ANG-LINUX-SDK_ROS User Guide

Revision History

Version number time		Revision Record	Revision Department
v1.1.0 2023/04/20	Initial release		Software Department
v1.1.1 2023/05/17	Added monitoring function		Software Department
v1.1.2 2023/06/08	Generalization of sample code		Software Department
v1.1.3 2023/06/21	Fix the problem of getting SN number and optimize demo logic		Software Department
v1.1.4 2023/06/28	Add timestamp and sequence number to point cloud messages in ROS/ROS2 demo		Software Department
v1.1.5 2023/07/03	Fix the problem that network device traversal may occur		Software Department
v1.1.6 2023/07/06	hp60c/hp60cn/nuwa/vega configuration files add the option to select ordered or unordered point clouds. Software Department		
v1.1.7 2023/07/11		1. Added hp60v depth correction 2. Change the naming of some options in the nuwa configuration file	Software Department
v1.1.8 2023/07/12		1. Optimize the Kunlun algorithm 2. The sample code adds the function of judging whether it is a virtual machine and whether it matches the nuwa device	Software Department
v1.1.9 2023/07/14	1. Fixed the problem of NUWA hp60v (sh58) getting internal and external parameters incorrectly 1. Added support for		Software Department
v1.1.10 2023/07/27		previewing color point clouds on ros/ros2 for hp60c/hp60cn modules 2. Fixed the problem of ros2 program crashing when exiting 3. Fix the firmware upgrade failure issue for kunlun-a / kunlun-c / hp60c / hp60cn 4. Fixed the crash issue when closing data stream in v4l2 module 5. Fixed the issue that the setting parameters of some nuwa series modules were not effective 6. The ros2 log is saved through the launch file settings, and the redirection generation is cancelled 7. Update and add nuwa configuration file options	Software Department
v1.2.0 2023/08/11		1. Add launch files for specific cameras; 2. Added the option of mono8 topic publishing; 3. Added the release of ros1 tf tree topic; 4. Added the ability to obtain the alignment status of RGB and depth.	Software Department
v1.2.1 2023/08/21		1. Added the release of ros2 tf tree topic; 2. Fixed some abnormal issues with publishing ros topics 1.	Software Department
v1.2.2 2023/08/22		Optimized x100 deep data processing algorithm 2. Added x100 frame rate configuration.	Software Department
v1.2.3 2023/08/29		1. Fixed the problem of kunlun series module switching between near and far view 2. Added sample code for using ros2 for Kunlun series modules	Software Department
v1.2.4 2023/08/31	1. Fixed the issue that nuwa camera may have errors when matching multiple nodes		Software Department
v1.2.5 2023/09/04	1 Relax the network camera hot plug detection time from 2s->6s		Software Department
v1.2.6 2023/09/07	Added XB40, HP60, and HP60V frame rate configurations.		Software Department
v1.2.7 2023/09/11		1. Fixed the bug that caused the depth denoising parameter to be incorrect 2. Delete the automatic restart function in the non-hot-swap demo 3. Fixed the bug of network camera monitoring mechanism	Software Department
v1.2.8 2023/09/13	1. Optimize log printing and network broadcasting		Software Department

v1.2.9 2023/10/23		1. Optimize the NUWA camera parameter configuration sequence logic 2. Add HP60CN frame loss filter detection	Software Department
v1.2.10 2023/11/10		1. Added upgrade and capability level acquisition for each NUWA model 2. Optimize the release package structure and size 3. Add internal reference data, SDK call restriction description, and release package usage instructions to the development documentation 4. Delete the non-hot-plug reference example and remove the listener keyword from the original hot-plug example file name Character 5. Fixed the problem of occasional failure in creating handles for network cameras 3.6 . Fix the abnormal publishing of ros/ros2 topics in Kunlun series	Software Department
v1.2.11 2023/12/12		1. Modify the HP60 module noise reduction parameters (maximum speckle size denoiseMaxSpeckleSize) from 100 to 800 2. Modify HP60C 》 HP60CN camera internal parameter processing: cyRgb value from source data (rawData[12]+6)/2.25, modified to (rawData[12]+6)/2.25 and assigned after parsing 3. Modify the configuration parameters of HP60C and HP60CN: Change the order from the original setting after opening the flow to opening the flow. After the device is installed, before the stream is opened 4. Modify the raw parsing order of KunlunA and KUNLUNC images, from the original upper left corner to the right, Parse line by line downwards, change to parse from the lower right corner to the left, and from bottom to top line by line	Software Department
v1.2.12 2023/12/28		1. KUNLUN-A/C implements the interface for obtaining resolution capability set 2. KUNLUN-A fixed the abnormal problem of obtaining firmware version number 3. KUNLUN-A/C changed the way of reading local calibration files to reading calibration data in the module 4. KUNLUN-A/C's width and height parameters for setting resolution are changed to need to be filled in the final output image Width and height 5. Update KUNLUN configuration file 6. Added the function of enumerating devices in WINDOWS monitoring function	Software Department
v1.2.13 2024/01/04	1. Fixed the problem that the depth and color images could not be aligned due to the error in the output of internal and external parameters of HP60CN.		
v1.2.14 2024/01/29		1. Update KUNLUN-A/C algorithm library V3_20240116 2. Fixed the problem of abnormal frame number returned by KUNLUN-A/C Windows version 3. Added ROS/ROS2 topic publishing name with node name as prefix 4. Added the function of starting the camera stream and publishing the corresponding topic after subscribing to the ROS/ROS2 topic 5. Fixed the abnormal issue of KUNLUN-A/C ROS/ROS2 CameraInfo topic 6. Modify the ROS/ROS2 topic number to distinguish the topic names of multi-module or multi-node startup 7. Repair HP60CN occasional failure to open the device 8. Fixed the occasional crash of HP60CN 9. Optimize KUNLUN-A/C image output time 1. Add KUNLUN-A default internal parameters	Software Department
v1.2.15 2024/02/21		1. Added support for hp60 320x240 resolution 2. Fixed the abnormal callback progress of KUNLUN-A firmware OTA upgrade 3. Fix the abnormal configuration of KUNLUN register 4. Modify KUNLUN default frame rate from 20 fps to 10 fps 5. Fixed the abnormal startup sequence number of ROS/ROS2 multi-module or multi-node 6. Added ROS1 nodelet functionality 7. Added HP60C/HP60CN/KUNLUN-C mjpeg format data output to callback, and linux-demo display	Software Department
v1.2.16 2024/02/23		1.Fixed the issue that the register setting of KUNLUN-A/C does not take effect after opening the stream, causing the depth map to flicker 2. Update KUNLUN-A/C algorithm library V3_20240223 to fix the problem of automatic integration function	Software Department

v1.2.17 2024/03/25	1. Optimize the gain setting method of HP60C		Software Department
V1.2.18 2024/03/28	1. Mirror HP60CN depth map and color map		Software Department
v1.2.19 2024/04/03		1. Update KUNLUN algorithm library 2. Fixed the abnormal preview image issue of linux demo	Software Department
v1.2.20 2024/04/15	1. Update KUNLUN algorithm library		Software Department
v1.2.21 2024/04/25	1. Optimize the firmware upgrade order of NUWA series modules		Software Department
v1.2.22 2024/05/16		1. Added interface for obtaining the space size of camera private data 2. Add an interface for writing private data to the camera 3. Added an interface for reading private data from the camera 4. Update KUNLUN-A/C configuration files	Software Department
V1.2.23 2024/07/18		1. Add VEGA, CHANGJIANG-B, TAISHAN, TANGGULA-A model modules 2. Added the function of retaining decimals for VEGA model depth output, which is disabled by default and can be set in the configuration file Modify Decimal to true to enable 3. Fixed ros2 component link error 4. Fixed occasional crash of network camera when shutting down the stream 5. Fixed the bug that caused the network camera to interfere with each other and cause the device to fail to open 6. Mirror HP60CN depth and RGB direction 1. Fixed the	Software Department
V1.2.24 2024/08/08		problem of multiple files in the OTA package failing to upgrade 2. Fixed the problem of incorrect width and height returned by the changjiangB module when outputting infrared images 1.	Software Department
V1.2.25 2024/08/25		Fixed the problem of AS_SDK_DestoryCamHandle accessing released memory 2. Fixed the issue of not releasing memory when exiting the demo of Linux/ros/ros2 3. TANGGULA-A adds the functions of reading image resolution from the module and depth shift from the configuration file. Bit number function 4. Added O2 camera (only supports WINDOWS) 5. Add TANGGULA-B module 6. Update changjiangB's depth stitching algorithm 7. Added OTA upgrade file type warp 8. Fixed the memory leak issue of the changjiangB module switch device	Software Department
v1.2.26 2024/09/03		1. Update TANGGULA-A configuration file 2. TANGGULA-B adds the function of simultaneous output of color image and IR (speckle) image	Software Department
v1.2.27 2024/09/12		1. TANGGULA-A and TANGGULA-B use the maximum frame rate by default 2. Fix the point cloud distortion problem after reducing the resolution of hp60c/hp60cn 1. Update TANGGULA-	Software Department
V1.2.28 2024/10/15		B configuration file, enable alignment by default 2. Added default internal and external parameters for TANGGULA-A/TANGGULA-B 3. Added support for XB40-V3 serial communication version firmware	Software Department
V1.2.28 2024/10/21	1. Corrected the incorrect description of ANG-LINUX-ROS User Manual 9.5		Software Department

Table of contents

1. Introduction

4.1 1.1 Overview

as_camera_node is a ROS node for the Ansijiang camera module, which can publish depth maps, point cloud maps, IR maps, PEAK maps, RGB maps, etc. Function.

4.2 1.2 Environmental requirements

1.2.1 Operating Environment

ARM/AARCH64, X86_64 architectures.

To run the samples provided by SDK, you need to install the following software

sudo apt-get install cmake git g++

1.2.2 System Requirements

Ubuntu16.04/Ubuntu18.04/Ubuntu20.04/Ubuntu22.04

2. SDK package file description

4.3 2.1 Package structure description

AngstrongCameraSdk_vx.x.x.20xxxxxx

ÿÿÿ CHANGELOG	SDK Update History
ÿÿÿ docs	SDK/ROS development instructions and other information table
ÿÿÿ sample	
ÿÿÿ linux_ros	
ÿÿÿ linux	Reference example of simple call under Linux system
ÿÿÿ ros	ROS1 reference example under ros system
ÿÿÿ src	
ÿÿÿ as_nodelet	Nodelet Example
ÿÿÿ ascamera	Node Example
ÿÿÿ configurationfiles	Camera parameter configuration file
ÿÿÿ include	demo header file
ÿÿÿ launch	ROS parameter configuration file
ÿÿÿ libs	
ÿ ÿÿÿ include	library header file
ÿ ÿÿÿ lib	Library versions for each compiler
ÿÿÿ aarch64-linux-gnu	
ÿÿÿ arm-linux-gnueabihf	
ÿÿÿ x86_64-linux-gnu	
ÿÿÿ src	Demo source file
ÿÿÿ ros2	ROS2 reference example under ros system
ÿÿÿ windows	Reference examples for Windows systems

The structure of Linux refers to "ANG-SDK-006_SDK Development and Use Specifications.pdf", and the structure of ros2 refers to the structure of ros1.

The methods can be used for reference.

Version change description: Starting from v1.2.10, the provided linux/ros/ros2 demos are all references for the implementation of the hot-swap mechanism.

Example, equivalent to the reference example with xxx_listener in previous versions.

4.4 2.2 Example Program Running Instructions

2.2.1 Camera Configuration File Description

The camera configuration file is in the directory configurationfiles. When calling the "Open Camera" interface, you need to pass in the configuration file path.

2.2.2 Run the sample program on Ubuntu (Linux) system

Enter the sample/linux_ros/linux directory, where build.sh is the compilation script and run_ascamera.sh is the sample program. First, execute build.sh to compile, and then execute run_ascamera.sh script to run the program. Executing the program requires root Permissions, the script will be run as sudo

Compile:

./build.sh

run:

./run_ascamera.sh

2.2.3 Run ROS system ROS1 sample program

Enter the sample/linux_ros/ros directory, where build.sh is the compilation script and run_ascamera_node.sh is the startup node. First, execute build.sh to compile, and then execute run_ascamera_node.sh script to start the node.

The method can be referred to as camera node. Starting the node requires root privileges, and the script will run in sudo mode.

Compile:

./build.sh

Start the node:

./run_ascamera_node.sh

Start the nodelet:

./run_ascamera_nodelet.sh

2.2.4 Run ROS system ROS2 sample program

Enter the sample/linux_ros/ros directory, where build.sh is the compilation script and run_ascamera_node.sh is the startup node. First, execute build.sh to compile, and then execute run_ascamera_node.sh script to start the node. To start the node, you need root privileges, the script will be run as sudo.

Compile:

./build.sh

Start the node:

./run_ascamera_node.sh

3. Topic description

The node publishes the following topics, mainly including camera info, image, pointcoud, etc.

4.5 3.1 TF tree topic

By default, the tf tree topic is published. Users can set it through the pub_tfTree parameter of the launch configuration file /ros/src/ascamera/launch/xxx.launch. For ros1, refer to hp60c.launch and for ros2, refer to hp60c.launch.py. When

publishing the tf tree topic, use RVIZ to view the image Fixed Frame. You can drop down to select the coordinates nodeNameSpace_camera_link_x, nodeNameSpace_depth_x, nodeNameSpace_color_x, where nodeNameSpace_colorx is only applicable to RGBD cameras; when the tf topic is not published, please fill in nodeNameSpace_ascamera_x for Fixed Frame . Where x corresponds to the serial number of the camera, and nodeNameSpace is the namespace of the current node.

Instructions for using RGBD (HP60C/HP60CN/VAGA) cameras: (1) If

the output color data and depth data are not aligned, the camera coordinate system is "nodeNameSpace_camera_link_x" and the depth data is The coordinates are "nodeNameSpace_depth_x", and the coordinates of the color data are "nodeNameSpace_color_x", which can be seen through the topic;

(2) If the output color data and depth data are aligned, the camera coordinate system is "nodeNameSpace_camera_link_x", and the coordinates of the depth data and color data are both "nodeNameSpace_color_x", that is, the depth data has been aligned to the color data internally, which can be seen through the topic.

4.6 3.2 Depth Image Camera Information Topic

Topic name: /xxx/depthx/camera_info The "x" in

depthx represents the serial number of the found camera, such as "0", "1", and the same below. Type:

sensor_msgs::CameraInfo

4.7 3.3 Depth Image Topic

Topic name: /xxx/depthx/image_raw Type:

sensor_msgs::Image

4.8 3.4 Point cloud topic corresponding to depth

Topic name: /xxx/depthx/points Type:

sensor_msgs::PointCloud2

4.9 3.5 RGB image information topic

Topic name: /xxx/rgbx/camera_info Type:

sensor_msgs:CameraInfo

4.10 3.6 RGB image topic

Topic name: /xxx/rgbx/image
Type: sensor_msgs::Image

4.11 3.6 IR image information topic

Topic name: /xxx/irx/camera_info
Type: sensor_msgs:CameraInfo

4.12 3.7 IR image topic

Topic name: /xxx/irx/image
Type: sensor_msgs::Image

4.13 3.8 PEAK image information topic

Topic name: /xxx/peakx/camera_info
Type: sensor_msgs:CameraInfo

4.14 3.9 PEAK Image Topic

Topic name: /xxx/peakx/image
Type: sensor_msgs::Image

5. launch file description

5.1 4.1 ros launch file

launch file name	Parameter name	Parameter Description
ascamera.launch (default universal launch file) confiPath		The path to the configuration file directory, via run_ascamera_node.sh
hp60c.launch	confiPath	Same as the generic launch confiPath parameter
	depth_width	Set the width of the camera's depth data
	depth_height	Set the height of the camera depth data
	rgb_width	Set the width of the camera RGB data
	rgb_height	Set the camera's RGB data height
	fps	Set the camera frame rate
	color_pcl	Set whether to enable the color point cloud function
	pub_mono8	Set whether to publish mono8 topic
	pub_tfTree	Set whether to publish tf tree topic
hp60cn.launch	confiPath	Same as the generic launch confiPath parameter
	color_pcl	Set whether to enable the color point cloud function
	pub_tfTree	Set whether to publish tf tree topic
kunlun_a.launch	confiPath	Same as the generic launch confiPath parameter
	usb_bus_no	Set to open the camera with the specified USB bus number (-1 means no limit)
	usb_path	Set to open the camera with the specified USB port number (null means no limit)
	depth_width	The width of the camera output depth image (-1 is the output default)
	depth_height	The camera outputs a depth image with (-1 is the output default)
	peak_width	The width of the PEAK image output by the camera (-1 is the output default)
	peak_height	The camera outputs the peak image. (-1 is the output default)
	fps	Frame rate (-1 is output default)
kunlun_c.launch	confiPath	Same as the generic launch confiPath parameter
	usb_bus_no	Set to open the camera with the specified USB bus number (-1 means no limit)

	usb_path	Set to open the camera with the specified USB port number (null means no limit)
	depth_width	The width of the camera output depth image (-1 is the output default)
	depth_height	The camera outputs a depth image with (-1 is the output default)
	peak_width	The width of the PEAK image output by the camera (-1 is the output default)
	peak_height	The camera outputs the peak image. (-1 is the output default)
	rgb_width	The width of the camera output RGB image (-1 is the output default)
	rgb_height	The camera outputs RGB images. (-1 is the output default)
	fps	Frame rate (-1 is output default)
nuwa.launch	confiPath	Same as the generic launch confiPath parameter
	usb_bus_no	Set to open the camera with the specified USB bus number (-1 means no limit)
	usb_path	Set to open the camera with the specified USB port number (null means no limit)

[Note] When opening multiple camera nodes in one launch file, refer to the writing method of hp60c.launch and set different node names.

To the topic published by the corresponding node.

5.2 4.2 ros2 launch file

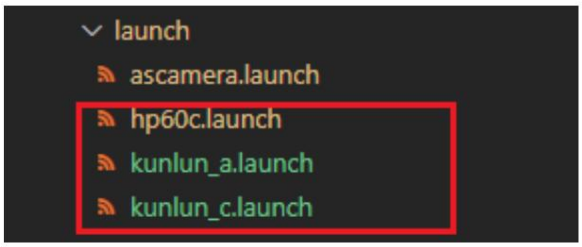
launch file name	Parameter name	Parameter Description
ascamera.launch.py (default general launch file)	confiPath	Path to the configuration file directory
hp60c.launch.py	confiPath	Path to the configuration file directory
	color_pcl	Set whether to enable the color point cloud function
	usb_bus_no	Set to open the camera with the specified USB bus number (-1 means no limit)
	usb_path	Set to open the camera with the specified USB port number (null means no limit)
	depth_width	Set the width of the camera's depth data
	depth_height	Set the height of the camera depth data
	rgb_width	Set the width of the camera RGB data
	rgb_height	Set the camera's RGB data height
	fps	Set the camera frame rate
hp60cn.launch.py	confiPath	Path to the configuration file directory
	color_pcl	Set whether to enable the color point cloud function

nuwa.launch.py	usb_bus_no	Set to open the camera with the specified USB bus number (-1 means no limit)
	usb_path	Set to open the camera with the specified USB port number (null means no limit)
	confiPath	Path to the configuration file directory
kunlun_a.launch.py	usb_bus_no	Set to open the camera with the specified USB bus number (-1 means no limit)
	usb_path	Set to open the camera with the specified USB port number (null means no limit)
	confiPath	Path to the configuration file directory
	depth_width	The width of the camera output depth image (-1 is the output default)
	depth_height	The camera outputs a depth image with (-1 is the output default)
	peak_width	The width of the PEAK image output by the camera (-1 is the output default)
	peak_height	The camera outputs the peak image. (-1 is the output default)
	fps	Frame rate (-1 is output default)
kunlun_c.launch.py	usb_bus_no	Set to open the camera with the specified USB bus number (-1 means no limit)
	usb_path	Set to open the camera with the specified USB port number (null means no limit)
	confiPath	Path to the configuration file directory
	depth_width	The width of the camera output depth image (-1 is the output default)
	depth_height	The camera outputs a depth image with (-1 is the output default)
	peak_width	The width of the PEAK image output by the camera (-1 is the output default)
	peak_height	The camera outputs the peak image. (-1 is the output default)
	rgb_width	The width of the camera output RGB image (-1 is the output default)
	rgb_height	The camera outputs RGB images. (-1 is the output default)
	fps	Frame rate (-1 is output default)

5.3 4.3 Notes

When using the launch file for a specific camera model (as shown in the figure), you need to modify the run_ascamera_node.sh script. The default script used is ascamera.lunch is a general launch file. Change the last line roslaunch ascamera ascamera.launch argPath:=$current_dir | tee $log_file to roslaunch ascamera xxx.launch argPath:=$current_dir | tee $log_file. (xxx is the launch file to be used.

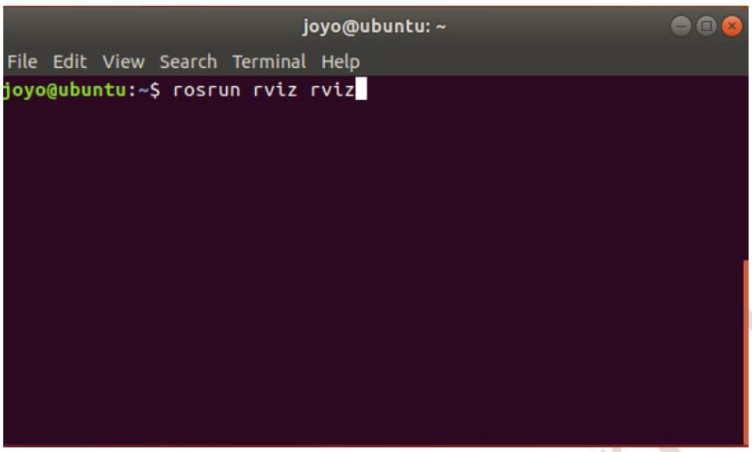
document)



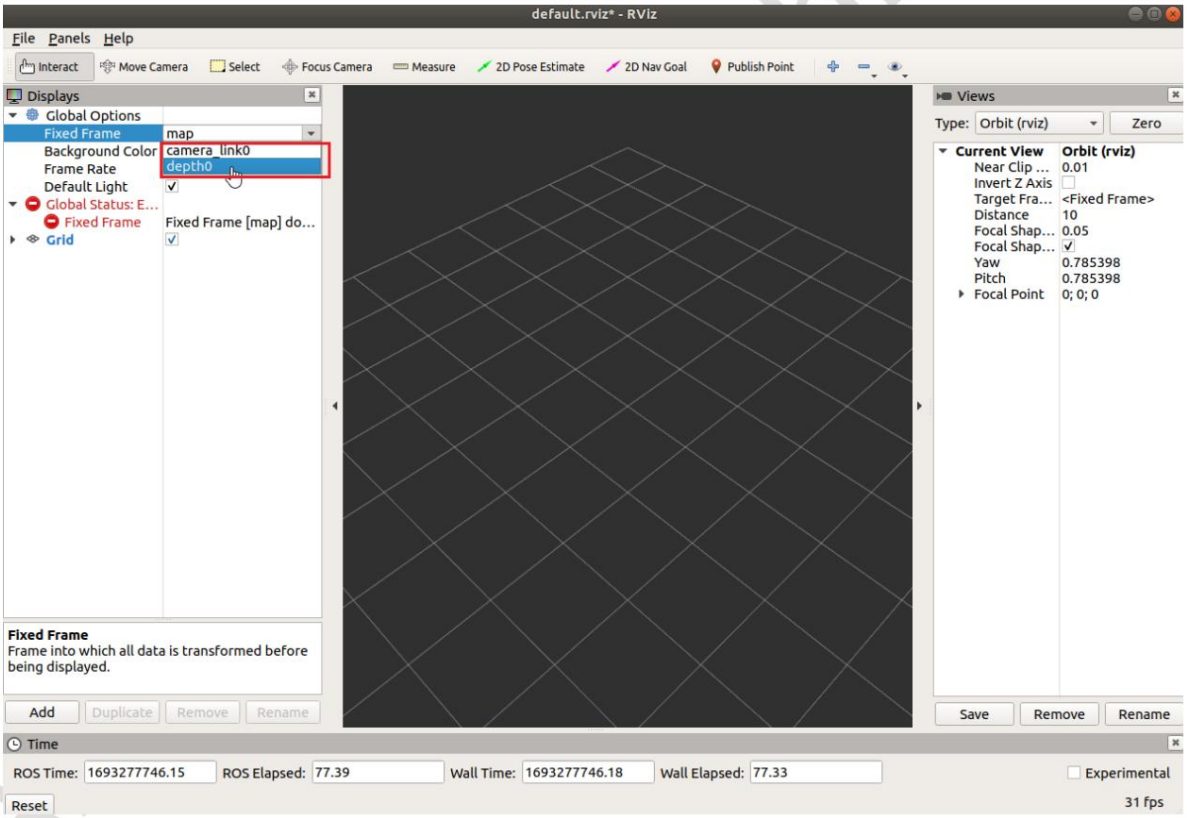

```
gcc_target=$(gcc -v 2>&1 | grep Target: | sed 's/Target: //g')
echo "Target: "${gcc_target}

cd src/ascamera/libs/lib/${gcc_target}
libPath=$(pwd)
export LD_LIBRARY_PATH=$libPath:$LD_LIBRARY_PATH
cd -

echo "lib path:"${libPath}

. devel/setup.bash

log_file=AngstrongsdkLog.txt

current_dir=$PWD"/src/ascamera/configurationfiles"

# rosrun ascamera ascamera_node
# roslaunch ascamera ascamera.launch argPath:=$current_dir | tee $log_file
roslaunch ascamera xxx.launch argPath:=$current_dir | tee $log_file
```

6. View images using RVIZ

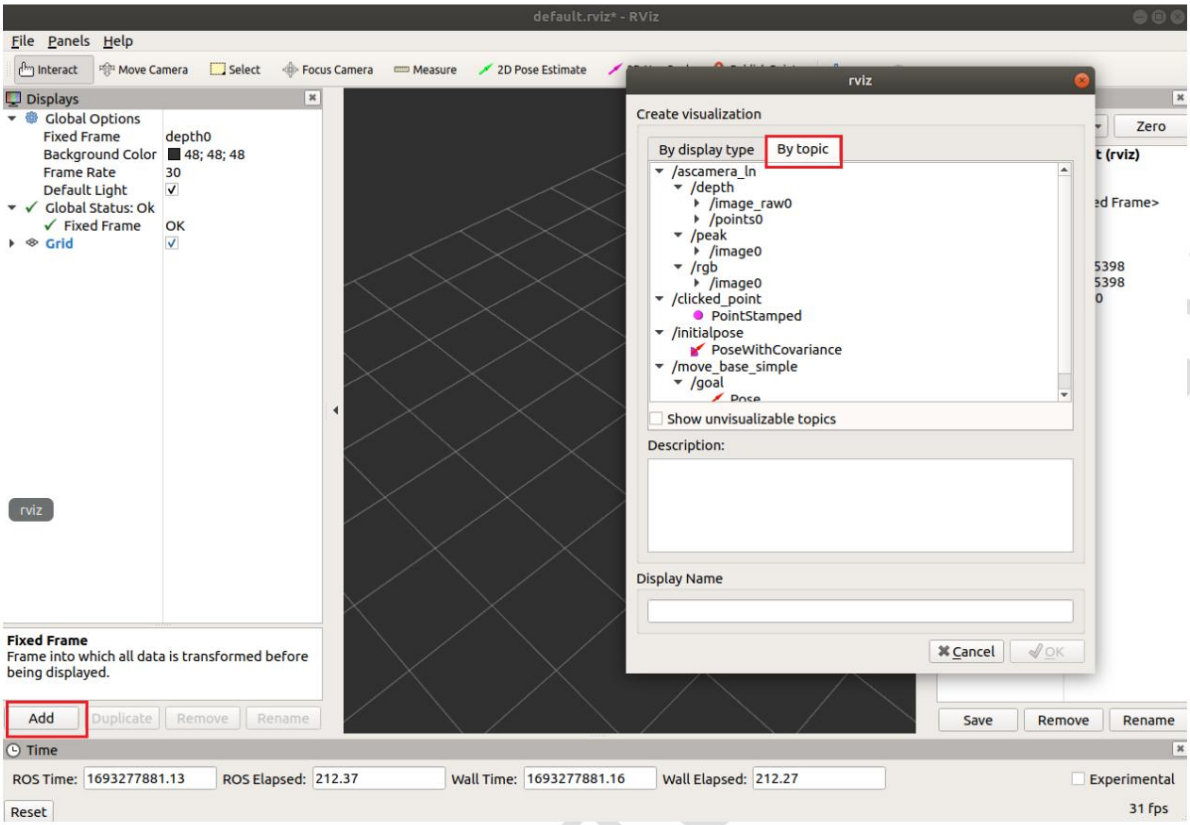
First, compile and run the ROS node according to the steps in 2.2.2. When the node is started, open rviz to receive and view the image. The steps are as follows: 1. ROS1 starts the rviz node with the following command: (Run the command under ROS2: ros2 run rviz2 rviz2)



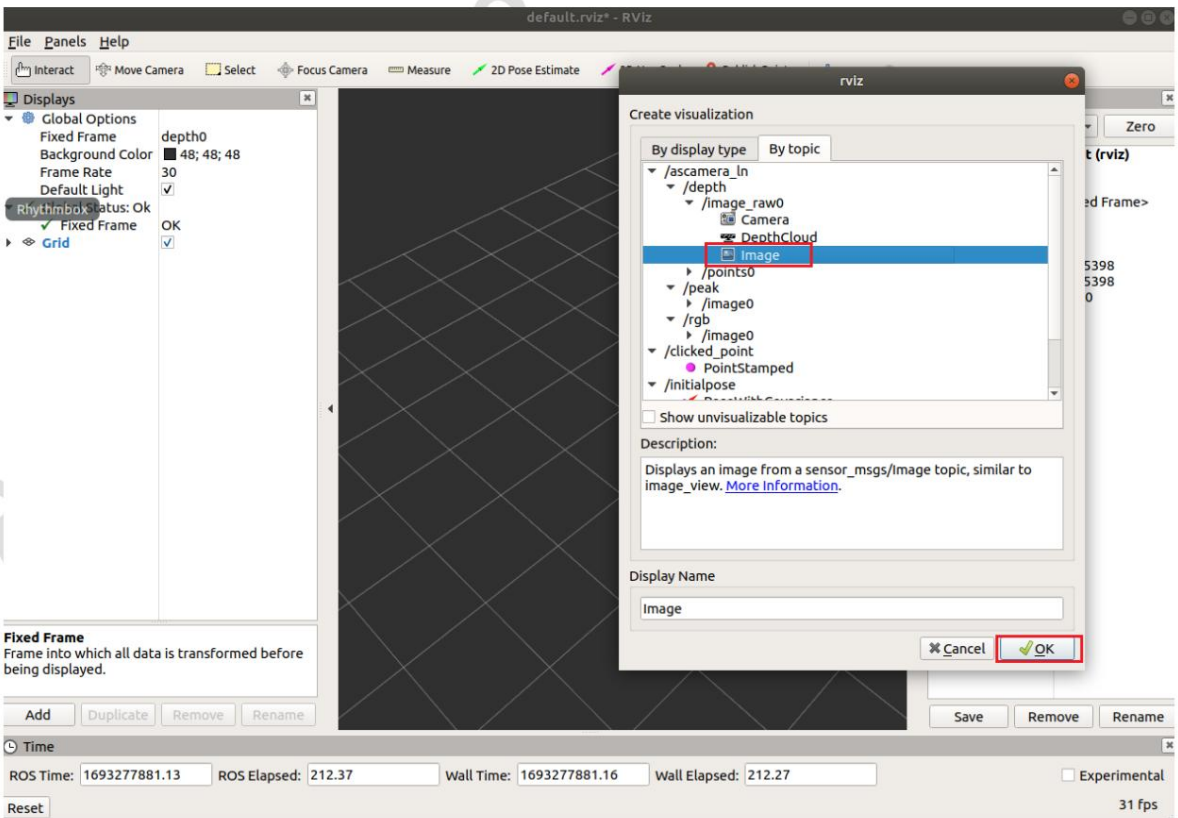
2. After starting RVIZ, select the Fixed Frame you need in the Globa Options option of Displays, as shown in the figure below.



3. Add display options by clicking Add in the lower left corner of RVIZ, selecting By topic in the pop-up window, and adding the topic you want to display.

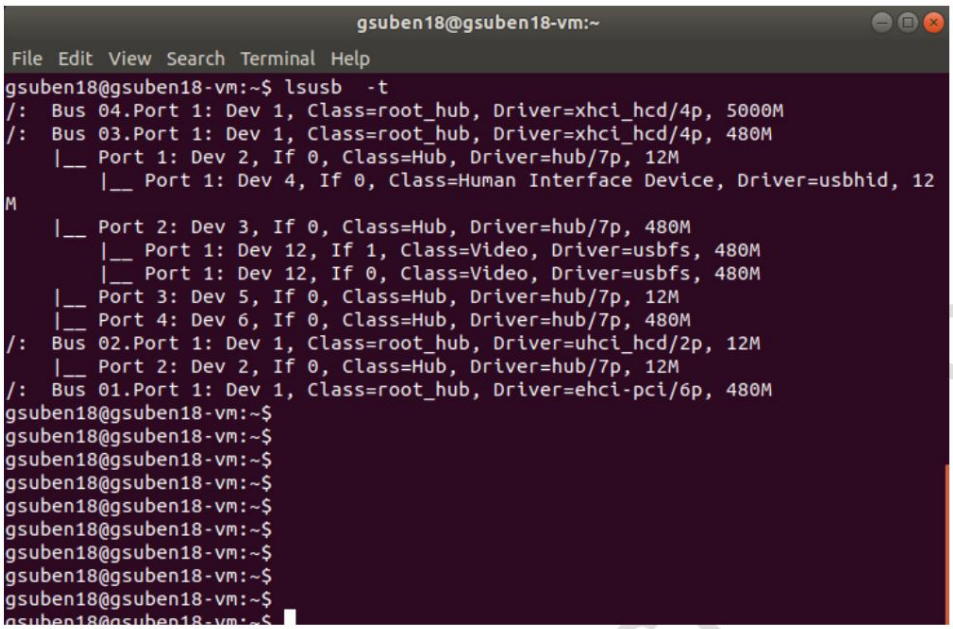


4. Taking viewing depth as an example, select Image under /xxx/depth/image_raw0 and click OK to add this topic to the display window.



7. Get device path information

After the device is mounted, enter lsusb -t to view the device node information. As shown in the following figure:



Among them, Bus corresponds to usb_bus_no in the launch file, and the Port under Bus corresponds to usb_path in the launch file.

Note that there are several layers of ports that need to be written in full. Take the above figure as an example, when the camera is turned on, the usb_bus_no is specified as "3" and the usb_path is "2.1".

8. Error code description

Error Code	describe
-80	libuvc initialization failed
-81	Failed to obtain camera list
-82	Failed to open the camera. The camera does not exist.
-83	Input parameter error, camera handle does not exist
-84	Failed to close the camera. The camera is not initialized or does not exist.
-85	Failed to open image stream, unsupported image type
-86	Failed to open the image stream, Libuvc stream opening error
-87	Failed to close the image stream, Libuvc failed to close the stream
-88	XU communication failed, setup request failed
-89	XU communication failed, get request failed
-90	Failed to obtain USB device information

9. FAQ

9.1 How to run a ROS node without root privileges

In Linux systems, root privileges are required to access devices. For programs that operate devices by operating device nodes in the /dev directory,

One way is to modify the permissions of the device node through the chmod command, but this is only temporary and has no effect on devices that are not operated through the /dev device node.

If the program is operating the device, the permissions cannot be changed through chmod. In this case, the permissions of the device can be managed through Linux udev and rules.

For Linux systems using the Ansjiang camera module, you can execute the scripts in the sample/linux_ros/ros/src/ascamera/scripts directory.

Create_udev_rules.sh script, you can run ROS node with normal permissions to access the Ansijiang camera. Modify sample/linux_ros/ros/

In the run_ascamera_node.sh script in the directory, comment out the statements that check and apply for root permissions.

```
CUR_DIR="$(dirname "$(realpath "${BASH_SOURCE[0]}")")"
# check for whitespace in $CUR_DIR and exit for safety reasons
grep -q "[[:space:]]" <<<"${CUR_DIR}" && { echo "\"${CUR_DIR}\" contains whitespace. Not supported. Aborting." >&2 ; exit 1 ; }

# if [ $EUID -ne 0 ]; then
#     echo -e "${RED}---This script requires root privileges, trying to use sudo${NORMAL}"
#     sudo "$CUR_DIR/run_ascamera_node.sh" "$@"
#     exit $?
# fi

if [ -f /opt/ros/melodic/setup.bash ]; then
    source /opt/ros/melodic/setup.bash
elif [ -f /opt/ros/kinetic/setup.bash ]; then
    source /opt/ros/kinetic/setup.bash
elif [ -f /opt/ros/noetic/setup.bash ]; then
    source /opt/ros/noetic/setup.bash
else
    echo "Error,Can't not found ros in /opt/"
fi

if [ "$(ps -aux | grep rosmaster | grep -v grep | wc -l)" -eq "0" ]; then
    roscore &
    sleep 3
fi
```

9.2 How to run a ROS2 node without root privileges

In Linux systems, root privileges are required to access devices. For programs that operate devices by operating device nodes in the /dev directory,

One way is to modify the permissions of the device node through the chmod command, but this is only temporary and will not work for devices that are not operated through the /dev device node.

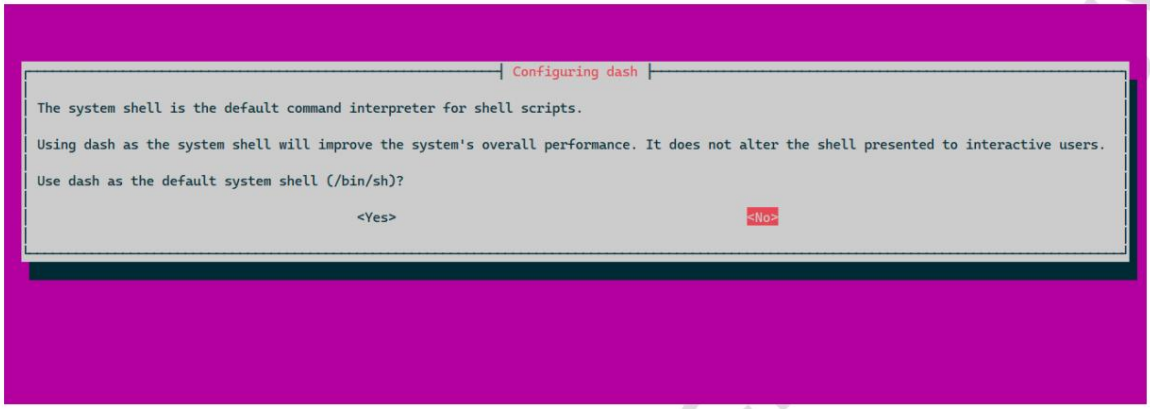
If the program is operating the device, the permissions cannot be changed through chmod. In this case, the permissions of the device can be managed through Linux udev and rules.

For Linux systems using the Ansjiang camera module, you can run the create_udev_rules.sh script in the sample/linux_ros/ros2/src/ascamera/scripts directory to run the ROS2 node with normal permissions to access the Ansjiang camera.

9.3 Executing the script provided by SDK and reporting an error

When executing the script provided by SDK, errors such as Bad substitution or Syntax error: redirection unexpected are reported. This is because the SDK provides The shell script for Ubuntu is bash shell, while the default shell parser for Ubuntu system is dash. Bash is more powerful than Dash, and some Bash syntax can be The default shell of Ubuntu can be changed to bash by following the steps below.

Execute sudo dpkg-reconfigure dash in the terminal and select NO in the pop-up window.



9.4 Compile error on ubuntu20.04/22.04 ROS

The following error is reported when compiling ROS (Noetic or higher) on ubuntu20.04/22.04.

Since our ROS node sample program is developed based on the melodic version of ROS on Ubuntu 18.04, when ported to ROS- When using Noetic, you can change -std=c++11 in the ros/src/ascamera/CMakeLists.txt file to -std==c++14 or -std==c++17

```
                 from /opt/ros/noetic/include/ros/time.h:58,
                 from /opt/ros/noetic/include/ros/ros.h:38,
                 from /home/boocax/roswork/src/as_nuwacam/src/as_nuwacam_node.cpp:29:
/usr/include/boost/mpl/aux_/preprocessed/gcc/minus.hpp:68:8: note:    'boost::mpl::minus'
   68 | struct minus
      |        ^~~~
In file included from /usr/include/pcl-1.10/pcl/point_types.h:44,
                 from /usr/include/pcl-1.10/pcl/common/impl/copy_point.hpp:41,
                 from /usr/include/pcl-1.10/pcl/common/copy_point.h:58,
                 from /usr/include/pcl-1.10/pcl/common/impl/io.hpp:45,
                 from /usr/include/pcl-1.10/pcl/common/io.h:586,
                 from /usr/include/pcl-1.10/pcl/io/file_io.h:41,
                 from /usr/include/pcl-1.10/pcl/io/pcd_io.h:44,
                 from /opt/ros/noetic/include/pcl_conversions/pcl_conversions.h:70,
                 from /home/boocax/roswork/src/as_nuwacam/src/as_nuwacam_node.cpp:38:
/usr/include/pcl-1.10/pcl/point_types.h:698:1: error: 'minus' is not a member of 'pcl::traits'
  698 | POINT_CLOUD_REGISTER_POINT_STRUCT (pcl::_PointDEM,
      | ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
/usr/include/pcl-1.10/pcl/point_types.h:698:1: note: suggested alternatives:
In file included from /usr/include/c++/9/string:48,
                 from /usr/include/c++/9/bits/locale_classes.h:40,
                 from /usr/include/c++/9/bits/ios_base.h:41,
                 from /usr/include/c++/9/ios:42,
                 from /usr/include/c++/9/ostream:38,
                 from /usr/include/c++/9/iostream:39,
                 from /home/boocax/roswork/src/as_nuwacam/src/as_nuwacam_node.cpp:21:
/usr/include/c++/9/bits/stl_function.h:177:12: note:    'std::minus'
  177 |     struct minus : public binary_function<_Tp, _Tp, _Tp>
      |            ^~~~
```

sample > ros > src > ascamera > CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0.2)
project(ascamera)

## Compile as C++11, supported in ROS Kinetic and newer
add_compile_options(-std=c++11)
add_definitions(-std=c++11)
# get gcc -v target
```

9.5 No published topics found using rviz2 on ros2

Cause: When method 9.2 is not executed, the script will run the program with root privileges, resulting in the failure of the script to run with root privileges on Ubuntu 22.04 or some versions.

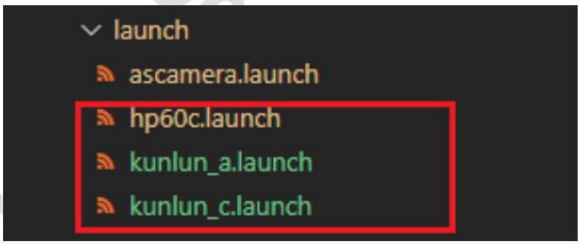
Running rviz2 with root privileges will result in failure to subscribe to topics published by the program.

Solution: Follow the method in 9.2 above.

9.6 Notes on using launch files for specific camera models

When using the launch file for a specific camera model (as shown in the figure), you need to modify the run_ascamera_node.sh script. The default script used is ascamera.lunch is a general launch file. Change the last line roslaunch ascamera ascamera.launch argPath:=$current_dir | tee $log_file to roslaunch ascamera xxx.launch argPath:=$current_dir | tee $log_file. (xxx is the launch file to be used.

document)



```
gcc_target=$(gcc -v 2>&1 | grep Target: | sed 's/Target: //g')
echo "Target: "${gcc_target}

cd src/ascamera/libs/lib/${gcc_target}
libPath=$(pwd)
export LD_LIBRARY_PATH=$libPath:$LD_LIBRARY_PATH
cd -

echo "lib path:"${libPath}

. devel/setup.bash

log_file=AngstrongsdkLog.txt

current_dir=$PWD"/src/ascamera/configurationfiles"

# rosrun ascamera ascamera_node
# roslaunch ascamera ascamera.launch argPath:=$current_dir | tee $log_file
roslaunch ascamera xxx.launch argPath:=$current_dir | tee $log_file
```

9.7 Modifying launch file parameters does not take effect

Possible causes: See 8.6

9.8 NUWA series cameras match abnormally in virtual machines

Since the NUWA series camera is composed of 2 USB devices combined into 1 camera (1 USB communication device + 1 UVC device),

Need to be matched; when running on a virtual machine, the USB device cannot be paired correctly because the device topology of the virtual machine may be incorrect, so

When using a virtual machine to run the program, only one NUWA series camera can be connected. If more than one camera is connected, the matching may fail due to the above reasons.

9.9 KUNLUN-A camera runs abnormally in the virtual machine

KUNLUN-A modules currently only support physical machines and cannot be used on virtual machines.

9.10 Abnormal streaming when running with multiple cameras

One possible cause of this problem is that the system kernel parameter usbfs_memory_mb is set too small. usbfs_memory_mb is a kernel parameter that specifies

Specifies the amount of memory used by the USB file system (usbfs). usbfs is a virtual file system used to transfer USB devices between user space and the kernel.

Data. This parameter specifies the memory size in MB that usbfs allocates to the USB device communication buffer. By default, this value is 16 MB.

By increasing or decreasing this value, you can adjust the memory size of the USB device communication buffer.

Temporary modification method (will become invalid after system restart): Change this kernel parameter to 64/128/512 or larger.

ÿÿÿecho "64" | sudo tee -a /sys/module/usbcore/parameters/usbfs_memory_mb

Customers can set the permanent modification method according to the system they are using.

9.11 Failed to open USB camera in Windows 10/Windows 11

If the camera fails to be opened on Windows, and the SDK log prints "SETUP – COM already setpup – threaded VI might not be possible". The reason for this problem is that the video stream capture SDK on Windows requires the use of SDK components, and the SDK is designed to use COM's multi-threaded mode. However, some software such as QT components also use COM components, but in single-instance mode. Different modes may cause the SDK camera to fail to open.

Because most of them need to disable the use of COM components, or use CoUninitializ(void) to deinitialize the COM components before using the SDK, and then Always start the SDK and let the SDK determine the mode of the com component.