Introduction to ROS
What is ROS?

• It's not an operating system (despite the name)!
  • Middleware
  • Sits between the user application code and the OS
• Provides an abstract API for hardware devices
  • takes care of irrelevant details (hopefully)
• Provides a communication and synchronization mechanism
• Includes a simulation facility (Gazebo)
What is ROS?

• A “Meta” Operating System.
  • Open source
  • Runs in Linux (esp. Ubuntu)
  • Ongoing Windows implementation

• Agent based (nodes)
• Message passing
  • Publish
  • Subscribe
  • Services via remote invocation

• Supports numerous programming languages (C++, Python, Lisp, Java)
What is ROS?

- Low level device abstraction
  - Joystick
  - GPS
  - Camera
  - Controllers
  - Laser Scanners
  - ...

- Application building blocks
  - Coordinate system transforms
  - Visualization tools
  - Debugging tools
  - Robust navigation stack (SLAM with loop closure)
  - Arm path planning
  - Object recognition
  - ...

Application building blocks
System software

ROS
What is ROS?

- Software management (compiling, packaging)
- Remote communication and control
ROS Philosophical goals

- “Hardware agnosticism”
- Peer to peer
- Tools based software design
- Multiple language support (C++/Java/Python)
- Lightweight: runs only at the edge of your modules
- Free
- Open source
- Suitable for large scale research and industry
Tools-based software design

Tools for:

- Building ROS nodes
- Running ROS nodes
- Viewing network topology
- Monitoring network traffic

Many cooperating processes, instead of a single monolithic program.
Multiple language support

• ROS is implemented natively in each language.
• Quickly define messages in language-independent format.

File: PointCloud.msg

Header header
Points32[] pointsXYZ
int32 numPoints
Lightweight

- Encourages standalone libraries with no ROS dependencies:
  
  *Don’t put ROS dependencies in the core of your algorithm!*

- Use ROS only at the *edges* of your interconnected software modules: Downstream/Upstream interface

- ROS re-uses code from a variety of projects:
  
  - OpenCV : Computer Vision Library
  - Point Cloud Library (PCL) : 3D Data Processing
  - OpenRAVE : Motion Planning

ROS Community

Carnegie Mellon
Peer to Peer Messaging

• Claim: No Central Server through which all messages are routed.
• “Master” service run on 1 machine for name registration + lookup
  • Undercuts that claim above.
• Messaging Types:
  • Topics: *Asynchronous* data streaming
  • Parameter Server
Peer to Peer Messaging

- **Master**: Lookup information, think DNS
  
  `roscore` command → starts master, parameter server, logging

- **Publish**: Will not block until receipt, messages get queued.

- **Delivery Guarantees**: Specify a queue size for publishers: If publishing too quickly, will buffer a maximum of X messages before throwing away old ones

- **Transport Mechanism**: TCPROS, uses TCP/IP

- **Bandwidth**: Consider where your data’s going, and how
Free & Open Source

- BSD License: Can develop commercial applications
- Drivers (Kinect and others)
- Perception, Planning, Control libraries
- MIT ROS Packages: Kinect Demos, etc
- Interfaces to other libraries: OpenCV, etc
ROS Debugging

- Shutdown “Object” node → re-compile → restart: won’t disturb system

- Logging (VIDEO)

- Playback (VIDEO)
Useful ROS Debugging Tools

• rostopic: Display debug information about ROS topics: publishers, subscribers, publishing rate, and message content.
  - rostopic echo [topic name] → prints messages to console
  - rostopic list → prints active topics
  - ... (several more commands)

• rxplot: Plot data from one or more ROS topic fields using matplotlib.
  - rxplot /turtle1/pose/x,/turtle1/pose/y → graph data from 2 topics in 1 plot
Useful ROS Debugging Tools

rxgraph
ROS Visualization

Visualize:

- Sensor data
- Robot joint states
- Coordinate frames
- Maps being built
- Debugging 3D markers

[VIDEO]
ROS Transformations

• “TF” = Name of Transform package
  “Tully Foote” == Person/Developer

• TF Handles transforms between coordinate frames : space + time

• tf_echo : print updated transforms in console

Example:
rosrun tf tf_echo [reference_frame] [target_frame]
Packages

• Perception
  • Point Cloud Library (PCL)
  • OpenCV
  • Kinect/OpenNI
ROS Resources

- http://www.ros.org
- http://wiki.ros.org