Feroze Naina M Dheen Mohamed Ismail

Team D - HARP

Teammates – Alex Brinkman, Rick Shanor, Abhishek Bhatia, Lekha Mohan

ILR06

Jan. 28, 2016

Individual Progress

After the Fall Validation Experiment, we made a decision to switch to the Universal Robots' UR5 robot platform. The UR5 robot has a larger workspace and is much faster than the PR2 for our use-case as the spinal and base movements are eliminated, giving us a competitive edge. This has also allowed us to upgrade to the better supported and well documented version of Moveit in ROS Indigo. Since the UR5 is fully ROS supported, most of our existing codebase was transitioned with minimal changes.

My main contribution was in implementing collision avoidance features from MoveIt. For this, I initially worked with the PR2 simulation and then switching to the UR5.

The collision object is added to the planning scene in MoveIt and the planner generates collision-free trajectories between the start and goal state. In our implmentation, the target position of the end effector is obtained from the vision pipeline and sent to a ROS server node (move_arm_server.cpp) which calls the MoveIt planner. The collision object was a mesh generated from the CAD model of the shelf.

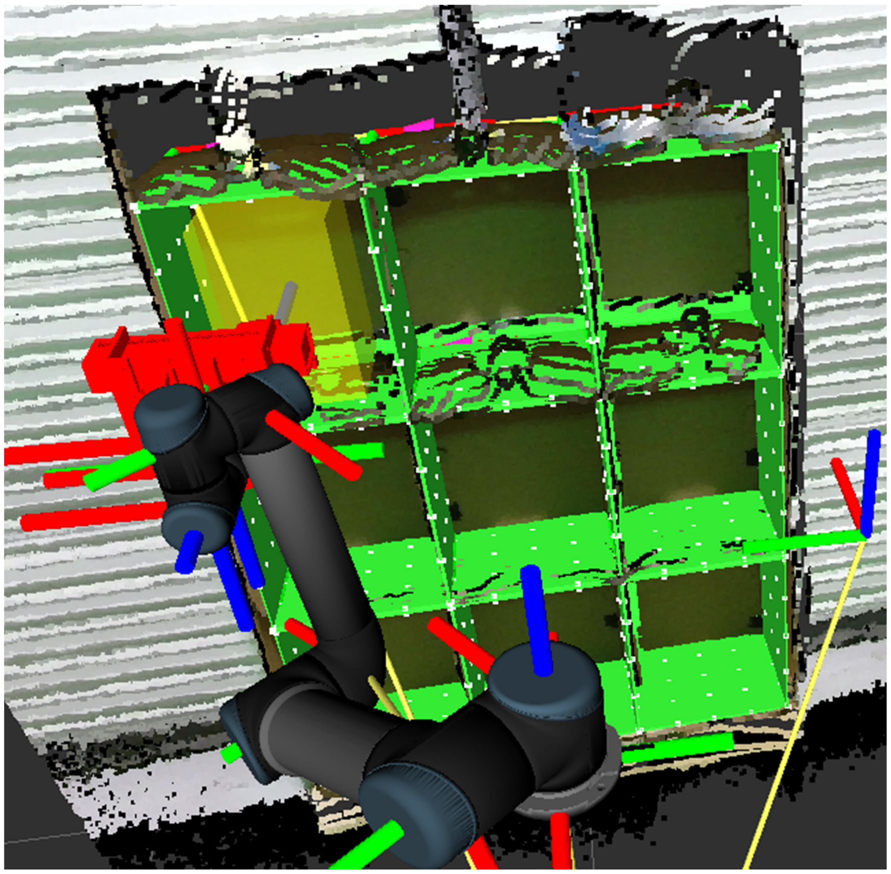

Fig 1 – Collision object and padded region

The above picture (Fig 1) shows the simulation of our system in Rviz. The green shelf is the collision object aligned after localization. The white points on the edges of the green shelf is the localized shelf pointcloud. The translucent yellow box inside the first shelf bin shows the allowable region for the grasp points generated by the vision pipeline. For the Project Review demo, we had provided 5cm padding on all sides. MoveIt uses the RRTConnect probabilistic algorithm to generate valid trajectories for the robot arm.

A key takeaway from was learning how to effectively use rosparam server for testing and debugging. The rosparam feature allows us to easily modify variables in runtime without recompiling the program. This could have saved us a lot of time if we had implemented it before.

I also installed an additional 512GB SSD for storing large 3D data-sets and upgraded the OS to Ubuntu 14.04 LTS. I setup up the ROS drivers for Kinectv2 and installed the NVIDIA drivers to fully utilize the computation power of the graphics card.

Challenges

While testing collision avoidance with the shelf and UR5, I directly used the points generated by the vision pipeline without first visualizing and verifying their pose in RViz. As a result, I repeatedly got collision errors for all items. I then successfully verified that MoveIt was capable of planning paths inside each shelf bin using interactive markers. Then, I bounded all the generated points inside a padded region in the shelf bin.

While generating the grasp point using the centroid of the item, we hadn't taken into account the width of the end effector. As a result, when items are touching the sides of the shelf, the generated grasp point causes the end-effector to collide with the shelf.

Another major issue we are facing is the visualization of generated paths by the default UR5 planner. The robot arm jumps from state to state in Rviz without clearly displaying the path taken. We have however verified that a valid collision-free path is being generated by observing the trajectory messages.

Team Work

Lekha worked on input handling of the item list. She parsed the JSON work order file and interfaced it with the smach state controller. She is currently working on making the perception system more robust by merging multiple pointclouds captured from different viewing angles to densify item pointclouds.

Alex worked on migrating the codebase from PR2 to UR5. Alex also modified the UR5 ROS packages and added a custom URDF for our suction end-effector (Fig 2). He is also collaborating with Andrew Dornbush from the Search Based Planning Lab for running the SBPL MoveIt plugin with the UR5 robot.

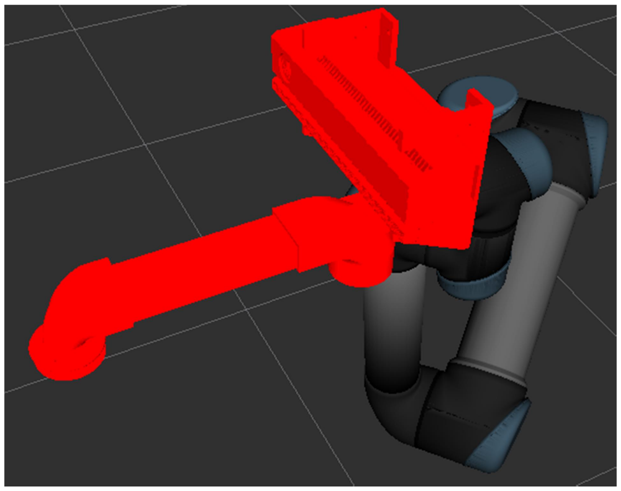

Fig 2 – Custom URDF for UR5 end-effector.

Rick worked on implementing shelf localization using Iterative Closest Point (ICP) algorithm. He also assembled a 9 bin shelf using cardboard boxes for testing the perception pipeline. Rick and Alex worked together in designing, fabricating and testing the new suction end-effector for the UR5. Rick also designed and created the Kinect mount for the UR5.

Bhatia worked on fixing the TFUtil package for ROS Indigo. Our system uses this package for calculating various transforms between frames. He also added a static transform broadcaster for each of the 9 shelf bins and assisted me with testing the collision avoidance system.

Plans

Our main focus for the next week is creating the demo video for the Amazon Picking Challenge (APC) 2016 application (deadline – February 5th). We would be submitting a video of our FVE demo on the PR2 as well as hardware-in-the-loop simulations using the UR5 platform. It is critical to demonstrate all our implemented work as the APC has become more competitive – the number of shortlisted finalists decreased from 25 teams last year to 16 teams.

I will be working to improve visualization, testing and debugging of the perception and planning pipeline by publishing poses for grasp points. I will be creating the collision object for the base platform of the UR5 from the CAD model.

Our new UR5 robot platform will be arriving next week and we would have to assemble it and integrate it with our system. Rick and Alex are also working on improving the end-effector and Kinectv2 mounts for the UR5.

Due to a change in the competition rules, we have to improve the perception system to handle up to 12 partially occluded items in each bin. For this, we are exploring Object Recognition Kitchen and SimTrack (kinectv2) packages. An additional stowage task has also been introduced and we will start planning towards it.