Alex Brinkman

Team D: Project HARP (Human Assistive Robotic Picker)

Teammates: Abhishek Bhatia, Lekha Mohan, Feroze Naina, Abhishek Bhatia

ILR #4: Progress Report

Submitted 11/13/2015

1. Individual progress

   The last 2 weeks I mainly developed of the transform (TF) tree of the robot and environment, modified the PR2 ROS robot description to model the suction end effectors, integrated the MoveIt! arm controllers in the SMACH state controller, and iterated on the suction hardware design.

   We needed to extend the PR2's default TF tree to include the Kinect2 camera frame and add a world coordinate frame. My exposure to the PR2_plugs package when I was developing the SMACH state controller taught me how the developers on that project handed the transforms through their state machine so I took on this task. I started by creating a static broadcaster whose parent coordinate frame was the PR2's head link and rotated the frame to comply with the ROS convention for camera frames. I also made a static TF publisher called map that the PR2's TF tree inherited. This map frame allows us to specify real world poses for desired end effector positions. We wanted to demonstrate control of both TF frames and arm control so we made interactive markers in Rviz that generated poses in the global map frame with the intent of commanding the arm back and forth between 2 interactive markers. As their name suggests, these markers can be manipulated in real-time in simulation. These markers can be seen in Figure 1 and a video of this demonstration can be found on our website. This took a while to generate the pose feedback from the interactive markers but the online ROS tutorials and documentation made this task manageable.

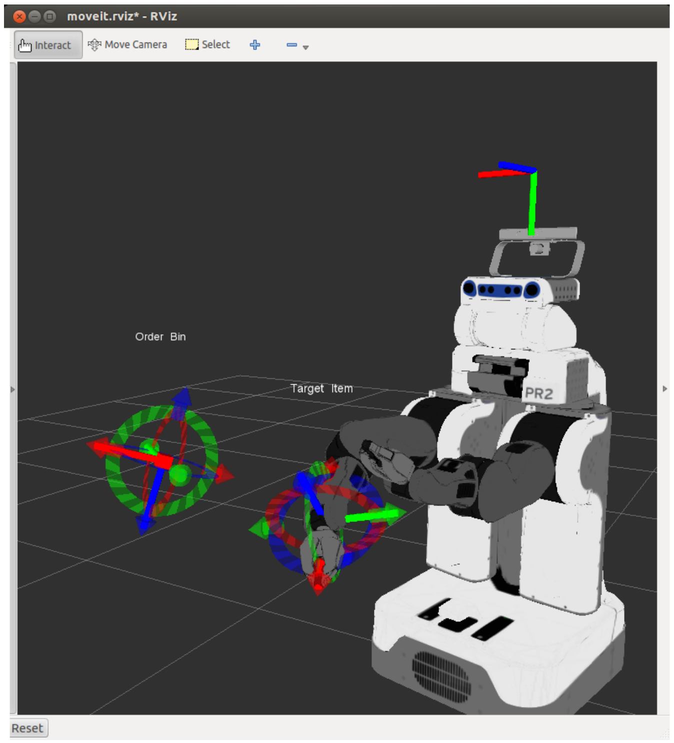

Figure 1: Rviz Simulation of PR2 with Interactive Markers

Next, I decided to work on modifying the PR2 Unified Robot Description Format (URDF) to create visual and collision models for the suction end effectors while Feroze worked on wrapping the MoveIt! arm controllers into a package we could use with the state controller. I forked a local copy of the PR2_Common package which contains the PR2 descriptions, meshes, and model constructor to make the modifications for our application and add the CAD files of our suction design. This seemingly simple task proved deceptively difficult but I was able to struggle through and successfully include these models as child elements of the right and left gripper tool frames shown in Figure 2.

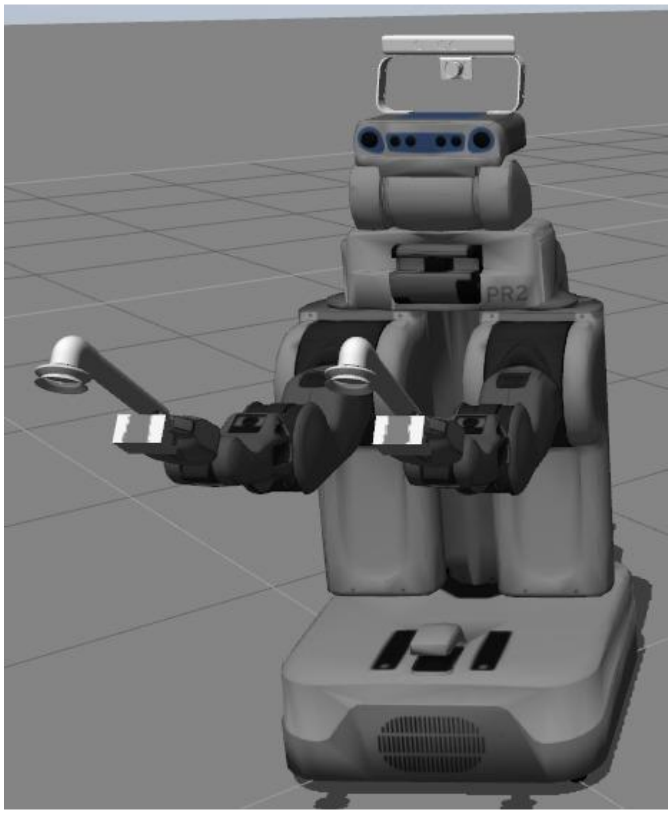

Figure 2: Gazebo Simulation of PR2 with Custom End-Effectors

Once Feroze successfully demonstrated the MoveIt! arm server, we worked toward the demonstration of moving the arm back and forth between two interactive markers. As expected, this integration did not go smoothly and took a while to debug. I had to figure out how to generate request callback for getting the pose of the interactive markers from the TF tree and pass them to the server. Once we were confident this was working correctly, we had to make minor modifications to the MoveIt! server to make the system work as intended. In the end, we were able to successfully demonstrate SMACH control of the PR2's manipulation capabilities and command the gripper to a location in the world.

We wanted to work toward a better iteration of the gripper design with the focus on final build quality. I generated CAD models, printed parts using the in-lab Makerbots, machined more grasping blocks for the PR2 hand holds, and ordered parts for the prototype. Figure 3 shows the final assembly complete with the PR2 grasping block.

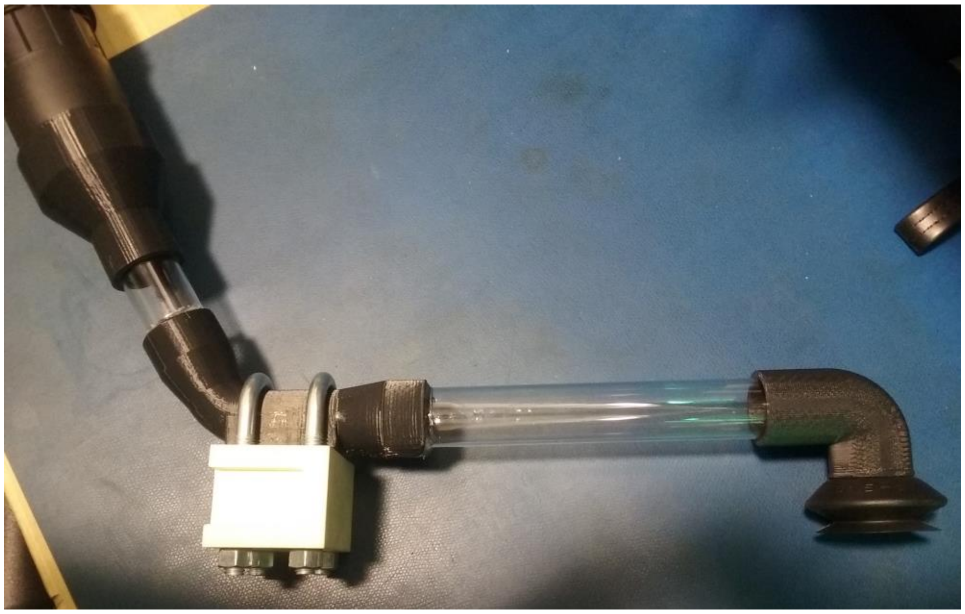
Figure 3: Latest Iteration of Suction End-Effector

2. Challenges

The main challenges we experienced were with the Groovy MoveIt!, modifying the URDF description of the PR2, and in integrating the PR2 simulation for the demonstration.

ROS Groovy has not received good support since 2013. In several cases, we have found that Groovy development stopped abruptly and developers focused on developing ROS Hydro packages, leaving out some key functions included in later versions of ROS. We discovered this is the case for the ROS Groovy MoveIt! package. Feroze and I struggled to get an arm controller working using a default package and were successful passing joint trajectories for the arm motion, however, we still needed an inverse kinematics solver in order to focus on our higher level executives. Noticing the C++ package was further developed than the python package, Feroze dug into the MoveIt! C++ package and made modifications and was eventually able get it to work.

Developing the URDF modifications for the PR2 required learning the URDF syntax. I dug into how the PR2 model was constructed and discovered a huge set of configurations and xacro files used in the PR2_common package. Xacro is the XML macro language parser that is used in ROS to help with XML scripting. I had to also learn how the xacros are called and how to use xacro scripts. Eventually I was able to get the PR2 model to build which was the bulk of the hardship. The .stl meshes needed to be scaled correctly and rotated WRT the tool parent TF frame but these issues did not take long to rectify.

Finally, the integration for the arm movement demo required a lot of last minute debugging. A few of these problems are listed below:

- The first version server only simulated the arm motion plan but did not execute the movement in some simulation configurations which took a while to understand and fix.
- The SMACH request callbacks to the MoveIt! server were only being called counter-intuitively when we started the controller instead of every time we entered the appropriate state. We consulted the API and made a minor change to one of the keyword arguments to fix this one.
- The MoveIt! server was looking for Pose messages but the TF utility was using PoseStamped messages.
- The roslaunch configuration worked on my laptop but failed on the workstation and required additional configuration to work on both.

We expected these integration problems and continued to work through these problems right up until 20 minutes before the demonstration, which worked flawlessly.

3. Teamwork

Developing the perception algorithm took longer than expected. Rick and Lekha have been working almost full time on this task. We needed to get a test script working to evaluate the perception system so we could see how we were progressing on our performance goal of 70% item recognition and pose estimation within 3 cm. We decided we needed to reprioritize some resources from PR2 control to the perception testing so Abhishek put his linux and C++ skills to work on automating this process. The true pose of each item was taken from the Rutgers 2015 dataset and compared to our prediction. Together Rick, Lekha, and Abhishek were able to create a testing suite that showed a 53% accuracy with no refinements on the initial working system.

Feroze and I split up the PR2- based work between us. Feroze took the MoveIt! integration and I worked on the other aspects described in the Individual Progress section. Feroze and I scheduled enough time and were in constant contact to keep each other updated on our progress. Feroze mainly worked out of a git ‘moveit’ branch while I worked out of a ‘urdf’ branch. Once the markers and URDF changes were complete and Feroze had an early version of the moveit server working, we merged ‘urdf’ into ‘moveit’ to keep things clear between us.

Finally, while working on the gripper prototype we almost all had a hand in the design, fabrication, and assembly which was done mainly ad hoc in the lab.

4. Plans

Our plans for the week of November 16th are as follows:

Suction

- Final PCB and enclosure.
- Create test circuit with components and build final hardware.

Perception

- Procure computer for Kinect2.
- Run algorithms directly grabbing real-time data from Kinect-2.

Platform

- Demonstrate a simulation that can move base along one axis, move the arm to the center of 9 shelf bins, and then move the arm to the top of the order bin.

This week, I will be extending the TF tree to include a representation of the shelf and work at integrating a move base server into the SMACH controller. Feroze and I will continue to work closely to accomplish these tasks.