Pioneer3DX Robot Operation
ROS indigo and Ubuntu 14.04 LTS (Trusty Tahr)

Jen Jen Chung

November 16, 2016

Abstract

This document describes the necessary ROS software packages for controlling a Pioneer P3-DX robot. This setup incorporates packages from the hardware manufacturers of the platform (Adept Mobile Robots) and the attached sensors (Slamtec, formerly RoboPeak), as well as built in and custom ROS packages. This setup has been tested on Ubuntu 14.04 LTS (Trusty Tahr) machines with the ROS indigo distribution. Future ROS distributions may have different requirements, users are encouraged to check online for the latest information at http://wiki.ros.org.

1 Installation

For ROS installation instructions see the relevant online documentation at http://wiki.ros.org/indigo/Installation/Ubuntu. A compiled set of instructions including details for downloading some of the custom ROS packages specific to this simulation can be found at http://people.oregonstate.edu/~chungje/Code/Pioneer3dx%20simulation/ros-indigo-gazebo2-pioneer.pdf.

Standard ROS packages that will also need to be installed include:

```
sudo apt-get install ros-indigo-navigation ros-indigo-gmapping ros-indigo-ros-control ros-indigo-ros-
    controllers ros-indigo-rviz
```

2 ROS Packages

Packages required for robot localisation and waypoint navigation:

- **rosaria**: https://github.com/amor-ros-pkg/rosaria.git
- **rplidar_ros**: https://github.com/robopeak/rplidar_ros.git
- **pioneer_2dnav**: https://github.com/JenJenChung/pioneer_2dnav
- **nav_bundle**: https://github.com/JenJenChung/nav_bundle
- **simple_navigation_goals**: https://github.com/JenJenChung/simple_navigation_goals
- **pioneer_test**: https://github.com/JenJenChung/pioneer_test.git

Git clone these repositories into your ~/catkin_ws/src folder and run `catkin_make`.

2.1 rosaria

Rosaria is developed by Adept Mobile Robots as a ROS interface to the Pioneer class robot platforms. This package contains the RosAria node which handles the message passing from the onboard sensors to the ROS master; it also subscribes to /cmd_vel messages and converts them to motor commands, which are executed by the platform. RosAria publishes the robot pose and sonar returns along with other diagnostic data such as the battery state.

2.2 rplidar_ros

This package is developed by Slamtec as a ROS interface to the RPLidar sensor. The rplidarNode node is initialised with parameters describing the serial port, frame name, etc., and publishes a LaserScan message under the topic name "scan". Note that the default serial port is /dev/ttyUSB0, however when using with the Pioneer P3-DX robots, this should be changed to /dev/ttyUSB1. The ROS computer should be connected **first to the platform and *then* to the lidar** to ensure the correct port ordering.

2.3 nav_bundle

This is a **launch file only** package; files in this package collate all nodes required to perform localisation and navigation functions. Typically this consists of nodes to handle localisation and nodes to move the platform to waypoints in a specified frame, see /nav_bundle/launch/nav_bundle.launch for a basic example. Each time you need a new configuration for an experiment (e.g. amcl vs. slam_gmapping, or unique costmaps parameters for move_base), write a new launch file, **do not overwrite existing files**.

If you are using the **map_server** to load a map, please also include the relevant launch files in this package. The convention used here is to keep the **map_server** nodes separate from the localisation and navigation nodes, this is a consideration for more streamlined launching when using multiple robots with a shared map. For more information, see http://wiki.ros.org/ROS/Tutorials/Roslaunch%20tips%20for%20larger%20projects.

2.4 gmapping

This is a built in ROS package that includes the slam_gmapping node. The slam_gmapping node subscribes to TFMessage and LaserScan messages (topic names: /tf and base_scan, respectively) to reconstruct the SLAM map online. This node publishes the map and updates the transform tree (between the odometry frame and the global map frame). See http://wiki.ros.org/gmapping for more information.

2.5 move_base

This is a built in ROS package that includes the move_base node. The node provides a number of services related to waypoint navigation and must be initialised with parameters which define the cost space, trajectory planner, etc. See http://wiki.ros.org/move_base for more information.

2.6 pioneer_2dnav

This is a **launch file only** package. This folder contains the file to launch move_base with the relevant configuration parameters loaded from the associated .yaml files in the same folder. The three "costmap" .yaml files contain parameters describing the planning space, for example, the sensor data from which to construct the obstacle map, the footprint of the robot, etc. The base_local_planner_params.yaml file defines the planner to be used and includes values relating to goal tolerances, maximum allowable velocities, etc. Be aware that many of these parameters much be loaded during the initialisation of move_base and cannot be changed dynamically through the command line.

2.7 simple_navigation_goals

This package contains two nodes, map_navigation_client and base_link_navigation_client, both of which generate an action client to the move_base service for sending waypoints in either the map frame or the base_link frame, respectively. The map_navigation_client subscribes to the map_goal topic and forwards the registered waypoint to the move_base node, similarly for the base_link_navigation_client. The action client also receives the move_base result and will print "Waypoint reached." or "The base failed to reach the waypoint." depending on the outcome.

2.8 pioneer_test

The generic test script, run-pioneer-robot, in the top level directory of this package will launch a basic set of nodes to operate the Pioneer P3-DX robot. This package also includes a PID controller node (currently only using P to slow down the forward velocity of the platform), and a python script to handle a commonly observed move_base error where the base is unable to turn sufficiently to execute the planned path. There are also a number of redundant tf broadcast and listener nodes, which can be ignored.