Project   
WLR (Wheeled & Legged Robots)

short line

Project Group Members:

**22BCE331** - Shikhar Panchal

**22BCE174** - Mann Patel

**22BCE041** - Jigar Bhoye

12th November, 2024

# Introduction

MATLAB Robot Simulation with Path Planning and Obstacle Avoidance

**Goal**

As a result of the simulation on MATLAB, a differential drive robot is determined to be capable of achieving its desired goal to navigate within a predefined map with obstacles using path planning and path optimization and Monte Carlo localization. The final task for the robot in this experiment is to have it start from an initial pose and end at a position pointed to by the target, making sure that it avoids obstacles along the way.

**Robot Setup**

The robot is described by a `differentialDriveKinematics` model with the following parameters:

Wheel radius, 0.05 m.

Wheel speed is unconstrained `[-Inf, Inf]`.

Track width, 0.5 meters.

Vehicle Inputs Vehicle speed and heading rate

**Simulation Environment**

As shown in Section A: Introduction, a binary map representing occupancy, `simpleMap`, is loaded, scaled with a resolution of 2 cells per meter, and obstacles set in the following specific regions: [1 3; 2 5; 4 8; 3 1.5; 4.5 3; 6 8].

**Path Planning and Optimization**

A `plannerHybridAStar` algorithm is used for path planning:

- Start Position: [1, 1, π/4]

- Goal Position: [8, 2, 0]

An initial path is created and optimized by the call to `optimizePath` in order to smoothness and obstacle safety. Optimizing options of the path are

Minimum turning radius : 2 meters

-Maximum Path States: Three times the initial number of path states.

- Safety margin of Obstacle: 1 m

These are some of the options we tuned for our robot can have efficient path so that it can easily travel and avoid collisons with obstacles.

**Sensor Model**

A rangefinder with a range of [0.1, 10] meters and a horizontal angle of [-π/3, π/3] is used as an obstruction detector within the field of view. The localization accuracy is also realized by using the Monte Carlo Localization algorithm applied with likelihoodFieldSensorModel, which relies on

Sensor Ranges: 0.1 to 10 meters

- 10 Beams for better environment sensing

**Monte Carlo Localization (MCL)**

The MCL algorithm uses 1000 particles for localization with an odometry motion model. The initial position along with the covariance matrix for the particles is defined, that provides robustness while moving toward the goal by estimation of its position.

**Navigation and Obstacle Avoidance**

The robot computes the heading error and count ticks based on what time it has been under changing mode to the target and adjusts its angular velocity, `omega`, to represent the heading error. When an obstacle falls within a `safeDistance` of 1 meter, the robot:

In other words, we need to adjust the omega and recovery values to get out of the way. And finally it returns to the original trajectory when out of obstacles

**Simulation Results and Visualization**

During the simulation, the following paths are visualized:

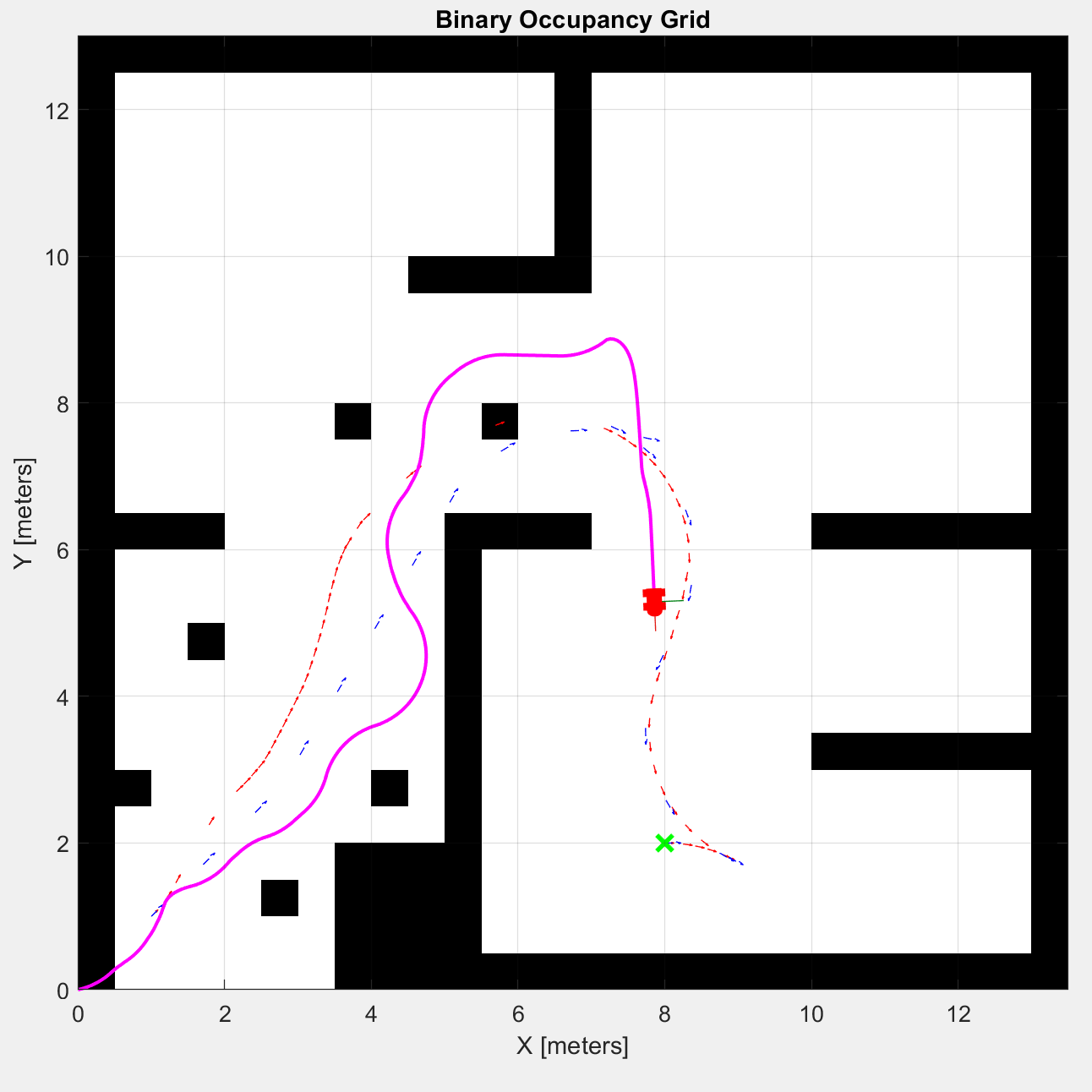
- Input Path: The initially generated path from the `Hybrid A\*` planner

- Optimized Path: The path after applying optimization

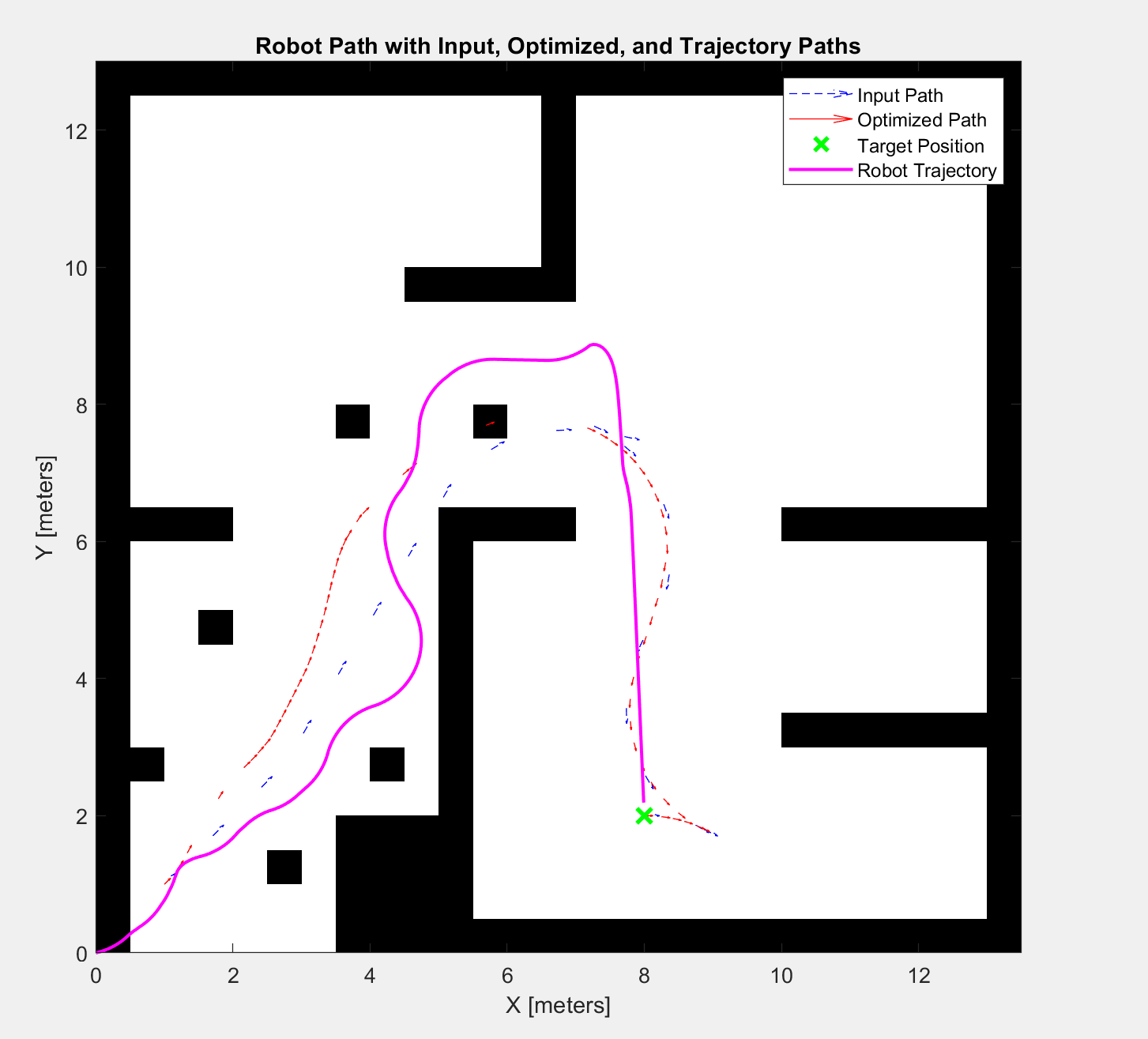
- Target Position: The target’s coordinates

- Robot Trajectory: The actual path taken by the robot, as updated in each loop iteration

A final figure displays these paths, allowing for performance evaluation against the planned and optimized paths.



*Fig 1. Robot Heading towards target.*



*Fig 2. The final path robot followed along with optimized path*

**Conclusion**

This simulation demonstrates how good the robot actually:

Hybrids A\* and Path optimization techniques should be used to plan and optimize a path.

Use reactive techniques to avoid obstacles.

Localize on the map using Monte Carlo Localization even in the presence of dynamic obstacles

**Scope of Future Enhancements**

1. Advanced Localization Accuracy:

Addition of Secondary Sensors: The localizing capability of the position in high precision can be obtained significantly enhanced by adding secondary sensors like LiDAR, ultrasonic sensors, or infrared sensors. Thus, the system under consideration can thus be improved using various kinds of multiple sensor fusion techniques to acquire a location of higher accuracy in diversified environments.

2. Obstacle Avoidance:

Machine Learning Obstacle Type Recognition: The system can be programmed to classify the types of obstacles by using machine learning algorithms and depending on them, select a specific avoidance strategy accordingly, like stopping in front of a human, but around static objects, one can turn.

These future improvements, when applied into the system, would probably be able to let it evolve into something smarter and more versatile that can deliver better performance in complex environments.

**References**:

- Optimized path & A\* ref: [optimizePath](https://in.mathworks.com/help/nav/ref/optimizepath.html)

-Monte carlo: [What Is Monte Carlo Simulation? - MATLAB & Simulink](https://in.mathworks.com/discovery/monte-carlo-simulation.html)

-Monte carlo Analysis: [Monte Carlo Analysis](https://in.mathworks.com/help/robust/monte-carlo-analysis.html)

-Robotic Toolbox: [Robotics System Toolbox](https://in.mathworks.com/help/robotics/)