Waymo Lidar ROS driver (package: hc)

Overview

The hc package provides a ROS driver for the Waymo lidar. Please consider that the package is in early development stage and subject to frequent changes.

Setup

Network

Connect the lidar to a free ethernet port of your workstation. The corresponding network card has to be configured with a static IP.

Workspace

Setup your ROS environment (http://wiki.ros.org/ROS/Tutorials/InstallingandConfiguringROSEnvironment). Add the directory which includes this file to your ROS_PACKAGE_PATH.

```
cd <directory-including-this-file>
export ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:$(pwd)
```

Nodes

hc_node

The hc_node is the main driver node. It detects and configures the laser automatically at startup.

Command-line Arguments

The hc_node supports the following command-line arguments:

- `--opensourcelicenses`: Prints the open-source license notices and exits.

Parameters

The hc_node supports the following parameters:

- `hostname`: The IP address or hostname to assign to your HC lidar device. By default the driver will automatically detect and configure the laser, but you can use this option to specify a particular device if you have multiple devices connected.
- `mac_address`: If you have multiple HC lidar devices on the same subnet, you can disambiguate which device you are configuring by specifying the MAC address.

These parameters can be set in several ways:

1. On the command-line, using the `_parameter_name:=value` syntax.
2. Using the `rosparam` command-line tool.
3. A `param` attribute inside the `node` element (see `dual_hc.launch` for an example).

Dynamic Parameters

The hc_node supports the following dynamically reconfigurable parameters:

- `frame_id`: The ROS frame_id for the generated point cloud.
- `fov`: Specify the horizontal field of view in degrees (0-360 degrees).
- `direction`: The center of the field of view in degrees (-180 - 180 degrees) zero is straight ahead.
- `spin_frequency`: The spin frequency of the device in Hz (5.0 - 15.0 Hz).
- `vertical_scan_frequency`: The rate at which the device sweeps out vertical points (1477.5 - 1500 Hz).
- `scans_per_point_cloud`: The maximum number of scans to include in each PointCloud2 message. The default, 0, means all of the scans in a single complete 360-

degree revolution of the Honeycomb device.

- `pitch_table`: The named pitch table used to define vertical shot pattern.
- `custom_pitch_table`: If `pitch_table` is set to `Custom`, then use this comma-separated list of angles (in degrees) as the pitch table.
- `interlaced`: Increase vertical resolution by interlacing shots between programmed pitch table entries.
- `sides`: Generate shots on the front-side only, back-side only, or both sides.
- `compute_normals`: Estimate normals at each point and publish them along with the point cloud.
- `drop_noise_returns`: Remove returns that the driver has determined to be artifacts from the point cloud.
- `publish_normal_markers`: Publish visualization markers for the normals. The parameter `compute_normals` must be enabled.
- `publish_imu_data`: Publish samples from the inertial measurement unit.
- `publish_spherical_coords`: Publish `yaw` and `pitch` with the point cloud.
- `publish_beam_side`: Publish which side of the device the shot was taken from.
- `publish_raw_intensity`: Publish raw intensity information in the point cloud in addition to the calibrated intensity value.
- `publish_pulse_width`: Publish pulse width information in the point cloud.
- `publish_return_index`: Publish the `return_index` in the point cloud to enumerate, temporally, the first, second, and third returns (if multiple returns were detected).
- `publish_return_state`: Publish the `noise` return state in the point cloud to distinguish between real returns and returns the driver has classified as artifacts.

Dynamic parameters can be manipulated while the driver is running in several ways:

1. Using the dynamic reconfigure GUI. Simply start the rqt_reconfigure node:

```
rosrun rqt_reconfigure rqt_reconfigure
```

2. Using the `dynparam` program. For more information, run:

```
rosrun dynamic_reconfigure dynparam
```

Topics

- /points (std_msgs/PointCloud2 (http://docs.ros.org/kinetic/api/sensor_msgs/html/msg/PointCloud2.html))

  The point data of a set of "scans". By default, the point cloud includes an entire spin's worth of scans (a single revolution of the Honeycomb Lidar). The PointCloud2 message is published wtih XYZ position and intensity values. Dynamic parameters provide the ability to include additional fields in the PointCloud2 message. The header timestamp represents the middle between the start and the end of a spin. Currently, measurements are not corrected for motion during data acquisition.

- /normals (visualization_msgs/Marker (http://docs.ros.org/kinetic/api/visualization_msgs/html/msg/Marker.html))

  Normals to the point cloud are estimated and published as visualization markers when both `compute_normals` and `publish_normal_markers` are enabled.

- /imu/data_raw (std_msgs/Imu (http://docs.ros.org/kinetic/api/sensor_msgs/html/msg/Imu.html))

  The Honeycomb incorporates an Inertial Measurement Unit (IMU) that provides linear acceleration in m/s^2 and angular velocity in radians/s.

- /diagnostics (diagnostic_msgs/DiagnosticStatus (http://docs.ros.org/kinetic/api/diagnostic_msgs/html/msg/DiagnosticStatus.html))

  The Honeycomb provides detailed diagnostic data on the /diagnostics topic. This data includes:

  - System information
  - System status
  - Detailed statistics
  - Error messages

Launch files

- launch/hc.launch

  The provided launch file starts the hc_node, dynamic reconfiguration GUI, and opens rviz with a pre-configured point visualization of the lidar data.

  ```
  roslaunch hc hc.launch
  ```

- launch/inverted_hc.launch

  Similar to hc.launch, but this launch file creates a `static_transform_publisher` to demonstrate how to run the lidar when it has been mounted upside-down.

  ```
  roslaunch hc inverted_hc.launch
  ```

- launch/dual_hc.launch

  This launch file demonstrates how to start two Waymo lidars at the same time.

  ```
  roslaunch hc dual_hc.launch
  ```

Common ROS Operations

Recording laser data to a file

With the lidar running (e.g. from one of the launch files above), you simply run the following command:

```
rosbag record -o <file prefix> /points
```

Playback laser data from a file

With `roscore` and `rviz` already running, run the following command:

```
rosbag play <bag file>
```

Converting captured laser data to CSV

If you have a bag file recorded, the following command will convert the point cloud stored in the file to comma-separated values (CSV):

```
rosrun hc pc2_to_csv <bag file>
```

Examining individual points in RViz:

1. Select **Panels** -> **Tools**, if the *Tools* panel is not visible.
2. Use the *Interact* tool to maneuver to the interesting points.
3. Switch to the *Select* tool.
4. Click and drag the selection rectangle to highlight the points you are interested in.
5. Select **Panels** -> **Selection** to open the *Selection* panel if it is not already open.
6. The individual point data is listed in the *Selection* panel.

Running the dynamic reconfigure GUI

If you started the laser with a launch file that doesn't include the reconfigure GUI, you can manually launch the GUI with the command:

```
rosrun rqt_reconfigure rqt_reconfigure
```

Alternatively, you can add the following line to your launch file to include the GUI in started on subsequent invocations of `roslaunch`:

```
<node name="rqt_reconfigure" pkg="rqt_reconfigure" type="rqt_reconfigure"/>
```

Troubleshooting

Can't run ROS commands, e.g. roslaunch

ROS is not in your `PATH` environment variable. Users typically run their ROS distribution's setup script to configure ROS for their shell. For example, in BASH you could include the following in your `.bashrc` file:

```
if [ -f /opt/ros/kinetic/setup.bash ]; then
  source /opt/ros/kinetic/setup.bash
fi
```

Can't find HC ROS package

If you get an error message similar to `[hc.launch] is neither a launch file in package [hc] nor is [hc] a launch file name`, it is likely that you haven't added the location of the HC ROS driver to your `ROS_PACKAGE_PATH` environment variable. Depending on where you installed the driver, run the commands:

```
cd <directory-including-this-file>
export ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:$(pwd)
```

Can't find HC device

If you get an error message that states `Unable to determine device hostname.`, the driver is not able to automatically detect the HC device. Double check to make sure that the device is connected to your computer as outlined in the manual, and that the interface has been configured with a static IP address.

License

Proprietary

Last updated 2020-05-28 UTC.