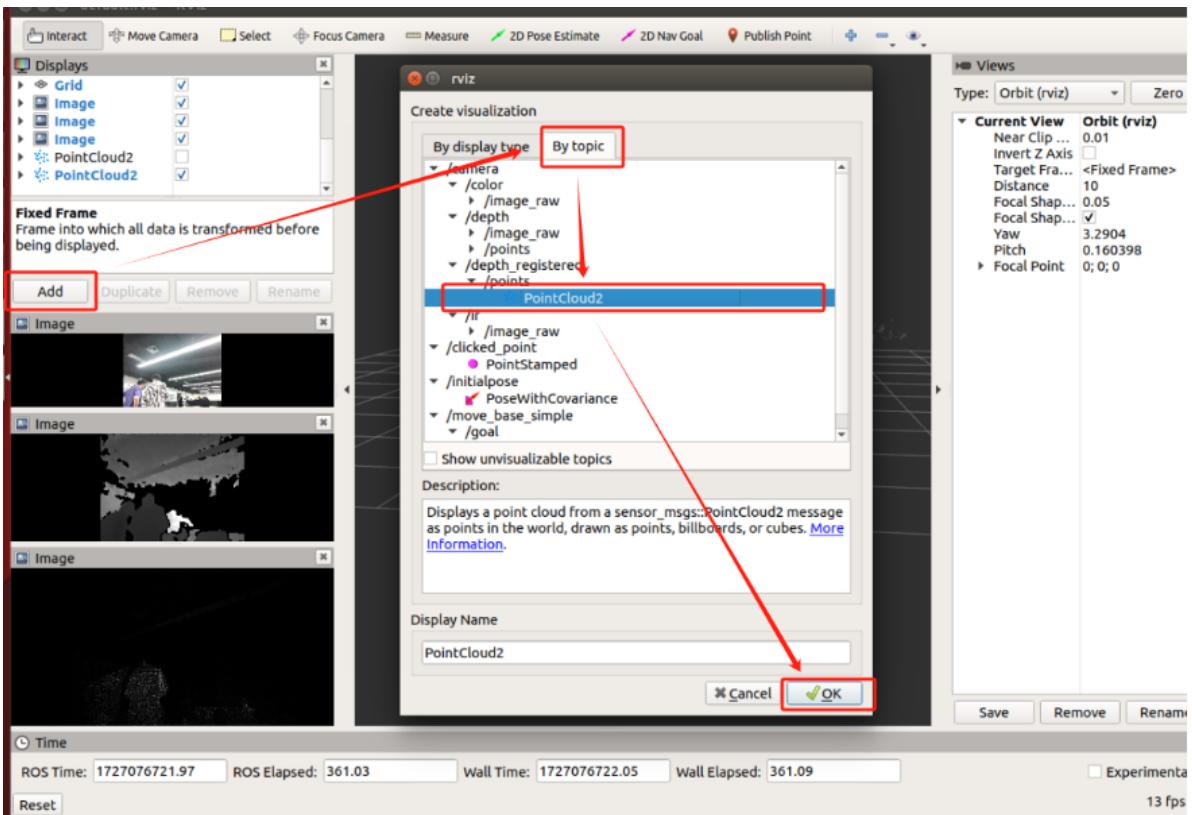
View depth point cloud data



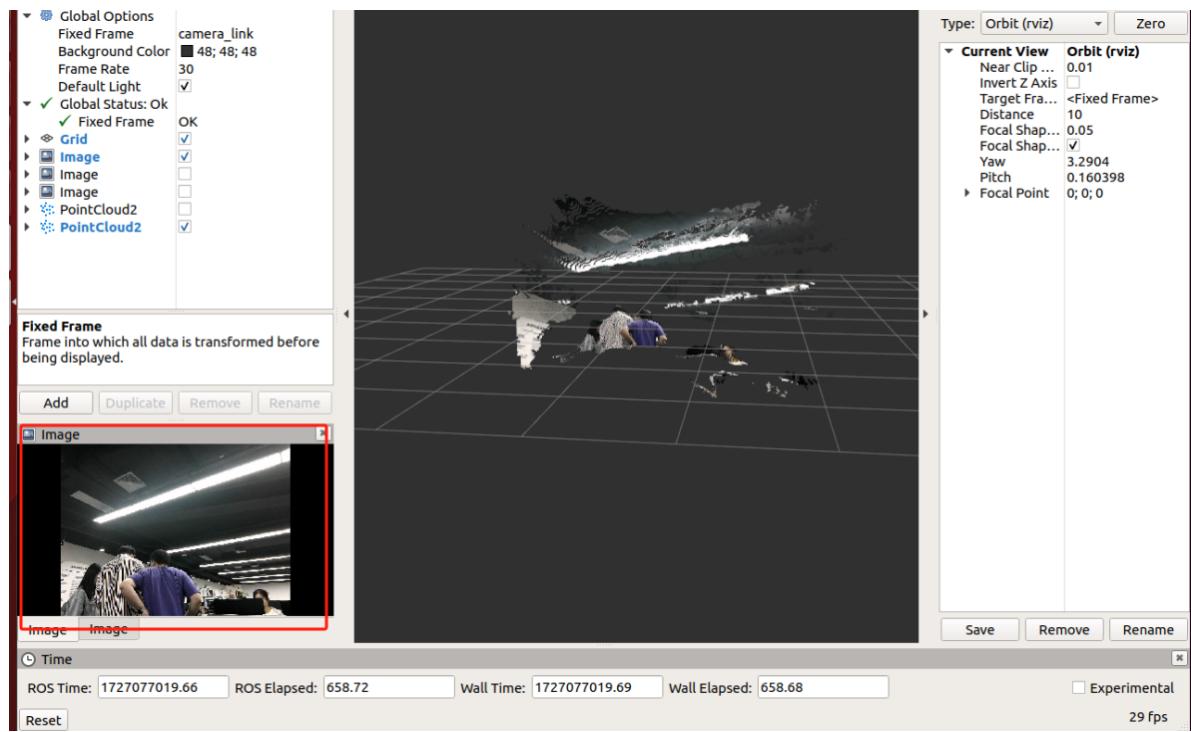
Click [Add], then click [By topic], select [/depth/points/PointCloud2], and finally click [ok] to confirm. The resulting depth point cloud image is as follows:



View color point cloud data



Click [Add], then click [By topic], select [/depth\_registered/points/PointCloud2], and finally click [ok] to confirm. The resulting color point cloud image is as follows:



### 3. Camera Parameters

#### 3.1. Hardware Parameter Table

DaBai DCW2/DW2 Depth Camera			
	Parameters	Specifications	Remark
Basic parameters	Name	DaBai DCW2/DW2	
	Model	DaBai DCW2: G10254-001 DaBai DW2: G11254-001	
	Working distance	Normal energy mode: 0.15~3m High energy mode: 0.15~5m	
	Device size	89.82mm*25.1mm*25.1mm	
	Power consumption	DaBai DCW2: Peak power consumption <7W; average power consumption <2.5W DaBai DW2: Peak power consumption <6W; average power consumption <2.0W	
	Baseline	40mm	
	Interface type	USB Type-C	
	Communication/power supply mode	USB 2.0	
	Working temperature	Normal energy mode: -10°C~50°C High energy mode: -10°C~40°C	
	Working humidity	5% ~ 95%RH	

#### 3.2. Camera Parameter Acquisition and Settings

The camera's initialization parameters have been set in the launched launch file. The launch file path is:

```
/home/jetson/dofbot_pro_ws/src/OrbbecSDK_ROS2-
main/orbbec_camera/launch/dabai_dcw2.launch.py
```

The content inside is as follows:

```
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument
from launch.substitutions import LaunchConfiguration
```

```

from launch_ros.actions import PushRosNamespace
from launch.actions import GroupAction
from launch_ros.actions import ComposableNodeContainer
from launch_ros.descriptions import ComposableNode
from launch_ros.actions import Node
import os

def generate_launch_description():
    # Declare arguments
    args = [
        DeclareLaunchArgument('camera_name', default_value='camera'),
        DeclareLaunchArgument('depth_registration', default_value='true'),
        DeclareLaunchArgument('serial_number', default_value=''),
        DeclareLaunchArgument('usb_port', default_value=''),
        DeclareLaunchArgument('device_num', default_value='1'),
        DeclareLaunchArgument('vendor_id', default_value='0x2bc5'),
        DeclareLaunchArgument('product_id', default_value=''),
        DeclareLaunchArgument('enable_point_cloud', default_value='true'),
        DeclareLaunchArgument('enable_colored_point_cloud',
            default_value='true'),
        DeclareLaunchArgument('cloud_frame_id', default_value=''),
        DeclareLaunchArgument('point_cloud_qos', default_value='default'),
        DeclareLaunchArgument('connection_delay', default_value='100'),
        DeclareLaunchArgument('color_width', default_value='640'),
        DeclareLaunchArgument('color_height', default_value='480'),
        DeclareLaunchArgument('color_fps', default_value='30'),
        DeclareLaunchArgument('color_format', default_value='MJPG'),
        DeclareLaunchArgument('enable_color', default_value='true'),
        DeclareLaunchArgument('flip_color', default_value='false'),
        DeclareLaunchArgument('color_qos', default_value='default'),
        DeclareLaunchArgument('color_camera_info_qos', default_value='default'),
        DeclareLaunchArgument('enable_color_auto_exposure',
            default_value='true'),
        DeclareLaunchArgument('color_exposure', default_value='-1'),
        DeclareLaunchArgument('color_gain', default_value='-1'),
        DeclareLaunchArgument('enable_color_auto_white_balance',
            default_value='true'),
        DeclareLaunchArgument('color_white_balance', default_value=' -1'),
        DeclareLaunchArgument('color_ae_max_exposure', default_value=' -1'),
        DeclareLaunchArgument('color_brightness', default_value=' -1'),
        DeclareLaunchArgument('color_sharpness', default_value=' -1'),
        DeclareLaunchArgument('color_saturation', default_value=' -1'),
        DeclareLaunchArgument('color_contrast', default_value=' -1'),
        DeclareLaunchArgument('color_gamma', default_value=' -1'),
        DeclareLaunchArgument('color_hue', default_value=' -1'),
        DeclareLaunchArgument('depth_width', default_value='640'),
        DeclareLaunchArgument('depth_height', default_value='400'),
        DeclareLaunchArgument('depth_fps', default_value='15'),
        DeclareLaunchArgument('depth_format', default_value='Y11'),
        DeclareLaunchArgument('enable_depth', default_value='true'),
        DeclareLaunchArgument('flip_depth', default_value='false'),
        DeclareLaunchArgument('depth_qos', default_value='default'),
        DeclareLaunchArgument('depth_camera_info_qos', default_value='default'),
        # /config/depthfilter/openni_device.json, need config path.
        DeclareLaunchArgument('depth_filter_config', default_value=''),
        DeclareLaunchArgument('ir_width', default_value='640'),
        DeclareLaunchArgument('ir_height', default_value='400'),
    ]

```

```

        DeclareLaunchArgument('ir_fps', default_value='15'),
        DeclareLaunchArgument('ir_format', default_value='Y10'),
        DeclareLaunchArgument('enable_ir', default_value='true'),
        DeclareLaunchArgument('flip_ir', default_value='false'),
        DeclareLaunchArgument('ir_qos', default_value='default'),
        DeclareLaunchArgument('ir_camera_info_qos', default_value='default'),
        DeclareLaunchArgument('enable_ir_auto_exposure', default_value='true'),
        DeclareLaunchArgument('ir_exposure', default_value='-1'),
        DeclareLaunchArgument('ir_gain', default_value='-1'),
        DeclareLaunchArgument('publish_tf', default_value='true'),
        DeclareLaunchArgument('tf_publish_rate', default_value='0.0'),
        DeclareLaunchArgument('ir_info_url', default_value=''),
        DeclareLaunchArgument('color_info_url', default_value=''),
        DeclareLaunchArgument('log_level', default_value='none'),
        DeclareLaunchArgument('enable_publish_extrinsic',
        default_value='false'),
        DeclareLaunchArgument('enable_soft_filter', default_value='true'),
        DeclareLaunchArgument('enable_ldp', default_value='true'),
        DeclareLaunchArgument('soft_filter_max_diff', default_value='-1'),
        DeclareLaunchArgument('soft_filter_speckle_size', default_value='-1'),
        DeclareLaunchArgument('ordered_pc', default_value='false'),
        DeclareLaunchArgument('use_hardware_time', default_value='false'),
        DeclareLaunchArgument('enable_depth_scale', default_value='true'),
        DeclareLaunchArgument('align_mode', default_value='HW'),
        DeclareLaunchArgument('laser_energy_level', default_value='-1'),
        DeclareLaunchArgument('enable_heartbeat', default_value='false'),
    ]
}

# Node configuration
parameters = [{arg.name: LaunchConfiguration(arg.name)} for arg in args]
# get ROS_DISTRO
ros_distro = os.environ["ROS_DISTRO"]
if ros_distro == "foxy":
    return LaunchDescription(
        args
        +
        [
            Node(
                package="orbbec_camera",
                executable="orbbec_camera_node",
                name="ob_camera_node",
                namespace=LaunchConfiguration("camera_name"),
                parameters=parameters,
                output="screen",
            )
        ]
    )
# Define the ComposableNode
else:
    # Define the ComposableNode
    compose_node = ComposableNode(
        package="orbbec_camera",
        plugin="orbbec_camera::OBCameraNodeDriver",
        name=LaunchConfiguration("camera_name"),
        namespace="",
        parameters=parameters,
    )
    # Define the ComposableNodeContainer
    container = ComposableNodeContainer(

```

```
        name="camera_container",
        namespace="",
        package="rclcpp_components",
        executable="component_container",
        composable_node_descriptions=[
            compose_node,
        ],
        output="screen",
    )
# Launch description
ld = LaunchDescription(
    args
    +
    [
        GroupAction(
            [PushRosNamespace(LaunchConfiguration("camera_name"))],
container]
    )
]
)
return ld
```

After the camera runs, we can use some services to get or set some parameters, such as getting the exposure value and setting the exposure value. First, let's check what services the camera node can call. Enter in the terminal:

```
ros2 node info /camera/camera
```

As shown in the figure below, these are some services provided after the camera node starts:

```

Service Servers:
/camera/camera/describe_parameters: rcl_interfaces/srv/DescribeParameters
/camera/camera/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
/camera/camera/get_parameters: rcl_interfaces/srv/GetParameters
/camera/camera/list_parameters: rcl_interfaces/srv/ListParameters
/camera/camera/set_parameters: rcl_interfaces/srv/SetParameters
/camera/camera/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
/camera/get_auto_white_balance: orbbec_camera_msgs/srv/GetInt32
/camera/get_color_exposure: orbbec_camera_msgs/srv/GetInt32
/camera/get_color_gain: orbbec_camera_msgs/srv/GetInt32
/camera/get_depth_exposure: orbbec_camera_msgs/srv/GetInt32
/camera/get_depth_gain: orbbec_camera_msgs/srv/GetInt32
/camera/get_device_info: orbbec_camera_msgs/srv/GetDeviceInfo
/camera/get_ir_exposure: orbbec_camera_msgs/srv/GetInt32
/camera/get_ir_gain: orbbec_camera_msgs/srv/GetInt32
/camera/get_ldp_measure_distance: orbbec_camera_msgs/srv/GetInt32
/camera/get_ldp_status: orbbec_camera_msgs/srv/GetBool
/camera/get_sdk_version: orbbec_camera_msgs/srv/GetString
/camera/get_white_balance: orbbec_camera_msgs/srv/GetInt32
/camera/reboot_device: std_srvs/srv/Empty
/camera/save_images: std_srvs/srv/Empty
/camera/save_point_cloud: std_srvs/srv/Empty
/camera/set_auto_white_balance: std_srvs/srv/SetBool
/camera/set_color_auto_exposure: std_srvs/srv/SetBool
/camera/set_color_exposure: orbbec_camera_msgs/srv/.SetInt32
/camera/set_color_gain: orbbec_camera_msgs/srv/.SetInt32
/camera/set_color_mirror: std_srvs/srv/SetBool
/camera/set_depth_auto_exposure: std_srvs/srv/SetBool
/camera/set_depth_exposure: orbbec_camera_msgs/srv/.SetInt32
/camera/set_depth_gain: orbbec_camera_msgs/srv/.SetInt32
/camera/set_depth_mirror: std_srvs/srv/SetBool
/camera/set_fan_work_mode: orbbec_camera_msgs/srv/.SetInt32
/camera/set_floor_enable: std_srvs/srv/SetBool
/camera/set_ir_auto_exposure: std_srvs/srv/SetBool
/camera/set_ir_exposure: orbbec_camera_msgs/srv/.SetInt32
/camera/set_ir_gain: orbbec_camera_msgs/srv/.SetInt32
/camera/set_ir_long_exposure: std_srvs/srv/SetBool
/camera/set_ir_mirror: std_srvs/srv/SetBool
/camera/set_laser_enable: std_srvs/srv/SetBool
/camera/set_ldp_enable: std_srvs/srv/SetBool
/camera/set_white_balance: orbbec_camera_msgs/srv/.SetInt32
/camera/switch_ir: orbbec_camera_msgs/srv/SetString
/camera/toggle_color: std_srvs/srv/SetBool
/camera/toggle_depth: std_srvs/srv/SetBool
/camera/toggle_ir: std_srvs/srv/SetBool

```

### 3.2.1. Camera Parameter Acquisition

Taking getting the exposure value as an example, call the service. Enter in the terminal:

```

ros2 service call /camera/get_color_exposure orbbec_camera_msgs/srv/GetInt32
'{}'

```

As shown in the figure below, the obtained exposure value is 100

```

jetson@yahboom:~$ ros2 service call /camera/get_color_exposure orbbec_camera_msgs/srv/GetInt32 '{}'
requester: making request: orbbec_camera_msgs.srv.GetInt32_Request()

response:
orbbec_camera_msgs.srv.GetInt32_Response(data=100, success=True, message='')

jetson@yahboom:~$ 

```

### 3.2.2. Camera Parameter Settings

Taking the auto exposure value switch as an example, call the service. Enter in the terminal:

```

ros2 service call /camera/set_color_auto_exposure std_srvs/srv/SetBool '{data:
true}'

```

```

jetson@yahboom:~$ ros2 service call /camera/set_color_auto_exposure std_srvs/srv/SetBool '{data: true}'
requester: making request: std_srvs.srv.SetBool_Request(data=True)

response:
std_srvs.srv.SetBool_Response(success=True, message='')

jetson@yahboom:~$ 

```

Manual setting

```
ros2 service call /camera/set_color_exposure orbbec_camera_msgs/srv/SetInt32
'{data: 50}'
```

```
jetson@yahboom:~/dofbot_pro_ws$ ros2 service call /camera/set_color_exposure orbbec_camera_msgs/srv/SetInt32 '{data: 50}'
Waiting for service to become available...
requester: making request: orbbec_camera_msgs.srv.SetInt32_Request(data=50)

response:
orbbec_camera_msgs.srv.SetInt32_Response(success=True, message='')
```

After setting the auto exposure value, the color image will change the exposure value according to the set environment changes.

