LAB 1 - R-IDE PRODUCT DESCRIPTION

Justin Tymkin

December 14, 2022

Old Dominion University

Professor J. Brunelle

CS410W

Contents

Figures

Introduction

The Robotics Operating System (ROS) is a software program created in 2007 by Eric Berger and Keenan Wyrobek, with the goal of assisting in building robots faster. They believed too little time was being spent building intelligent robots because of the time required to reimplement the programs for their robotics applications. Since its creation prestigious companies such as NASA have openly used ROS to build robots to send to space such as the Astrobee, and are currently using the newest version of ROS, ROS 2, to build VIPER a mobile robot to explore the south pole of the moon. With ROS being the software of choice the past ten years, the robotics industry has grown to be worth upwards of $249.2 million and estimates to be worth a little under double that mark in ten years.

Societal Problem

Big corporations have exposable income, unlimited time, and infinite manpower to create robots to visit the moon. Our first customer, Dr. Belfore, has a college budget and two semesters to teach new students how to use and implement ROS to compete in the Intelligent Ground Vehicle Competition (IGVC) once a year. This upcoming IGVC marks 30 years of competition for students in concentrations of electrical engineering, computer engineering, and mechanical engineering, from 33 different Universities across North America. Dr. Belfores efforts have been geared towards competing in the autonomous vehicle competition, combining aspects of all fields, to get his own vehicle to drive itself through a course designed by the IGVC. When Dr. BeIfore expresses frustrations with ROS, the question arises, if ROS is

good enough for NASA to create robots used in space, why is it not good enough for Dr. Belfore and his students competition in IGVC?

ROS is the best tool for building robotics, but it has a steep learning curve to master in order to be productive. For students with multiple obligations and limited time with ROS, many of them don't often begin to fully grasp ROS until a month before the IGVC. The timeline consists of, fall semester Dr. Belfore goes through ROS tutorials with students, then spring semester students get their hands on attempting to program with ROS on the Monarch 1, Old Dominion Universities IGVC autonomous vehicle. This cycle has led Dr. Belfore to seek ideas from outside his department.

Solution

ROS works in a multitude of developer environments, but often these environments are overwhelming and frustrating for new users. Programming packages for robotics using ROS on a Linux system can require multiple terminal windows, leading to a messy UI/UX. R-IDE will create an integrated environment for ROS users by developing an extension accessible through VSCode marketplace to lower the learning curve of ROS and empower new ROS developers by offering a friendly user environment.

R-IDE Product Description

The uniqueness of VSCode resides in their user marketplace which offers users flexible options to create your own IDE, as VSCode does not advertise itself as an Integrated Development Environment (IDE), but as a text editor. R-IDE has identified extensions already existing within VScode that will complement the R-IDE extension features. R-IDE itself will

provide a wizard for the user to create important components within a ROS package, along with visual tools like rviz, and various descriptive tools from rqt.

Key Product Features

Providing an extension within VSCode means the user will have everything they need at a click of a button, making for a friendly UI/UX. VScode offers the option to have buttons on the side of your window to easily navigate between our product, the text editor, and the debugger. VSCode also offers support in opening new windows for the user, allowing ROS developers to program for multiple nodes, or debug multiple packages at the same time.

Being able to create a ROS node, msg, or srv through the R-IDE wizard offers a unique tool to new ROS developers allowing them to quickly program their robotics. These are three things VScodes ROS extension does not currently offer to ROS developers. The wizard will walk through the steps of creation, then send generic templates to the text editor for the developer to begin creating their robotics application.

Allowing the user to record and play back a Rosbag at the touch of a button will offer more flexibility for learning what is happening within their product. A Rosbag records the raw data collected by the robot using the program by the user. With the option to play it back, this offers a seamless experience for new users to quickly understand any mistakes they may have made.

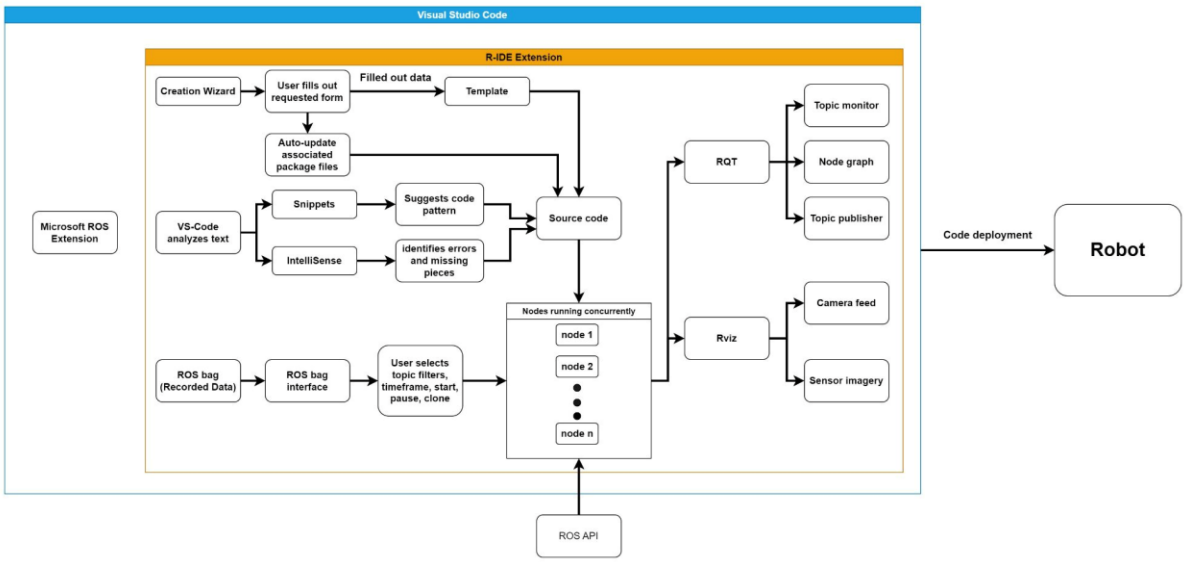
The addition of tools from rviz and rqt will offer tools for the user to develop a deeper understanding of what and how they are programming. Rviz is a 3d visual tool that replays what the user has programmed the robot to see, so the user can develop an idea of what they have

programmed it to see. Rqt is an extension for ROS which offers a variety of tools for ROS developers, we believe we have identified three important tools and plan to add them to our extension. The first tool from rqt being the node graph which is a graph that shows how your current node you are programming is communicating with other nodes. The second tool from rqt is the topic monitor which allows the user to select which ROS topic they would like to look at, and shows the types, bandwidth, and Hz of all topics. The last tool from rqt is the message publisher, like the topic monitor but for ROS messages.

Major Components

The major components of R-IDE can be seen in the figure below, the Major Function Component Diagram (MFCD). Inside of VScode, along with VSCodes ROS extension, the R-IDE extension will offer the key features in the flow below.

Figure 1



Major Functional Component Diagram

There are three starting points for a ROS developer inside of R-IDE, creating your node, msg, or srv, developing the application for the robot inside the text editor, or using the Rosbag to record data. These three options all stem from the creation of a node, once a node is created the user can utilize rviz or rqt. The R-IDE extension has no hardware requirements outside of the users choice of robot they are wishing to build. The software requirements are the same for any ROS developers needs, the ability to run ROS, rviz, and OpenGL.

Identification of Case Study

R-IDE is designed for new ROS developers and is being created inside of VSCode to encourage additional features be added to it as time goes on. ROS is always changing and adding new features and R-IDE wants to change with it, through the help of the community. The first iteration of R-IDE is focused on Dr. Belfores students, but we hope it has a broader appeal to ROS enthusiasts. If this is successful, maybe Dr. Belfore will have recommendations for future iterations.

Product Prototype Description

Prototype Architecture (Hardware/Software)

Prototype Features and Capabilities

Prototype Development Challenges

Glossary

Robot Operating System (ROS): ROS is a set of software libraries that helps to build robot applications. Ranging from drivers, to algorithms, and powerful developer tools, ROS is the preferred tool for robotics projects.

ROS Node: A node is a process that performs computation. Nodes are combined together and communicate with one another using streaming topics, RPC services, and the Parameter Server. These nodes are meant to operate at a fine-grained scale; a robot control system will usually comprise many nodes. For example, one node controls a laser range-finder, one Node controls the robot's wheel motors, one node performs localization, one node performs path planning, one node provides a graphical view of the system, and so on.

ROS Bag: A bag is a file format in ROS for storing ROS message data. These bags have an important role in ROS, and a variety of tools have been written to allow you to store, process, analyze, and visualize them. Bags are the primary mechanism in ROS for data logging, which means that they have a variety of offline uses.

ROS Master: The ROS Master provides naming and registration services to the rest of the nodes in the ROS system. It tracks publishers and subscribers to topics as well as services. The role of the Master is to enable individual ROS nodes to locate one another. Once these nodes have located each other they communicate with each other peer-to-peer. The Master also provides the Parameter Server and is most commonly run using the roscore command, which loads the ROS Master along with other essential components.

ROS Parameter Server: A parameter server is a shared, multivariate dictionary that is accessible via network APIs. Nodes use this server to store and retrieve parameters at runtime. As it is not designed for high-performance, it is best used for static, non-binary data such as configuration parameters. It is meant to be globally viewable so that tools can easily inspect the configuration state of the system and modify it if necessary.

ROS Messages: Nodes communicate with each other by publishing messages to topics. A message is a simple data structure, comprising typed fields. Standard primitive types (integer, floating point, boolean, etc.) are supported, as are arrays of primitive types. Messages can include arbitrarily nested structures and arrays. Nodes can also exchange a request and response message as part of a ROS service call.

ROS Services: Request / reply is done via a Service, which is defined by a pair of messages: one for the request and one for the reply. A providing ROS node offers a service under a string name, and a client calls the service by sending the request message and awaiting the reply. Client libraries usually present this interaction to the programmer as if it were a remote procedure call.

ROS Topics: Topics are named buses over which nodes exchange messages. Topics have anonymous publish/subscribe semantics, which decouples the production of information from its consumption. In general, nodes are not aware of who they are communicating with. Instead, nodes that are interested in data subscribe to the relevant topic; nodes that generate data publish to the relevant topic. There can be multiple publishers and subscribers to a topic. Topics are intended for unidirectional, streaming communication. Nodes that need to perform remote procedure calls, i.e. receive a response to a request, should use services instead. There is also the Parameter Server for maintaining small amounts of state.

Autonomous Machine: A machine capable of sensing its environment, carrying out computations to make decisions, and performing actions in the real world.

Rviz: (Ros Visualization) A 3D visualizer for displaying sensor data and state information from ROS.

Rqt: A QT-based framework for GUI development for ROS. It contains tools that support ROS topics, bags, node graphs, and many other tools for visualization and manipulation of ROS nodes.

Wizard: A user interface that presents dialog to lead a user through a sequence of steps. Often used to configure a service for the first time or to simplify a complex or unfamiliar process.

Template: An editable text or code snippet that can be filled with given values from another tool like a wizard.

Tutorial: A method of transferring knowledge that teach via example and supplies the information to complete a given task.

VSCode Extension: A tool designed to ass additional features and capabilities to VSCode. It can create new dialogues, add functions, or change the appearance of VSCode.

Refences

Ackerman, E. (2021, June 24). Microsoft announces experimental release of ROS for Windows 10. Retrieved November 3, 2022, from https://spectrum.ieee.org/microsoft-announces-experimental-release-of-ros-for-windows-10

Belfore, L. (jul 8, 2022). *Software Design Report for the ODU Monarch I* (Vol. 1.3, p. 3, Tech.).

Cardona, M., & Manzanares, J. (2020). *COVID-19 Pandemic Impact on Mobile Robotics Market* (pp. 1-4) (A. Palma, Ed.). IEEE. doi:10.1109/ANDESCON50619.2020.9272052

Fujita, T. (2018). Tomoya Fujita R&D center Sony Corporation - roscon.ros.org. Retrieved November 3, 2022, from https://roscon.ros.org/2018/presentations/ROSCon2018_Aibo.pdf

Gerkey, B. (2014, September 1). Ros running on ISS. Retrieved November 3, 2022, from https://www.ros.org/news/2014/09/ros-running-on-iss.html

Global Robot Operating System Market Research Report 2022(status and outlook). (2022, June 29). Retrieved November 3, 2022, from https://www.marketreportsworld.com/global-robot-operating-system-market-21185690

Guerry, M., Müller, C., Kraus, W., & Bieller, S. (2021, October 28). IFR International Federation of Robotics. Retrieved November 3, 2022, from https://ifr.org/downloads/press2018/2021_10_28_WR_PK_Presentation_long_version.pdf

Lunar Rover. (2021). Retrieved November 3, 2022, from https://www.openrobotics.org/customer-stories/lunar-rover

Robot Operating System Market. (2022, August). Retrieved November 3, 2022, from https://www.futuremarketinsights.com/reports/robot-operating-system-market

Robot operating system. (n.d.). Retrieved November 3, 2022, from https://www.ros.org/

Wessling, B. (2022, February 11). Open robotics developing space ROS with Blue Origin, NASA. Retrieved November 3, 2022, from https://www.therobotreport.com/open-robotics-developing-space-ros/