Localization and Motion Control for a Differential-Drive Mobile Robot

Student: Almansur Kakimov
Instructor: Sanzhar Kusdavletov
Discipline: Introduction to Intelligent Systems
Department of Intelligent Systems and Cybersecurity
Astana IT University

February 2025

Abstract

The paper describes the implementation of localization and motion control algorithms for a differential-drive mobile robot. The project focused on solving the point stabilization problem with numerous goal points while also incorporating short-term reactive navigation to avoid obstacles. The system is simulated in MATLAB, and the results show that the robot can navigate through a dynamic environment while avoiding obstacles. The material begins with a deep theoretical foundation, covering kinematics, control laws, and obstacle avoidance algorithms, before moving on to implementation details and simulation results.

1 Introduction

Autonomous mobile robots are widely utilised for a variety of tasks, including warehouse automation, search and rescue, and environmental monitoring. A major challenge in robotics is allowing a robot to dynamically explore an environment, reach precise objective points, and avoid impediments. This study solves these issues by incorporating localisation and motion control algorithms into a differential-drive mobile robot. The robot is simulated in MATLAB, and its performance is assessed in a 2D grid environment with obstacles and several objective locations.

2 Theoretical Foundations

2.1 Kinematics of a Differential-Drive Robot

A differential-drive robot consists of two wheels mounted on a common axis, each driven by an independent motor. The robot's motion is determined by the velocities of the two wheels, v_L (left wheel) and v_R (right wheel). The robot's linear velocity v and angular velocity ω are given by:

$$v = \frac{v_R + v_L}{2}, \quad \omega = \frac{v_R - v_L}{L},$$

where L is the distance between the two wheels. The robot's pose (x, y, θ) in the global coordinate frame is updated using the following kinematic equations:

$$\dot{x} = v\cos(\theta), \quad \dot{y} = v\sin(\theta), \quad \dot{\theta} = \omega.$$

These equations describe the robot's motion in continuous time. For discrete-time simulation, the pose is updated as:

$$x_{k+1} = x_k + v\cos(\theta_k)\Delta t, \quad y_{k+1} = y_k + v\sin(\theta_k)\Delta t, \quad \theta_{k+1} = \theta_k + \omega\Delta t,$$

where Δt is the time step.

2.2 Localization

Localization is the process of determining the robot's stance in the surroundings. In this project, we assume complete localization, which means that the robot always knows its exact position and orientation. Localization in real-world circumstances can be accomplished using sensors such as encoders, IMUs, or cameras, in conjunction with algorithms such as the Extended Kalman Filter (EKF) or particle filters.

2.3 Point Stabilization

Point stabilization involves controlling the robot to reach a specific goal point (x_g, y_g) from its current position (x, y). The error between the robot's position and the goal is defined as:

$$e = \sqrt{(x_g - x)^2 + (y_g - y)^2}.$$

The robot's orientation error is:

$$\theta_e = \text{atan2}(y_g - y, x_g - x) - \theta.$$

A proportional controller is used to adjust the linear and angular velocities of the robot:

$$v = K_p e, \quad \omega = K_\theta \theta_e,$$

where K_p and K_θ are control gains. These gains determine the robot's response speed and stability.

2.4 Obstacle Avoidance

Obstacle avoidance is achieved using a reactive navigation approach. The robot detects obstacles within a radius of proximity r_p and adjusts its trajectory to avoid collisions. The repulsive force F_r from an obstacle is modeled as:

$$F_r = \begin{cases} K_r \left(\frac{1}{d} - \frac{1}{r_p}\right) \frac{1}{d^2} & \text{if } d \leq r_p, \\ 0 & \text{otherwise,} \end{cases}$$

where d is the distance to the obstacle, and K_r is a repulsive gain. The robot's new direction is computed as:

$$\theta_{\text{new}} = \theta + \Delta\theta,$$

where $\Delta\theta$ is the angle adjustment to avoid the obstacle. The robot's velocity is also reduced near obstacles to ensure safe navigation:

$$v_{\text{new}} = v\left(1 - \frac{d}{r_p}\right).$$

2.5 A* Pathfinding

The A* algorithm is used for global path planning. It minimizes the cost function:

$$f(n) = g(n) + h(n),$$

where $g(n)$ is the cost from the start node to node n, and $h(n)$ is the heuristic estimate of the cost from node n to the goal. The heuristic used is the Euclidean distance:

$$h(n) = \sqrt{(x_g - x_n)^2 + (y_g - y_n)^2}.$$

The algorithm explores the grid map, avoiding obstacles, and returns the optimal path to the goal.

3 Implementation

3.1 System Overview

The system consists of the following components:

- Map initialization with obstacles and target points.
- A* pathfinding for global planning.
- Reactive obstacle avoidance for dynamic navigation.
- Smooth motion control for point stabilization.

3.2 Simulation Environment

The simulation environment is a 2D grid map of size 10×10. Obstacles are randomly generated and goal points are placed in random locations, ensuring that they do not coincide with obstacles. The robot's initial position is $(1, 1)$, and its speed is set to 0.2 units per time step.

3.3 Algorithm Workflow

1. The A* algorithm computes the optimal path from the robot's current position to the goal.
2. The robot follows the path while continuously checking for obstacles within its proximity radius.
3. If an obstacle is detected, the robot adjusts its trajectory using the reactive avoidance algorithm.
4. The robot pose is updated using the kinematic equations, and the control loop ensures smooth motion toward the goal.

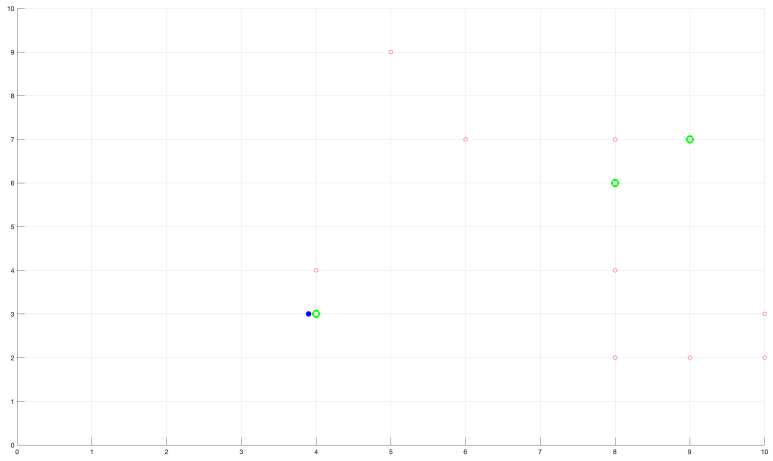

Figure 1: Simulation of the robot navigating through the map. The robot successfully reaches all goal points while avoiding obstacles.

4 Results

The simulation demonstrates the robot's ability to navigate through the map, reach multiple goal points, and avoid obstacles. Figure 1 shows the robot's path during the simulation.

5 Conclusion

The project has successfully implemented localization and motion control algorithms for a differential-drive mobile robot. Using A* pathfinding and reactive navigation, the robot moves through a dynamic environment, achieves numerous goals, and avoids obstacles. Future work may involve improving the obstacle avoidance algorithm, integrating more advanced localization techniques, and testing the system in a real-world setting.

6 Future Work

This section outlines potential directions for future research and improvements based on the current work:

- **Realistic Localization:** The paper assumes complete localization. Future studies could incorporate sensor noise and use localization algorithms such as the Extended Kalman Filter (EKF) or particle filters.
- **Dynamic Obstacles:** The current simulation includes static obstacles. Introducing dynamic impediments could improve the simulation's realism.
- **Advanced Control Laws:** The study employs a basic proportional controller. Implementing more advanced control rules (e.g., PID, MPC) may increase the robot's performance.

References

1. S. B. Niku, *Introduction to Robotics: Analysis, Control, Applications*, 2nd edn. Hoboken, NJ: Wiley, 2020.

2. Introduction to Intelligent Systems, Moodle, *Lectures, study materials, and labs.*