Jacob Jose[1], Kanav Sharma[2], Nihal Singh[3], Venkata Kanaka[4], Krishna Pandya[5], Jay Patel[6]

Software Engineering, Arizona State University, Tempe, Arizona

{jjose14,ksharm53,nnolas27,vgudimet,kpandya5,jpatel89}@asu.edu

Abstract

An analysis and comparison of two distinct platforms used for development of autonomous rovers which can traverse the given maze and send feedback to the operator. Rover development being the niche area, have very few resources to refer and a new user/future users spends a lot of time in environment setup, system configuration, validating tools compatibility. Limitations of sticking to specific versions of dependencies and constant reducing support for non LTS versions, make development even difficult. This paper. Nevertheless, very little is explored about the use and suitability of specific platforms for rover development. We fill this gap as we have already explored two different platforms and will detail out practice, comparison, perceived benefits, and related tools used.

Keywords: Unity, ROS, Linux, Gazebo, CLI, GUI, Automation Scripts, RViz, SLAM

Introduction

The Robot Operating System 2 (ROS) is a set of software libraries and tools that help us build robot applications. With a set of algorithms and powerful developer tools, this open-source software can be implemented through different modes and platforms. As the saying goes, a good start is half the battle won. Similarly choosing the correct platform and deciding initial strategy is the key in rover development and robotics. Therefore, selecting tools should be handled with a deeper understanding about the differences between the same. If properly analyzed and brainstormed, those steps can be automated and can be used to improve the next generation of users. In this paper, we focused on two such approaches for rover development

- Unity 3D with ROS2
- Ubuntu (Linux) with ROS2

Related Work

https://github.com/Unity-Technologies/Robotics-Nav2-SLAM-Example

Above reference is for exploration of an environment generated with the Robotics Warehouse randomizable environment.

PROS:

- Everything is manually set up which gives a detailed overview for every step.
- Detailed explanation of visualization and exhaustive documentation.

CONS:

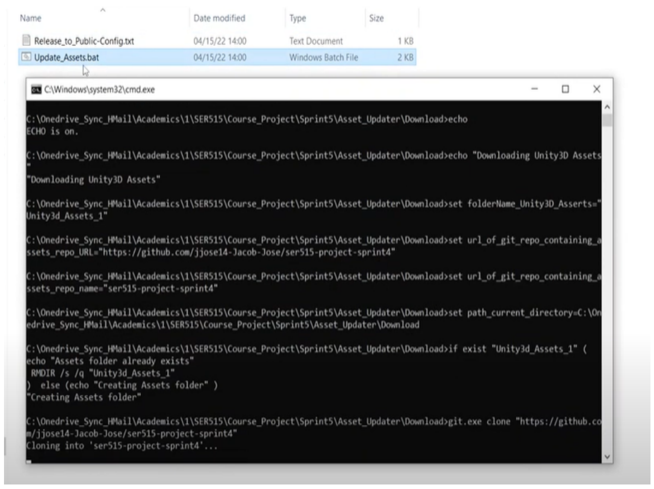
- Longer time to set up the environment.
- Many issues for a perfect flow (No Detailed explanation of integration of individual components).
- Inability of customization of rover as per user requirements.

Our Solution

Our solution focuses on omitting the time-consuming part of setup of environment for rover development, providing flexibility to choose desired configurations of the rover and environment & providing one stop shop for detailed documentation of how to make rover up and running. This helps users get ready with feature developments in a few clicks. We achieved it via three below enhancements:

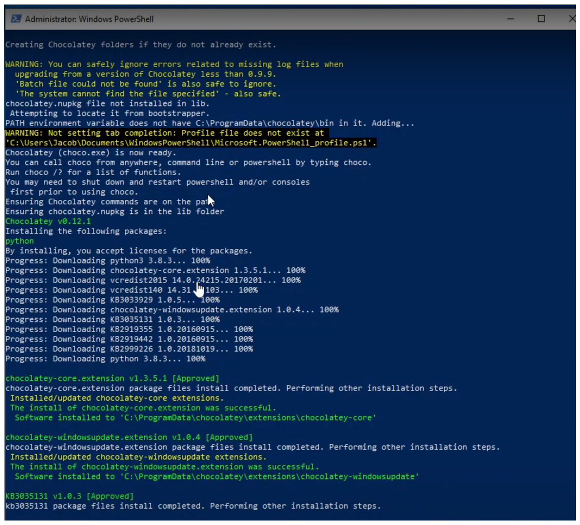
- ***Automation scripts**:* With implementation of syntactic sugar by creating automation scripts we were able to greatly reduce time needed for environment setup, downloading all required dependencies, managing LTS versions and ability to update dependencies with few changes. Setting up ROS2 on windows required ~20 different dependencies as below which was cumbersome for users and diverted from the actual goal of rover development.
  - Python
  - Visual C++ Redistributable
  - OpenSSL
  - Visual Studio
  - OpenCV
  - CMake
    - asio.1.12.1.nupkg
    - bullet.2.89.0.nupkg
    - cunit.2.1.3.nupkg
    - eigen-3.3.4.nupkg
    - tinyxml-usestl.2.6.2.nupkg
    - tinyxml2.6.0.0.nupkg
    - log4cxx.0.10.0.nupkg
  - Qt5
  - ROS2
  - Gazebo

With Automation script in place, all above dependencies are installed and configured within a few clicks.

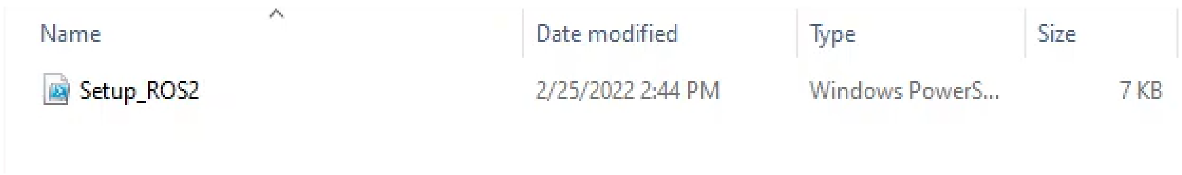


```
Name                         Date modified    Type                Size
Release_to_Public-Config.txt 04/15/22 14:00   Text Document       1 KB
Update_Assets.bat            04/15/22 14:00   Windows Batch File  2 KB
```

```
C:\Windows\system32\cmd.exe

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>echo
ECHO is on.

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>echo "Downloading Unity3D Assets
"
"Downloading Unity3D Assets"

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>set folderName_Unity3D_Asserts="
Unity3d_Assets_1"

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>set url_of_git_repo_containing_a
ssets_repo_URL="https://github.com/jjose14-Jacob-Jose/ser515-project-sprint4"

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>set url_of_git_repo_containing_a
ssets_repo_name="ser515-project-sprint4"

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>set path_current_directory=C:\On
edrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>if exist "Unity3d_Assets_1" (
echo "Assets folder already exists"
 RMDIR /s /q "Unity3d_Assets_1"
)  else (echo "Creating Assets folder" )
"Creating Assets folder"

C:\Onedrive_Sync_HMail\Academics\1\SER515\Course_Project\Sprint5\Asset_Updater\Download>git.exe clone "https://github.co
m/jjose14-Jacob-Jose/ser515-project-sprint4"
Cloning into 'ser515-project-sprint4'...
```

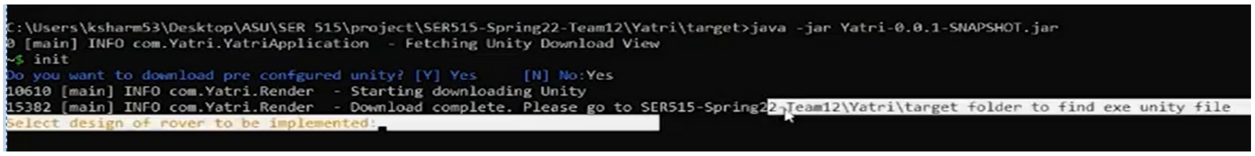


```
Administrator: Windows PowerShell

Creating Chocolatey folders if they do not already exist.

WARNING: You can safely ignore errors related to missing log files when
  upgrading from a version of Chocolatey less than 0.9.9.
  'Batch file could not be found' is also safe to ignore.
  'The system cannot find the file specified' - also safe.
chocolatey.nupkg file not installed in lib.
 Attempting to locate it from bootstrapper.
PATH environment variable does not have C:\ProgramData\chocolatey\bin in it. Adding...
WARNING: Not setting tab completion: Profile file does not exist at
'C:\Users\Jacob\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1'.
Chocolatey (choco.exe) is now ready.
You can call choco from anywhere, command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
 first prior to using choco.
Ensuring Chocolatey commands are on the pat
Ensuring chocolatey.nupkg is in the lib folder
Chocolatey v0.12.1
Installing the following packages:
python
By installing, you accept licenses for the packages.
Progress: Downloading python3 3.8.3... 100%
Progress: Downloading chocolatey-core.extension 1.3.5.1... 100%
Progress: Downloading vcredist2015 14.0.24215.20170201... 100%
Progress: Downloading vcredist140 14.31 103... 100%
Progress: Downloading KB3033929 1.0.5... 100%
Progress: Downloading chocolatey-windowsupdate.extension 1.0.4... 100%
Progress: Downloading KB3035131 1.0.3... 100%
Progress: Downloading KB2919355 1.0.20160915... 100%
Progress: Downloading KB2919442 1.0.20160915... 100%
Progress: Downloading KB2999226 1.0.20181019... 100%
Progress: Downloading python 3.8.3... 100%

chocolatey-core.extension v1.3.5.1 [Approved]
chocolatey-core.extension package files install completed. Performing other installation steps.
 Installed/updated chocolatey-core extensions.
 The install of chocolatey-core.extension was successful.
  Software installed to 'C:\ProgramData\chocolatey\extensions\chocolatey-core'

chocolatey-windowsupdate.extension v1.0.4 [Approved]
chocolatey-windowsupdate.extension package files install completed. Performing other installation steps.
 Installed/updated chocolatey-windowsupdate extensions.
 The install of chocolatey-windowsupdate.extension was successful.
  Software installed to 'C:\ProgramData\chocolatey\extensions\chocolatey-windowsupdate'

KB3035131 v1.0.3 [Approved]
kb3035131 package files install completed. Performing other installation steps.
```

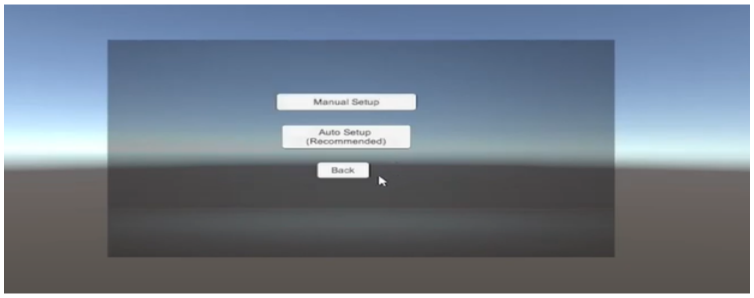


```
Name             Date modified        Type              Size
Setup_ROS2       2/25/2022 2:44 PM    Windows PowerS...  7 KB
```

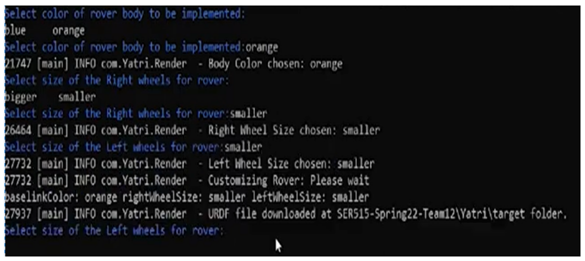
- ***Command Line Interface and User Interface:*** Having the environment setup complete using automation scripts, next we ensured users have flexibility to choose rover environment (Maze, terrain) and rover specification (Color, Size and number of rover wheels and body) of his/her choice. Having CLI and GUI will further help users to bypass configurations and focus on business needs of the rover. We implemented factory & bridge design patterns in JAVA, which helps avoiding multiple config files for each modification and have optimized implementation.
    - ***Configuring Environment:***



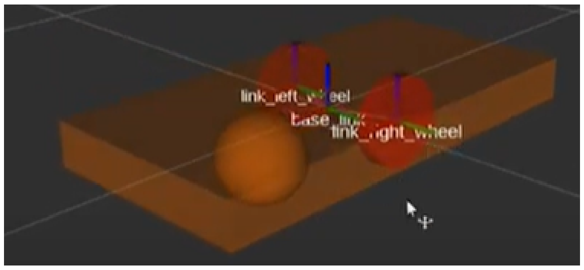
```
C:\Users\ksharm53\Desktop\ASU\SER 515\project\SER515-Spring22-Team12\Yatri\target>java -jar Yatri-0.0.1-SNAPSHOT.jar
0 [main] INFO com.Yatri.YatriApplication  - Fetching Unity Download View
-$ init
Do you want to download pre configured unity? [Y] Yes     [N] No:Yes
10610 [main] INFO com.Yatri.Render  - Starting downloading Unity
15382 [main] INFO com.Yatri.Render  - Download complete. Please go to SER515-Spring22-Team12\Yatri\target folder to find exe unity file
Select design of rover to be implemented:
```



- *<u>Customizing Rover Specifications:</u>*



```
Select color of rover body to be implemented:
blue    orange
Select color of rover body to be implemented:orange
21747 [main] INFO com.Yatri.Render  - Body Color chosen: orange
Select size of the Right wheels for rover:
bigger    smaller
Select size of the Right wheels for rover:smaller
26464 [main] INFO com.Yatri.Render  - Right Wheel Size chosen: smaller
Select size of the Left wheels for rover:smaller
27732 [main] INFO com.Yatri.Render  - Left Wheel Size chosen: smaller
27732 [main] INFO com.Yatri.Render  - Customizing Rover: Please wait
baselinkColor: orange rightWheelSize: smaller leftWheelSize: smaller
27937 [main] INFO com.Yatri.Render  - URDF file downloaded at SER515-Spring22-Team12\Yatri\target folder.
Select size of the Left wheels for rover:
```


link_[illegible]eft_w[illegible]eel
base_link
link_[illegible]ight_wheel

- ***<u>Exhaustive documentation:</u>***

https://github.com/ksharm53/SER515-Spring22-Team12#readme

The README section of GitHub repository acts as a one stop shop for all details required for downloading, building, and executing scripts, command line interface and user interface. Along with setup, README file acts as a library for each component used in the rover development, contributors and LTS versions to be used.

- ***<u>Unity Independent Maze and Terrain generation scripts:</u>***

Team emphasized on creating components which are independent of any version of Unity used. Although automation scripts and CLI does allow downloading LTS Unity 3D, creating C# scripts helps importing the same to another platform (Ubuntu) as well.

Comparison Table		
Evaluation Criteria	**Linux**	**Unity 3D**
Installation	● Installation of Unix / virtual box ● Installation of ROS2 ● Installation of Gazebo ● Running the python scripts written by the developer and executing the ezRos to view all the scripts and the configuration of the project	● Automation script for the installation of Unity3D editor and ROS 2 environment completely configured. ● By running the docker container the connection between ROS2 and unity editor will be established ● Operator receives the feedback from the rover mapping the maze ● Laser Scan sensor is being visualized to map the rover ● User Interface and Command Line Interface for modification of rover

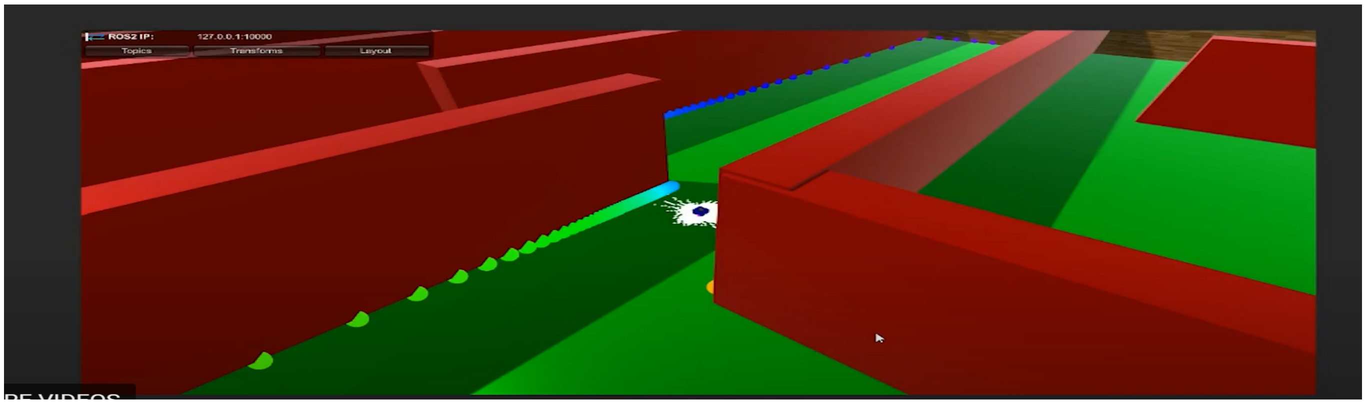
Ease of Use	● Command Line Interface (CLI) for rover customization. ● Installation setup test scripts.	● Command Line Interface (CLI) for rover customization. ● UI for rover customization. ● Extensive documentation. ● Unity independent C# scripts. (Maze, terrain, wheel, and bodies) ● Automation scripts.
Documentation	● Limited documentation available.	● Exhaustive Documentation (Taiga and GitHub) which could also work as a user manual.
Automation Script	● Install the packages locally.	● Set up the environment completely.
Possible Configurations	● Software allows simulating the rover without a laser and/or camera sensor. ● Dynamic maze generation. Maze can be saved.	● Three different sizes and numbers for the rover's wheels. ● Rover body and maze color customization . ● Dynamic maze generation.
System Visual Performance	● Maze generated is not as real-world as in Unity 3D. ● Inputs more responsive than Unity3D due to Gazebo's lesser GPU requirement.	● A real world environment can be rendered in Unity3D and the rover can map the room using SLAM. ● Unity 3D's high GPU requirement slightly affects system performance.
Host OS	● Project supports Ubuntu OS using Python. ● Project uses Linux Shell Script.	● Project supports Windows OS in Windows PowerShell. ● Project uses Windows Shell Script.
Rover Configuration Specification File	● Rover customization using Spatial Data File(SDF).	● Rover customization using Unified Robotics Description Format(URDF).
Rover Movement Visualization	● Uses ROS2, SLAM, RViz	● Uses ROS2, SLAM, RViz, Laser Scan.
Say Do Ratio (Agile metrics)	● 30:29	● 17:13
Project Compatibility with changing environment	● As the project is based on Ubuntu, the scripts for rover configurations like wheels, sensors and the maze are compatible with any stable version of ubuntu like ubuntu 18.04 ● The project uses python 3 script for CLI prompts to perform rover configuration, and hence python 3 is necessary	● The project is compatible with any version of Unity. ● Unity independent maze generation capabilities ● Unity independent Terrain, rover body and rover wheels ● Easing out the future users' tasks by providing them with the ability to customize the environment through CLI

Major Drawbacks faced	● Very limited online resources available for ROS2 compared to ROS1,hindering integration process ● Even number of wheels	● Connection of ROS2 and unity editor ● ROS TCP connection ● Generating and Validating URDF files

Result

In 6 sprints, the team was able to deliver a working rover with multiple ease of use features for future rover developers. Below are the capabilities our rover has post sprint 5

- o *Unity Independent Maze and terrain.*
- o *Rover body and wheels with jump capabilities.*
- o *Manual movement of the rover across a given maze.*
- o *Autonomous movement of the rover across maze.*
- o *Automation scripts for environment setup.*
- o *CLI and GUI for environment and rover customization.*
- o *Detailed documentation for rover setup, build & implement.*



Future Work

1. Functionality to add different sensors.
2. Ability to get feedback from multi rovers with cross rover communication.
3. Integrating with IOT.
4. Mobile application support.