4632B - Mobile Manipulation
Microsoft Robotic Studio

Henrik I Christensen
hic@cc.gatech.edu
Outline

- Background
- What is MSR?
- The basic components
- Simulations
- Services
- Where to find more information
- Things to consider
Background

- Construction of systems requires “glue” for integration
- What is needed?
  - Communication framework
  - Standard library of components
  - A control/synchronization model
  - Tools to support development
Communication

- Requirements
  - Language neutrality
  - Handling of structured data
  - Efficient
  - Multiple control models
    - Push, Pull, Subscription Model
Std Components Library

- Interfaces to standard set of sensors and actuators
  - Camera, Laser Scanner, Sonar, Gripper, GPS
  - Mobile Platform, Manipulators, ....
- Essential for rapid prototyping
Tools for development

- Access to data on the fly
- Facilities for logging
- Code templates / reference implementations
- Simulations to allow offline evaluation
- Visualization of operation / performance
Microsoft Robotic Studio

- An attempt by Microsoft to enter systems integrator market segment
- A framework for process synchronization (CCR)
- A communication framework (DSS)
- Simulation - PhysX (AGEIA)
- Visual programming framework
Microsoft CCR

- Concurrency & Coordination Runtime
- C# library for process coordination
- Supervisor for process coordination
- Separating I/O and computing
- Asynchronous IO management
- Light weight process management
Microsoft DSS

- A communication library in C# for serialization of data
- Includes standard communication services
  - Yellow Page Service
  - Push of data, Pull/Query and Subscription
- Interfaces to a number of std languages
  - C#, C++, Java, Python, Visual Basic
Web based status
Trivial Example
More realism
Push / Pull

- Client Service
- Target Service
- Change
- Subscriber
- Change
- Subscriber
- Change
- Subscriber
Visual Programming
MSR Simulation

- PhysX to have physics based simulation
- For prototyping
- Nice system, but requires HW to generate RT solutions
- Limits availability
Interfacing

- Has interfaces for
  - iRobot Create
  - LEGO NXT
  - Parallax
  - Kondo “toy humanoid”
  - Pioneer P3 / WXP
- More devices in progress
Interfaces

- Interfaces to a number of standard IO devices
  - Camera (USB)
  - GPS - NMEA serial
  - Binary sensors
  - ...

Robotics and Intelligent Machines @ Georgia Tech
Getting to know MSR

- A series of tutorials on the web
  - Service design
  - VPL, Language interfaces
  - Reading state, changing state, ...
- The documentation does not seem to be structured (yet)
- Similar to other projects
- Check www.microsoft.com/robotics
Competition

- OROCOS (Open Robot Control System)
  - Manipulation oriented - GPL
- ORCA (Open Robot Control Architecture)
  - Mobile Systems Oriented - GPL
- Saphira (Mobile Robotics)
- Player-Stage
  - Device interface library
# Brief comparison

<table>
<thead>
<tr>
<th></th>
<th>Arch</th>
<th>Lang</th>
<th>Devices</th>
<th>Doc</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MRS</strong></td>
<td>HD</td>
<td>C#, VB, Python</td>
<td>Some</td>
<td>+</td>
</tr>
<tr>
<td><strong>OROCOS</strong></td>
<td>React</td>
<td>C++</td>
<td>Manip</td>
<td>++</td>
</tr>
<tr>
<td><strong>ORCA</strong></td>
<td>HD</td>
<td>C++/Java</td>
<td>mobile,</td>
<td>++</td>
</tr>
<tr>
<td><strong>Saphira</strong></td>
<td>“React”</td>
<td>C, C++</td>
<td>Pioneer</td>
<td>+</td>
</tr>
<tr>
<td><strong>Player</strong></td>
<td>?</td>
<td>C++, Java, ...</td>
<td>MANY</td>
<td>+++</td>
</tr>
</tbody>
</table>
Summary

- Systems integration is a key technology for success
- MRS offers a number of interesting options
- Python might be a good integration language
- Documentation poses a challenge
- MRS is an option it is not a required part of any project
More Organization

- Wednesday
  - Recognition - SIFT Based
- Friday
  - Visual Servoing
- Monday
  - Early presentation of design