

ROS2 Package User Guide

This package is the ROS2 driver for CCH YB laser scanners. It supports acquiring scan data and configuring scanner parameters. The driver publishes standard `sensor_msgs::msg::LaserScan` messages, which applications can subscribe to in order to receive scan results.

This guide describes how to build and run the package, configure the network, display scan results, adjust parameters, and subscribe to scan data.

1. Building and running the package

Follow the standard ROS2 package build procedure:

1. Change to the workspace directory:

```
cd ~/dev_ws/
```

2. Build the package. If ROS2 is installed correctly, the build should complete successfully. If problems occur, check the ROS2 installation.

```
colcon build
```

3. Source the workspace setup file so that ROS2 can find the package:

```
source install/setup.sh
```

4. Optionally, run the following command and check that `qjkh_yb_udp2` is listed:

```
ros2 pkg list
```

5. Start the driver using the corresponding launch file:

```
ros2 launch qjkh_yb_udp2 qjkh_yb_udp_with_1_lidar_launch.py
```

Setup and visualization

1. Building and running (continued)

6. If compilation fails, insufficient permissions on the driver package files may be the cause. From `~/dev_ws/src`, change the permissions of the affected files as indicated by the error messages, or use the command below if the package folder is named `qjkh_yb_udp2`:

```
sudo chmod 775 -R qjkh_yb_udp2/
```

7. The launch file uses topic remapping by default. For a single scanner, the default topic name is `qjkh_scan`. Select the correct topic when displaying scan results in RViz2.

2. Network configuration

Before running the ROS2 package, configure the network connection as needed so that ROS2 can receive data from the scanner.

1. Change to the package directory:

```
cd ~/dev_ws/src/qjkh_yb_udp2
```

2. Open the relevant launch file and identify the YAML file it references in the config directory. Edit the required parameters in that YAML file. Refer to the product manual for the required values.

3. Viewing scan results in RViz2

Use RViz2 to visualize and debug scan results:

1. Start the scanner and the ROS2 driver.
2. Open another terminal and start RViz2:

```
ros2 run rviz2 rviz2
```

3. In the Displays panel on the left, click Add.
4. Select the By Topic tab in the dialog.
5. Select LaserScan under `/qjkh_scan`, then click OK.
6. Under Global Options in the Displays panel, set Fixed Frame to laser.

The scan should now appear in the RViz2 view.

4. Parameter configuration

Edit the YAML files in `~/dev_ws/src/qjkh_yb_udp2/config` to change the package parameters. After making changes, rebuild the package and restart the ROS2 driver.

Subscribing to scan data

5. Subscribing to scan data

The `qjkh_yb_udp2` driver publishes standard `sensor_msgs::msg::LaserScan` messages. The launch file remaps the scan topic to `qjkh_scan`. Subscribe to this topic to receive the scanner data. The example below logs the received message metadata.

```
#include <functional>
#include <memory>
#include <string>
#include "rclcpp/rclcpp.hpp"
#include "sensor_msgs/msg/laser_scan.hpp"

class Subscribe : public rclcpp::Node
{
public:
    // Constructor: the argument specifies the node name.
    Subscribe(std::string name) : Node(name)
    {
        subscription_ = this->create_subscription<
            sensor_msgs::msg::LaserScan>(
                "qjkh_scan", 10,
                std::bind(&Subscribe::command_callback, this,
                    std::placeholders::_1));
    }

private:
    // Subscription object.
    rclcpp::Subscription<sensor_msgs::msg::LaserScan>::SharedPtr
        subscription_;

    void command_callback(const sensor_msgs::msg::LaserScan msg)
    {
        RCLCPP_INFO(this->get_logger(),
            "QJKH_YB - Laser Scan message received:");
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - frame_id: %s",
            msg.header.frame_id.c_str());
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - timestamp: %d",
            msg.header.stamp.sec);
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - angle_min: %f",
            msg.angle_min);
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - angle_max: %f",
            msg.angle_max);
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - angle_increment: %f",
            msg.angle_increment);
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - time_increment: %f",
            msg.time_increment);
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - scan_time: %f",
            msg.scan_time);
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - range_min: %f",
            msg.range_min);
        RCLCPP_INFO(this->get_logger(), "QJKH_YB - range_max: %f",
            msg.range_max);
    }
};
```

Subscriber example and scan angle

```
int main(int argc, char **argv)
{
    rclcpp::init(argc, argv);
    // Create a shared pointer to the node.
    auto node = std::make_shared<Subscribe>("qjkh_yb_demo");
    // Run the node until shutdown is requested.
    rclcpp::spin(node);
    rclcpp::shutdown();
    return 0;
}
```

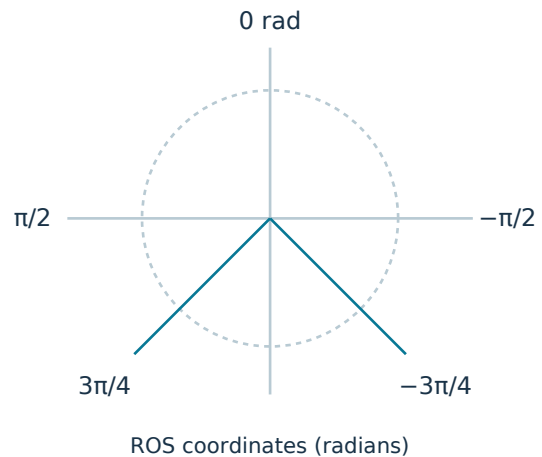
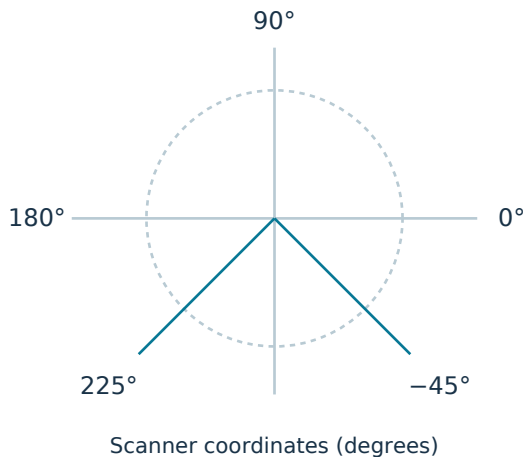
This code is provided for reference only. Implement the data processing required by your application.

6. Adjusting the scan angle

These settings control the angular range output by the ROS2 package. The default scanner angle range is -45° to 225° . Edit the YAML configuration in `~/dev_ws/src/qjkh_yb_udp2/config`, then rebuild and restart the driver.

The default configuration range is -2.356194 to 2.356194 radians (-0.75π to 0.75π), corresponding to scanner angles of -45° to 225° .

The scanner scans counterclockwise from -45° to 225° in its degree-based coordinate system. Standard `sensor_msgs::msg::LaserScan` messages use radians, with 0 radians pointing forward. The two coordinate systems have a 90° ($\pi/2$ radian) offset.



Apply the coordinate offset when converting between scanner degrees and ROS radians:

$$\text{Radians} = (\text{Degrees} \times \pi / 180) - \pi/2$$

$$\text{Degrees} = (\text{Radians} \times 180 / \pi) + 90$$

For example, to set the scanner angle range to 0° through 180° :

$$\text{Start angle: } (0 \times \pi / 180) - \pi/2 = -\pi/2 \text{ rad}$$

$$\text{End angle: } (180 \times \pi / 180) - \pi/2 = \pi/2 \text{ rad}$$