CS 378 - Autonomous Vehicles in Traffic I

Week 9b - Launch Files and Command Line Tools
Announcements

- The CNS Summer Fellowship deadline is approaching soon (this Friday 3/23)!
- No class on Monday 3/26. We'll instead devote the time to project meetings
- Short lecture on Wednesday 3/28 (about 30 minutes). After that we'll talk about the project literary/software review and finalize any last remaining details about the projects.
Today

● We'll go over the ROS Launch system
● We'll review ROS command line tools
● At the end of the class, we'll try and get closer to finalizing groups.
ROS Launch

- As we have seen across a number of examples, in ROS a number of nodes need to be brought up to run a system.
- In most of the examples in the slides, as well as the instructions in the assignments, we have been starting all the nodes independently.

- This is a inconvenience because of 2 reasons:
  - You need to document what all nodes needs to be started for the system to work. Furthermore, you need to run a number of commands to get these nodes up.
  - In some instance of debugging, it can be tiresome if you need to restart all your nodes again and again.
Enter *roslaunch*

- *roslaunch* is a tool for easily launching a bunch of ROS nodes
- *roslaunch* uses launch files written in XML that indicate the nodes that need to be started up
- You can create these launch files inside a package
  - Jack and I typically tend to store these launch files in a directory called *launch* inside the package

- Programming Assignment 5 will deal with launch files

[http://ros.org/wiki/roslaunch]
The standard method of calling roslaunch is:
  - `roslaunch <package-name> <launch-filename> [args]`

The main argument that you guys might use is:
  - `--screen`
    - This argument prints the output of all the nodes on the screen, as you would expect during rosrun

Other arguments and a detailed explanation is in the link below

By default a launch file will start roscore as well
  - For now it does not matter - but we'll see why this is important when we take a look at the Parameter Server

Let's take a look at an example
  - `roslaunch art_run auto_stage.launch`

[http://ros.org/wiki/roslaunch/Commandline%20Tools]
There is a specified format in which launch files need to be written.

A detailed description of the format is given in the link below, which can be used for reference.

I'll explain the format through some of the launch scripts that we have written to launch the car code.

- Launch files in art_run

Some tutorials on roslaunch are provided here:

The tutorial on using `gdb` or `valgrind` with one of the nodes in the launch file is very useful
  ○ If one of the nodes is misbehaving, you can simply tell the launch file to run that node under `gdb`
  ○ I generally use the following launch-prefix
    ■ `xterm -e gdb --args`
ROS command line tools

- The best way to review the command line tools is through the [ROS CheatSheet](https://www.ros.org/cheatsheet/).
Projects

- By the end of the day, send an email to cs378-spr12-submit, giving the names of your group members and your selected topic.
  - If you have had some trouble forming a group or selecting a topic then let me know.
  - If you have already collected additional information about your topic, do put it in your proposal.

- Next week, we'll have our first meeting. The Doodle Link for deciding on meeting:
  - [http://www.doodle.com/m3y4krafpubdq358](http://www.doodle.com/m3y4krafpubdq358)

- Now, to the projects spreadsheet!