Robotic Operating System - ROS

Salomão Rodrigues Jacinto

June 14, 2016

Quick Guide and Explanation

We are currently using the Indigo version of ROS which is the last LTS release and has support for the needed packages until now. (ROS Jade has support too).

The basic concept of ROS is the message-passing between processes that is made by topics, for example we have topics publishing informations as odometry and we can also have topics that are subscribed to the odometry one to receive this information.

As we go through the packages used, this will be more easy to understand. You can find in the links below information about ROS, ROS topics and the page where you can find the tutorials made by ROS.

You probably can not copy and paste from this document because of the indentation, so you can follow the links given and copy from there, or can type yourself, beware of underscores and dashes.

- ROS Wikipedia
- ROS Topics
- ROS Tutorials

Installation

The complete installation guide can be found at:

- ROS Installation

As said before we are using Indigo version and to install it you will need Ubuntu 13.10 or 14.04. Ubuntu is the supported operating system, you can try

install in other distributions but beware you might need to downgrade or install aditional libraries.

1. Installation
   All repositories need to be allowed, "restricted", "universe" and "multiverse".

   (a) Setup sources.list and keys

   ```
   sudo sh -c '. /etc/lsb-release && echo "deb http://
       ros.fei.edu.br/archive-ros/packages.ros.org/ros/
       ubuntu $DISTRIB_CODENAME main" > /etc/apt/sources.
       list.d/ros-latest.list'
   sudo apt-key adv --keyserver hkp://ha.pool.sks-
       keyservers.net --recv-key 0xB01FA116
   sudo apt-get update
   ```

   (b) Install

   For full intallation use:

   ```
   sudo apt-get install ros-indigo-desktop-full
   ```

   For the robot installation, with no GUI tools use:

   ```
   sudo apt-get install ros-indigo-ros-base
   ```

   If you want to search for a packages you can use:

   ```
   sudo apt-cache search ros-indigo-{PAKCAGENAME}
   ```

   To easily install system dependencies we need to initialize rosdep:

   ```
   sudo rosdep init
   rosdep update
   ```

2. Configuration

   (a) To update the ROS environment variables automatically in every new shell:

   ```
   echo "source /opt/ros/indigo/setup.bash" >> ~/.bashrc
   source ~/.bashrc
   ```

   (b) Create a workspace and initialize it:

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws/src
catkin_init_workspace
cd ..
catkin_make
echo "source ~/catkin_ws/devel/setup.bash" >> ~/.
    bashrc
```

3. Robot and PC connection

   Both PC and the robot need to be connected to the same network and some variables need to be set for them to communicate with each other. Some command lines will be useful:

```
hostname #to find your pc name
hostname -I #to find your IP address
```

   And you will need to change this two files, " /.bashrc" and "/etc/hosts". In the bash file you add the lines below, and in the hosts you add the IP's followed by their hostnames. The bash and hosts files need to be updated every time beacuse UFSC has a dynamic IP.

   (a) ROS Variables on PC

```
ROS_IP={YOUR-IP}
ROS_HOSTNAME{YOUR-HOSTNAME}
ROS_MASTER_URI=http://{ROBOT-IP}.11311
```

   (b) ROS Variables on the Robot

```
ROS_IP={ROBOT-IP}
ROS_HOSTNAME{ROBOT-HOSTNAME}
ROS_MASTER_URI=http://{ROBOT-IP}.11311
```

   The first thing you will run is the main ros node, "roscore", with that running you can run the nodes from the packages below.

Packages Used

The RosAria and Sicktoolbox Wrapper need to run on the robot, so you can do that on the robot itself or through ssh.

I RosAria

The RosAria package is used to communicate ROS with ARIA which is the P3-DX library. It creates topics such as "cmdvel" to publish robot speed and "pose" to receive robots odometry information.

1. Installation

   (a) First you need to install ARIA, you can download in the link below and install using "sudo dpkg -i NAME-OF-PACKAGE".

   - ARIA Page

   (b) Then you go to your source workspace directory and clone the RosAria repository to it, then you build the package.

```
cd ~/catkin_ws/src
git clone https://github.com/amor-ros-pkg/rosaria.git
cd ~/catkin_ws
catkin_make #if this doesn't work you can try
    catkin_make --force-cmake
```

2. Using RosAria

   (a) You need to give write and read permissions to the serial port that the microcontroller is connected to (in this case 0), and then start the RosAria node.

```
sudo chmod a+rw /dev/ttyS0
rosrun rosaria RosAria _port:=/dev/ttyS0
```

   (b) The laser topic of RosAria doesn't work properly, for that we will use sicktoolbox.

II Sicktoolbox Wrapper

The Sicktoolbox Wrapper is used to publish laser information in the "scan" topic properly for further use in the gmapping package.

1. Installation

   (a) First you will need to install some dependencies.

```
sudo apt-get install ros-indigo-sicktoolbox
sudo apt-get install ros-indigo-sicktoolbox-wrapper
```

(b) Then you can clone the repository into your source directory of ROS workspace and build the package.

```
cd catkin_ws/src
git clone https://github.com/ros-drivers/
    sicktoolbox_wrapper
cd catkin_ws
catkin_make
```

2. Using Sicktoolbox Wrapper

(a) You need to change the permission of the serial port that the laser is connected to (in this case 2), and change it's IRQ number.

```
sudo chmod a+rw /dev/ttyS2
sudo setserial /dev/ttyS2 irq 10
```

(b) Then you can run with the following parameters.

```
rosrun sicktoolbox_wrapper sicklms _port:=/dev/ttyS2
    _baud:=38400 _connect_delay:=40
```

(c) We need this connection delay because the laser becomes available after turning on after a few seconds.

III Gmapping

The Gmapping is used to generate a map from a laser scan.

1. Installation

(a) First you will need to install some dependencies.

```
sudo apt-get install ros-indigo-gmapping
sudo apt-get install ros-indigo-slam-gmapping
sudo apt-get install ros-indigo-openslam-gmapping
```

(b) Then you can clone the repository into your source directory and build the package.

```
cd catkin_ws/src
git clone https://github.com/ros-perception/
    slam_gmapping
cd catkin_ws
catkin_make
```

2. Using Gmapping

   (a) Fisrt we need the laser reference frame, the RosAria node already publishes two frames, the "odom" frame and the "base link" frame, we need a frame called "laser". The frame transformation below with everything with '0' is just to see if it's working. We can improve it's precision by doing a right transformation from the "base link" frame to the "laser one", following this site ROS tf Setup.

```
rosrun tf static_transform_publisher 0 0 0 0 0 0 /
    base_link /laser /100
```

   (b) Now you can start the Gmapping node and see the mapping on the GUI tool rviz.

```
rosrun gmapping slam_gmapping scan:=scan
rosrun rviz rviz
```

   (c) In the rviz you can add topics to see what they are publishing in the add button on the bottom left, and in the right tab of the opening window. To see the map that is being producing by Gmapping you can add the "/map" topic.

   (d) Gmapping does not need to be running on the robot, you can start in your PC if the ROS enviroment variables are correct.

   (e) To move the robot around to test what it is mapping you can use the "rosaria client" package, after you clone it's repository you will need to edit the "print state" file deleting everything referencing to "Bumper State". Then you can build it corretly and start the teleop node to use the arrow keys to move around and use the space bar to stop. The repository is: RosAria Client And to run the teleop node:

```
rosrun rosaria_client teleop
```

IV Map Server

1. Installation

2. Using Map Server