Assignment B.Sc - Mobile Robotics

Turtlesim Automata (BA MECH 22)

Bachelor program - Mechatronics

4th semester

Lecturer: Daniel McGuiness

Group: BA-MECH-22

Author: Mats Pohle

July 5, 2024

Contents

1 TurtelSim Automata

1.1 SHORT INTRODUCTION

The goal of this assignment was to develop an autonomous turtle in the ROS2 simulation environment TurtleSim, which reacts to collisions with walls. After a collision, the turtle should turn 90 degrees clockwise. This required the implementation of a ROS2 node, publisher, and subscriber, as well as handling messages and timer events. Everything should be autonomous because of a Bash script.

1.2 DISCUSSION

At the beginning of the assignment, the development environment was set up, and the required dependencies were installed. Subsequently, a new ROS2 node named TurtleSimAutomata was created, responsible for controlling the turtle. A publisher was set up to send movement commands (Twist messages) to the turtle, and a subscriber to receive the turtle's pose messages and react to wall collisions.

One problem was the correct synchronization of backward movement and turning. Initially, there were difficulties in reliably turning the turtle clockwise. The angle calculation and adjustment of the angular velocity had to be modified several times. Additionally, there were issues with multiple "Edge detected" messages when the turtle hit a wall again during the turn. These were resolved by using state flags and more precise condition checks. Synchronizing the backward movement and turning required several adjustments to ensure that the turtle performed the actions sequentially and not simultaneously.

An additional problem was the folder structure. Due to an incorrect folder structure, the Bash often executed the wrong folder, which led to the entire structure having to be restarted multiple times.

The turtle was tested in the TurtleSim simulation environment. Various scenarios were run to ensure that collision detection, backward movement, and turning worked correctly. Several collisions at different points in the simulation environment were conducted to check the turtle's response. Finally, the turtle was run for an extended period to ensure it reliably responded to collisions and moved around the environment.

1.3 RESULTS

The implementation of the autonomous turtle in the TurtleSim simulation environment was successful. The turtle reliably responds to wall collisions, moves backward, and turns 90 degrees before continuing in a new direction. The challenges of turning direction and synchronizing movements were resolved through iterative adjustments and tests. The developed methods and the use of ROS2 components enabled a robust and reliable implementation.

1.4 LEARNING OUTCOME

Throughout this project, significant learning outcomes were achieved. The practical experience of setting up and working with ROS2 nodes, publishers, and subscribers provided a deep understanding of the ROS2 framework. The iterative process of debugging and resolving issues related to movement synchronization and angle calculations enhanced problem-solving skills and reinforced the importance of careful condition checking and state management.

Moreover, handling the folder structure and ensuring the correct execution paths highlighted the importance of a well-organized project directory, which is crucial for larger projects and collaborative environments. Working with Linux commands and the Bash shell provided valuable insights into effective project management and execution within the Linux operating system. The ability to adapt and troubleshoot effectively when faced with unexpected challenges is a key takeaway from this project.

Overall, this assignment not only achieved its technical objectives but also provided valuable insights into the practical application of ROS2 in robotic simulations and the efficient use of Linux for project management, fostering a deeper appreciation for the complexities and intricacies involved in autonomous robotic control.

2 Screenshots

The results can be seen in Figure 2.1.

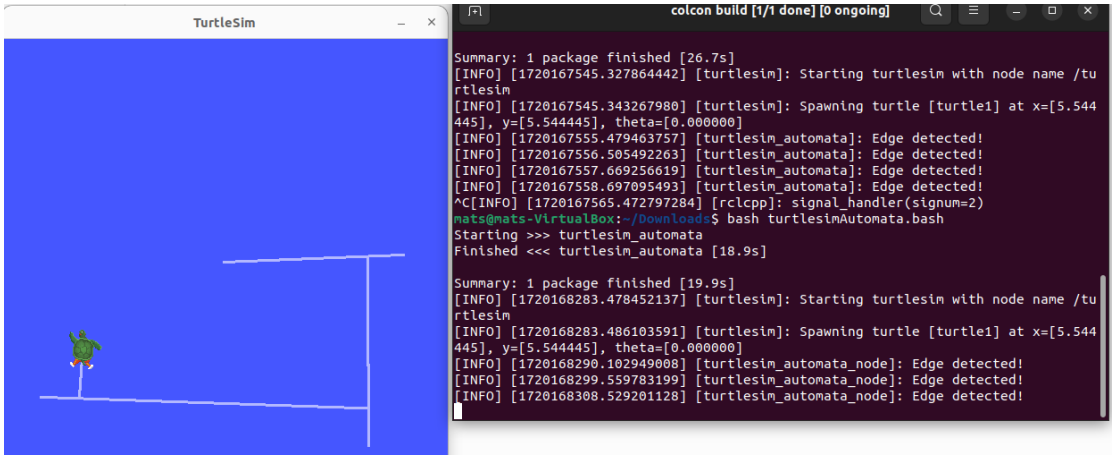

Figure 2.1: Turtlesim und Terminal

List of Figures