ROS User Manual for 0x series robots

Nex Robotics Pvt. Ltd.

Version 2.1

Read sentences carefully which are marked with⚠caution symbol.

⚠Important:

User must go through hardware and software manuals before using robot.
This manual is intended for 0x series of robots.

⚠Safety precautions:

✔ Robot’s electronics is static sensitive. Use robot in static free environment.

✔ Do not access any part of the robot unless robot is in the anti static environment and user is wearing anti static strap.

✔ If robot’s battery low buzzer starts beeping, immediately charge the batteries.

✔ To prevent fire hazard, do not expose the equipment to rain or moisture.

✔ Refrain from dismantling the unit or any of its accessories once robot is assembled.

✔ Charge the battery only with the charger provided with the robot.

✔ Charge the battery in the open area and on the concrete or ceramic flooring.

✔ Never allow battery to deep discharge. If it is deep discharged, charger will refuse to charge the battery because of safety concerns.

✔ Mount all the components with correct polarity.

✔ Keep wheels away from long hair or fur.

✔ Keep your hands away from the wheels. Do not wear loose clothes while operating the robot. Loose cloth may get entangled in robot’s wheels and can cause serious injury.

✔ Keep the robot away from the wet areas. Contact with water will damage the robot.

✔ To avoid risks of fall, keep your robot in a stable position.

✔ Do not attach any connectors while robot is powered ON.

✔ Never leave the robot powered ON when it is not in use.

✔ Before operating the robot, make sure that you have access to at least “Class A/B” type fire extinguisher.

⚠Inappropriate Operation:

Inappropriate operation can damage your robot. Inappropriate operation includes, but is not limited to:

- ✔ Dropping the robot, running it off an edge, or otherwise operating it in an irresponsible manner.
- ✔ Interfacing new hardware without considering compatibility
- ✔ Overloading the robot above its payload capacity.
- ✔ Exposing the robot to wet environments.
- ✔ Continuing to run the robot after hair, yarn, string, or any other item has become entangled in the robot's axles or wheels.
- ✔ All other forms of inappropriate operation.
- ✔ Using robot in areas prone to static electricity.

Notice

The contents of this manual are subject to change without notice. All efforts have been made to ensure the accuracy of contents in this manual. However, should any errors be detected, NEX Robotics welcomes your corrections. You can send us your queries / suggestions at info@nex-robotics.com

- **Robot's electronics is static sensitive. Use robot in static free environment.**
- **Read the Robot's manual completely before start using this robot**

Recycling:

Almost all of the robot parts are recyclable. Please send the robot parts to the recycling plant after its operational life. By recycling we can contribute to cleaner and healthier environment for the future generations.

⚠User must go through hardware and software manuals before using 0x series of Robots.

Commonly Used Abbreviations in manual:

Short Forms	**Full Forms**
IR	*Infrared*
GPS	*Global Positioning System*
DGPS	*Differential Global Positioning System*
GPIO	*General Purpose Input Output*
S	*Success*
F	*Failure*
MSB	*Most Significant Byte*
LSB	*Least Significant Byte*
UL	*Ultrasonic*
WL	*White Line*
GUI	*Graphical User Interface*

⚠Important notes related to safety feature of robot:

Behavior during robot safety ON feature

1. Max velocity of robot gets limited to a safe value. This safe max velocity for 0x Delta robot is 0.25 m/Sec.

Behavior during robot safety OFF feature

1. Max velocity of robot can be altered up to robot's maximum reachable velocity.
2. Max reachable linear velocity of 0x Delta robot is +/- 1.282 m/Sec.
3. Max reachable angular velocity of 0x Delta robot is +/- 4.596 radians/Sec.

Table of Contents

1 Introduction

ros0xrobot provides ROS interface for most Nex Robotics robot bases in 0x series (pronounced as Ox) supported by 0xRobotCpp library. Information from the robot base, and velocity and acceleration control, is implemented via a ros0xrobot node, which publishes topics providing data received from the robot's embedded controller by 0xRobotCpp library, and sets desired velocity, acceleration and other commands in robot when new commands are received from command topics.

2 Installation and Getting Started

Instructions on installing ROS can be found here.

3 Reporting bugs and making feature requests

Bugs and requests can be reported here: rosdev@nex-robotics.com

4 ros0xrobot node

ros0xrobot is available under topic name `ros0xrobot`/topic name. You can remap these to different names if necessary. Please refer Names, Remapping Arguments and the <remap> tag for launch files for more information. The ros0xrobot node requires *0xRobotCpp* library for establishing communication with robot base. The library is available in the */lib* folder of source provided.

4.1 Subscribed topics

`cmd_vel` (geometry_msgs/Twist)

Receives new velocity commands. The received velocities will be updated and maintained by robot. There is no need to send the velocity repeatedly to maintain the velocity. If the robot is kept in timeout mode, then any communication/command needs to be sent before timeout to keep the robot moving.

4.2 Published topics

`pose` (nav_msgs/Odometry)

publishes odometry information (rate depends on the robot, normally 10Hz)

`sonar` (sensor_msgs/PointCloud)

publishes sonar readings. Requires enableSonar parameter to be set to true.

imu (sensor_msgs/Imu)

publish imu readings. Requires enableSonar parameter to be set to true.

4.3 Parameters

~port (string, default: /dev/ttyUSB0)
Serial port device the robot is is connected to. Set to /dev/ttyUSBx depending on port on which robot is connected.

CountsPerRev (`int`, default: 3840)
Set counts per revolution of the robot. Refer hardware manual of robot for appropriate value.

WheelDiameter (`float`, default: 100.0)
Set wheel diameter of robot in mm.

AxelLength (`float`, default: 290.0)
Set axle length in mm.

enableSonar (bool, default: false)
Enable or disable Sonar. This will cause the node to start publishing sonar messages when enabled.

enableImu (bool, default: false)
Enable or disable IMU. This will cause the node to start publishing imu messages when enabled.

4.4 Running ros0xrobot node

```
$rosrun ros0xrobot ros0xrobotNode
```

5 ROS node for Hokuyo laser range finder

The ROS node to provide access to SCIP 2.0-compliant Hokuyo laser range finders (including 04LX) is available on http://wiki.ros.org/hokuyo_node. To run this node a launch file named ros0xrobot_hokuyo.launch is provided in launch folder. To launch this node...

```
$roslaunch ros0xrobot  ros0xrobot_hokuyo.launch
```

You can modify the transform location in launch file to suit the location of sensor as per your choice.

6 ROS node for SICK LMS 1xx laser range finder

The lms1xx package that contains a basic ROS driver for the SICK LMS1xx line of LIDARs is available on http://wiki.ros.org/LMS1xx. To run this node a launch file named ros0xrobot_lms1xx.launch is provided in launch folder. To launch this node...

```
$roslaunch ros0xrobot  ros0xrobot_lms1xx.launch
```

The IP address of LMS 1xx and topic name for publishing messages can be specified in the launch file.

You can modify the transform location in launch file to suit the location of sensor as per your choice.

7 ROS node for IMU

The 9DOF imu unit can be accessed in ROS by using razor_imu_9dof node. Install the node by following the procedure given on ROS wiki here http://wiki.ros.org/razor_imu_9dof#Install_ROS_razor_imu_9dof_Package.
Connect the unit to host running ROS. It will be available over a communication port /dev/ttyUSBn, where n is the port number. Set the correct port number in config file named my_razor.yaml.

1. Go to imu node directory by

```
$roscd razor_imu_9dof/config
```

2. Open the config file named my_razor.yaml

```
$sudo nano my_razor.yaml
```

3. Modify the port parameter. Save the file.

A launch file is provided to run the node.

To run the node,

1. Open a terminal window and

```
$roslaunch razor_imu_9dof razor-pub.launch
```

2. The published messages can be seen by echoing the imu topic as follows

```
$rostopic echo /imu
```

8 ROS node for GPS

GPS access is provided through *nmea_navsat_driver.* For more information please refer http://wiki.ros.org/nmea_navsat_driver . We use a GPS module with 38400 baud rate and update rate of 5 Hz. A launch file is provided to run the node.

1. Open the terminal window and

```
$roslaunch ros0xrobot ros0xrobot_gps.launch
```

2. The published messages can be seen by echoing the gps topic as follows

```
$rostopic echo /fix
```

You can modify the transform location in launch file to suit the location of sensor as per your choice.

for using GPS for localization, please refer ...

http://wiki.ros.org/robot_localization/Tutorials/GPS%20Integration

9 ROS node for bumblebee XB3

The bumblebee XB3 package that contains a basic ROS driver for the Point Grey bumblebee XB3 stereo camera is available on https://github.com/ravich2-7183/bumblebee_xb3. To run this node a launch file named ros0xrobot_BumblebeeXB3.launch is provided in launch folder. To launch this node...

1. Open a new terminal window and type

```
$roslaunch ros0xrobot ros0xrobot_BumblebeeXB3.launch
```

2. Now running

```
$rostopic list
```

should show you image streams, which you can view with rqt_image_view

3. Now start the stereo_image_proc node in any of the THREE stereo_camera namespaces, say LEFT-CENTRE:

```
$ROS_NAMESPACE=/camera/stereo_camera_LC/ rosrun stereo_image_proc stereo_image_proc
```

You should see that the point cloud topic is now published as
/camera/stereo_camera_LC/points2

4. To view the point cloud, start rviz:

```
$rosrun rviz rviz
```

You can modify the transform location in launch file to suit the location of sensor as per your choice.

10 Launch files

The launch files are available for various devices present on robot. They are present in launch folder. The launch files will work only if the corresponding device is present on the robot.

1. ros0xrobot_minimal.launch : This launch file launches a basic robot node.
2. ros0xrobot_hokuyo.launch : This will launch robot node and hokuyo laser range finder node.
3. ros0xrobot_imu.launch : This will launch the robot node with imu enabled.
4. ros0xrobot_sonar.launch : This will launch robot node with sonar enabled.

5. ros0xrobot_gmapping_demo.launch : This will launch the gmapping demo for creating map of surrounding area. A laser ranging device is required for this.
6. amcl_demo.launch : This will launch the amcl node. It requires map file as input argument. This demonstrated the amcl based navigation of robot with known map.
7. ros0xrobot_stargazer.launch : This will launch the robot node with stargazer node.
8. ros0xrobot_gps.launch : This will launch the robot node with gps node.
9. ros0xrobot_all.launches : This will launch all the sensors on robot (sonar, imu, gps, stargazer, hokuyo). This requires all sensors to be present and switched ON.

11 Demos

11.1 Network setup and remote operation

11.1.1 Network setup

ROS requires bidirectional networking between all computers attached to the network and does not have security built in. Network for 0x robot and remote pc can be configured with their IP addresses. It can also be done with nameserver or /etc/hosts. Ultimately you will need to configure ROS_MASTER_URI and ROS_HOSTNAME correctly to ensure the ros communication channels can find each other.
ROS master always runs on 0x robot. Therefore the ROS_MASTER_URI of remote workstation must be set to ROS_MASTER_URI=http://<0xROBOT_IP>:11311

For Ex.

- On 0x robot:

ROS_MASTER_URI=http://localhost:11311
ROS_HOSTNAME=<0xROBOT_IP>

- On remote workstation:

ROS_MASTER_URI=http://<0xROBOT_IP>:11311
ROS_HOSTNAME=<WORKSTATION_IP>

If all the configuration is correct, the command **rostopic list** on workstation should display the topics running on 0x robot.

Note : If you get an error "Couldn't find an AF_INET address for [<WORKSTATION_NAME>]", then set ROS_IP=<WORKSTATION_IP> on workstation.
It will be convenient to add these network setting to respective *.bashrc* file on robot and workstation.

11.1.2 Launching the node from remote computer

An ROS node can be launched from remote PC by simply connecting to 0x robot PC over ssh. After connecting, go to launch folder and launch the desired node / launch file from command line.

11.2 Building a map with SLAM

This tutorial explains how to generate a map using gmapping.

11.2.1 Launch the gmapping app on robot

- Start a terminal Bring up the robot

```
$roslaunch ros0xrobot ros0xrobot_minimal.launch
```

- Run the gmapping demo app

```
$roslaunch ros0xrobot ros0xrobot_gmapping_demo.launch
```

11.2.2 On your workstation

This assumes you have ROS on your workstation and ROS_MASTER_URI has been set to point to your 0x Robot. Please remember to source the setup.bash from present workspace on workstation. In terminal on workstation, run source command
$source /home/*yourhome*/catkin_ws/devel/setup.bash. Also, make the ros0xrobot node on your workstation using catkin_make in your workspace. The workstation should have equivalent setup as that on the robot.

- Launch rviz

```
$roslaunch ros0xrobot build_map.launch
```

11.2.3 On the robot

- Drive the robot around by launching the tele-operation node using the launch file ros0xrobot_teleop.launch

```
$roslaunch ros0xrobot ros0xrobot_teleop.launch
```

- Save the map to file by

```
$rosrun map_server map_saver -f /home/nex/.../ros0xrobot/maps/map0
```

Note: Do not close the gmapping launch until saving the map.

11.3 Autonomous Navigation of a Known Map with 0x Robot

This demo assumes that you have a map of your working area available.

11.3.1 Launch the amcl app

- Bring up the robot

```
$roslaunch ros0xrobot ros0xrobot_minimal.launch
```

- Run the navigation demo app passing in your map file.

```
$roslaunch ros0xrobot amcl_demo.launch
map_file:=/home/nex/../ros0xrobot/maps/map0.yaml
```

11.3.2 On your workstation

This assumes you have ROS on your workstation, ros0xrobot node is present on workstation and is built using *catkin_make* and ROS_MASTER_URI has been set to point to your 0x Robot.
Go to launch folder. Launch rviz:

```
$roslaunch ros0xrobot ros0xrobot_navigation.launch
```

Note : Ideally you don't need the ros0xrobot node present on the workstation. All you need is the rviz configuration file in /rviz folder named ros0xrobot_navigation.rviz. You can then launch rviz directly as follows.

```
$rosrun rviz rviz -d /path/to/rviz/navigation/config/ ros0xrobot_navigation.rviz
```

11.3.3 In RVIZ

11.3.3.1 *Localize the 0x robot*

When starting up, the 0x robot does not know where it is. To provide him the approximate location on the map:

- Click the "2D Pose Estimate" button
- Click on the map where the 0x robot approximately is and drag in the direction the 0x robot is pointing.

You will see a collection of arrows which are hypotheses of the position of the 0x robot. The laser scan should line up approximately with the walls in the map. If things don't line up well you can repeat the procedure.

11.3.3.2 *Teleoperation*

The teleoperation node can be run simultaneously with the navigation stack. It will override the autonomous behavior if commands are being sent. It is often a good idea to teleoperate the robot after seeding the localization to make sure it converges to a good estimate of the position.

11.3.3.3 *Send a navigation goal*

With the 0x robot localized, it can then autonomously plan through the environment.

To send a goal:

- Click the "2D Nav Goal" button
- Click on the map where you want the 0x robot to drive and drag in the direction the 0x robot should be pointing at the end.

This can fail if the path or goal is blocked.

If you want to stop the robot before it reaches it's goal, send it a goal at it's current location.

Note : All the 0x robots have powerful drive trains. Running them into obstacles can cause severe damage to obstacles as well as robots. Use emergency stop button on robot in case of any such event. Most importantly, be careful.