CS 378: Autonomous Intelligent Robotics (FRI)

Dr. Todd Hester
Are there any questions?
Logistics

- CS mentoring in Kinsolving and Jester dining halls

- First homework assignment (due class time Thursday)

- Talks Friday
  - Dr. Mohan Sridharan
  - Towards Autonomy in Human-Robot Collaboration
    - 11 am, ACES 2.402
  - Integrating Answer Set Programming and Probabilistic Planning on Robots
    - 3 pm, ACES 2.402
Dr. Xiaofeng Ren's talk

- Summary
- Can we apply it to our project?
- What won't will apply to our project?
Today
Robot Operating System (ROS)

Readings
• High level overview
• Advantages of using ROS?
• Disadvantages of using ROS?
ROS

(adapted from slides by Prof. Chad Jenkins and Piyush Khandelwal)
Intelligent Complete Robot

Sensing, modeling the world

Behaviors, action selection, planning, learning
Multi-robot coordination, teamwork
Response to opponent, multi-agent learning

Motion, navigation, obstacle avoidance

Perception → Cognition → Action

Sensors → Perception

Actuators → Action

External World
Example: iRobot Create based robot

[adapted from slide by Chad Jenkins]
Software Architecture

- From wikipedia: "The **software architecture** of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both."

- Software architecture is important for
  - creating reusable code
  - ensuring portability between different devices and platform

- Important for robotics because
  - Large code-bases
  - Integration of many different and a dynamic set of devices
  - Many different options for a single component
Controlling robots using code

[adapted from slide by Chad Jenkins]
Straightforward approach

● Just write and compile a program to perform robot's "cognitive" functions
● This program will include
  ○ Code to interface with the camera and the iRobot Create
  ○ Code to understand the images and the environment and control the Create
● Once implemented, the system works well and efficiently

[adapted from slide by Chad Jenkins]
Straightforward approach

However this approach suffers from a problem. Any ideas?

[adapted from slide by Chad Jenkins]
An example problem...

- After implementing my program, I realized the create is too slow (0.5 m/s).
- How easy it is to use a segway robot instead (1.7 m/s)?

Could I have implemented my code differently to make this transition easier?
Enter robot middleware

- Provide an abstraction layer and drivers between computation and embodiment.

- This is similar to how hardware abstraction allows your program to work independent of the actual hardware.
  - i.e. the hardware abstraction layer in the operating system.

- Using a middleware package might seem a subtle difference right now, but it is a fundamentally different approach to developing robot applications. Let's look at an example.

[adapted from slide by Chad Jenkins]
Using robot middleware

● Looks about the same. So what's the advantage?

[adapted from slide by Chad Jenkins]
Using robot middleware

[adapted from slide by Chad Jenkins]
The advantages

- **Reusability**
  - Reuse existing drivers and code written for other robots, platforms and research projects.

- **Portability**
  - Easier to switch to another robotic platform.

- Easier to expand functionality
ROS (Robot Operating System)

- A very popular robot middleware package
- Peer-to-peer architecture among nodes over a network
- Robot functionality split over multiple nodes (processes)
- Nodes subscribe to and publish messages on "topics"
  - ROS Master runs topic registry
- Topics are named channels over which messages are exchanged

[adapted from slide by Chad Jenkins]
Robot Example

Let's say we have a camera, a laptop, and a create, and we want to move the robot based on detected objects in the camera image.

- What nodes might we use?
- What messages would they send?
How it works - Create example

- Lets say we split up the code into 4 functional components
  - Camera Driver - produces images from the camera
  - Create Driver - accepts forward and angular velocity and makes the Create move
  - Blobfinder node (cmvision) - takes an image and returns the positions of different colored blobs on the screen
  - Control node - takes the position of the orange blob and calculates the velocities required to reach it.

[adapted from slide by Chad Jenkins]
How it works

- cmvision node
- control node
- camera node
- create node

USB

USB-Serial

[adapted from slide by Chad Jenkins]
How it works

- **cmvision node**
  - I will receive images on topic "image" and publish blobs on topic "blobs"

- **camera node**
  - I will publish images on topic "image"

- **control node**
  - I will receive blobs on topic "blobs" and publish velocities on topic "cmd_vel"

- **create node**
  - I will receive velocities on topic "cmd_vel"

(from slide by Chad Jenkins)
How it works

- cmvision node
- control node
- camera node
- create node

ROS Master

- images on "image"
- blobs on "blobs"
- velocities on "cmd_vel"

Sets up communication

USB
USB-Serial
How it works

● These message formats for inter-node communication are well defined. We'll see more of these in upcoming weeks.

● All this communication is done over TCP or UDP. This allows one of your nodes to be in China if you want.

● In many cases, all these nodes are running on a single machine.
ROS Nodes

• A node is a process that performs some computation.
• Typically we try to divide the entire software functionality into different modules - each one is run over a single or multiple nodes.
• Nodes are combined together into a graph and communicate with one another using streaming topics, RPC services, and the Parameter Server.
• These nodes are meant to operate at a fine-grained scale; a robot control system will usually comprise many nodes.

[http://www.ros.org/wiki/Nodes]
ROS Topics

- Topics are named buses over which nodes exchange messages.
- Topics have anonymous publish/subscribe semantics - A node does not care which node published the data it receives or which one subscribes to the data it publishes.
- There can be multiple publishers and subscribers to a topic:
  - It is easy to understand multiple subscribers.
  - Can't think of a reason for multiple publishers.
- Each topic is strongly typed by the ROS message it transports.
- Transport is done using TCP or UDP.
ROS Messages

• Nodes communicate with each other by publishing messages to topics.
• A message is a simple data structure, comprising typed fields. You can take a look at some basic types here:
  ○ `std_msgs/Bool`
  ○ `std_msgs/Int32`
  ○ `std_msgs/String`
  ○ `std_msgs/Empty` (huh?)
• Messages may also contain a special field called header which gives a timestamp and frame of reference.

[http://www.ros.org/wiki/Messages]
ROS Naming

- Subscription is to particular named topic
- No knowledge of actual node you are connecting to

- Also compiling or running packages
  - rosmake
  - rosrun
  - roscd
  - roslaunch

- name of the Package that the resource is in plus the name of the resource

- rosrnn segbot_gazebo segbot_mobile_base.launch
Open-Source Code / Collaboration

http://www.ros.org/wiki/Repositories
- Repository: Contains all the code from a particular development group (We have 3 repositories from utexas)
- Stack: Groups all code on a particular subject / device
- Packages: Separate modules that provide different services
- Nodes: Executables that exist in each model (You have seen this already)
ROS command line tools

- The best way to review the command line tools is through the [ROS CheatSheet](#).
ROS: Goals

Main goals of ROS
● Provide a robotics platform designed for code reuse
● Provide a code and file structure for easier collaborative development
● Provide a number of tools for visualization and monitoring
● Encourage modularization of drivers and different functional units.

These goals and their benefits will become clearer as this semester progresses
Example 1 - Publisher and Chatter

- The first example is directly from ROS Tutorials
- I *highly recommend* going through these tutorials on your own time
- We'll take a look at C++ tutorial today (Tutorial 11)
- If you are interested in using ROS in Python go through the Python tutorial (Tutorial 12). The tutorials are fairly similar
First Assignment Due Thursday!