Multi-Robot Control (MRC)

W1: Design Process and Tools
Plan of the Day

Overview of robotic control tools
• Linux, ROS, MATLAB and Git – Oh My!

Assignment 1 Brief

Lab Introduction – Read lab etiquette!
• Setup accounts
Working Hypothesis
• Students focusing on unmanned systems should be able to...
  – Proficient users of ROS
    Which means they need to be users of Linux, Git, etc.
  – Proficient developers in MATLAB

This is a hypothesis; I'm interested in your experience and observations throughout the course.
Implementation Tools

- Linux, Ubuntu
- ROS
- MATLAB
- Git
What is ROS?

- ROS = Robot Operating System
- Framework for robot command and control
  - Originally developed at Stanford AI Lab
  - Currently maintained by Willow Garage
- Supported Operating Systems
  - Linux
  - Mac OS X
  - FreeBSD
- Large user base;
- ROS wiki: [http://wiki.ros.org](http://wiki.ros.org)
Why ROS?

Inter-process communication
• Modularization by enabling programs (nodes) to communicate

Very large user community
• We don't have to re-invent the wheel
• Community support/knowledge
• Increasing commercial compliance

Introspection and tools

Alternatives
• JAUS, LCM, MOOS, etc.
A Simple ROS Architecture Example

- iRobot Create with a Hokuyo URG-04LX laser
Simulation in ROS with Gazebo

- 3D, physically based simulation

![Diagram of simulation components]

- MATLAB Controller
- `scan` and `odom`
- `cmd_vel`
- `sensor_msgs` LaserData
- `nav_msgs` Odometry
- `geometry_data` Twist
- `laser model`
- `robot model`
- `world model`

Gazebo
Linux

Why Linux?

- Standard support for development
- Despite the steep learning curve, software is simpler and more powerful
- ROS (and most other frameworks) are much easier to use in a Linux environment.

Ubuntu

- One of many distributions of Linux
- Simplified software distribution (package management)
- Well integrated with ROS
The Linux Command Line

Why?

• A complementary user interface model
• The art of Unix
  – Using collections of small programs to do big jobs
  – Avoiding manual tasks
• “Necessary” for ROS
The Standard NPS Robotics Environment

Ubuntu 14.04 (Long-term support)

ROS Indigo Igloo

MATLAB 2015b
Git

Version Control Software (VCS)
• Seemlessly keep all edits
• Collaboration and sharing
• Revert to prior edits

Git vs. Subversion vs. Mercurial, etc.

Why?
• Knowing the basics provides access to lots of community code.
• Pervasive use by ROS community
“Simple” Git Workflows

Existing Repository
- Clone (get a working copy)
- Make changes...
- Stage and Commit (local check-in)
- Push (remote check-in)

In the Future
- Pull
- Make changes...
- Stage/commit
- Push

New Repository
- Make new space gitlab server via web interface
- Init local copy
- Stage/Commit
- Push to server

In the Future
- Pull
- Make changes...
- Stage/commit
- Push
Assignment 1

Edit the Robotics Computation Wiki

Linux Command Line
- File/directory manipulation
- A word about text editors

Git
- Cloning a repository
  - Workflow: Clone/pull, change, commit, push
- Creating a repository

If possible, do the reading in front of a computer!
Next Steps

Today
• Everyone should have an account and start on Assignment 1

Tomorrow
• Meet in CAVR
• Goal is to finish Assignment 1!