NATIONAL UNIVERSITY OF SIENCE & TECHNOLOGY

School of Mechanical & Manufacturing Engineering

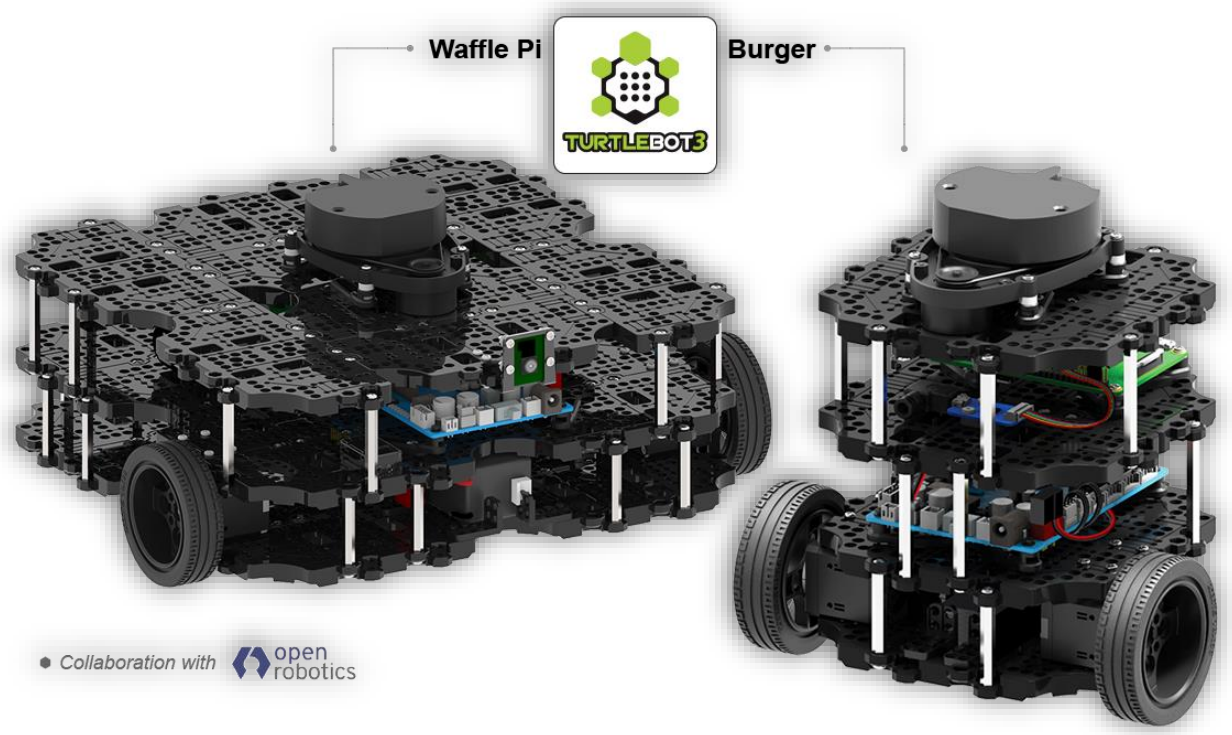
Department of Robotics and Intelligent Machine Engineering

Name	Muhammad Bilal Khan
Course	Mobile Robotics
Semester	2nd
Submitted to	Dr Yasar Ayaz
Date	25_04_2024

Examination of ROS and Gazebo Simulation Environment with the TurtleBot3 Robot

- **Abstract:**

  This paper assesses the usability of ROS (Robot Operating System) and Gazebo simulation environment through experiments on a TurtleBot3 robot. It also discusses ROS Development Studio and presents an overview of key ROS concepts and teleoperation implementation. Key terms include ROS, TurtleBot3, Gazebo, rviz, and ROS Development Studio.



➢ Gazebo:

A Powerful Tool for Tuning Robot Performance and Behavior

Gazebo's strength lies in its ability to create a highly realistic simulated environment for robots. This environment allows developers to tune various aspects of a robot's behavior before deploying it in the real world. Here is how:

- **Physics Engine**: Gazebo boasts a powerful and modular physics engine. This enables developers to tweak parameters like friction, gravity, and contact dynamics to simulate different terrains and scenarios. By adjusting these settings, developers can analyze how a robot responds in various conditions, allowing them to fine-tune its movement and performance.
- **Sensor and Actuator Models**: Gazebo provides highly detailed models of various sensors (like LiDAR or cameras) and actuators (like motors or wheels). These models allow developers to assess different sensor configurations and control strategies. They can adjust sensor noise levels and actuator response times to understand how these factors affect the robot's behavior and decision-making.
- **Plugin Ecosystem**: Gazebo's extensive plugin ecosystem opens doors for even more granular control. Developers can leverage plugins to simulate specific robot components, environmental factors (like wind or rain), or even introduce custom behaviors for testing purposes. This allows for highly targeted tuning of specific aspects of the robot's performance.
- **Data Logging**: Gazebo allows developers to record sensor data, control signals, and robot states during simulations. This data can be analyzed to identify areas for improvement and gauge the effectiveness of tuning efforts.
- **Visualization Tools**: Tools like rviz integrate seamlessly with Gazebo, allowing for real-time visualization of sensor data and robot motion within the simulation. This visual feedback helps developers understand how parameter changes translate into actual behavior, making it easier to identify and correct issues.
- **Benefits of Tuning in Gazebo**: Reduced Development Time: Testing and tuning in a safe, simulated environment eliminate the risk of damaging real robots during experimentation. This significantly reduces development time and allows for faster iteration cycles.
- **Improved Performance**: Precise tuning of robot parameters based on simulated data leads to optimized performance in the real world. Robots can move more efficiently, oversee complex environments better, and react more accurately to their surroundings.
- **Cost-Effectiveness**: Gazebo eliminates the need for expensive physical prototypes and real-world testing environments. This saves resources and allows for exploration of a wider range of design possibilities without significant financial constraints.

In conclusion, Gazebo empowers developers to refine and optimize robot behavior through its powerful physics engine, diverse models, and comprehensive data analysis capabilities. This simulation-based approach streamlines development, reduces costs, and leads to the creation of more capable and robust robots.

➢ Turtle Bot3:

TurtleBot3: Your Friendly Neighborhood Robot for Learning and Fun!

Imagine a little robot, small enough to scoot around your house, but powerful enough to learn and explore its surroundings. That is the magic of TurtleBot3! This affordable and easy-to-use robot is perfect for students, researchers, hobbyists, and anyone who wants to tinker with robotics.

- **Two Flavors to Choose From**:

TurtleBot3 comes in two versions: Burger and Waffle. Think of them like different car models. They both offer the same core features, but you can customize them with various parts to suit your needs. Want to add a camera or a more powerful computer? No problem!

- **Packed with Potential**:

Even though it is small, TurtleBot3 is a powerhouse. It can map its surroundings (like creating a mini map of your room!), navigate around obstacles, and even pick things up with the help of a special attachment called Open MANIPULATOR (think tiny robot arm!).

- **Learning Made Easy**:

TurtleBot3 is all about making robotics accessible. Control it with your laptop, a gamepad, or even your smartphone! It works seamlessly with ROS, a popular robot operating system, making it a great platform to learn the basics of robot programming.

- **Part of a Robotics Legacy**:

TurtleBot3 is not the first robot in its family. It follows in the footsteps of TurtleBot1 and TurtleBot2, each generation building on the successes of the last. This means you are getting a robot that has been refined and improved over time, making it even more dependable and user-friendly.

So, whether you are a curious student or a seasoned robotics enthusiast, TurtleBot3 is a wonderful way to get started in the exciting world of robots!

➢ ROS-GAZEBO IMPLEMENTATION EXAMPLE WITH TurtleBot3

I. **Using ROS and Gazebo with TurtleBot3**

This section explores controlling a TurtleBot3 robot with keyboard commands (teleoperation) using ROS (Robot Operating System) and Gazebo, a robot simulator. It then dives into the benefits of using ROS and Gazebo.

- **Teleoperation in Gazebo**

Launch Simulation: We use roslaunch to start a Gazebo simulation with a TurtleBot3.

Control the Robot: A program sends "Twist messages" (movement instructions) based on keyboard input.

Understanding Messages: These messages are published on a topic ("cmd_vel") for robots to listen to.

Making the Robot Move: "TurtleBot3-diff-drive" receives messages and translates them to control wheel speed and direction.

Visualization (Figures 5 & 6): These show the Gazebo simulation and software communication diagrams.

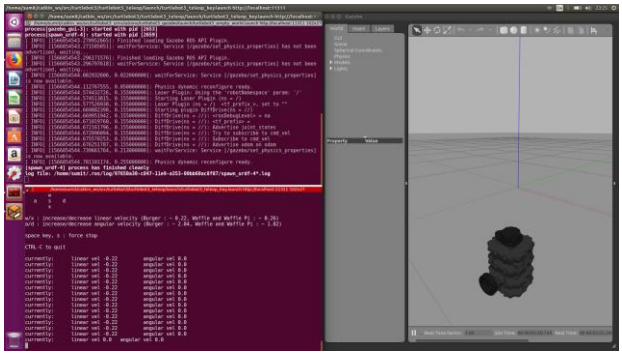


Fig. 5: Teleoperation simulation on Gazebo

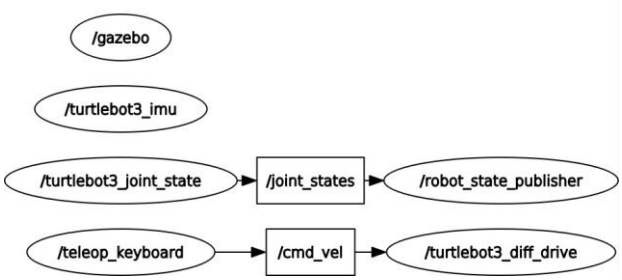


Fig. 6: RQT graph of teleoperation simulation

- **Teleoperation on Real Robot**

Sending Control Messages: A program sends Twist messages.

Receiving Messages: A program running on the robot receives the messages.

Starting Robot Software: A "bringup" node initializes ROS and makes the robot listen for messages.

Visualization (Figure 7): Shows software communication on the real robot.

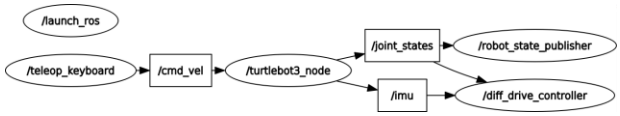


Fig. 7: rqt graph of teleoperation implementation on Turtle- Bot3

- **Benefits of ROS and Gazebo**

Simplified Communication: ROS manages communication between robot software components, saving time and improving organization.

Modular Design: ROS promotes modularity, making code development and reuse easier.

Data Recording & Replay: ROS allows recording and replaying sensor data for testing robot behaviors in different scenarios.

Coordinate Frame Transformations: The tf library helps manage data interpretation from different robot parts.

Robot Modeling: ROS provides tools to describe robots, making it easier for other ROS software to interact with them.

Powerful Development Tools: ROS offers tools for debugging, visualizing, and understanding the state of a robot system.

Visualization with Rviz: Rviz visualizes sensor data, helping developers understand how the robot perceives its environment.

Gazebo Simulation: Gazebo provides a powerful robot simulator with a realistic physics engine for safe virtual testing.

Learning Curve: ROS has a learning curve, but development becomes easier once familiar with the basics.

➢ Conclusion

This evaluation assessed ROS (robot brain) and Gazebo (simulator) for a real-world robot application. It evaluated on a TurtleBot3 in simulation and on the real robot, potentially using ROS Development Studio (RDS) for development. While ROS is powerful for robotics, its original design lacked security. To address this, ROS2 was created, utilizing the DDS standard for secure communication.

➢ REFERENCES

1) M. Quigley, B. Gerkey, and W. Smart, *Programming Robots with ROS*.
2) Sebastopol, CA: OReilly, 2015.
3) ROS, "About ROS", ROS.org. [Online]. Available: https://www.ros.org/about-ros [Accessed: March 05, 2020]
4) MathWorks Help Center," Exchange Data with ROS Publishers and Subscribers", The MathWorks Inc. [Online]. Available: https://in.mathworks.com/help/ros/ug/exchange-data-with-ros-publishers-and-subscribers.html [Accessed: March 08, 2020]
5) ROS," Core Components", ROS.org. [Online]. Available: https://www.ros.org/core-components [Accessed: March 05, 2020]
6) ROS Robots, "OpenManipulator", ROS.org. [Online]. Available: https://robots.ros.org/openmanipulator [Accessed: April 01, 2020]