Path Planning Algorithms of the Mobile Robots Control

Arbnor Pajaziti[1](✉), Lavdrim Panxha[1], Xhevahir Bajrami[1], Shaban Buza[2], and Mirlind Bruqi[3]

[1] Faculty of Mechanical Engineering, Department of Mechatronics, University of Prishtina, Prishtina, Republic of Kosovo
{arbnor.pajaziti,xhevahir.bajrami}@uni-pr.edu, lavdrim.panxha1@student.uni-pr.edu

[2] Department of Engineering Design and Vehicles, University of Prishtina, Prishtina, Republic of Kosovo
shaban.buza@uni-pr.edu

[3] Department of Industrial Engineering and Management, University of Prishtina, Prishtina, Republic of Kosovo
mirlind.bruqi@uni-pr.edu

Abstract. The purpose of this paper is to provide a complete analysis and comparison of the most used algorithms to plan the trajectories of mobile robots. By examining the principles, functions, and applications of these algorithms, it is intended to see their particular advantages and limitations. Each algorithm is analyzed in detail, including basic principles, mathematical fundamentals, and applications in the real world. Further, a comparative analysis has been developed, assessing algorithms based on various criteria, such as performance, applicability, limitations, computational requirements, and suitability. By examining these factors, we aim to provide a thorough understanding of the advantages and disadvantages of each algorithm. There are a wide range of algorithms and models available for controlling mobile robots. Depending on the specific application needs, various algorithms such as A*, RERT, Pure Pursuit, MPC, etc., as well as mathematical and Artificial Intelligence-based models for robot control, object tracking, obstacle avoidance, and many more can be used.

Keywords: Path planning · Mobile robot · Control · Simulation · Algorithm

1 Introduction

In recent years, the field of robotics has experienced a rapid evolution marked by significant strides in automation and autonomous systems. An essential driver of this progress is the development and integration of algorithms that enable robots to navigate and function in intricate and dynamic environments. These algorithms are pivotal in tasks encompassing robotic control, path planning, obstacle avoidance, and object recognition, among others. Evaluating the merits and drawbacks of these algorithms is pivotal in crafting efficient and dependable robotic systems.

G. Guxho et al. (Eds.): ITC-ICEE 2023, LNMIE, pp. 391–400, 2024.
https://doi.org/10.1007/978-3-031-48933-4_38

This paper's primary objective is to conduct a comprehensive analysis and comparison of the most prevalent algorithms in the realm of robotics. By scrutinizing the principles, functions, and applications of these algorithms, we aim to elucidate their specific strengths and limitations. Our methodology involves a meticulous algorithm selection process based on their ubiquity, relevance, and impact in robotics. Furthermore, we concentrate on algorithms that address fundamental challenges in robotics, including control, path planning, localization, perception, and learning.

Each chosen algorithm undergoes an in-depth examination encompassing its underlying principles, mathematical foundations, and real-world utility. Additionally, we perform a comparative analysis, assessing the algorithms across various criteria, such as performance, applicability, limitations, computational demands, and adaptability. Through this comprehensive evaluation, we endeavor to provide a profound comprehension of the advantages and disadvantages associated with each algorithm.

1.1 Types of Algorithms Used in Advanced Robotics

Recent years have seen remarkable advancements in robotics, enabling their application in diverse and complex scenarios, largely attributed to the utilization of advanced algorithms. These algorithms unify mathematics, engineering, AI, and computer science [1].

This paper offers a comprehensive overview of advanced robotics algorithms, elucidating their impact on enhancing robot capabilities and performance. Subsequently, we will delve into an assessment of widely utilized algorithms in the realm of advanced robotics, considering their applications [2, 3].

2 Robot Control Algorithms

Control algorithms are pivotal in directing and managing a robot's behavior by utilizing information from its sensors to determine actions required to meet predefined objectives. These algorithms find application in tasks such as executing precise movements, object manipulation, and trajectory planning. The following sections present evaluations of different algorithm types, highlighting their respective pros and cons.

2.1 PID Control

Various robot control algorithms are employed in advanced robotics, with the PID (Proportional-Integral-Derivative) algorithm being among the most prevalent. PID, a classic control system, integrates three components: the Proportional (P), Integral (I), and Derivative (D) terms [4].

Utilizing feedback from the controlled system, the PID algorithm computes and modifies control inputs, such as motor position or speed (Fig. 1). This algorithm comprises three components: the proportional component, dependent on the current error; the integral component, reliant on the accumulated error over time; and the derivative component, contingent on the error's rate of change. By fine-tuning these component parameters, the PID controller can achieve precise control behavior.

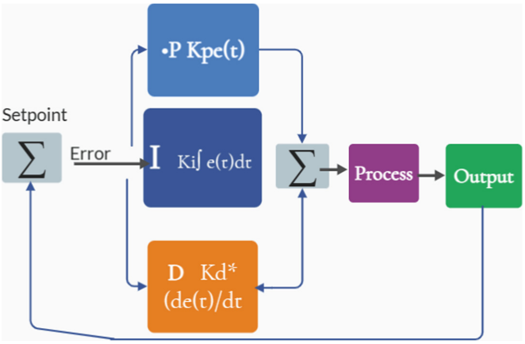

Fig. 1. Block diagram of PID [4].

PID represents a feedback-driven control algorithm that adapts control inputs in response to the current error, accumulated error over time, and the error's rate of change. While it offers simplicity in implementation and adjustment, it may encounter challenges when handling intricate systems and constraints.

2.2 MPC Control

Model Predictive Control (MPC) is a control system that employs a mathematical model of the controlled system to forecast future behavior. In the realm of robotics, MPC finds application in trajectory control, where the objective is to precisely track a predefined path (Fig. 2). This algorithm relies on a dynamic mathematical model of the system to anticipate its future evolution [5].

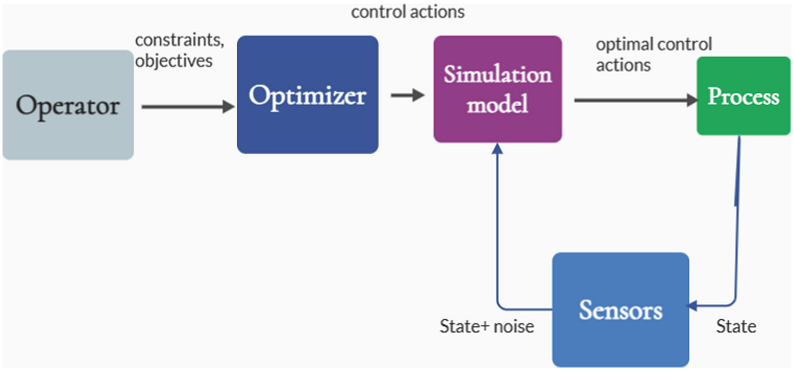

Fig. 2. Basic scheme of MPC [5].

MPC involves addressing an optimization challenge with the aim of minimizing an objective function that encapsulates the execution of control actions to meet predefined conditions and objectives. This optimization problem's resolution is influenced by control conditions, including action restrictions, load limitations, and trajectory requirements. MPC stands as an advanced control approach, especially well-suited for systems entailing constraints, as seen in domains like chemical processes, power generation, and industrial operations (Fig. 3).

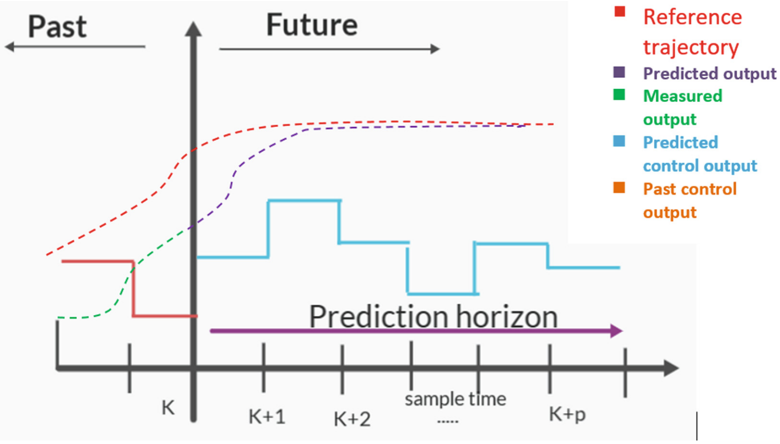

Fig. 3. MPC action method [5].

In MPC control, a process model is harnessed to forecast the future performance of the system, and these predictions guide the optimization of control actions to attain a desired level of performance. The model relies on the physical attributes of the system and aligns with the control objectives, and it can assume either linear or non-linear form. The MPC algorithm leverages this model to optimize control actions across a time span known as the prediction horizon [6].

A key benefit of MPC lies in its ability to handle constraints, encompassing input and output restrictions, while optimizing control actions to attain desired performance levels within these boundaries. Moreover, MPC exhibits resilience to model uncertainty and disturbances, rendering it well-suited for intricate and ever-changing processes. In essence, MPC represents a sophisticated control strategy that employs a system's mathematical model and future behavior predictions to optimize control actions for achieving desired performance outcomes.

2.3 Fuzzy Logic Control

Fuzzy Logic serves as a method for modeling information that accommodates a spectrum of values, not limited to binary "yes" or "no" distinctions. In the field of robotics, Fuzzy Logic has found application in addressing intricate challenges linked to robot perception, operation, and control within environments characterized by ambiguity or complexity (Fig. 4) [7].

In contrast to traditional binary logic, which relies on "yes" or "no" cutoff values to denote statement accuracy, Fuzzy Logic employs variables representing the degree of

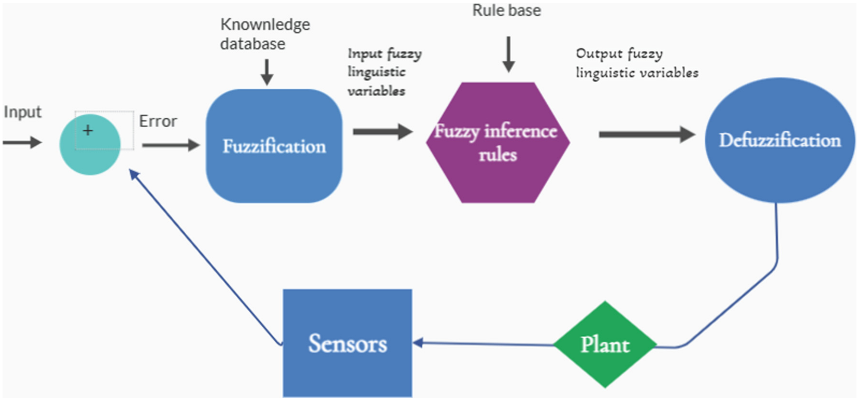

Fig. 4. A platform with Fuzzy Logic Controller [7].

truth or certainty in a statement. These variables manifest as functions that describe the connection between input and output values.

In the realm of robotics, Fuzzy Logic proves valuable for modeling robot perception, decision-making, and behavior control. A Fuzzy Logic Controller (FLC) can intake data from diverse sensors, such as ultrasonic sensors for obstacle distance detection and global positioning systems for determining the robot's position. Through fuzzy inference, the FLC generates an output that enables the robot to distinguish environmental changes and execute designer-desired actions. For instance, a wheeled robot may adjust its rotation speed to navigate corners and avoid obstacles, while multi-legged robots could utilize inputs like roll angle or forward speed.

Leveraging Fuzzy Logic in robotics offers numerous benefits. It facilitates the management of ambiguous and unstructured information, enhancing a robot's adaptability in challenging environments. Furthermore, FLC can handle intricate, non-linear relationships between inputs and outputs, enabling the execution of sophisticated behaviors.

3 Path Planning and Obstacle Avoidance

A path is a sequence of poses connecting a start configuration to an end configuration while adhering to specific requirements, such as obstacle avoidance or force profiles. It differs from a trajectory, which represents the execution of a path over time.

Path planning and obstacle avoidance are critical facets of robot control within a given environment, ensuring safe and efficient robot movement by finding optimal paths while avoiding obstacles.

Path planning entails identifying the best route to a destination. The robot utilizes environmental data and its goal to determine the most appropriate path, considering factors like open spaces, obstacles, and employing search algorithms to optimize based on criteria such as speed, length, or convenience.

Obstacle avoidance focuses on detecting and evading obstacles in the robot's path, safeguarding the robot's integrity and enabling obstacle-free navigation. Sensors like cameras, LIDAR, or ultrasonic sensors provide data for obstacle detection and trigger actions like path adjustments, speed reduction, or turning to ensure safe robot movement.

During path planning and obstacle avoidance, the robot utilizes environmental inputs, including obstacle proximity, type, movement limitations, and the defined goal. The choice of planning algorithm can lead to paths with varying degrees of optimization, considering factors like path length, turns, or braking.

Implementing these algorithms typically involves advanced mathematical and computational techniques, such as pathfinding algorithms and AI. The choice of algorithm depends on the robot's application and the environment's characteristics.

In essence, path planning algorithms find optimal routes for robots, addressing path mapping and obstacle identification. Obstacle avoidance algorithms help robots detect and navigate around obstacles, ensuring safe and efficient movement in diverse environments. These algorithms leverage environmental data to determine suitable paths and effectively avoid obstacles.

3.1 Shortest Path Search Algorithm

The shortest path search algorithm is a technique utilized to discover the most efficient route between two defined points within a graph. In the context of robotics, this algorithm is instrumental in determining the quickest path for a robot to reach a specified destination while avoiding obstacles in its course. This algorithm plays a pivotal role in robotics by enabling robots to pinpoint the shortest and most efficient way to reach their goals. Its application proves especially valuable in planning a robot's route through diverse environments while adeptly navigating around obstacles.

Multiple variants of the shortest path search algorithm exist, including:

Dijkstra's Algorithm: Dijkstra's algorithm is a renowned and straightforward method used to identify the shortest path within a graph that features diverse weights on its nodes and edges (Fig. 5) [8]. It commences at the initial points, computes the shortest distance to all other points in the graph, and through successive iterations, selects the point with the smallest distance. This process continues until the destination point is reached.

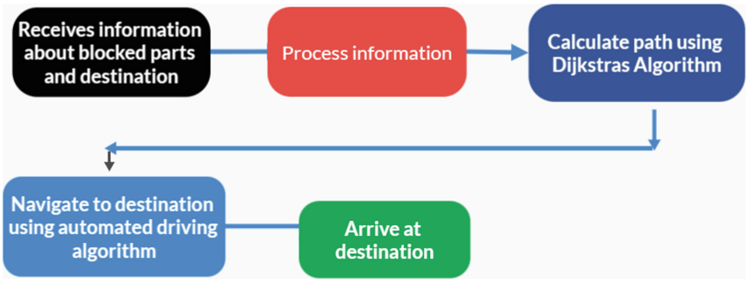
Receives information about blocked parts and destination
Process information
Calculate path using Dijkstras Algorithm
Navigate to destination using automated driving algorithm
Arrive at destination

Fig. 5. Dijkstra's algorithm [8].

Bellman-Ford Algorithm: It finds the shortest path in graphs with negative weights by iteratively updating distances between points. It initializes distances, then updates them by comparing new distances to the existing ones. It can detect negative cycles and is used in robotics for path planning and obstacle avoidance.

A* Algorithm: This algorithm combines graph search with heuristics to find the shortest path. It maintains open and closed lists of visited points and selects points based on a function combining distance and a heuristic estimate. A* is valuable in robotics for efficient path planning in complex environments [9].

Rapidly-Exploring Random Tree (RERT) Algorithm: RERT explores a robot's configuration space by building a spanning tree through random point selection and expansion. It efficiently plans paths in complex spaces, making it suitable for various robotic applications.

Pure Pursuit Algorithm: Pure Pursuit guides autonomous robots along planned paths by determining waypoints, calculating angles, linear velocity, and angular velocity, and implementing control commands. It adapts to changes in speed and orientation [10].

Algorithm with LIDAR: LIDAR-based algorithms use LIDAR sensor data for tasks like object segmentation, path identification, determining target points, and integrating with the Pure Pursuit algorithm for path following. They enable robots to adapt to changing environments, including obstacle detection [11].

Probabilistic Roadmaps (PRM): PRM is used in robotics for motion planning in obstacle-filled spaces. It samples robot configurations, checks for collisions, connects collision-free configurations, and constructs a roadmap of potential paths through the environment while avoiding obstacles.

4 Simulations of Mobile Robot and Discussions

In this section we will present simulations of the movement of mobile robots by comparing the algorithms described above [12, 13].

In the first case, a comparison was made of the efficiency of the algorithms: RERT and LIDAR for following the route and target points while avoiding various obstacles.

The RERT algorithm is a tree-based algorithm that explores space.

This algorithm helps in finding the optimal path to reach the target point efficiently. By using the RERT algorithm, the robot can find the shortest and most optimized path to reach the target point by avoiding obstacles and using only the available information.

The LIDAR algorithm is a technology to detect obstacles and help track the road. With the help of LIDAR, the robot can detect objects and obstacles in its surroundings and react efficiently to avoid them. The use of LIDAR can provide a more detailed view of the surrounding environment and help track the route and target points with greater precision.

After testing and analyzing both algorithms under different conditions, the results showed that the RERT algorithm was more efficient and had better results in tracking the path and target points. This algorithm provided a more stable, faster, and more accurate tracking, minimizing the distance traveled and the time required to reach the target point. However, it has been estimated that the use of LIDAR as an additional tool can improve the results and increase the level of safety in following the road, detecting objects and obstacles efficiently.

According to the performance analysis and evaluation, the RERT algorithm was more efficient and provided better results in tracking the path and target points.

The results achieved and comparisons in the following Figs. 6 and 7.

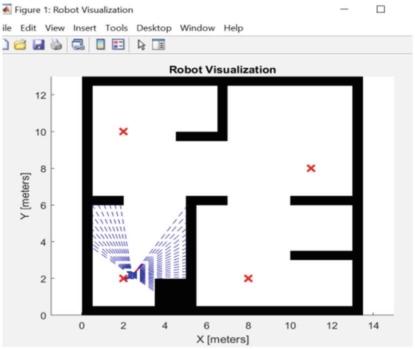
Figure 1: Robot Visualization
ile Edit View Insert Tools Desktop Window Help
Robot Visualization
Y [meters]
0
2
4
6
8
10
12
X [meters]
0
2
4
6
8
10
12
14

Fig. 6. Robot movement with LIDAR algorithm.

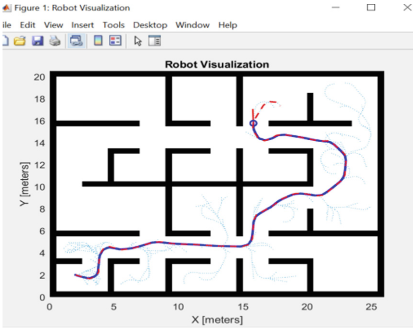
Figure 1: Robot Visualization
ile Edit View Insert Tools Desktop Window Help
Robot Visualization
Y [meters]
0
2
4
6
8
10
12
14
16
18
20
X [meters]
0
5
10
15
20
25

Fig. 7. Robot movement with RERT algorithm.

In the second case, a comparison was made of the efficiency of the algorithms: PP and MPC for following the path and target points. The PP algorithm is a method that follows the path using a target point as a target. This algorithm uses a defined trajectory and finds the closest position of the robot relative to this trajectory. After that, use a strategy to move towards the target point using direct control. The PP algorithm is relatively simple to implement and requires a limited responsibility in data processing.

MPC is an algorithm that aims to find an optimized trajectory given a system model and a set of defined rules. This algorithm makes continuous predictions of the system's behavior in a future period and makes control decisions in an optimized way to follow the intended path. MPC requires an accurate system model and continuous computation to generate control decisions.

After testing and analyzing both algorithms under different conditions, the results showed some fundamental differences. The PP algorithm provided good performance on the directed path to the target point and was simple to implement. However, it can have problems when disruptions occur in the road, such as obstacles or unforeseen interruptions, requiring a quick adaptation to follow the new path.

On the other hand, MPC showed a good performance in complicated trajectories and unpredictable situations. Through the system model and continuous predictions, the MPC can adapt and determine control decisions in an optimized way to reach the target point. Although MPC requires an accurate system model and continuous computation, it offers a high level of performance and precision in path tracking.

According to the performance analysis and evaluation, the PP offered a simple and efficient solution for tracking the route and target points, while the MPC offered an advanced performance suitable for complex trajectories and unforeseen situations. The choice of the most appropriate algorithm will depend on the specific system requirements and application conditions.

The results achieved and comparisons in the following Figs. 8 and 9:

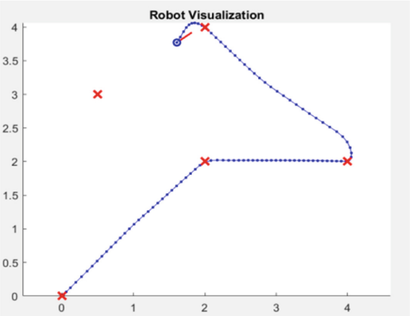

Fig. 8. Robot movement with MPC algorithm.

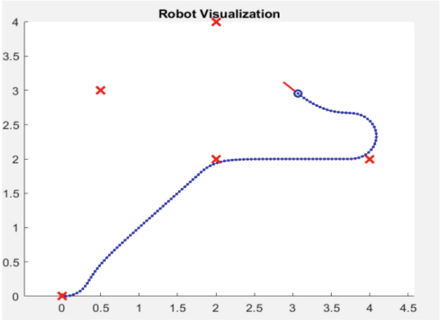

Fig. 9. Robot movement with PP algorithm.

5 Conclusions and Future Work

In the realm of mobile robot control, a multitude of algorithms and models stands ready to serve diverse purposes, from tracking objects to evading obstacles. The effectiveness of these solutions hinges on environmental conditions, path intricacies, tracking precision, robot constraints, and computational speed. Furthermore, the selection of sensors, be it LIDAR, cameras, or GPS, and the data they provide, including size, direction, and distance, play pivotal roles in determining the suitability and performance of these algorithms and models. Choosing the right algorithm depends on the robot's intended objective. For instance, the Pure Pursuit (PP) algorithm suits trajectory tracking, while the A* algorithm excels at navigating obstacle-laden terrain. Evaluating these options through rigorous testing enables the identification of strengths and weaknesses, driving enhancements and superior solutions for mobile robotics. Future decisions regarding mobile robot control should factor in application specifics, environmental nuances, available resources, and defined objectives. Meticulous evaluation and results-driven testing pave the way to selecting the most effective combination of algorithms and models to optimize mobile robot performance.

References

1. Borenstein, J., Everett, H.R., Feng, L., Wehe, D.: Mobile robot positioning: sensors and techniques. J. Robot. Syst., Special Issue on Mobile Robots **14**(4), 231–249 (1997)
2. Noreen, I., Khan, A., Habib, Z.: Optimal path planning using RRT based approaches a survey and future directions. Int. J. Adv. Comput. Sci. Appl. **7**(11) (2016)
3. Maulana, M.J.S., Riky, D.P., Wahyu, C.: Artificial potential field path planning algorithm in differential drive mobile robot platform for dynamic environment. Int. J. Robot. Control Syst. **3**(2), 161–170 (2023)
4. https://github.com/ivmech/ivPID. Accessed on 11 Sept 2023
5. https://towardsdatascience.com/ai-for-industrial-process-control-intro-to-control-strategies-part-1-b855fb6df61d. Accessed on 11 Sept 2023
6. Agrawal, M.: What is model predictive control? Technical Article, Control Automation, August 10 (2020). Accessed on 11 Sept 2023
7. https://www.tutorialspoint.com/fuzzy_logic/fuzzy_logic_control_system.htm. Accessed on 11 Sept 2023
8. Dijkstra's shortest path algorithm - a detailed and visual introduction. https://www.freecodecamp.org/news/dijkstras-shortest-path-algorithm-visual-introduction/. Accessed on 11 Sept 2023
9. Introduction to A*. http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html. Accessed on 11 Sept 2023
10. Introduction to pure pursuit tracking algorithm. https://vinesmsuic.github.io/robotics-purepersuit/. Accessed on 11 Sept 2023
11. Raber, R.B.: Lidar guidebook: concepts, project design, and practical applications. https://www.urisa.org/clientuploads/directory/Documents/Books%20and%20Quick%20Study/lidar_guidebook.pdf. Accessed on 11 Sept 2023
12. Conduraru, I., Doroftei, I., Conduraru, A.: Localization methods for mobile robots - a review. Adv. Mater. Res. (2014)
13. Matlab & Simulink: https://www.mathworks.com/products/simulink.html. Accessed on 11 Sept 2023