

ROS Package User Guide

This package is the ROS driver for CCH YB laser scanners. It supports acquiring scan data and configuring scanner parameters. The driver publishes standard `sensor_msgs::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 obtain debugging information.

1. Building and running the package

Follow the standard ROS package build procedure:

1. Change to the catkin workspace directory:

```
cd ~/catkin_ws/
```

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

```
catkin_make
```

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

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

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

```
rospack list
```

5. Start the driver using the corresponding launch file:

```
roslaunch qjkh_yb_udp qjkh_yb_udp_nodelet.launch
```

Setup and visualization

1. Building and running (continued)

6. If compilation fails, insufficient permissions on the driver package files may be the cause. From `~/catkin_ws/src`, try the command below, or change the permissions of individual files as indicated by the error messages:

```
sudo chmod 775 -R qjkh_yb_udp/
```

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 RViz.

2. Network configuration

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

1. Change to the launch file directory:

```
cd ~/catkin_ws/src/qjkh_yb_udp/launch
```

2. Open the relevant launch file and modify the hostname and port parameters. Refer to the product manual for the required values.

3. Viewing scan results in RViz

There are several ways to view scan results in ROS. The following procedure uses RViz for visualization and debugging.

1. Start the scanner and the ROS driver.
2. Open another terminal and start RViz:

```
roslaunch rviz rviz
```

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 RViz view.

Configuration and debugging

4. Parameter configuration

The package provides two types of parameter configuration: configure settings and launch settings. Configure settings are generally used for basic scanner parameters, while launch settings define startup parameters.

Configure defaults can be edited directly in the following file:

```
~/catkin_ws/src/qjkh_yb_udp/cfg/QjkhYb.cfg
```

Rebuild the package after changing this file for the changes to take effect. The driver also supports ROS dynamic reconfiguration. Run the command below to adjust settings at runtime, such as the scan angle or whether debug output is enabled:

```
roslaunch rqt_reconfigure rqt_reconfigure
```

Launch settings are primarily used to configure startup parameters. Restart the ROS package after changing the launch file.

5. Obtaining debugging information

Two types of debugging information are available: the driver's debug output and ROS logging output. Currently, the driver's debug output contains only scan results received from the scanner.

Enable or disable driver debug output in the configure settings. You can toggle it dynamically using:

```
roslaunch rqt_reconfigure rqt_reconfigure
```

Use `rqt_logger_level` to change the ROS logging level dynamically. Available levels are Debug, Info, Warn, Error, and Fatal. The default level is Info. Select a more detailed level if additional information is required.

6. Subscribing to scan data

The `qjkh_yb_udp` driver publishes standard `sensor_msgs::LaserScan` messages. Applications can receive scan data by subscribing to the published topic.

The driver's base topic is `scan`. The default launch file remaps it to `/qjkh_scan`, which is the topic used in the example on the next page.

Subscriber example

The following example subscribes to scan messages and prints the received data.

```
int main(int argc, char **argv)
{
    /* Initialize ROS system and node */
    ros::init(argc, argv, "qjkh_yb_demo");
    ros::NodeHandle nhQjkhYbMsg;
    ros::Subscriber subQjkhYbMsg = nhQjkhYbMsg.subscribe(
        "/qjkh_scan", 100, &OnQjkhYbMessageReceived);

    /* Let ROS take over */
    ros::spin();
}
```

A simple implementation of the callback `OnQjkhYbMessageReceived` is shown below:

```
void OnQjkhYbMessageReceived(const sensor_msgs::LaserScan &msg)
{
    ROS_INFO("qjkh_yb - Laser Scan message received:");
    ROS_INFO("qjkh_yb - frame_id: %s", msg.header.frame_id.c_str());
    ROS_INFO("qjkh_yb - timestamp: %d.%d",
        msg.header.stamp.sec, msg.header.stamp.nsec);
    ROS_INFO("qjkh_yb - angle_min: %f", msg.angle_min);
    ROS_INFO("qjkh_yb - angle_max: %f", msg.angle_max);
    ROS_INFO("qjkh_yb - angle_increment: %f", msg.angle_increment);
    ROS_INFO("qjkh_yb - time_increment: %f", msg.time_increment);
    ROS_INFO("qjkh_yb - scan_time: %f", msg.scan_time);
    ROS_INFO("qjkh_yb - range_min: %f", msg.range_min);
    ROS_INFO("qjkh_yb - range_max: %f", msg.range_max);

    /* Dump data */
    int dataCount = msg.ranges.size();
    ROS_INFO("qjkh_yb - Data Count: %d", dataCount);
    for(int index = 0; index < dataCount; index++)
    {
        if(((index + 1) % 60 == 0) && (index != 0))
        {
            printf("\n");
        }
        printf("%.2f ", msg.ranges[index]);
    }
    printf("\n\n");
}
```

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

Adjusting the scan angle

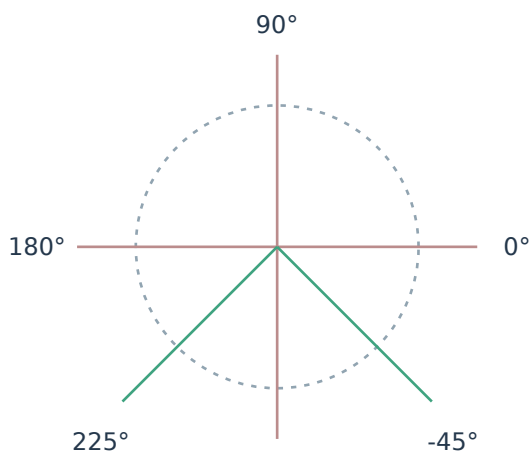
The angle settings described here control the angular range output by the ROS package. The default scanner angle range is -45° to 225° . Defaults are defined in:

```
~/catkin_ws/src/qjkh_yb_udp/cfg/QjkhYb.cfg
```

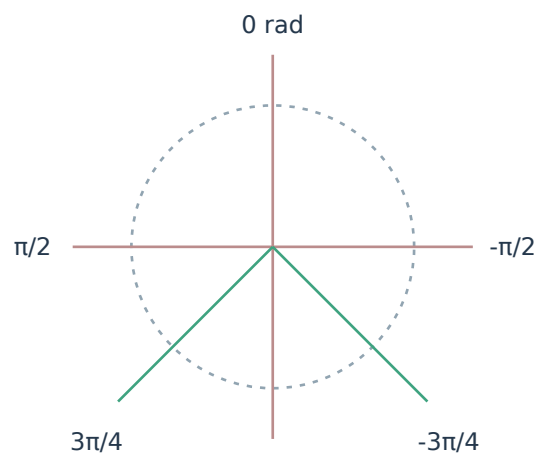
Run `roslaunch rqt_reconfigure rqt_reconfigure` to adjust the range dynamically. The default configuration range is -2.356194 to 2.356194 radians (-0.75π to 0.75π), corresponding to scanner angles of -45° to 225° .

Drag the start-angle and end-angle controls in `rqt_reconfigure` to observe the scan range changing in RViz. Alternatively, edit the configuration file above and rebuild the package.

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



Scanner coordinates (degrees)



ROS coordinates (radians)

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° :

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

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