Autonomous Navigation of a Differential Drive Robot Using Lidar-Based Obstacle Avoidance

Students: Beilassan Hdewa, Lana Al wazzeh, Zain Alabidin Shbani.

1. Abstract

This report presents the development of a robotic simulation project focusing on autonomous navigation in a static environment with obstacles. A differential drive robot equipped with a lidar sensor has been modeled and controlled to reach designated goals while avoiding collisions. The system integrates perception, mapping, odometry, and control within the Robot Operating System (ROS) framework. The potential field method is implemented as the navigation strategy, providing real-time obstacle avoidance and goal-oriented motion.

2. Introduction

Autonomous mobile robots must be capable of perceiving their environment, localizing themselves within it, and making navigation decisions that ensure safe movement toward a target. This project demonstrates these principles through the design and simulation of a differential drive robot fitted with a lidar sensor. The robot operates in a static environment containing various obstacles. By combining odometry, occupancy grid mapping, and an artificial potential field (APF) controller, the robot achieves autonomous goal-directed behavior.

3. System Design

3.1 Robot Model

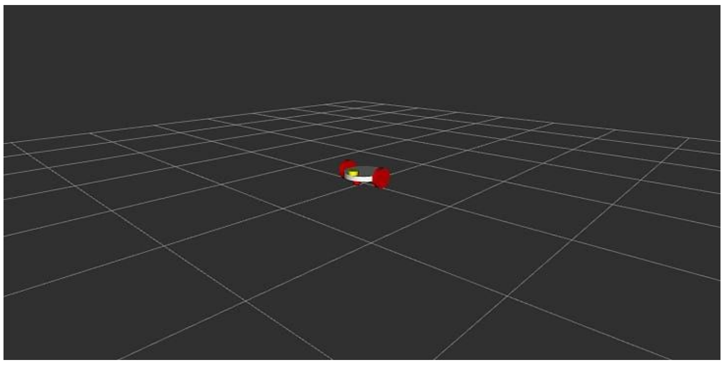

Figure 1: Differential drive wheels and a lidar sensor mounted on the robot's chassis in RViz

A Unified Robot Description Format (URDF) file was created to define the robot's physical structure. The model includes the differential drive wheels and a lidar sensor mounted on the robot's chassis. This configuration provides the necessary kinematic and sensing capabilities for obstacle detection and navigation.

3.2 Environment and Obstacles

A dedicated node, **complex_obstacles**, publishes various obstacles within the simulated environment using the visualization_msgs/MarkerArray message type. These obstacles form the static environment through which the robot must navigate.

3.3 Sensor Processing

The **lidar** node publishes real-time lidar readings of the surrounding environment. The laser scan data is used both for mapping and for the obstacle avoidance strategy applied in the control stage.

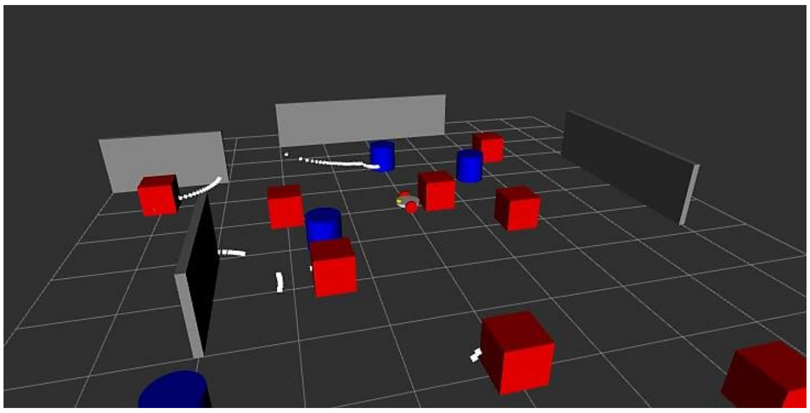

Figure 2: The simulated environment in RViz

3.4 Mapping

The **map** node constructs an occupancy grid based on the lidar scans. Cells corresponding to detected obstacles are marked as occupied, while unexplored

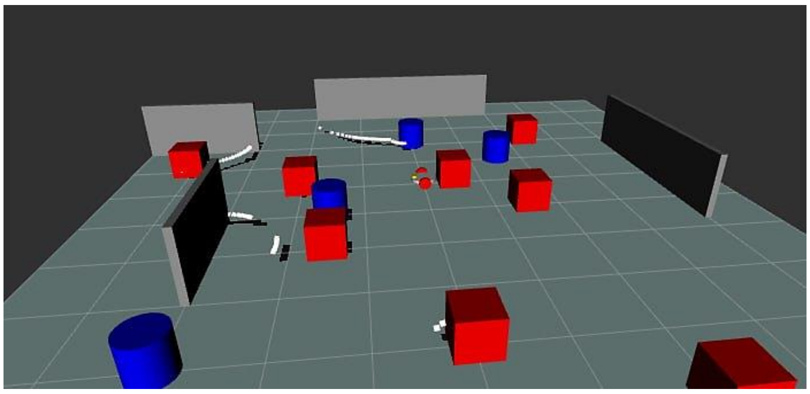

Figure 3: Spatial representation of the environment using the LIDAR's scan

areas remain unknown. The resulting map provides a spatial representation of the environment, essential for visualization and navigation.

3.5 Odometry

The **odom** node estimates the robot's position and orientation through the integration of wheel velocities. Odometry information is published in the standard nav_msgs/Odometry format, making it accessible to other nodes such as the controller.

3.6 Control

The **robot_controller** node implements the Artificial Potential Field (APF) method. Attractive forces are generated toward the goal, while repulsive forces are applied away from obstacles. The resultant force vector determines the robot's velocity commands, which are sent to the /cmd_vel topic to drive the robot. In addition, the transformation tree (TF) is employed to align sensor data with the robot's frame of reference, ensuring that motion vectors and goal positions are calculated consistently in the global coordinate system.

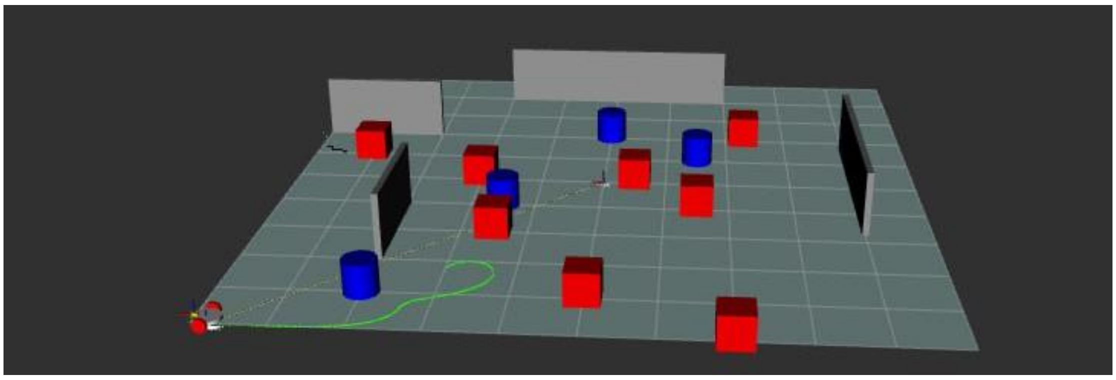

Figure 4: The Simulated environment in RVIZ with the robot's path and the LIDAR's scan

3.7 Hardware Abstraction

A **fake_arduino** node was implemented to simulate the functions of a motor driver and sensor interface. This node publishes lidar data on behalf of the robot's sensor and subscribes to velocity commands (/cmd_vel) issued by the controller. In this way, it emulates the role of hardware components that bridge sensor inputs and actuator outputs in a physical robot.

---

4. Implementation and Workflow

1. The simulation is launched, initializing the robot and environment.
2. The lidar node begins publishing scans, which are processed by the mapping node to update the occupancy grid.

3. Obstacles are simultaneously published through the complex_obstacles node for visualization.
4. Odometry information is computed by the odom node, providing the robot’s current pose.
5. A navigation goal is defined through the /set_goal service.
6. The robot_controller node applies the APF algorithm, balancing attractive and repulsive forces to generate motion commands.
7. The robot moves toward the goal while dynamically avoiding obstacles until the target location is reached.

---

5. Results and Discussion

The system successfully demonstrated obstacle avoidance and target-oriented navigation within the static simulated environment. The occupancy grid representation provided clear visualization of free and occupied regions. The APF method enabled smooth trajectory generation while ensuring that the robot did not collide with obstacles. Furthermore, the robot’s motion was visualized in RViz as a continuous trajectory trace, allowing the path taken toward the goal to be observed and evaluated. However, as is typical with APF approaches, potential issues such as local minima could occur in more complex environments.

---

6. Conclusion

This project highlights the integration of multiple robotics components within a ROS-based simulation. Through URDF modeling, sensor processing, mapping, odometry, and APF control, a differential drive robot was able to autonomously navigate toward goals while avoiding obstacles. The modular design of the package supports further extensions, such as dynamic environments, improved localization, or the use of global path planning strategies.
