Object Recognition using Point Cloud Library

Mikhail Medvedev
University of Massachusetts Lowell
mmedvede@cs.uml.edu

ABSTRACT
The goal of robotics is to build systems that can autonomously perform some task in an environment. Such a system can not perform complex tasks without internalizing the outside world to some degree and then manipulating objects in the world. The project described in this paper uses point cloud to detect an object and ROS interactive markers to manipulate the object.

Author Keywords
PCL, pointcloud, Kinect, RANSAC, ROS, Bilibot, Interactive markers

Introduction
Object recognition and manipulation is something that seems very easy for a human and is hard for a robot. This project uses Random Sample Consensus (RANSAC)[1] algorithm from PCL library [2] to aid the object extraction from a pointcloud. The pointcloud is obtained from the Microsoft’s Kinect sensor. Once the object model is extracted from the point cloud, it is visualized in Rviz. Rviz also contains interactive markers that control the robot and its arm.

Project description
The object recognition from the pointcloud is achieved using the PCL filter pipeline. The input is the point cloud. The output is the model. In this case the model is a sphere, so the output produces the coordinates of the sphere (in the same frame as the point cloud) and the radius.

We do not use the raw pointcloud as it comes from the sensor, as it is too much data to process. So the first step is throttling down the rate to 2 Hz, this is fast enough for the purpose of the project. But even with the lower rate, the RANSAC segmentation algorithm would take a considerable time on the full cloud, so we use the VoxelGrid filter. The filter allows to “prune” extra points from the cloud, bringing the number of points down considerably. The grid size is chosen to be 1 cm. This size leaves enough points to detect the ball with radius of 7 cm. VoxelGrid filter also removes points farther than 1.5 meters from the sensor.

RANSAC filter in the PCL library works better with less outliers in the data set. The easiest way to remove uninteresting points is to remove the points that belong to the flat surfaces. So the reduced point cloud is pruned further using SACSegmentation node with the planar model. This filter is run 3 times. The result is that the 3 largest flat planes are removed from the pointcloud.

Finally, The SACSegmentation node is used with the spherical model for sphere segmentation. Additional nodes has been written to display the object and robot model in the Rviz interface. The interactive markers are used for visualization. This allows to add context menus to the detected object as well as to the representation of the robot. See Figure 2.

The tracking autonomous behavior was implemented. When enabled, it controls the robot base to turn whenever the object is moved.

Analysis of results
During the conducted tests, the system was able to detect the spherical object most of the time. When the cloud was not cluttered and there was only one spherical object in the field of view, the detection rate was very high, close to 100% accuracy. Otherwise, the rate could fall down lower than
The visualization worked well and it was easy to teleoperate the robot using the interactive markers. The implementation of autonomous object grasping proved to be outside of the project time constraints.

Discussion
The task of extracting a meaningful information from a point-cloud seemed straightforward, as there are existing libraries that do exactly that. But it still took a considerable amount of time to connect the pieces together and to find parameters that do work (for some conditions). I found out that even on the fast computer, the segmentation does take a lot of time. I have learned the structure of ROS PCL library and how to use it. It does not give a “turn-key” object recognition, but does provide all the necessary algorithms. The PCL is similar to OpenCV.

Rviz interactive markers proved to be a very useful and fast way to implement a variety of controls for the system.

Conclusions
The system showed good results of sphere segmentation for some conditions. The recognition for other objects (cylinders, cubes) would require running additional PCL filter nodes, but is possible. It is also possible to improve the performance by adding distance clustering and working with each point cluster separately.

In addition to improving the object detection performance, the implementation of autonomous grasping is the next logical step of the project. It would rely on the quality of the transform tree so the modeled hand correctly represents the real hand and the motion calculations can be carried out correctly.

Acknowledgments
The work described in this paper was conducted as part of a Spring 2012 robotics course, taught in the Computer Science department of the University of Massachusetts Lowell by Prof. Fred Martin.

Part of this project would not be possible without Eric McCann’s fix to the Bilibot arm firmware.

REFERENCES