CS 309: Autonomous Intelligent Robotics
FRI I

Lecture 12:
Publish/Subscribe & ROS Topics Part 2
Remote Procedure Call & ROS Services

Instructor: Justin Hart

http://justinhart.net/teaching/2018_spring_cs309/

A couple of quick notes

- Homework 2: Is due March 1.
  - PDDL
  - Questions on the homework build on each other.
    - You could, in theory, use only one domain file for the entire assignment
    - But multiple fact files
  - Any questions?

Publish/Subscribe (Recap)

- ROS Topics use a Publish/Subscribe architecture
  - Publishers
    - Broadcast data on a “topic”
  - Subscribers
    - Multiple subscribers can read data from a topic simultaneously
  - Example
    - rviz example from Kinect v2 data

ROS Topics

- Connect over a network socket
  - Stored in $ROS_MASTER_URI
  - Default is http://localhost:11311
  - You probably won't use it in this class, but for RoboCup@Home, we use a network of 3 or 4 computers to drive the robot
    - To manage this, we use $ROS_MASTER_URI to coordinate communications

ROS Topics

- Last time, we quickly ran through a ROS Topics tutorial. Let’s revisit this:
  - http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics

Understanding ROS Topics

- http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics
- Stop at part 3.1 for more explanation

Writing our first subscriber

- Event-driven programming
  - Main loops are used in event-driven programming
  - The idea is to wait for something to happen, and then act on the thing that happened
    - The thing that happened is called an event

Writing our first subscriber

- The main loop allows the program to check if something has happened repeatedly
  - ros::spinOnce() does the checking
  - Subscribers check to see if a message has been published on a topic
- At the start of the program, a function called a “callback” is “registered” with the event-driven framework
- How this works will become more clear when we try it out

Writing our first subscriber

- ROS main loop processing calls
  - ros::spin();
  - ros::spinOnce();
  - ros::MultiThreadedSpinner spinner(4);
    - Spins in several threads
  - ros::AsyncSpinner spinner(4);
    - Another multi-threaded spinner
- These maintain the “callback queue”
  - Every event in ROS goes into the callback queue
  - spin() and its cousins simply process this queue
    - That is to say, a list of things that happens is kept, and spin takes items off of that list

Writing our first subscriber

- Back to:
  - http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29

Using custom messages

- Example
  - TwoInts.msg
    - Changes to CmakeLists.txt
    - Changes to package.xml
  - sendint.cpp
  - recint.cpp

Remote Procedure Call (RPC)

- RPC is a method of calling a function in another program
  - Traditionally, this is across a network, but can be on the same machine.
  - In ROS, this also uses roscore
  - Also serviced by an event-driven model on the server side