Orbbec ROS SDK

Orbbec ROS SDK is a wrapper for OrbbecSDK that supports ROS Kinetic, Melodic, and Noetic distributions.
It enables smooth integration of Orbbec 3D cameras into ROS projects.

Table of Contents

Install Dependencies

ROS

- Please refer directly to the ROS wiki for installation instructions.

Other Dependencies

- Install dependencies (be careful with your ROS distribution):

```
# Assuming you have sourced the ROS environment, same below
sudo apt install libgflags-dev ros-$ROS_DISTRO-image-geometry
ros-$ROS_DISTRO-camera-info-manager \
ros-$ROS_DISTRO-image-transport ros-$ROS_DISTRO-image-publisher libgoogle-
glog-dev libusb-1.0-0-dev libeigen3-dev
```

Create ROS Workspace and Build

Create a ROS workspace (if you don't have one):

```
mkdir -p ~/ros_ws/src
```

Get the source code:

```
# Go to where your OrbbecSDK_ROS_xxx.tar.gz is located, xxx is the version, which
may be different
tar -xvf OrbbecSDK_ROS_xxx.tar.gz -C ~/ros_ws/src/
```

Build the package:

```
cd ~/ros_ws
catkin_make
```

Install udev rules:

```
cd ~/ros_ws
source ./devel/setup.bash
roscd orbbec_camera
cd scripts
sudo cp 99-obsensor-libusb.rules /etc/udev/rules.d/99-obsensor-libusb.rules
sudo udevadm control --reload && sudo udevadm trigger
```

Start the Camera

In terminal 1:

```
source ./devel/setup.bash
roslaunch orbbec_camera gemini2R.launch
```

In terminal 2:

```
source ./devel/setup.bash
rviz
```

Select Topics and Control the Camera

Check topics, services, and parameters (open a new terminal):

```
rostopic list
rosservice list
rosparam list
```

Get camera parameters (*MUST* start stream first):

```
rosservice call /camera/get_camera_params "{}"
```

Check camera parameters (please refer to the ROS documentation for the meaning of specific fields in
the camera info):

```
rostopic echo /camera/depth/camera_info
rostopic echo /camera/color/camera_info
```

Check device information:

```
rosservice call /camera/get_device_info "{}"
```

Get the SDK version (includes firmware and Orbbec SDK versions):

```
rosservice call /camera/get_sdk_version "{}"
```

Set/get (auto) exposure:

```
rosservice call /camera/set_color_auto_exposure '{data: false}'
rosservice call /camera/set_left_ir_auto_exposure "{data: false}"

# Setting exposure values (be careful with the data range; the following example
may not be correct)
rosservice call /camera/set_left_ir_exposure "{data: 2000}"
rosservice call /camera/set_color_exposure "{data: 2000}"

# Get exposure
rosservice call /camera/get_left_ir_exposure "{}"
rosservice call /camera/get_color_exposure "{}"
```

Set/get gain:

```
# Get gain
rosservice call /camera/get_color_gain '{}'
rosservice call /camera/get_left_ir_gain '{}'

# Setting the gain (be careful with the data range; the following example may not
be correct)
rosservice call /camera/set_color_gain "{data: 200}"
rosservice call /camera/set_left_ir_gain "{data: 200}"
```

Set/get (auto) white balance:

```
rosservice call /camera/set_auto_white_balance "{data: false}"
rosservice call /camera/get_auto_white_balance "{data: false}"
```

Turn on/off laser:

```
rosservice call /camera/set_laser '{data: true}' # Turn on
rosservice call /camera/set_laser '{data: false}' # Turn off
```

Save images:

```
rosservice call /camera/save_images "{}"
```

Save point cloud:

```
rosservice call /camera/save_point_cloud "{}"
```

NOTE: The images are saved under `~/.ros/image` and are only available when the sensor is on.

Available Services for Camera Control

The service names intuitively reflect their purposes. It's crucial to understand that services related to setting or getting parameters—denoted as `set_*` and `get_*`—become available only when the respective `enable_*` parameters are activated. For instance, enabling features such as left infrared (IR) with `enable_left_ir`, right IR with `enable_right_ir`, depth sensing with `enable_depth`, or color processing with `enable_color` (refer to Launch Parameters) is a prerequisite for their corresponding services to be operational. This configuration ensures that services are accessible only when their specific stream is enabled in the launch file's stream argument.

- `/camera/get_auto_white_balance`
- `/camera/get_camera_params`
- `/camera/get_color_auto_exposure`
- `/camera/get_color_camera_info`
- `/camera/get_color_exposure`
- `/camera/get_color_gain`
- `/camera/get_depth_auto_exposure`
- `/camera/get_depth_camera_info`
- `/camera/get_depth_exposure`
- `/camera/get_depth_gain`
- `/camera/get_device_info`
- `/camera/get_device_type`
- `/camera/get_left_ir_auto_exposure`
- `/camera/get_left_ir_camera_info`
- `/camera/get_left_ir_exposure`
- `/camera/get_left_ir_gain`
- `/camera/get_serial`
- `/camera/get_sdk_version`
- `/camera/get_white_balance`
- `/camera/reset_color_exposure`
- `/camera/reset_color_gain`
- `/camera/reset_depth_exposure`
- `/camera/reset_depth_gain`
- `/camera/reset_left_ir_exposure`
- `/camera/reset_left_ir_gain`
- `/camera/reset_white_balance`

- `/camera/save_images`
- `/camera/save_point_cloud`
- `/camera/set_auto_white_balance`
- `/camera/set_color_auto_exposure`
- `/camera/set_color_exposure`
- `/camera/set_color_gain`
- `/camera/set_depth_auto_exposure`
- `/camera/set_depth_exposure`
- `/camera/set_depth_gain`
- `/camera/set_flood`
- `/camera/set_left_ir_auto_exposure`
- `/camera/set_left_ir_exposure`
- `/camera/set_left_ir_gain`
- `/camera/set_laser`
- `/camera/set_white_balance`

Available Topics

- `/camera/color/camera_info`: The color camera info.
- `/camera/color/image_raw`: The color stream image.
- `/camera/depth/camera_info`: The depth camera info.
- `/camera/depth/image_raw`: The depth stream image.
- `/camera/depth/points`: The point cloud, only available when `enable_point_cloud` is `true`.
- `/camera/depth_registered/points`: The colored point cloud, only available when `enable_colored_point_cloud` is `true`.
- `/camera/left_ir/camera_info`: The left IR camera info.
- `/camera/left_ir/image_raw`: The left IR stream image.
- `/camera/right_ir/camera_info`: The right IR camera info.
- `/camera/right_ir/image_raw`: The right IR stream image.

Launch Parameters

The following launch parameters are available:

- `connection_delay`: The delay time in milliseconds for reopening the device. Some devices require
  a longer time to initialize, and reopening the device immediately can cause firmware crashes when hot-plugging.
- `enable_point_cloud`: Enables the point cloud.
- `enable_colored_point_cloud`: Enables the RGB point cloud.
- `color_width`, `color_height`, `color_fps`: The resolution and frame rate of the color stream.

- `left_ir_width`, `left_ir_height`, `left_ir_fps`: The resolution and frame rate of the left IR stream.
- `right_ir_width`, `right_ir_height`, `right_ir_fps`: The resolution and frame rate of the right IR stream.
- `depth_width`, `depth_height`, `depth_fps`: The resolution and frame rate of the depth stream.
- `enable_color`: Enables the RGB camera.
- `enable_depth`: Enables the depth camera.
- `enable_left_ir`: Enables the left IR camera.
- `enable_right_ir`: Enables the right IR camera.
- `depth_registration`: Enables hardware alignment of the depth frame to the color frame. This field is required
  when `enable_colored_point_cloud` is set to `true`.
- `log_level` for OrbbecSDK controls console log verbosity, with levels `none`, `info`, `debug`, `warn`, `fatal`. Logs
  save in `~/.ros/Log`. For file logging, adjust `<FileLogLevel>` in
  `config/OrbbecSDKConfig_v1.0.xml`.
- `oredered_pc`: Whether the point cloud should be organized in an ordered grid (`true`) or as an unordered set of points (`false`).
- `device_preset`: The device preset options are `Default` or `High Accuracy`.
- `enable_decimation_filter`: This filter effectively reduces the depth scene complexity. The filter runs on kernel sizes [2x2] to [8x8] pixels. The image size is scaled down proportionally in both dimensions to preserve the aspect ratio.
- `enable_hdr_merge`: This filter is used jointly with depth HDR function. By merging consecutive depth images of alternating exposure values, we can over come challenges in acquiring depth values for under-illuminated and over-illuminated objects simultaneously.
- `enable_sequenceID_filter`: This filter is used jointly with depth HDR function and outputs only the sequence with specified sequence ID.
- `enable_threshold_filter`: This filter preserves depth values of interest and omits depth values out of scope.
- `enable_noise_removal_filter`: This filter removes speckle noise in clusters and gives rise to a less-filled depth map.
- `enable_spatial_filter`: This filter performs multiple iterations of processing as specified by the magnitude parameter, to enhance the smoothness of depth data. It is also capable of filling small holes in depth maps.
- `enable_temporal_filter`:This filter is intended to improve the depth data persistency by manipulating per-pixel values based on previous frames. The filter performs a single pass on the data, adjusting the depth values while also updating the tracking history.
- `enable_hole_filling_filter`: This filter fills all holes in the depth map using the specified mode.

Frequently Asked Questions

No Picture from Multiple Cameras

- It's possible that the power supply is insufficient. To avoid this, do not connect all cameras to the same hub and use
  a powered hub instead.
- It's also possible that the resolution is too high. To resolve this, try lowering the resolution.

Error: "Failed to start device: usbEnumerator createUsbDevice failed!"

- The current device does not have permission to access the device. Check the PID of the current device:

  ```
  lsusb | grep 2bc5
  ```

  Your output should look like this:

  ```
  Bus 002 Device 007: ID 2bc5:your_pid_here
  ```

  Edit `/etc/udev/rules.d/99-obsensor-libusb.rules` and add the following line:

  ```
  SUBSYSTEM=="usb", ATTR{idProduct}=="your_pid_here", ATTR{idVendor}=="2bc5",
  MODE:="0666", OWNER:="root", GROUP:="video", SYMLINK+="your_device_name_here"
  ```

  Replace `your_pid_here` with the PID of your device and `your_device_name_here` with the name you want to create for the device (e.g., `gemini2R`).

  Then restart the udev service:

  ```
  sudo udevadm control --reload-rules && sudo service udev restart && sudo
  udevadm trigger
  ```

License