Yu Liu

EC601-Mini Project 2

Report of ROS

1 Introduction

The Robot Operating System (ROS) is a set of software libraries and tools that can help on building robot applications. It aims to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

ROS owns numerous libraries and packages for nearly all robotics application, and it is open source. The developers of robots help build upon the foundation of ROS to give it additional functions by adding their own tools and libraries for other to use.

2 Summary of References

2.1 ROS Architecture

The ROS framework uses the concept of packages, nodes, messages, topics, etc. A node is a piece of software responsible for a specific task, and its output may be redirected to another node's input. The information which moves from node to node is called a message. Messages travel anonymously via special ports called topics. All related nodes are combined in one package that can easily be compiled and ported to other computers. The packages are necessary to build a complete ROS-based autonomous robot control system [1].

2.2 Integration with open source projects

ROS provides integration with open source projects. It has a message passing system that allows developers to swap out different data sources [2].

Integration between ROS and Gazebo (a 3D multi-robot simulator) is provided by a set of Gazebo plugins that support many existing robots and sensors.

Integration between ROS and OpenCV (a computer vision library) allows users to easily feed data published by cameras of various types into OpenCV algorithms, such as segmentation and tracking.

Integration between ROS and MoveIt (a motion planning library), allows plans to be executed via the ROS control system.

Integration with the Point Cloud Library, ROS Industrial, and other open source projects are also provided by ROS.

2.3 Core Components of ROS

Communication features include Message Passing, Recording and Playback of Messages, Remote Procedure Calls and Distributed Parameter System.

Robot-Specific Features include Standard Message Definitions for Robots, Robot Geometry Library, Robot Description Language, Preemptable Remote Procedure, Diagnostics, Pose Estimation, Localization, Navigation and Mapping.

The strongest feature of ROS is the powerful development toolset, which supports debugging and visualizing the state of the system. “rviz” provides three-dimensional visualization of sensor data types, while “rqt” can be used for developing graphical interfaces[3].

.

2.4 ROS-based Autonomous Navigation Wheelchair

The paper to be discussed is “ROS-based Autonomous Navigation Wheelchair using Omnidirectional Sensor” [1]. This paper presents work on integrating a method of image-based geo-localization in a powered wheelchair, mainly consisting of four parts: wheelchair instrumentation, vision-based geo-localization algorithm, collision avoidance algorithm, and the integration within a ROS framework. It focused on integrating the geo-localization algorithm within the ROS framework. Tests were conducted by using a camera fixed on the wheelchair control system, and the results show low control errors in straight line and curved paths.

3 Analysis of Results

3.1 Analysis of the paper

The work presented on the paper integrated two independent algorithms within a ROS framework. According to the paper, the estimated error was computed from the difference between the truth positions and the ones from their test runs. When it was in a straight-line path, the average error was almost zero. However, the error was much larger when the path

was complicated. Thus, the complexity of the path is negatively affecting the accuracy of the algorithms.

3.2 Pros and Cons of ROS

Pros:

1) Most of the common programing languages are supported by ROS, such as Python, C++, MySQL.
2) It is open source, with an active community that are still posting software.
3) Codes can also be integrated in other robotic software framework.
4) An ideal tool for prototyping and research.

Cons:

1) Lack of security features.
2) Incompatible with Windows OS.

4 Conclusions

ROS is a powerful tool for robotic development. Developers specialized in different fields of robotics can upload their packages for others to use. Also, ROS is compatible with many programing languages, which gives people the flexibility to program with a language that fits with their project. In my point of view, ROS is strong enough to create a self-driving wheelchair when the path is not complicated. However, there are more factors that needs to be considered about as the road condition becomes more complex, and ROS might not be reliable enough in this case.

References

[1] ROS-based Autonomous Navigation Wheelchair using Omnidirectional Sensor, International Journal of Computer Applications (0975 – 8887) Volume 133 – No.6, January 2016.

[2] https://www.ros.org/integration/

[3] https://www.ros.org/core-components/

[4] http://wiki.ros.org/

Team Paper Review

Shengyao Shao

This report mainly introduced the ROS-based Autonomous Navigation Wheelchair project. It summarized the hardware and software components of the wheelchair, together with the test results of the project. The author also analyzed the pros and cons of ROS, and would highly recommend using ROS for robotic projects.

Yanyu Zhang

This report firstly provided an introduction of the ROS features. Then it focused on the main concepts of ROS, such as Master and Node. It also presented entry-level instructions on how to install and run ROS, which is helpful to beginners. At the last part of the report the author discussed the Autonomous Navigation Wheelchair project as an example of the application of ROS.

(Paper review is not finished yet…)

Matthew Boyd

Haik Martirosyan