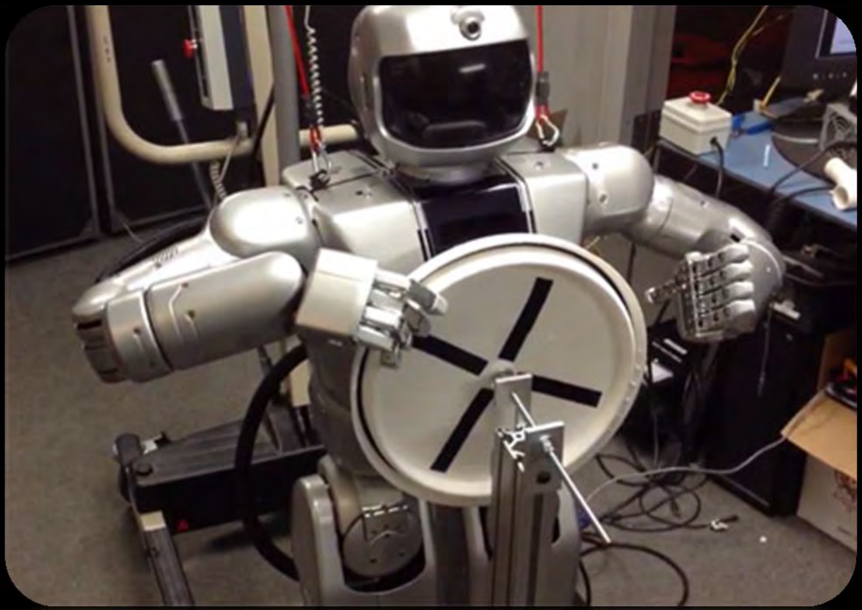
The Valve Turning Task

Jim Mainprice, Nicholas Alunni, Calder Phillips-Grafflin, Halit Bener Suay,
Sonia Chernova, Robert W. Lindeman, Dmitry Berenson

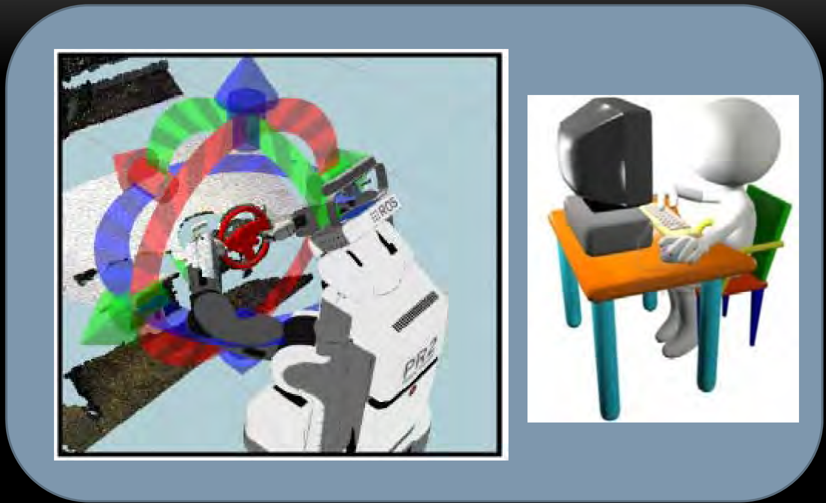
Valve Turning task

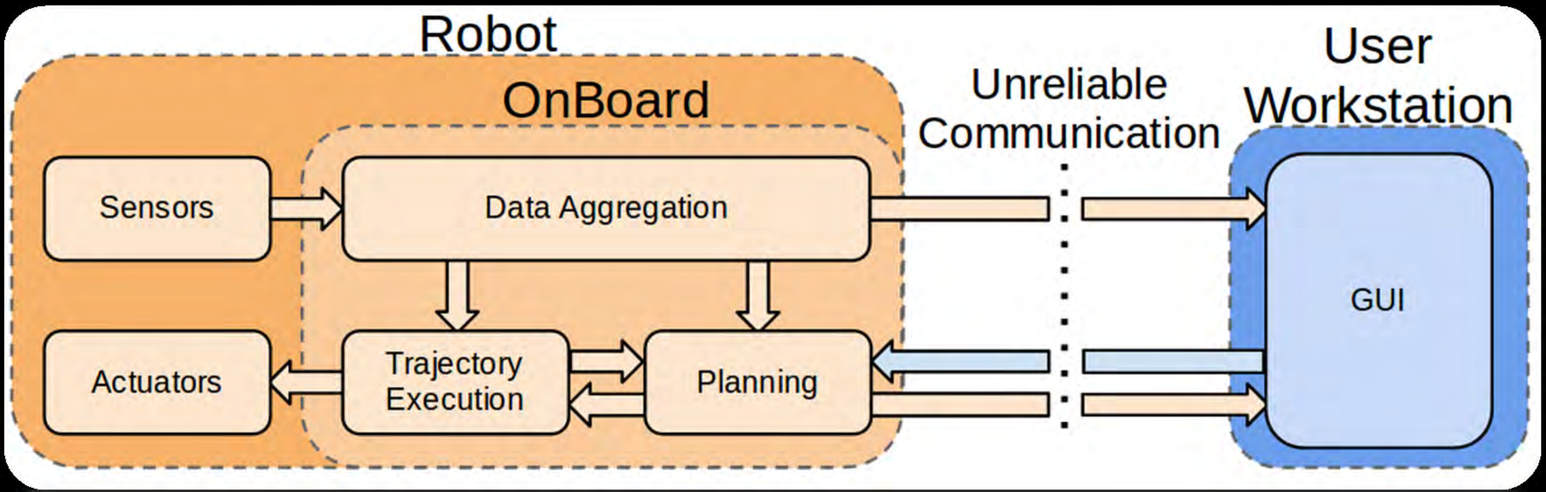


Challenges :

- Valve localization
  - Unreliable communication
  - Unknown environment
- Motion planning
  - Placement, IK for manipulation
  - Kinematic and Torque restriction
- Execution
  - Assess the success/failure of trajectory execution

User Guided Approach





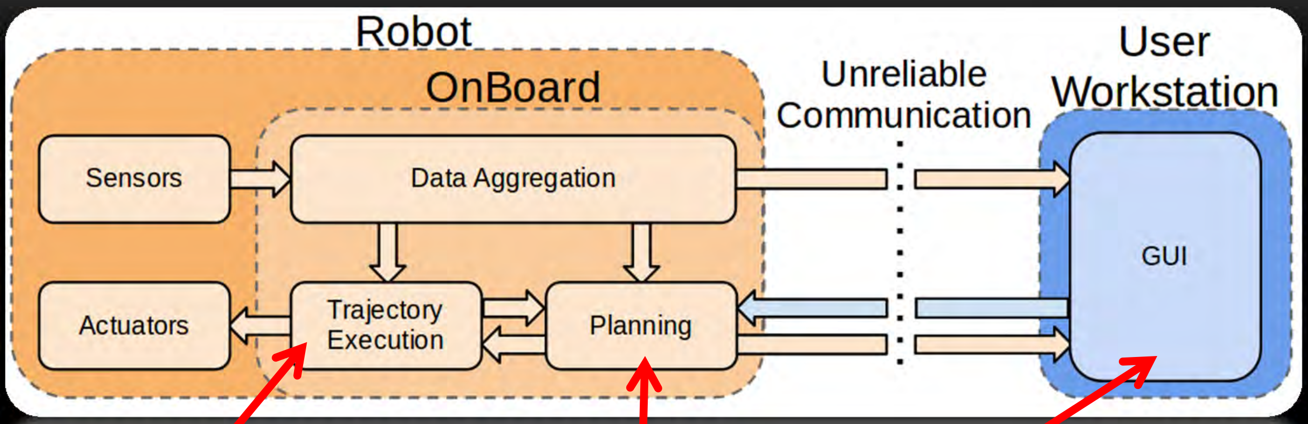
User Guided Approach

User-guided Valve-Turning on the PR2

Nick Alunni
Calder Phillips-Grafflin
Halit Bener Suay
Rob Lindeman
Sonia Chernova
Dmitry Berenson

Worcester Polytechnic Institute (WPI)

User Guided Approach : Outline



1 - User Interface
Monitoring the robot state and sending commands

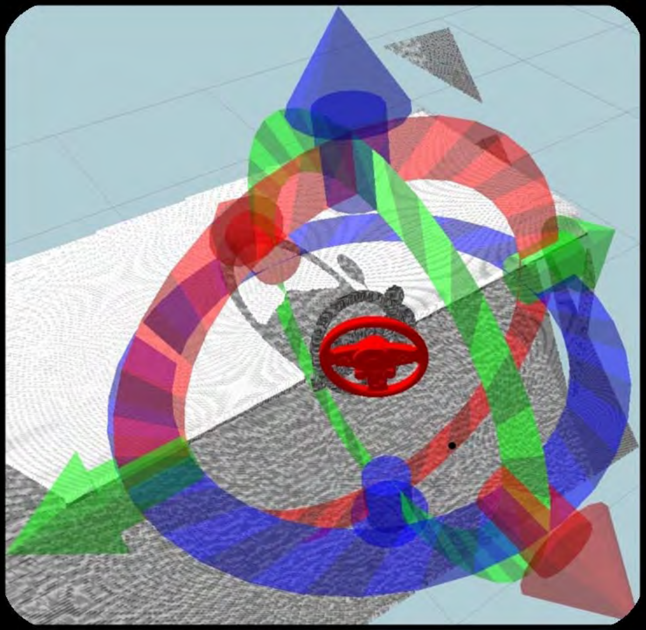
2 - Motion Planning
High level goal and valve turning specific constraints

3 - Trajectory Execution
Error detection

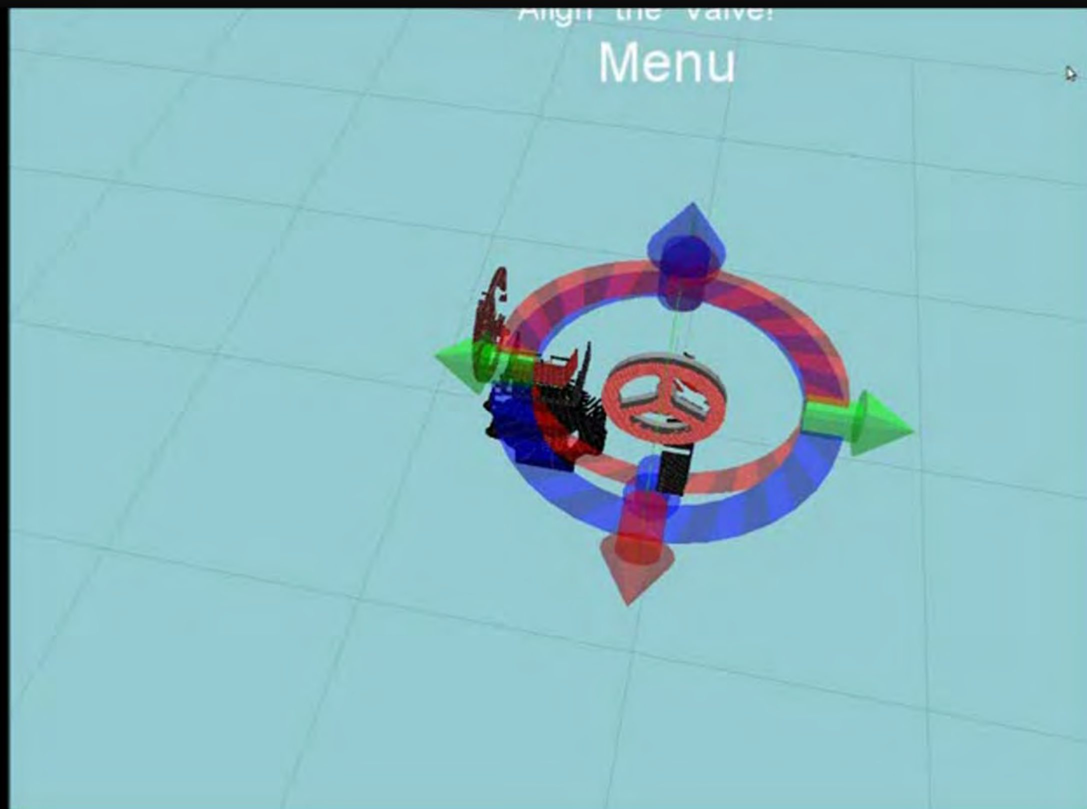
4 - Framework Infrastructure

User Guided Valve Alignment

- The user “hints” at the valve’s location (6 Dof)
- Asynchronous communication with low bandwidth
- User through GUI:
  - Places an interactive marker to indicate the position of an object of interest
  - Sends the point cloud aligned pose and to the planning software

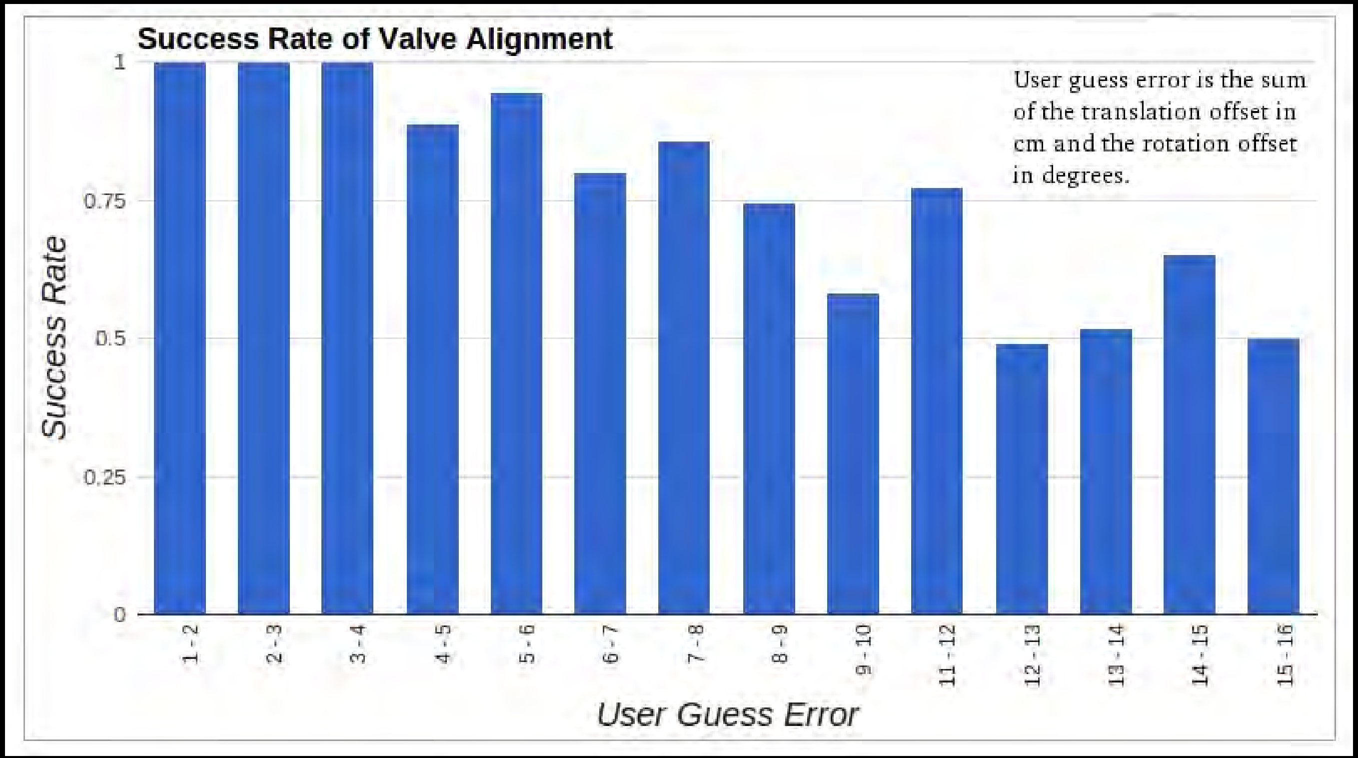


Valve Alignment To Point Cloud



Iterative Closest Point (ICP) aligns the interactive marker to the point cloud provided by the robot sensors

Valve Alignment Success Rate



Error combines translation and rotation, the farther the initial guess the lower the success rate

User Interface : Ongoing Work

- More autonomous valve identification

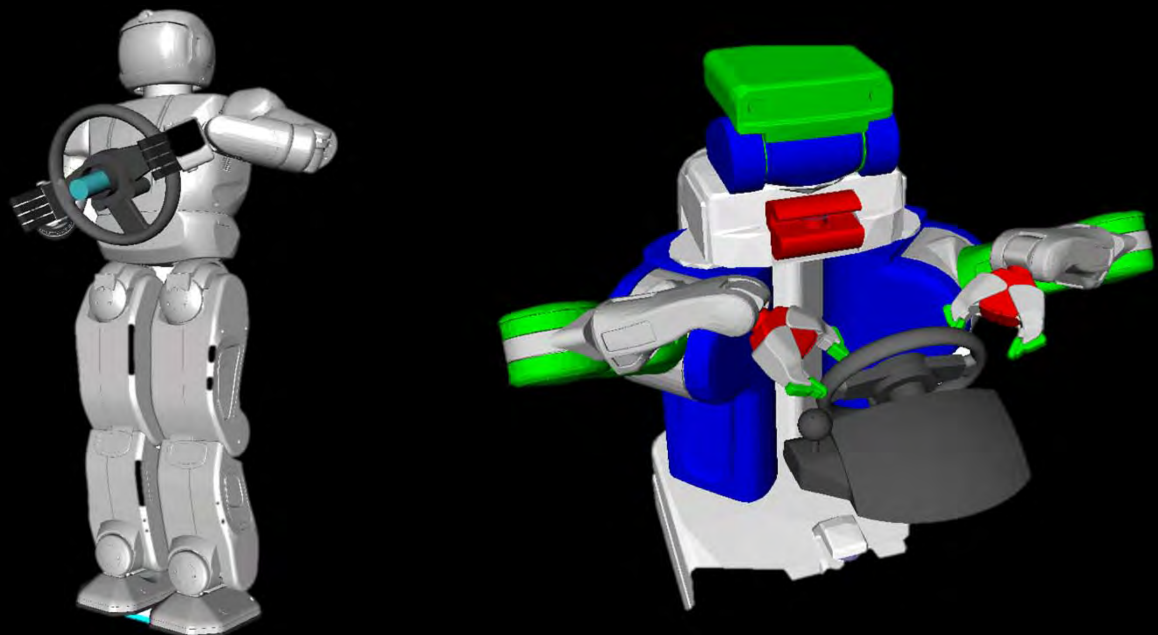
- Make use of feedback
  - Improve visual, modularity for task switching
  - Multi-sensory (touch, sound, tactile, …)

- User studies
  - Validation of the UI

Motion Planning For Valve Manipulation

Robot agnostic motion planning applied to PR2 and Hubo

CBiRRT[1] (Constraint-Based Bi-RRT) in OpenRave

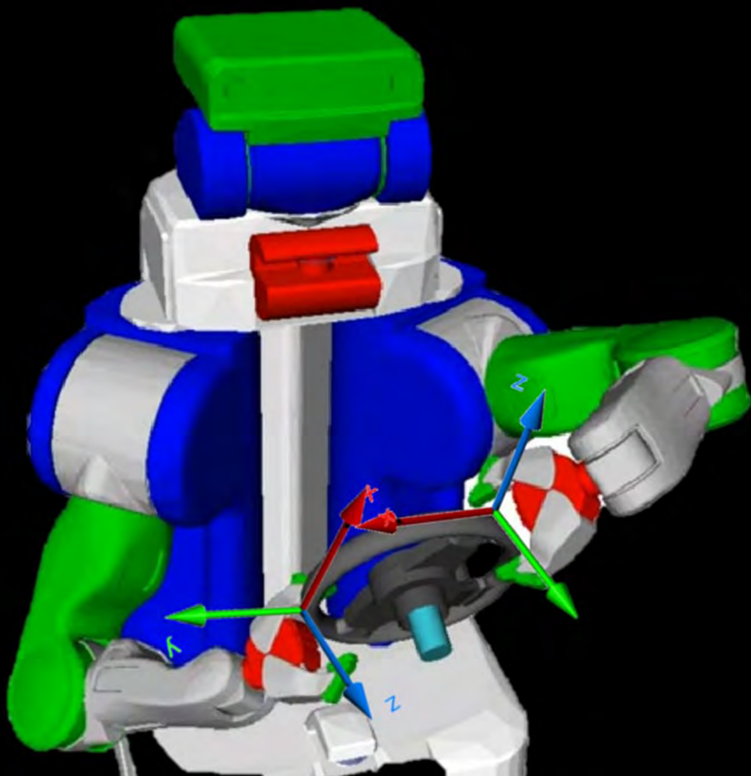


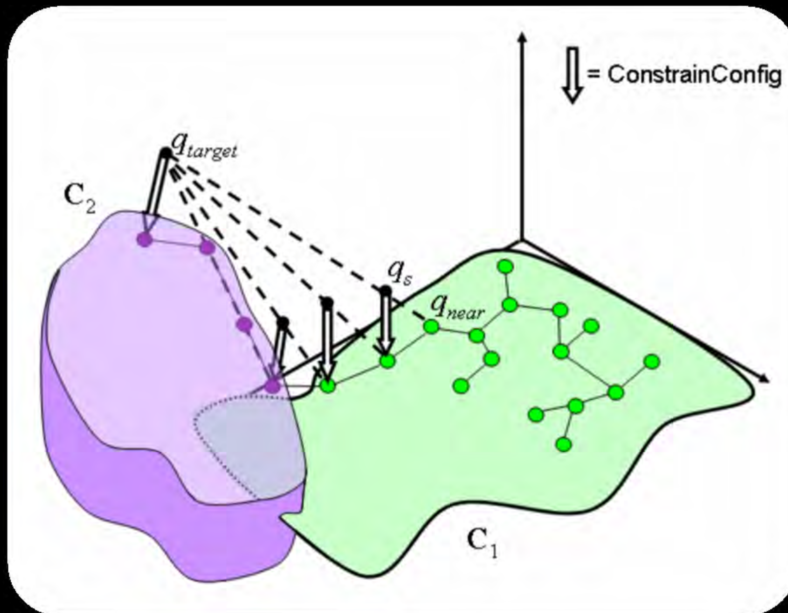
(1) Berenson et. al, *Task Space Regions: A Framework for Pose-Constrained Manipulation Planning,* IJRR, October, 2011

Motion Planning For Valve Manipulation

CiBiRRT[1] for two handed object manipulation with two constraints,

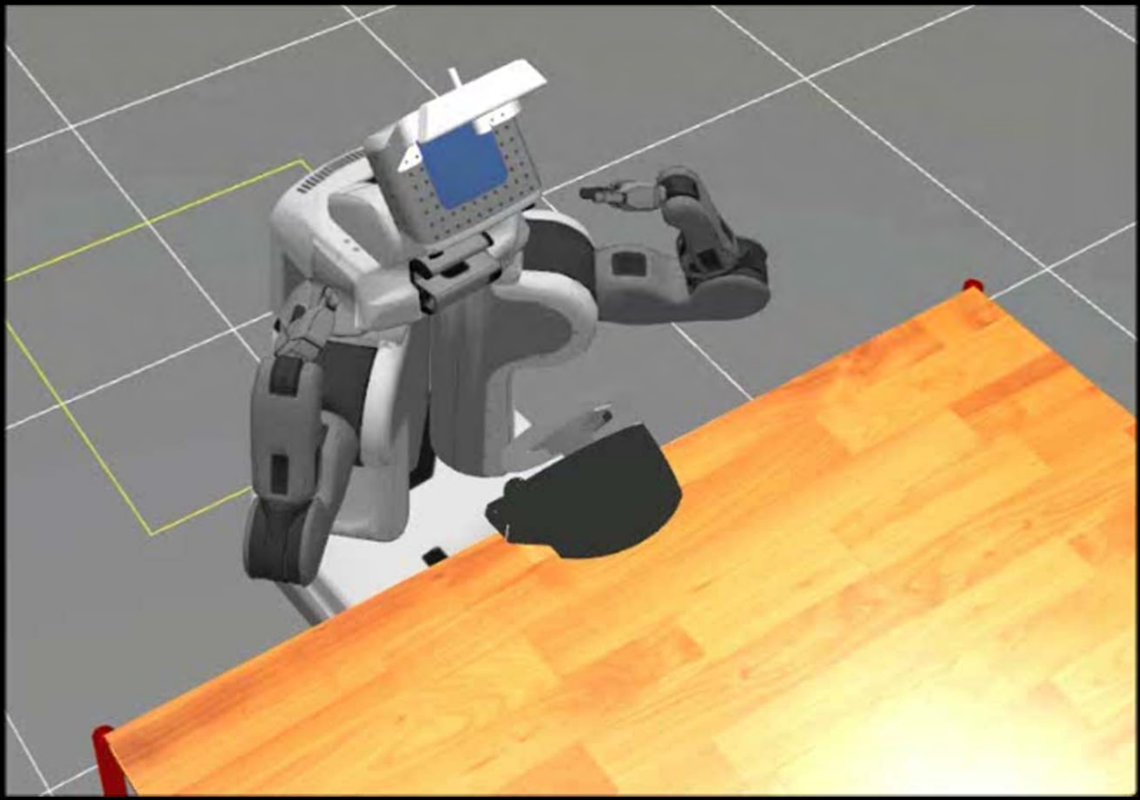
One for each arm, chain of TSRs





(1) Berenson et. al, *Task Space Regions: A Framework for Pose-Constrained Manipulation Planning,* IJRR, October, 2011

Motion Planning In Simulation



This trajectory is successful on the real robot

Motion Planning In Simulation

- Gazebo is unable to model correctly friction on the wheel
- Other artifacts make simulation difficult
  - The PR2 is drifting despite its weight and friction
  - Prolonged testing hard or impossible!
  - We don’t know what is happening
- Need a better simulator!

Valve Manipulation In Real World

After generating and testing the arm trajectories in OpenRAVE,

we ran them on the real PR2 and HUBO

December 2012

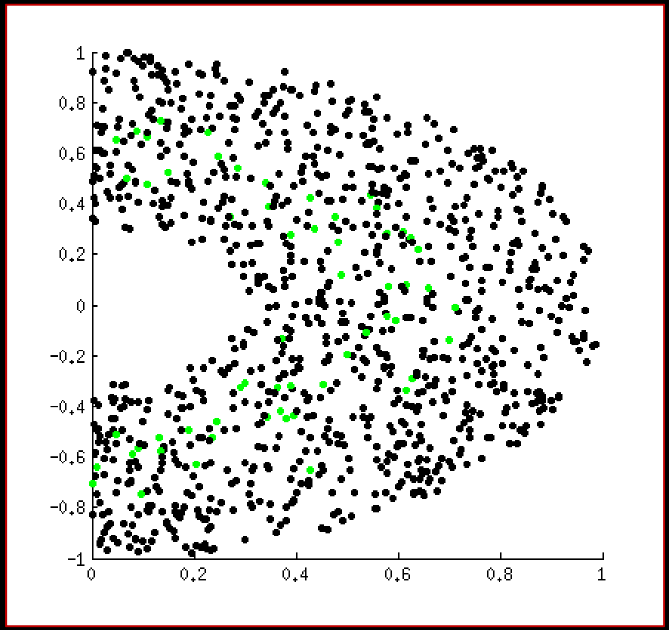
January 2013

Planning system (based on CBiRRT) required minimal changes to define task

Planning takes into account quasi-static balance for Hubo

Motion Planning : Ongoing Work

A learning approach to predict good initial configurations, under different constraints, given a valve pose



We randomly sample different robot and valve configurations.

We save the results, and label them by success or failure to reach a start IK and a goal IK solution.

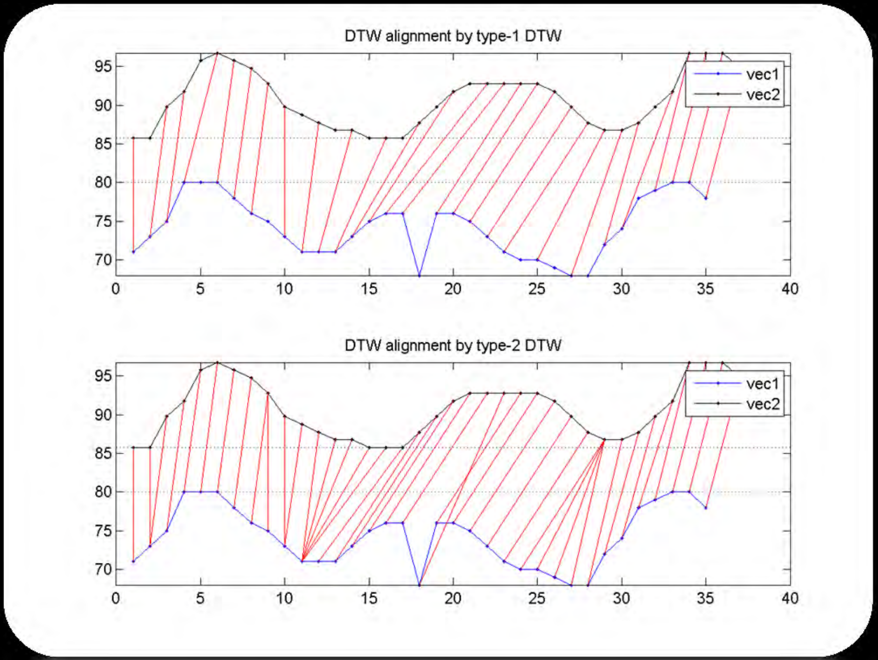
Error Detection And Execution

Slip one hand

Miss both hands

Error Detection And Execution

- A library Approach:
  - Compare current trajectory to a library of success and failure
  - Classify good and bad using Dynamic Time Warping (DTW) that gives a cost metric
  - Good : both hands grasped the wheel throughout the entire trajectory



- Features:
  - Pose-space trajectory analysis
  - Requires no visual feedback
  - Requires no feedback from end effectors

Error Detection : Performance

Library-based algorithm

- Sample executed trajectory
- Compute DTW metric to all trajectories in the library
- Return good or bad depending on best match

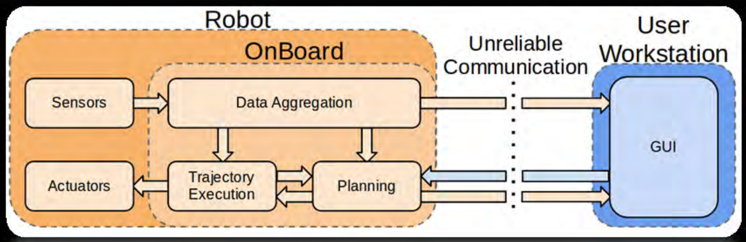
- Library : 526 tagged example trajectories (186 good, 340 bad)
- Detects grasp quality during valve turning (both hands on the valve)
- Leave one out testing, 88% overall correct

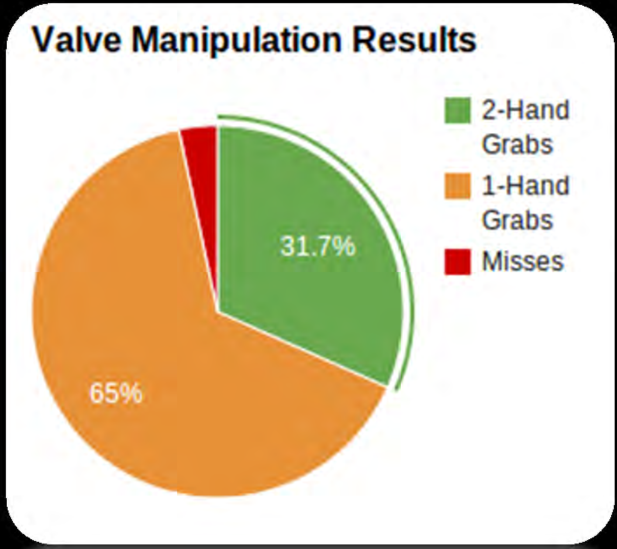
Error Detection : Ongoing work

- Analysis online, during trajectory execution
- More error conditions (slip, …)
- Increased dynamic time warping performance
  - Faster
  - Better accuracy
- Automatic error recovery

Framework : Performance

- The PR2 is able to grasp the wheel in 97% of the total tests
- Most often grasp with one hand :
  - Misalignment between the robot placement and the wheel
  - Error in Kinect measurements
  - Calibration error (Kinect pose not well determined)
  - Arm controller inaccuracy
- Solutions :
  - Better placement of the robot (account for uncertainty)
  - More robust grasping strategy
  - Visual and force feedback (hand)





Framework : Integration With ROS

Why ROS?

- Demonstrated on multiple platforms
- Easy to learn and use and develop
  - Presented framework built by 3 students with mixed ROS experience in 2 months
- Easy to integrate with other frameworks
  - OpenRave, Hubo-ACH, …
- Core is Documented, Integrated debug tools

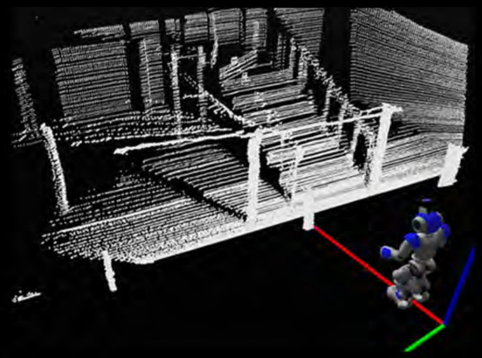





Framework : Integration With ROS

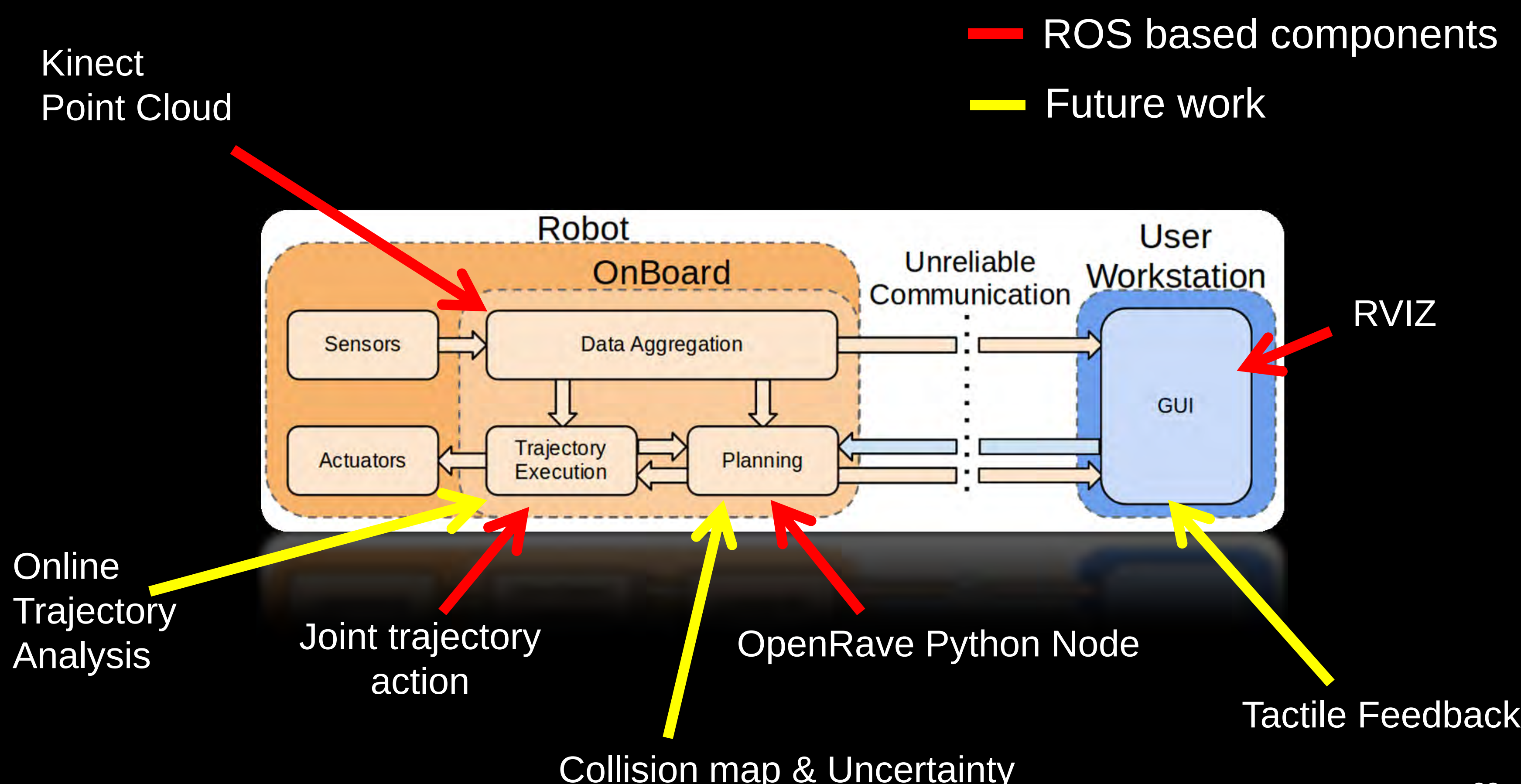
Why ROS?

- Large developer community and major libraries available
- Multi-language support
  - C++, Python, Java, Lisp, MATLAB, …
- Flexibility
  - Synchronous, Asynchronous, Pre-emptible Commands
- Robust Networking
  - Works across wired and wireless links
  - Presented framework tested using 1, 2, and 3 networked workstations
  - Demonstrated with Hubo



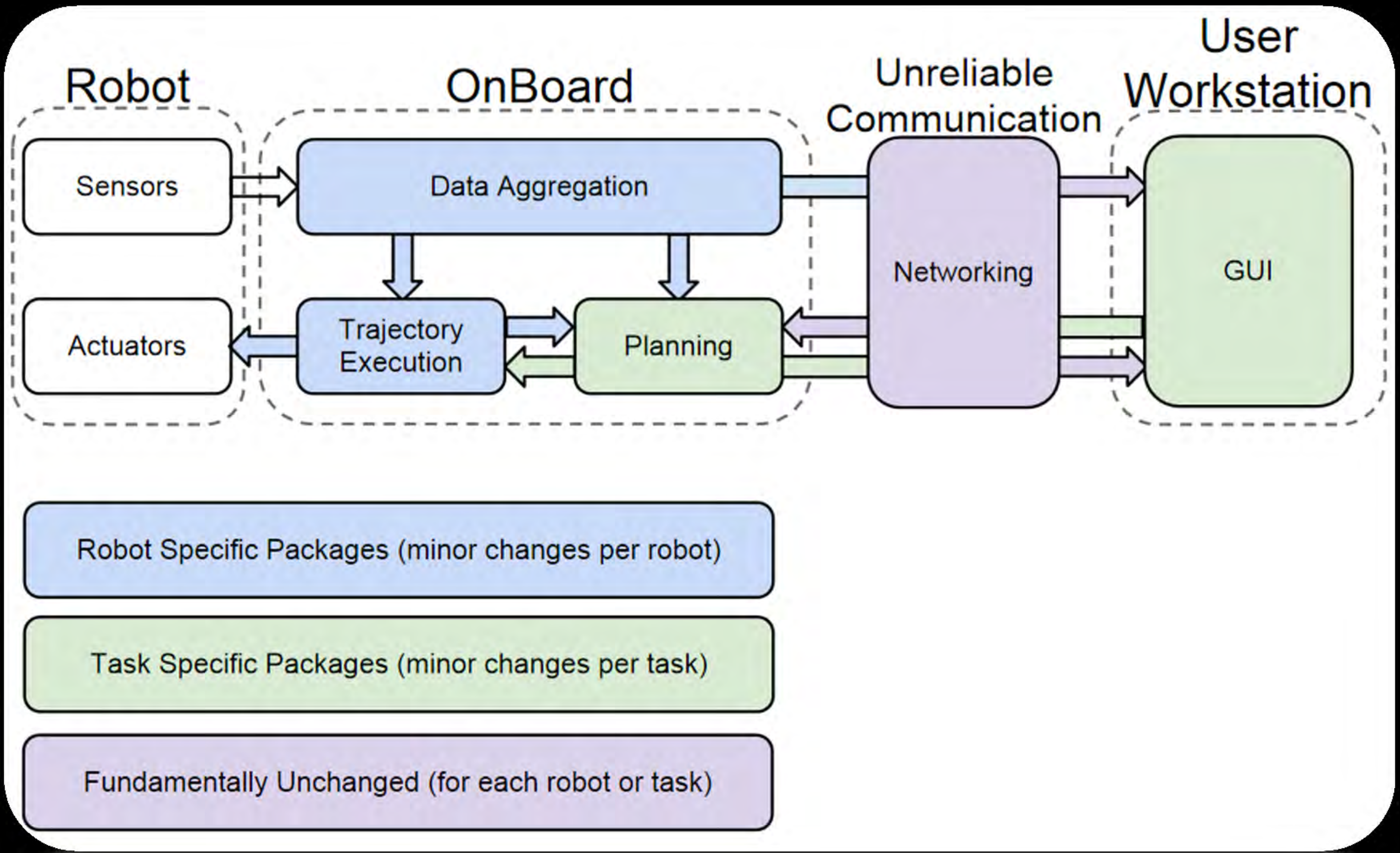
Architecture & Future Work

We chose to use ROS as much as possible



Framework : Integration With ROS

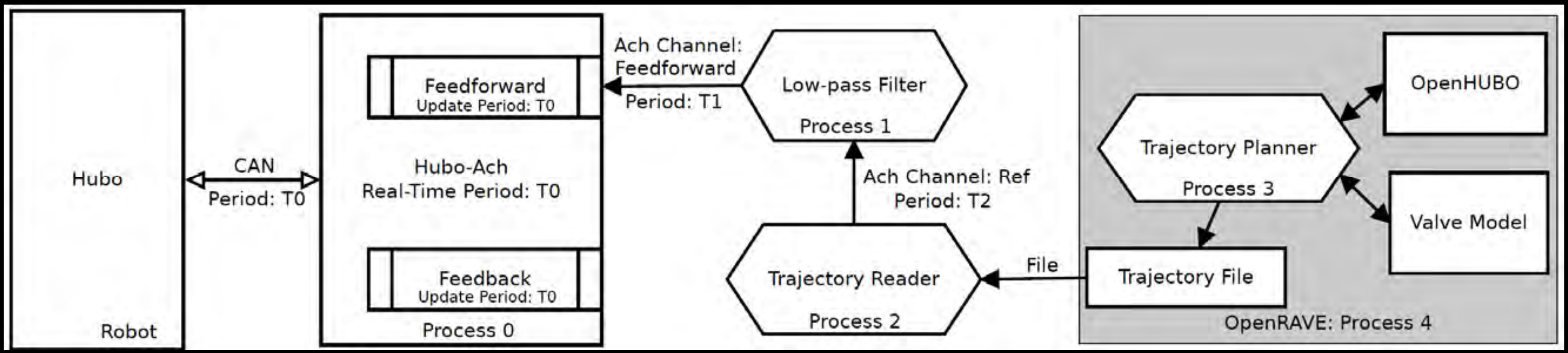
Packages can be reused for other projects



Framework : Integration with Hubo

- The Pr2 implementation relies on ROS components
  - RVIZ
  - Kinect Point Cloud
  - Arm controller

- The Hubo implementation builds on a real-time controller in ACH

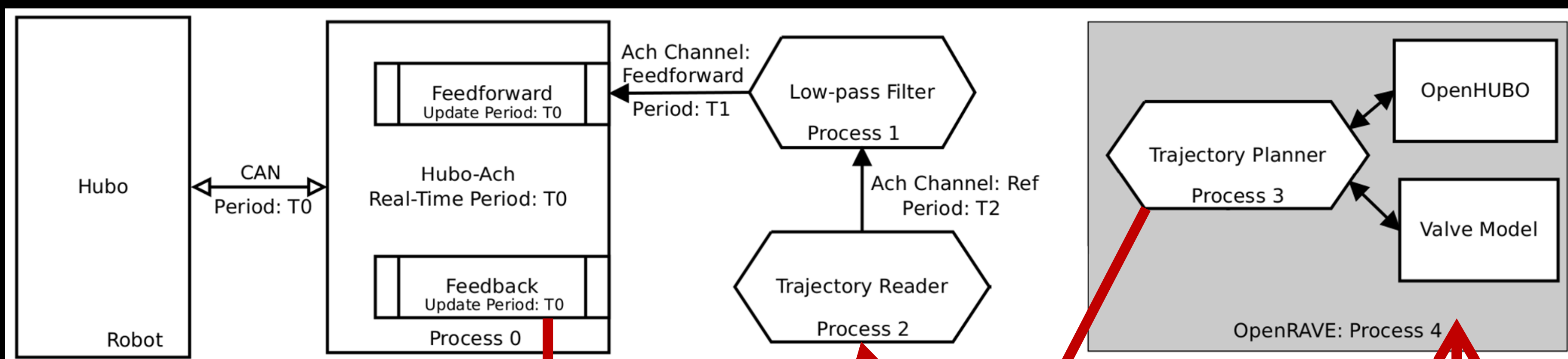


Schema : Dan Lofaro

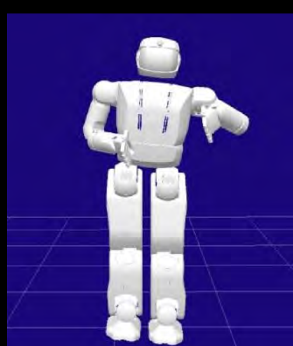
Framework : Integration with Hubo

The ROS-ACH bridge directly converts data streams and

Hubo state is displayed in RVIZ

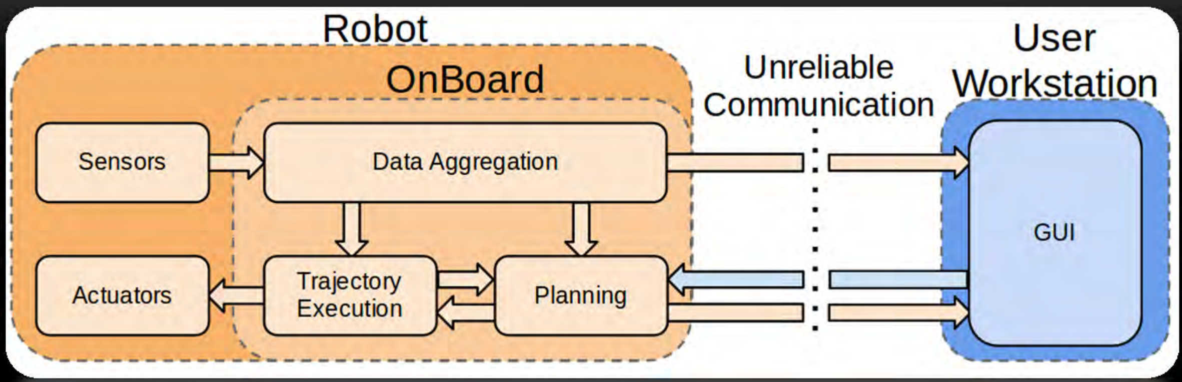


ROS-ACH Bridge



Hubo in RVIZ

Summary



- The User Guided Approach
  - Valve localization based on ICP and interactive marker
- Motion Planning
  - CiBiRRT, easy portable to different robots
  - Motion executed on Hubo and Pr2
- Error Detection
  - Library of trajectories
  - Do not requires visual feedback
- ROS implementation, ACH-bridge

Challenges and Future work

- Enhance the framework
  - More intuitive user interaction
  - Better placement of the robot, account for uncertainty in planning
  - More robust grasping

- Carry out the task on Hubo
  - ACH <-> ROS bridge
  - Available (Hubo-Ros, Hubo-Rviz)
  - Waiting for ACH simulator for testing
  - Use force feedback from wrists
  - Waiting for decision on perception hardware
  - Waiting on hand design

DRC : Valve Turning Task

Questions!!!

Links To Videos

PR2 Gazebo to OpenRAVE pipeline:

https://www.youtube.com/watch?v=heilzou0J6M

Hubo at MIT:

https://www.youtube.com/watch?v=DF2RDKGqnJk

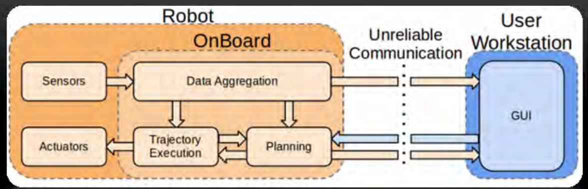
Hubo in OpenRAVE:

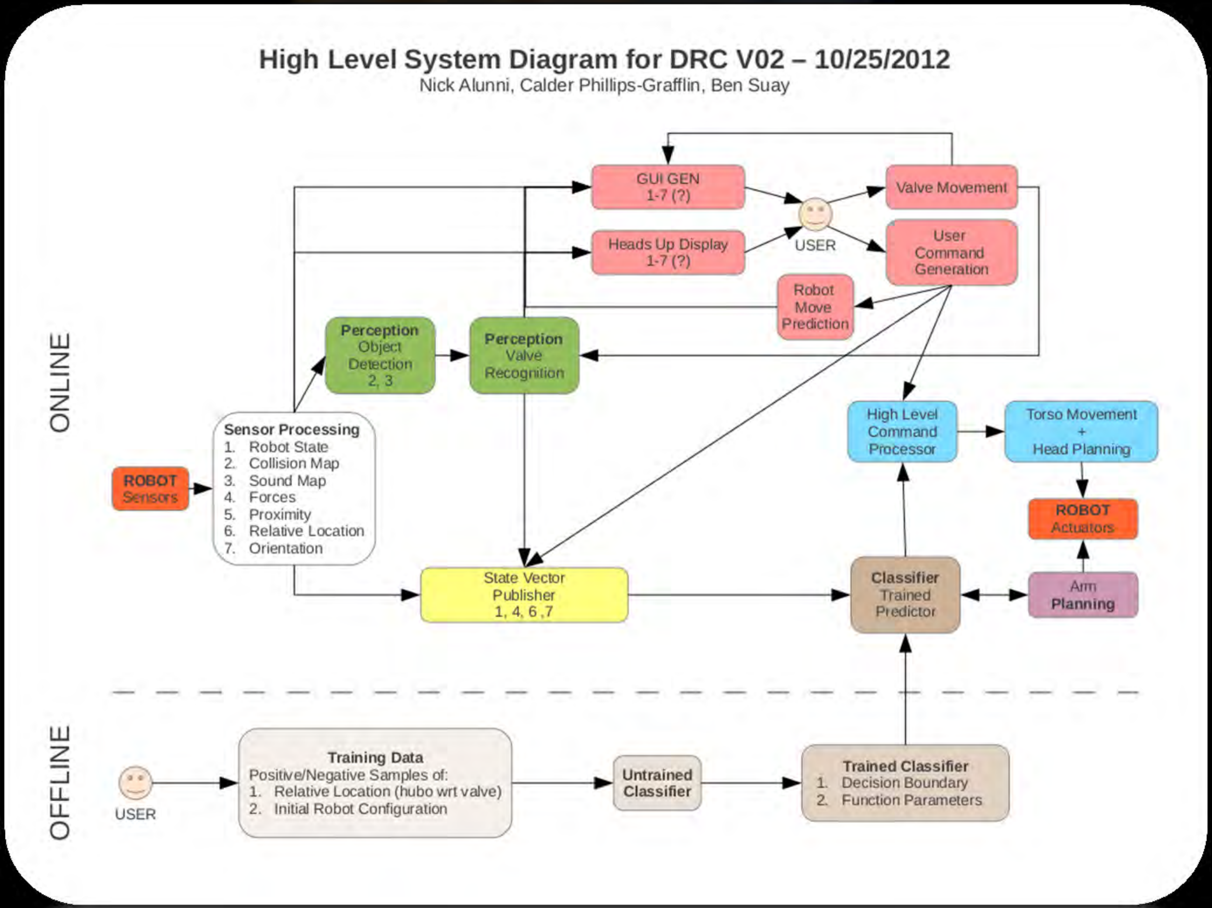
https://www.youtube.com/watch?v=LPzyJvbxHRY

User Guided Valve Turning:

https://www.youtube.com/watch?v=xRcUO2mXt3s

Framework : Evaluation





The user interacts with:

- The valve pose prediction
- Sends the command to the motion planner