YDLIDAR HP60C Compact Lidar Sensor User Manual

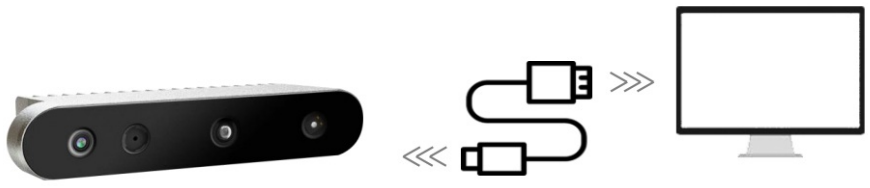
YDLIDAR HP60C Compact Lidar Sensor User Manual

Contents

OPERATION UNDER WINDOWS

Device Connection

When HP60C is evaluated and developed under windows, HP60C and PC need to be interconnected. The specific process is as follows:



How to Use EAIViewer

YDLIDAR provides EAIViewer, a visualization software for HP60C real-time scanning. Users can use this software to visually observe the HP60C scanning map. Visualization software download link: https://www.ydlidar.com/dowfile.html?cid=29&type=5

Data Display

After opening the software, RGB and Depth data will be displayed automatically, and the Depth information of the mouse position will be displayed in the lower left corner of the Depth window, as shown in the figure below:

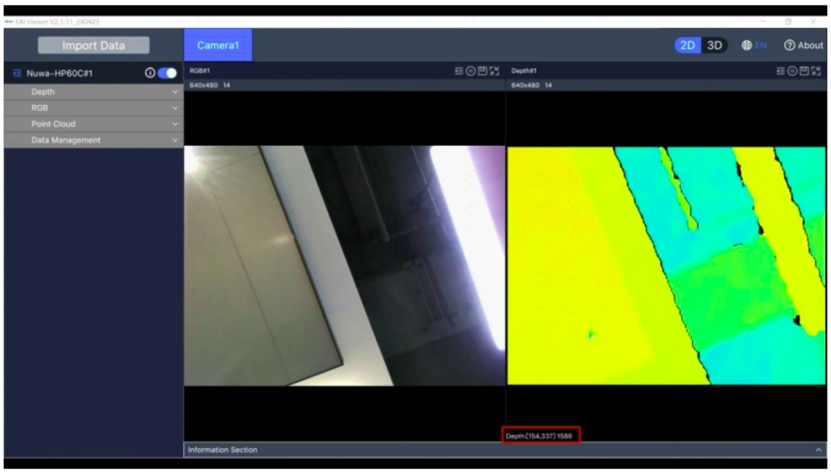
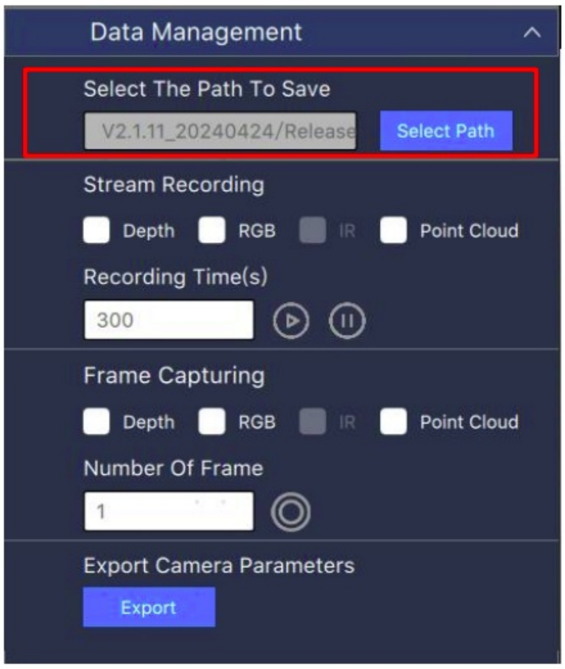


FIG 2 SOFTWARE INTERFACE

Data Management – Change The Save Address
Set the address to save the data.

By default, the data was stored in the same directory folder named “Nuwa – HP60C”.



Select The Path To Save: V2.1.11_20240424/Release
Stream Recording
Recording Time(s): 300
Frame Capturing
Number Of Frame: 1
Export Camera Parameters

FIG 3 CHANGE THE DATA SAVE ADDRESS

Record Data
Users can record Depth, RGB, Point Cloud data, and save it in the peer directory by default.
Save address Change refers to Data Management – Change the saved address.

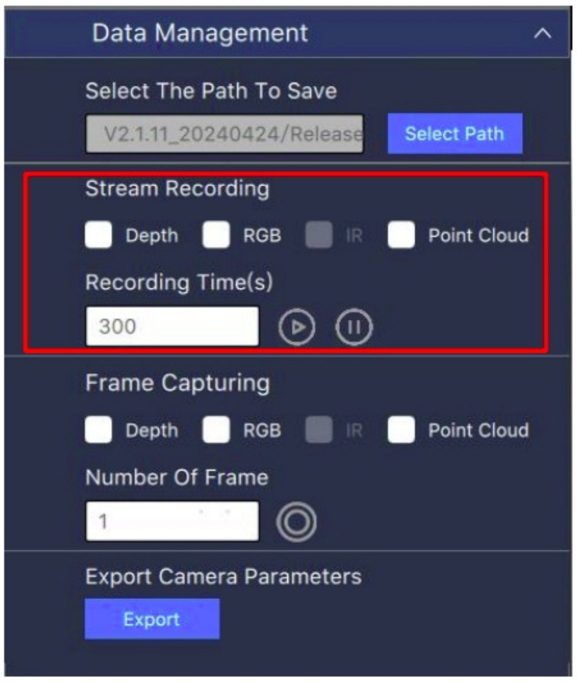


FIG 4 DISPLAY MEAN AND STANDARD DEVIATION

Save Image
Depth, RGB, and Point Cloud data can be saved in the peer directory by default; Save address Change refers to Data Management – Change the saved address.

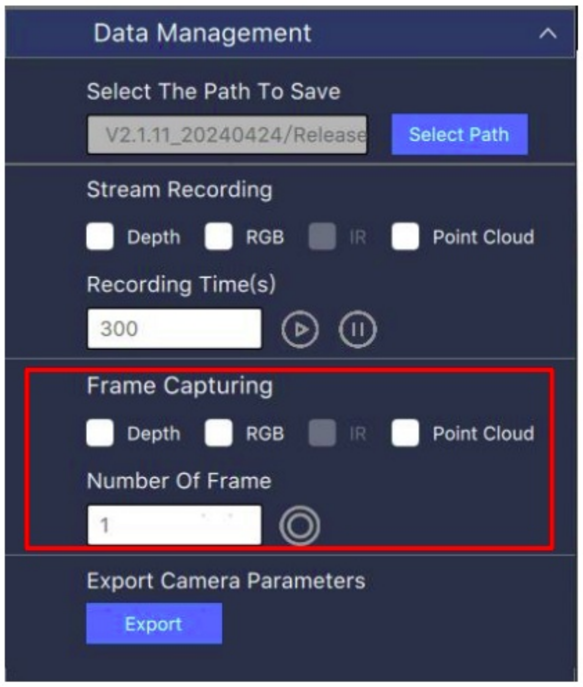
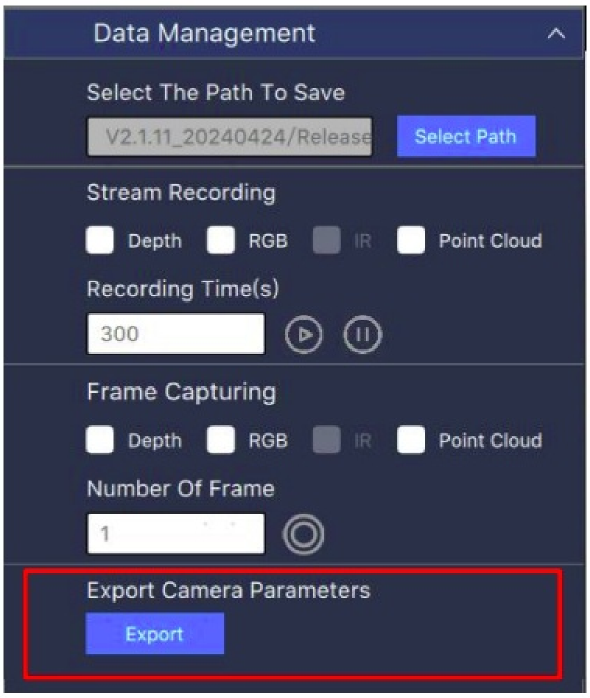


FIG 5 SAVE IMAGE

Save Camera Parameters
Camera parameters are saved in text format.


Data Management

Select The Path To Save

V2.1.11_20240424/Release

Stream Recording

Depth RGB IR Point Cloud

Recording Time(s)

300

Frame Capturing

Depth RGB IR Point Cloud

Number Of Frame

1

Export Camera Parameters

FIG 6 SAVE CAMERA PARAMETERS

LINUX ROS OPERATION

Run Ubuntu (Linux) System Example Program

Go to the sample/linux directory, where build.sh is the compile script and run_ascamera.sh is the script to run the sample program. root permissions are required to execute the program, which is run in sudo mode in the script.

Compile:
$ ./build.sh

Start node:
$ ./run_ascamera.sh

Run ROS System Sample Program

Go to the sample/ros directory, where build.sh is the compile script and run_ascamera.sh is the script for starting the node. Starting a node requires root permission, which is run in sudo mode in the script.

Compile:
$ ./build.sh

Start node:
$ ./run_ascamera.sh

Run ROS2 System Sample Program

Go to the sample/ros2 directory, where build.sh is the compile script and run_ascamera.sh is the script for starting the node. Starting a node requires root permission, which is run in sudo mode in the script.

Compile:
$ ./build.sh

Start node:
$ ./run_ascamera.sh

RVIZ Image Checking

When the node is started, open rviz to receive and view the image. The steps are as follows:

1. **Start the rviz node with the following command:**

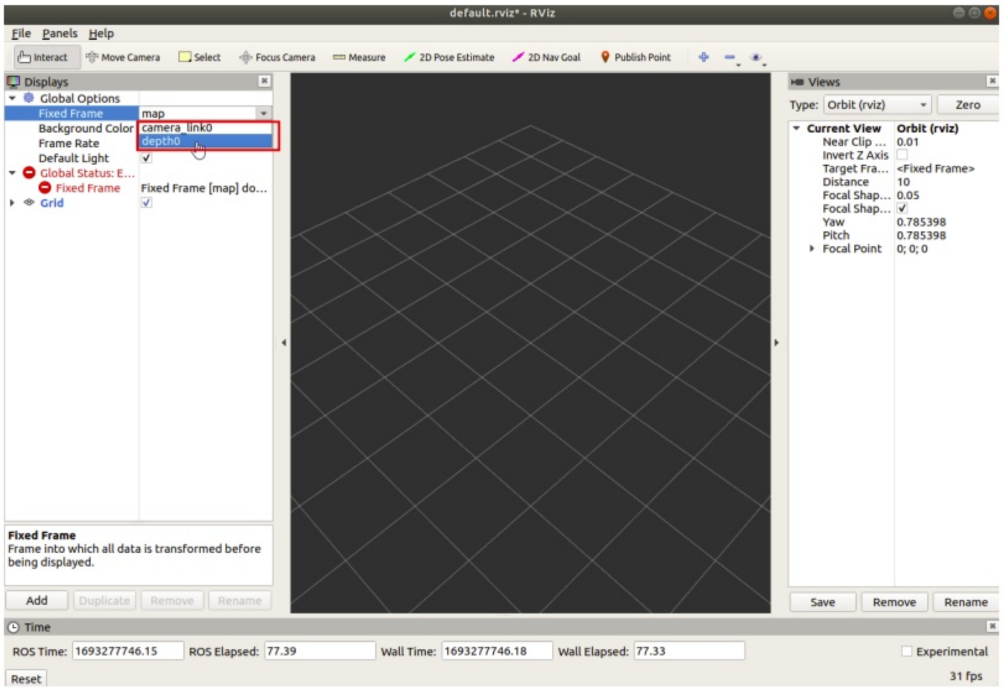
   ROS1:

   $ rosrun rviz rviz

   ROS2:

   $ ros2 run rviz2 rviz2

2. Select the Fixed Frame the Users want in the Globa Options of displays in RVIZ after startup, as shown below.



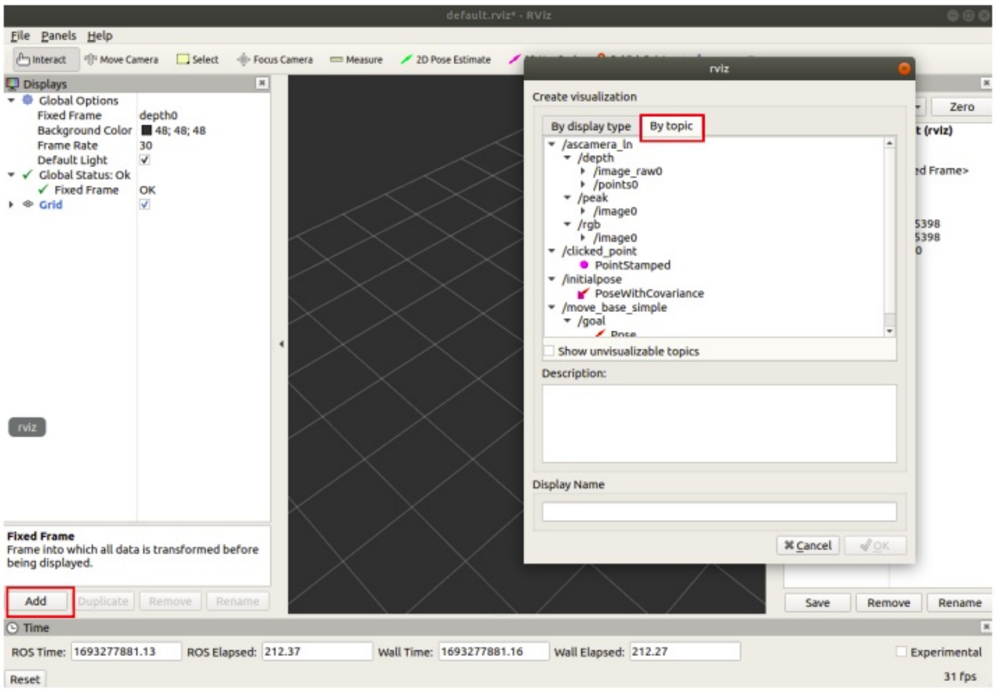
```
Global Options
  Fixed Frame        map
                     camera_link0
                     depth0
  Background Color
  Frame Rate
  Default Light      ✓
Global Status: E...
  Fixed Frame        Fixed Frame [map] do...
Grid                 ✓
```

Fixed Frame
Frame into which all data is transformed before being displayed.

```
Type: Orbit (rviz)
Current View    Orbit (rviz)
  Near Clip ...   0.01
  Invert Z Axis
  Target Fra...   <Fixed Frame>
  Distance        10
  Focal Shap...   0.05
  Focal Shap...   ✓
  Yaw             0.785398
  Pitch           0.785398
  Focal Point     0; 0; 0
```

```
ROS Time: 1693277746.15   ROS Elapsed: 77.39   Wall Time: 1693277746.18   Wall Elapsed: 77.33
31 fps
```

3. To Add display options, click [Add] in the lower left corner of RVIZ and select [By topic] in the pop-up window to add the topic Users want to display.



```
Global Options
  Fixed Frame        depth0
  Background Color   48; 48; 48
  Frame Rate         30
  Default Light      ✓
Global Status: Ok
  Fixed Frame        OK
Grid                 ✓
```

Fixed Frame
Frame into which all data is transformed before being displayed.

Create visualization

By display type [✓ By topic]

```
/ascamera_ln
  /depth
    /image_raw0
    /points0
  /peak
    /image0
  /rgb
    /image0
/clicked_point
    PointStamped
/initialpose
    PoseWithCovariance
/move_base_simple
  /goal
    Pose
```

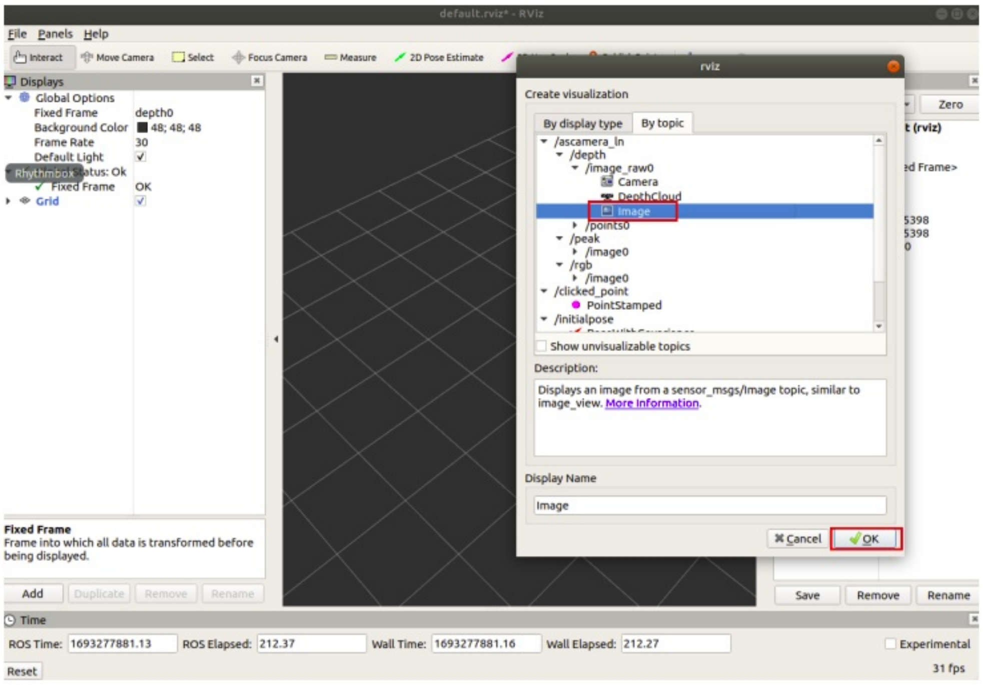
Show unvisualizable topics

Description:

Display Name

```
ROS Time: 1693277881.13   ROS Elapsed: 212.37   Wall Time: 1693277881.16   Wall Elapsed: 212.27
31 fps
```

4. To view depth as an example, select Image under /xxx/depth/image_raw0 and click [OK] to add the topic to the display window.

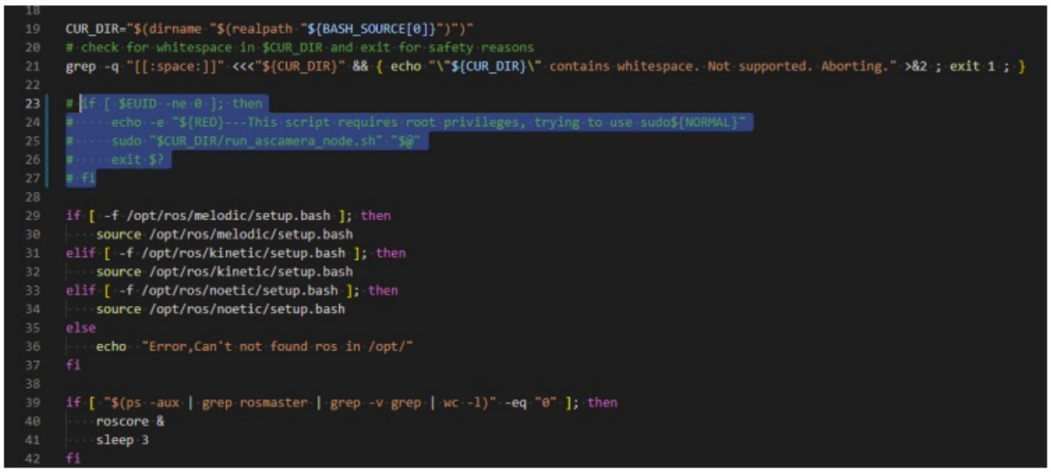


FAQ

How To Run ROS Nodes Without Root Permissions

In Linux, the root permission is required to access the device. For programs that operate the device node in the /dev directory, one way to change the device node permission is to run the chmod command, but this is only temporary. For programs that do not operate the device node in the /dev directory, the root permission is required. Permissions cannot be changed by chmod. In this case, users can use Linux udev and rules to manage device permissions.

For Linux system using HP60C phase module, the sample can be performed through/linux_ros/ros/SRC/ascomata create_udev_rules/scripts directory. Sh scripts, can operate by ordinary jurisdiction ros node access HP60C camera. Modify the run_ascamera_node.sh script in the sample/linux_ros/roes/ directory to check and apply for the comment of the root permission. As shown below



```
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
```

How to Run The Ros2 Node Without Root Permissions

In Linux, the root permissions is required to access the device. For programs that operate the device node in the /dev directory, one way to change the device node permission is to run the chmod command, but this is only temporary. For programs that do not operate the device node in the /dev directory, the root permission is required. Permissions cannot be changed by chmod. In this case, users can use Linux udev and rules rules to manage device permissions.

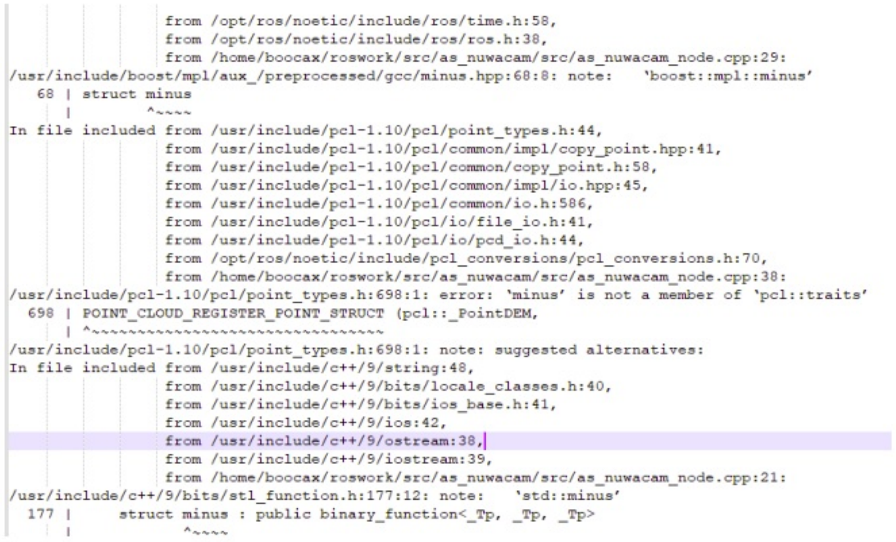
For Linux system using HP60C phase module, by holding the sample/linux_ros/ros2 / SRC/ascamera create_udev_rules/scripts directory. Sh scripts, The HP60C camera can be accessed by running the ROS2 node with normal permissions.

Executing Script Errors Provided By SDK

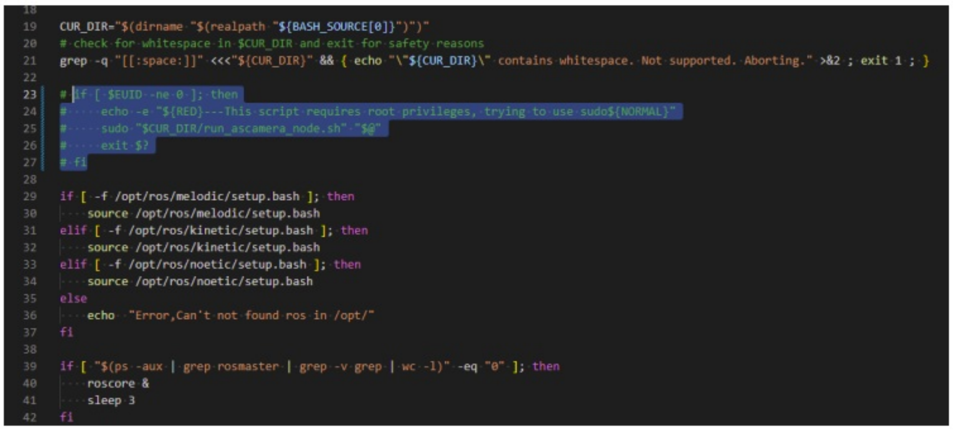
Execute errors such as script: Bad substitution or Syntax error: redirection unexpected provided by the SDK. This is because the shell script provided by the SDK is the bash shell, and the default shell parser for ubuntu is dash. Bash is more powerful than Dash, and some Bash syntax may not be parsed by dash. Users can change the default shell of ubuntu to bash by following these steps.
The terminal execution sudo DPKG to reconfigure the dash, select [NO] in a pop-up window.

Compilation Error On Ubuntu 20.04/22.04 ROS

ROS (Noetic or later) in Ubuntu 20.04/22.04 reported the following error. As a result of our node example program under the ROS is based on ubuntu18.04 melodic development version of ROS, when transplanted into ROS – Noetic, Ros can be/SRC/ascomata/CMakeLists. TXT file – STD = c + + 11 changes to – STD = = 14 c + + 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>
      |            ^~~~~
```



```
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
```

Using RVIZ2 On ROS2 Did Not Find The Posted Topic

Cause: When method 3.2 is not executed, the script will run the application with root permission, resulting in ubuntu22.04 or some versions of rviz2 without root permission will not be able to subscribe to the application published topics.

Solution: to perform the above 3.2.

Depth Camera Matching Exception Problem In Virtual Machine

Since HP60C camera is combined by two USB device for a camera (1 USB communication equipment + 1 UVC equipment), so users need to match; When running on a virtual machine, USB devices cannot be correctly paired because the device topology of the virtual machine may be incorrect. Therefore, when running a program on a virtual machine, only one HP60C camera can be connected. If more than one camera is used, the matching may fail due to the above reasons.

Multiple Cameras Run Abnormal Flow

One possible reason for this issue is that the system kernel parameter usbfs_memory_mb is set too small. Usbfs_memory_mb is a kernel parameter used to specify the memory size used by the USB file system (USBFS). USBFS is a virtual file system used to transfer USB device data between user space and the kernel. This parameter specifies the memory size, in MB, allocated by usbfs 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 fail after system restart): Increase this kernel parameter to 64/128/512 or higher.
Command: echo “64” | sudo tee – a/sys/module/usbcore/parameters/usbfs memory_mb The permanent modification method can be set by customers according to the system they are using.

Revise

Date	Version	Content
2024-06-05	1.0	The 1st release

YDLIDAR

Documents / Resources

YDLIDAR HP60C USER MANUAL	[YDLIDAR HP60C Compact Lidar Sensor](#) [pdf] User Manual HP60C, HP60C Compact Lidar Sensor, Compact Lidar Sensor, Lidar Sensor, Sensor

References

- User Manual