CS 378 - Autonomous Vehicles in Traffic I

Week 2a - Introduction to ROS

(adapted from slides by Prof. Chad Jenkins)
Logistics

- Some updates to class webpage
  - Percentage points for every assignment are now included in the schedule
  - Office hours are up and effective from now on
  - I am adding a couple more sections at the bottom of the webpage with some interesting links
  - I have put up a printable version of the slides as well.

- If you enrolled in the class after Wednesday's class (I think there are 2-3 students), then let me know if you have any questions about the course structure
  - Course structure is available in Week 1 slides
Logistics

- If you have not enrolled to the mailing list cs378-spr12-announce@utlists.utexas.edu, please do so immediately!
  - Effective immediately all announcements will be made here
  - Do not sign up for cs378-spr12-submit

- Instructions for joining up the mailing list cs378-spr12-announce (thanks Kevin!)
  - Go to https://utlists.utexas.edu/sympa/info/cs378-spr12-announce
  - If you've never had a mailing list account before, click the First login link in the top left and enter your email address to get a password. Then just log in and subscribe to the list.
Logistics

- In December, a reporter from the UT Alumni magazine approached us to write about this research stream.
- I was just told that the story is going to run in their March/April issue.
- On Friday they are planning on running a photo-shoot to emphasize the fact that undergrad students are working on the car.
- We still need at least 2-3 more students to participate. If any of you are free from 3-4PM (plus some time to get to PRC and back), please let me know after class. I will send an announcement out tonight.
Today

- Autonomous Intelligent Agents
- Importance of software architecture in robotics
- Robot middleware
- What is ROS?

*Remember to ask questions wherever necessary!*
Autonomous Intelligent Agents

● What makes an agent?
  ○ They must sense their environment
  ○ They must decide what action to take (i.e., think)
  ○ They must act in their environment

● What makes a complete agent?
  ○ Interact with other agents (Multi-agent Systems)
  ○ Improve performance from experience (Learning)

● A robot is an artificial agent that interacts with the physical environment through sensors and actuators.

● What is an example of a non-artificial agent?
Intelligent Complete Robot

Sensing, modeling the world

Perception

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

Cognition

Motion, navigation, obstacle avoidance

Action

Sensors

External World

Actuators

[slide by Manuela Veloso]
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

[adapted from slide by Chad Jenkins]
ROSG (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
- Later in the semester we'll talk about
  - Nodelets - Each node is no longer its own process (Why?)
  - Services - Request-reply communication

[adapted from slide by Chad Jenkins]
How it works - Create example

- Let's 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

[adapted from slide by Chad Jenkins]
How it works

- **cmvison node**: I will receive images on topic "image" and publish blobs on topic "blobs"
- **camera node**: I will publish images on topic "image"
- **create node**: I will receive velocities on topic "cmd_vel"
- **control node**: I will receive blobs on topic "blobs" and publish velocities on topic "cmd_vel"

[adapted from slide by Chad Jenkins]
How it works

[adapted from slide by Chad Jenkins]
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.

- Let's look at how communication between 2 nodes is setup at the packet level.

[adapted from slide by Chad Jenkins]
Matchmaking at a lower level

[adapted from slide by Chad Jenkins]
More about matchmaking

● Subscription to a topic is typically non-blocking
  ○ This means that a node can do other work while it is waiting for a subscription to be fulfilled

● A node can also unsubscribe whenever it wants to
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
Review

- ROS is a peer-to-peer robot middleware package
- We use ROS because it allows for easier hardware abstraction and code reuse
- In ROS, all major functionality is broken up into a number of chunks that communicate with each other using messages
- Each chunk is called a node and is typically run as a separate process
- Matchmaking or bookkeeping between nodes is done by the ROS Master
Learning ROS

● Here are some links that provide an overview of ROS
  ○ http://courses.csail.mit.edu/6.142/wiki/images/0/05/Icraoss09-ROS.pdf
  ○ http://www.ros.org/wiki/ROS/Introduction

● ROS Cheatsheet

● These links assume a decent understanding of robotics and sufficient programming experience.

● Our goal this semester is to go through all these concepts using a number of examples and assignments.
Reading Assignment 1

• Reading Assignment 1 is due tomorrow night at 10PM by email.
  ○ The reading response should be in plain-text
  ○ I will go through your responses tomorrow night and discuss some of the common questions in class on Wednesday
  ○ As I get time, I'll send out a more complete list on the class mailing list.