ROS Made Easy
3: Launch Files and the Parameter Server

Aniruddh K Budhgavi
Enigma, IIIT-B

June 25, 2020

1 Important Note

This tutorial was created for **ROS1 Melodic Morenia** on **Ubuntu 18.04 Bionic Beaver**, in **June 2020**. I expect them to become rapidly out of date. It is my hope that Team Enigma will continually maintain and update these tutorials.

This tutorial assumes that you are running Ubuntu, and have at least an elementary grasp of Python 2.7 and C/C++ .

All the code for this tutorial is available at `https://github.com/aniruddhkb/enigmatutorials`.

The aim of this tutorial is to make you *functional* in ROS, not to make you a master. For that, look elsewhere.

2 Launch files

2.1 Introduction

1. One thing that has been very clunky so far is that we have to manually start the master and every node in its own individual terminal window. This is fine for small environments like Turtlesim, but completely unsuitable for larger projects. Further, it is apparent that on many occasions, there are nodes which naturally work together and must be spawned together as a group. It is clear that we need a process for launching nodes in bulk, instead of messing around with individual terminals.
2. All this functionality is provided by **launch files**.

2.2 A basic launch file

1. Create a package `intro2launch` which depends on `turtlesim`.
2. Within this package, create a folder called `launch`.
3. Within this folder, create `firstLaunch.launch`:

```
<?xml version="1.0"?>
<launch>
    <node name="turtlesim" pkg="turtlesim" type="turtlesim_node"/>
    <node name="teleop" pkg="turtlesim" type="turtle_teleop_key"
     launch-prefix="gnome-terminal -e" />
</launch>
```

To spawn a node, the syntax is `<node name=... pkg=... type=... />`. The `launch-prefix="gnome-terminal --"` option is used to make the node output display in a new terminal.

4. There are a few more options you should familiarize yourself with, including **required**, **respawn**, **ns** and **output**. You should also take a look at how to pass arguments. All this can be found at `http://wiki.ros.org/roslaunch/XML`. – have a look at the tag reference. Also look at the relevant chapter of *A Gentle Introduction to ROS*.

5. To run this launch file:

```
roslaunch intro2launch firstLaunch.launch
```

   You may notice that this automatically starts a master if no master has been found. To shut down all nodes started by this, use Ctrl+C.

2.3 Including other launch files

1. Now, create `secondLaunch.launch`.

```
<?xml version="1.0"?>
<launch>
    <include file="$(find intro2launch)/launch/firstLaunch.launch"/>
</launch>
```

   Run this in the same way as before. You can also pass args in the include file. Find out how by reading the docs and seeing examples.

3 Looking ahead

You can now use launch files to launch multiple nodes at once. This section was especially brief because I wish to only give a survey of ROS before spending much more time on simulation and control.

Now, we are ready to start creating models of our robots for simulation.