A hybrid motion planning framework for three-wheeled mobile robots based on improved A* and nonlinear model predictive control

Quang-Duy Do Nguyen[1,2], Xuan-Minh Dinh[3,4], Phuc-Lam Dang[5], Thanh-Loan Pham[2], Thanh-Trung Duong[2], Xuan-Hai Le[6]*

[1]*Hanoi University of Industry, Hanoi, Vietnam*
[2]*Hanoi University of Mining and Geology, Hanoi, Vietnam*
[3]*Faculty of Mechanical Engineering and Mechatronics, Phenikaa University, Duong Noi, Hanoi 12116, Vietnam*
[4]*HTI UAS Joint Stock Company, Hanoi, Vietnam*
[5]*Hanoi University of Science and Technology, Hanoi, Vietnam*
[6]*International School, Vietnam National University, Hanoi, Xuan Thuy Campus: Buildings E5 and G7, 144 Xuan Thuy, Cau Giay, Hanoi, Vietnam*
**Corresponding author E-mail: hailx@vnu.edu.vn*
DOI: https://doi.org/10.64032/mca.v30i2.414

Abstract

This paper proposes a hybrid motion planning framework for three-wheeled mobile robots (3WMRs), integrating an improved A-star (A*) algorithm for global planning with Nonlinear Model Predictive Control (NMPC) for local planning. The improved A* algorithm is designed with a two-stage adaptive heuristic strategy that combines Manhattan and Octile distances. This approach significantly reduces the number of expanded nodes during the search process while mitigating the tendency of trajectories to graze obstacle boundaries through a safety buffer expansion mechanism within the grid map construction. The resulting global trajectory is smoothed using Cubic Spline interpolation and serves as the reference trajectory for the NMPC controller at the local planning layer. This ensures precise trajectory tracking, smooth motion, and strict adherence to kinematic constraints, non-holonomic constraints, workspace limits, and the robot's control limits. Furthermore, the paper proposes a real-time replanning mechanism based on the improved A* algorithm to handle sudden obstacles in dynamic environments. The effectiveness of the proposed method is validated through simulation scenarios in MATLAB R2023b across both static and dynamic environments, including quantitative comparisons with classical algorithms such as Dijkstra and conventional A*. The results demonstrate that the proposed method substantially decreases the number of expanded nodes while ensuring reliable collision avoidance and high tracking accuracy. These findings confirm the efficiency, robustness, and practical feasibility of the proposed hybrid motion planning framework for mobile robots in complex environments.

Keywords*: A* Algorithm; Nonlinear Model Predictive Control; Three-wheeled Mobile Robots; Motion Planning; Cubic Spline.*

Abbreviations

3WMRs	Three-Wheeled Mobile Robots
A*	A-star
RRT	Rapidly Exploring Random Tree
MPC	Model Predictive Control
NMPC	Nonlinear Model Predictive Control
CBFs	Control Barrier Functions
TEB	Timed Elastic Band
FLC	Fuzzy Logic Control
HA	Hedge Algebra
RK4	Fourth-order Runge-Kutta
NLP	Nonlinear Programming
OCP	Optimal Control Problem

1. Introduction

In recent years, mobile robots have played an increasingly vital role in various sectors, including industrial automation, intelligent services, and defense and security. Due to their simple design and cost-effectiveness while maintaining high maneuverability, 3WMRs have become a preferred platform for researching and testing motion planning and control algorithms. To ensure the efficient operation of 3WMRs in real-world environments, the development of intelligent path planning methods is essential. These methods must guarantee reliable collision avoidance, path optimality, smooth motion, and strict adherence to constraints.

To date, a wide range of motion planning methods has been investigated and proposed, as documented in [1-17]. These methods are generally classified into two main categories: global planning and local planning. Global planning methods are exemplified by graph-based search algorithms such as Dijkstra [1] and A* [2], as well as sampling-based methods like RRT* [3], along with their enhanced variants, including A* with adaptive heuristic strategies [4] and A* with multi-neighbor search mechanisms [5]. These global planners effectively leverage prior knowledge of the environment map to find feasible, collision-free paths while optimizing criteria such as path length or the number of expanded nodes during the search process. In addition to graph-based approaches, sampling-based methods like RRT-the precursor to RRT*-are widely utilized due to their simple architecture and high computational speed. However, RRT often suffers from low planning efficiency, high stochasticity, and poor path quality [6]. Furthermore, since RRT prioritizes space exploration over optimality, the path cost does not necessarily improve as the number of samples increases.

Although global planning methods exhibit high efficiency in pathfinding within known maps, most of them fail to fully account for kinematic characteristics, physical constraints, non-holonomic constraints, and the robot's workspace limitations. Furthermore, conventional global planners

Received: 3 February 2026; Revised: 18 March 2026; Accepted: 2 April 2026.

typically assume a static and predefined environment, leading to limited adaptability when encountering sudden obstacles or local environmental changes. To improve obstacle avoidance, several studies have proposed integrating replanning mechanisms based on environment map updates and the re-execution of global planning algorithms. For instance, in [7], an optimal collision avoidance algorithm based on an improved Dijkstra was proposed, where the robot continuously monitors for obstacles during motion, updates the graph upon detecting new obstacles, and subsequently re-executes the planning algorithm. However, this approach requires a full re-execution of the algorithm, which increases computational latency and compromises the real-time responsiveness of the system due to Dijkstra's inherent limitations regarding the number of expanded nodes. This necessitates more efficient trajectory replanning strategies that maintain feasibility for practical implementation.

In contrast to global planning, local planning methods focus on generating trajectories and control signals over a short horizon, based on real-time sensor data and the current state of the robot. Among these, methods rooted in optimal control, notably MPC and its nonlinear variants such as NMPC [8], have garnered significant attention. This is due to their ability to directly incorporate nonlinear kinematics, nonholonomic constraints, actuator saturation or control limits, and obstacle avoidance requirements into the robot's mathematical model. NMPC enables the prediction of robot behavior over a finite time horizon while optimizing a cost function to ensure smooth, stable, and safe motion. Given its capacity to handle multiple simultaneous constraints, NMPC is particularly suited for robotic platforms with complex kinematics or those requiring high precision. However, the performance of NMPC is heavily dependent on the design of the cost function and the initial reference trajectory. In complex, cluttered environments or in the absence of clear global guidance, NMPC may converge to locally optimal trajectories that fail to achieve global optimality in terms of path length or travel time. Nevertheless, numerous experimental studies, such as [9], have demonstrated the feasibility and effectiveness of NMPC in real-world mobile robot applications. In recent years, the integration of NMPC with CBFs [10] has emerged as a robust approach to enhance the safety guarantees of mobile robots in dynamic and uncertain environments. CBFs provide a rigorous mathematical framework to enforce safety constraints as involvement inequalities, ensuring that the system states consistently remain within a predefined safe set, such as maintaining a minimum distance from obstacles or avoiding forbidden zones. This approach addresses a critical limitation of conventional NMPC, where safety is often only indirectly managed via penalty functions in the cost function or Euclidean distance-based constraints. Such indirect methods often struggle to balance target tracking, motion smoothness, and collision avoidance. Despite these advantages, NMPC–CBFs faces certain challenges. Incorporating CBF constraints into the optimization problem increases computational complexity, particularly in environments with numerous dynamic obstacles or a high count of CBFs. Furthermore, the efficacy of this method relies heavily on the selection of appropriate CBF candidates, their associated parameters, and the ability to solve the optimization problem in real-time. Consequently, designing efficient NMPC–CBFs architectures that reduce computational overhead while ensuring practical deployability remains a highly active and significant research direction.

Beyond standalone global or local planning methods, hybrid approaches that integrate both layers have been widely adopted in mobile robotic systems to leverage their respective advantages. Specifically, studies [11–13] proposed combining global search algorithms, such as A* and Dijkstra, with the TEB method, forming A*–TEB and Dijkstra–TEB planning frameworks. In these frameworks, A* or Dijkstra generates a global reference trajectory based on the environment map, while TEB serves as the local planner to refine the trajectory for smoothness, time-optimality, and obstacle avoidance. Although TEB [14] is capable of generating trajectories consistent with the robot's motion characteristics and is effective in many practical scenarios, its output quality can sometimes be unstable and may violate boundary conditions or kinematic requirements in complex situations. To address this, research [15] proposed integrating TEB with FLC, resulting in the FLC–TEB method. This approach incorporates additional objectives related to motion smoothness and jerk during trajectory optimization to yield smoother paths and more stable control. Experimental results indicate that while FLC–TEB slightly increases the total travel time, it provides more stable, smoother, and shorter trajectories compared to the classical TEB. Furthermore, study [16] applied HA theory to robot motion control. Simulation results demonstrated higher control efficiency and faster computational speeds compared to conventional fuzzy set theory-based controllers for autonomous navigation. Despite these significant improvements in trajectory quality, methods based on fuzzy logic and hedge algebra are inherently limited by their heavy reliance on expert knowledge for constructing and fine-tuning fuzzy rule sets. This poses challenges for scalability and application in complex or rapidly changing environments.

Consequently, model-based optimization methods, which enable the automated handling of constraints and trajectory quality objectives within a unified mathematical framework, are garnering increasing attention in hybrid motion control systems. Following this approach, study [17] proposed an A*–NMPC framework where the global trajectory generated by A* serves as the reference for the NMPC controller. The results indicated that NMPC significantly improved tracking performance compared to TEB. However, the NMPC in [17] did not directly integrate obstacle avoidance mechanisms into the optimization problem, nor did it consider global trajectory replanning strategies for dynamic environments. As a result, the system's adaptability remains limited when encountering sudden obstacles or local environmental changes. This underscores the need for hybrid planning frameworks that more tightly integrate global guidance, local optimization, and real-time safety guarantees. Furthermore, since conventional A* primarily optimizes for path length on discrete grid maps, the resulting global trajectories often tend to graze the boundaries of obstacles, posing potential collision risks in practical scenarios [18]. In contrast, improved A* algorithms utilizing adaptive heuristic strategies significantly reduce the number of expanded nodes during the search process. This efficiency enables the integration of replanning mechanisms to update the path upon detecting new obstacles

while maintaining computational efficiency for real-world applications.

Based on the aforementioned analysis, it is evident that an effective motion planning framework for mobile robots must simultaneously satisfy three primary requirements: (i) providing reliable global guidance based on environment map information; (ii) ensuring kinematically feasible, safe, and smooth local trajectories during execution; and (iii) possessing the capacity for flexible adaptation when the environment undergoes changes or sudden obstacles appear. The studies analyzed above typically address only one or two of these requirements effectively, while remaining limited in terms of efficient replanning and real-time safety guarantees. For instance, while the A*–NMPC framework [17] shows potential in integrating global planning and local control, the use of conventional A* can result in global trajectories that graze obstacle boundaries. Meanwhile, the NMPC layer remains heavily dependent on the reference trajectory and lacks tight integration with obstacle avoidance or replanning mechanisms. These shortcomings diminish the system's adaptability when deployed in real-world environments.

Against this background, this paper proposes a hybrid motion planning framework for 3WMRs, integrating an improved A* algorithm for global planning with NMPC for the local control layer. Specifically, the improved A* algorithm employs a two-stage heuristic strategy-combining Manhattan and Octile distances-which significantly reduces the number of expanded nodes during the search process. Simultaneously, it mitigates the tendency of trajectories to graze obstacle boundaries by incorporating a safety buffer expansion mechanism during the grid map construction. The resulting global trajectory is smoothed via Cubic Spline interpolation and serves as the reference trajectory for the NMPC controller at the local planning layer. This ensures precise trajectory tracking, smooth motion, and strict compliance with kinematic, non-holonomic, workspace, and robot's control limits. Furthermore, the paper introduces a real-time trajectory replanning mechanism based on the improved A* algorithm to handle sudden obstacles within the operating environment. The effectiveness of the proposed framework is validated through simulation scenarios in MATLAB R2023b across both static and dynamic environments, including quantitative comparisons with classical algorithms such as Dijkstra and conventional A*. The results demonstrate that the proposed method substantially decreases the number of expanded nodes while ensuring reliable collision avoidance and high tracking accuracy. These findings confirm the efficiency, robustness, and practical feasibility of the proposed hybrid motion planning framework for mobile robots in complex environments.

The remainder of this paper is organized as follows: Section 2 presents the mathematical modeling and the research objectives. Section 3 provides a detailed description of the proposed hybrid motion planning framework, including the grid map construction, the improved A* algorithm for global planning, and the NMPC for local planning, as well as the real-time trajectory replanning mechanism for handling sudden obstacles. Section 4 presents the simulation scenarios and a detailed analysis of the results. Finally, Section 5 provides the concluding remarks and suggests potential directions for future work.

2. Mathematical modeling and research objectives

The nonlinear mathematical model of the 3WMR moving on the OXY plane is illustrated in Fig. 1, consisting of two independent rear driving wheels powered by DC motors and a front castor wheel. Here, r denotes the driving wheel radius; v_L (m/s) and v_R (m/s) represent the linear velocities of the left and right wheels, respectively; ω_L (rad/s) and ω_R (rad/s) are the corresponding angular velocities; and $2R$ is the distance between the two rear driving wheels. To comprehensively describe the motion of the 3WMR, this study simultaneously employs two reference frames: the global coordinate system OXY and the local (body-fixed) coordinate system CX_CY_C . The global frame OXY serves as a fixed reference to represent the absolute position of the robot on the map. In this frame, the state of the 3WMR is defined by the vector $\chi = [x, y, \theta]^T \in \mathbb{R}^{3\times1}$, where (x, y) denotes the robot's position coordinates in meters (m), and θ represents the robot's orientation angle in radians (rad) relative to the OX axis. The local coordinate system CX_CY_C is rigidly attached to the robot body to define input control variables and establish internal kinematic constraints, ensuring consistency with the physical characteristics of the system. In this setup, the CY_C axis lies along the lateral line connecting the two driving wheels, while the CX_C axis coincides with the robot's heading direction. In this research, the center of mass C is designed to coincide with the midpoint P of the rear axle, resulting in a zero offset distance ($d = 0$). Furthermore, the control input is described by the vector $u = [v, \omega]^T \in \mathbb{R}^{2\times1}$, where v (m/s) is the linear velocity and ω (rad/s) is the angular velocity of the robot.

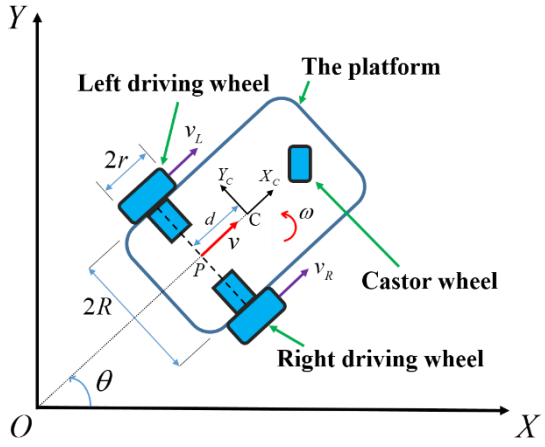

Figure 1: Mathematical model of the three-wheeled mobile robot [10].

Based on the geometric relationships, the kinematic equations of the 3WMR under ideal operating conditions are established as follows [19]:

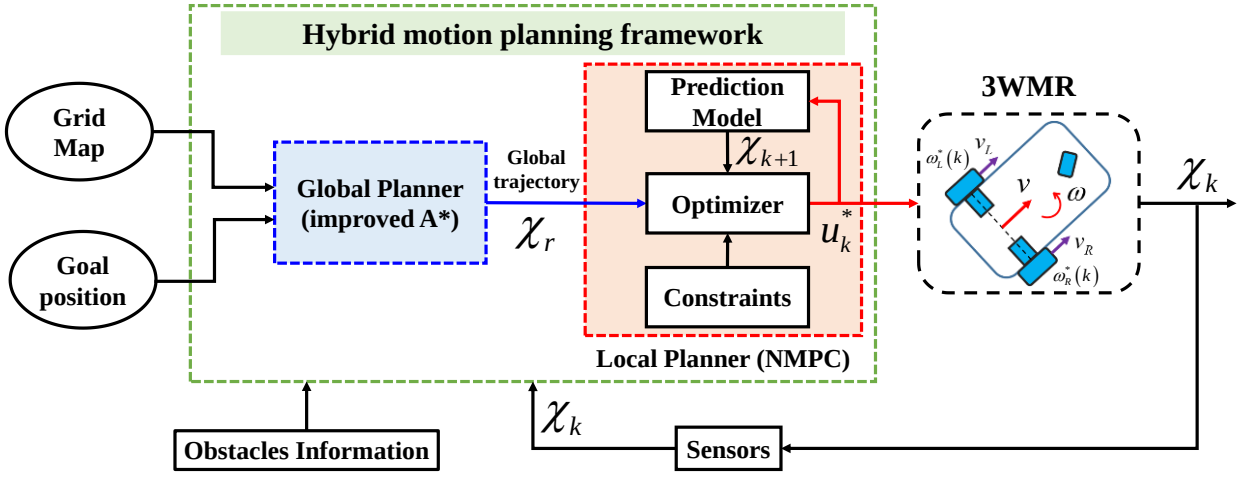

Figure 2: Proposed hybrid motion planning framework.

$$\dot{\chi} = f(\chi,u) = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ \omega\end{bmatrix} = \begin{bmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\frac{v_L+v_R}{2}\\ \frac{v_R-v_L}{2R}\end{bmatrix} = \begin{bmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\frac{r}{2} & \frac{r}{2}\\ \frac{r}{2R} & -\frac{r}{2R}\end{bmatrix}\begin{bmatrix}\omega_R\\ \omega_L\end{bmatrix} \quad (1)$$

The non-holonomic constraint equation is defined as follows [10]:

$$\dot{x}\sin(\theta) - \dot{y}\cos(\theta) = 0 \quad (2)$$

Remark 1: Based on the kinematic model (1), the relationship between the control inputs, the velocities in the local coordinate system, and the state variations of the 3WMR in the global coordinate system is explicitly established. Due to the torque and power limitations of the DC driving motors, the angular velocities ω_L and ω_R are always bounded within a specific range. This leads to saturated control input constraints for the robot: $|v| \le v_{\max}$ and $|\omega| \le \omega_{\max}$, where $v_{\max}$ and $\omega_{\max}$ denote the maximum allowable speeds. Furthermore, according to the non-holonomic constraint (2), the robot is unable to achieve instantaneous lateral motion along the CY_C axis. In addition, the robot's physical workspace is limited by the operational area and the presence of environmental obstacles. Therefore, in this study, our objective is to establish a hybrid motion planning framework for 3WMRs, enabling the robot to navigate safely and efficiently in obstacle-cluttered environments while satisfying the kinematic constraints (1), the non-holonomic constraints (2), control input constraints, and the robot's physical workspace limitations.

3. Proposed hybrid motion planning framework

Fig. 2 illustrates the proposed hybrid motion planning framework for 3WMRs, which tightly integrates global trajectory planning with local optimal control to ensure feasibility, safety, and efficiency in the operating environment. Specifically, the environment map in the form of a Grid Map, along with the Goal position, is fed into the global planner. Based on this information, the global planner utilizes an improved A* algorithm to search for a feasible, collision-free global reference trajectory that guides the robot from its initial position to the goal. The enhancement of the improved A* algorithm using an adaptive heuristic strategy enables the search for optimal and safe paths while reducing the number of redundant expanded nodes, thereby creating a better-oriented global trajectory in complex environments. The global trajectory, smoothed using the Cubic Spline technique [21], serves as the reference input for the NMPC-based local planner.

At the local level, at each discrete time step k, the process of generating optimal control signals $u_k^* = \left[v_k^*, \omega_k^*\right]^T$ to track the global reference trajectory χ_r is performed by the NMPC. Specifically, the optimizer solves the optimization problem (12) at every time step k, utilizing a prediction model discretized by the fourth-order Runge-Kutta (RK4) method (10) to predict the robot's future state χ_{k+1}. This optimization process continuously calculates an optimal control sequence at each iteration to minimize the objective function over a finite prediction horizon N, while strictly satisfying constraints such as non-holonomic kinematics, system control limits, and the robot's physical workspace boundaries. According to the receding horizon control principle, only the first optimal control value in the sequence, u_k^* (14), is extracted and applied directly to the robot. For practical hardware implementation, these high-level optimal control signals can be converted into the optimal angular velocities of the two driving wheels, including the left wheel $\omega_L^*(k)$ and the right wheel $\omega_R^*(k)$ (15), through the kinematic model (1). The actual state of the robot $\chi = \left[x, y, \theta\right]^T$ is collected via sensors and subsequently fed back to the hybrid motion planning framework. This feedback, combined with obstacle information, allows the local planner to continuously update the optimization problem based on the robot's current state, while also providing the basis for global path replanning when the environment changes or new obstacles appear.

Under normal operating conditions, the hybrid planning process occurs sequentially, where the global trajectory is constructed first and subsequently tracked by the NMPC at the local level. However, upon the emergence of unexpected obstacles, the two planning layers operate in parallel: the NMPC maintains real-time control based on current information, while the global planner executes path replanning based on the updated map. Once a new trajectory is generated, it replaces the previous one and is fed into the NMPC for subsequent cycles. By integrating global guidance from the improved A* and local optimization via NMPC, the proposed framework leverages the advantages of both layers. It ensures long-term optimality and orientation of the trajectory while maintaining precise tracking capability, smooth motion, and strict adherence to constraints during practical operation.

3.1 Grid map construction

In this study, the workspace of the three-wheeled mobile robot is modeled as a two-dimensional grid map on the OXY plane, as illustrated in Fig. 3. The grid-based method is widely adopted due to its intuitive representation, simplicity, and ease of implementation. The workspace is discretized into square grid cells of a fixed size, where each cell represents a small region in the physical environment. Utilizing a grid map provides a visual representation of the environment while establishing a favorable foundation for applying graph-based global path planning algorithms. Static obstacles within the environment are modeled as clusters of occupied grid cells, represented by dark gray regions. Unlike conventional grid maps, the proposed method incorporates a safety buffer surrounding each obstacle, represented by light gray regions, to ensure operational safety in real-world environments. This safety buffer is established to compensate for practical factors, including the robot's geometric dimensions, sensor noise, and control uncertainties. Consequently, the resulting global trajectory tends to be safer, avoiding grazing obstacle edges, and is more suitable for refinement and trajectory tracking at the local level using the NMPC controller.

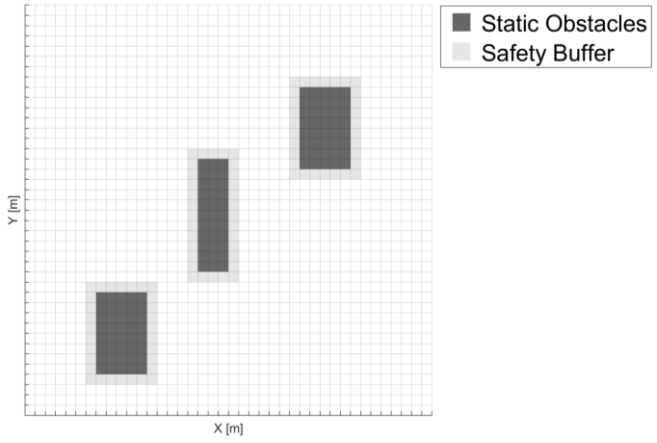

Figure 3: Grid map construction in the proposed method.

3.2 Global planning based on the improved A* algorithm

A* is a graph-based path planning algorithm widely utilized in global path planning due to its computational speed, high efficiency, and stability. To date, A* is considered one of the most classical and effective path planning algorithms [20]. However, a significant drawback of A* is the requirement to store and access a vast number of redundant nodes during the search process. As the map size increases, this leads to substantial waste in memory and computational time, thereby diminishing the pathfinding efficiency. Consequently, in this study, we propose an improved A* algorithm featuring an adaptive heuristic function strategy. This strategy aims to enhance search efficiency, ensure path optimality and safety, and simultaneously reduce the number of expanded nodes.

The evaluation function in the conventional A* algorithm is established as follows [20]:

$$f(n) = g(n) + h(n) \tag{3}$$

Where: $g(n)$ represents the actual cost accumulated from the start node to the current node n, $h(n)$ is a heuristic function representing the estimated cost from the current node n to the target node, and $f(n)$ is the evaluation function representing the total estimated cost from the start node to the target node through node n.

The heuristic function plays a crucial role in guiding the search process and directly influences the algorithm's efficiency. An unsuitable heuristic function can lead to the expansion of too many redundant nodes or degrade the optimality of the resulting path. Commonly used heuristic functions include Manhattan distance, Euclidean distance, and Octile distance. In this study, we define (x_n, y_n) as the coordinates of the current node n, and (x_G, y_G) as the coordinates of the target node or goal node.

Manhattan distance is calculated as follows [20]:

$$h_M(n) = |x_G - x_n| + |y_G - y_n| \tag{4}$$

Where: $h_M(n)$ is the heuristic function using Manhattan distance.

Euclidean distance is calculated as follows [20]:

$$h_E(n) = \sqrt{(x_G - x_n)^2 + (y_G - y_n)^2} \tag{5}$$

Where: $h_E(n)$ is the heuristic function using Euclidean distance.

Octile distance is calculated as follows [20]:

$$h_O(n) = \left(|x_G - x_n| + |y_G - y_n|\right) + \left(\sqrt{2} - 2\right)\min\begin{pmatrix}|x_G - x_n|,\\ |y_G - y_n|\end{pmatrix} \tag{6}$$

Where: $h_O(n)$ is the heuristic function using Octile distance.

Manhattan distance is particularly effective during the initial phase of the search process in grid-based environments, as it facilitates rapid node expansion and covers a broader search space. By prioritizing fast exploration, Manhattan distance helps mitigate the risk of the algorithm becoming trapped in local minima [4]. Meanwhile, Octile distance combines the characteristics of both Euclidean and Manhattan distances, allowing for a more accurate estimation of the actual travel cost, thereby enhancing search efficiency [20]. Consequently, Octile distance contributes to reducing the number of

expanded nodes and improving trajectory quality, especially during the path refinement stage near the goal.

Based on the aforementioned analysis, this study proposes a two-stage heuristic strategy that integrates Manhattan and Octile distances to leverage the advantages of both functions. When the current node is far from the target, the heuristic function employs Manhattan distance to prioritize rapid exploration and extensive search space coverage, enabling the algorithm to quickly identify feasible routes. In contrast, the heuristic function transitions to the Octile form during the final stage to more accurately approximate the true travel cost, thus reducing redundant nodes. The Manhattan–Octile two-stage heuristic strategy allows the improved A* algorithm to maintain high search speeds in the early phase while ensuring precision and trajectory quality in the later phase. As a result, the number of expanded nodes is significantly reduced compared to conventional A*.

Mathematically, our adaptive heuristic function is designed as follows:

$$h_A(n)=\begin{bmatrix} h_M(n) & if \quad h_E(n)>\delta \\ h_O(n) & if \quad h_E(n)\le\delta \end{bmatrix} \tag{7}$$

Where: δ is the distance threshold for switching between heuristic functions, and $h_A(n)$ is the adaptive heuristic function.

In this research, the cost function of the A* algorithm is extended to simultaneously consider trajectory length, movement direction, and safety level relative to obstacles. Specifically, the proposed evaluation function at each node is defined as:

$$f(n)=g(n)+\alpha h_A(n)-\beta\cos(\theta_n)+C(n) \tag{8}$$

Where: $g(n)$ represents the total actual cost accumulated from the start node to the current node n, $h_A(n)$ is the adaptive heuristic function, $\cos(\theta_n)$ reflects the alignment of the movement direction relative to the goal, and $C(n)$ is the risk cost formulated based on the distance to obstacles. Additionally, α and β are tuning parameters used to balance the influence of the heuristic and orientation components within the cost function. Consequently, the proposed evaluation function not only prioritizes shorter trajectories but also encourages smooth motion and maintains a safe distance from obstacles during the path planning process.

The risk cost function is defined as follows:

$$C(n)=\begin{bmatrix} \dfrac{\eta}{\min\limits_{o\in\Omega_o}\sqrt{(x_o-x_n)^2+(y_o-y_n)^2}+\varepsilon} & if \quad n\notin\Omega_o \\ \infty \quad if \quad n\in\Omega_o \end{bmatrix} \tag{9}$$

Where: (x_n,y_n) are the coordinates of the current node n, (x_o,y_o) are the coordinates of the obstacle, Ω_o denotes the set of obstacles, η is the risk constant, and ε is a small constant to prevent division by zero.

Remark 2: Although the improved A* algorithm enables the construction of a safe and optimal global trajectory on the grid map, the obtained trajectory remains discrete in nature and does not directly provide the robot's heading angle information. Therefore, in this study, we apply the Cubic Spline interpolation technique [21] to convert the sequence of discrete waypoints into a continuous, smooth trajectory. This method simultaneously allows for the derivation of continuous orientation angle states along the path. This approach generates a more feasible reference trajectory for the local planning stage.

3.3 Local planning based on NMPC and trajectory replanning mechanism

Once the global reference trajectory $\chi_r=\begin{Bmatrix}\chi_{r_1},\chi_{r_2},\\ ...,\chi_{r_M}\end{Bmatrix}$, consisting of M discrete reference points, is generated by the improved A* algorithm and smoothed via Cubic Spline interpolation, the local control layer utilizes NMPC to perform precise tracking. At each sampling period k, the generation of the control signal sequence $u=\begin{Bmatrix}u_k,u_{k+1},\\ ...,u_{k+N-1}\end{Bmatrix}$ essentially involves solving a Nonlinear Programming (NLP) problem over a finite prediction horizon N. Specifically, within the NMPC block, the Prediction Model employs the nonlinear kinematic model of the 3WMR, discretized using the RK4 method (10). Based on the actual state χ_k fed back from sensors, the Optimizer determines the control command sequence to minimize a cost function (12). This cost function is designed to balance primary objectives, including the elimination of tracking errors relative to the reference χ_r and the optimization of control effort to ensure smooth motion, thereby preventing abrupt velocity changes that could damage the mechanical system. Throughout the optimization process, the calculated control commands must strictly adhere to constraints, including non-holonomic kinematics, system control saturation, and the robot's physical workspace boundaries. After finding the optimal solution sequence, according to the receding horizon control principle, the system extracts only the first control element u_k^* (14) to be applied directly to the robot. For practical hardware implementation, these high-level optimal control signals are converted into optimal angular velocities for the two driving wheels, $\omega_L^*(k)$ and $\omega_R^*(k)$ (15), through the kinematic model (1). This entire procedure is repeated at each time step with continuously updated state feedback, allowing the system to track the trajectory flexibly and accurately while maintaining real-time adaptability to environmental fluctuations and the sudden appearance of obstacles.

First, the kinematic equation (1) is discretized using the RK4 method as follows [10]:

$$\chi_{k+1}=\chi_k+\frac{\Delta_T}{6}(R_1+2R_2+2R_3+R_4) \tag{10}$$

Where: $R_i\,(i=1,...,4)$ represents the slope estimates at different points within a sampling step, used to calculate the next state of the nonlinear system, and Δ_T is the sampling period.

The slope estimate values in (10) are determined as follows [17]:

$$\begin{cases} R_1 = f\left(\chi_k, u_k\right) \\ R_2 = f\left(\chi_k + \dfrac{\Delta_T}{2} R_1, u_k\right) \\ R_3 = f\left(\chi_k + \dfrac{\Delta_T}{2} R_2, u_k\right) \\ R_4 = f\left(\chi_k + \Delta_T R_3, u_k\right) \end{cases} \tag{11}$$

To solve the Optimal Control Problem (OCP), in this study, we apply the multiple shooting method [22] to convert the OCP into a Nonlinear Programming (NLP) problem. Accordingly, the optimization problem of the NMPC controller at each sampling period is formulated as follows:

$$\begin{aligned} \min_{\chi,u}\left(J_N\left(\chi_k, u_k\right)\right) &= \min_{\chi,u}\left(\begin{array}{l} \sum_{j=0}^{N-1}\left(\left\|\chi_{k+j} - \zeta_{k+j}\right\|_{\mathbf{Q}}^2 + \left\|u_{k+j}\right\|_{\mathbf{R}}^2\right) \\ + \left\|\chi_{k+N} - \zeta_{k+N}\right\|_{\mathbf{P}}^2 \end{array}\right) \\ &= \min_{\chi,u}\left(\begin{array}{l} \sum_{j=0}^{N-1}\left(\begin{array}{l} \left(\chi_{k+j} - \zeta_{k+j}\right)^T \mathbf{Q}\left(\chi_{k+j} - \zeta_{k+j}\right) \\ + \left(u_{k+j}\right)^T \mathbf{R}\left(u_{k+j}\right) \end{array}\right) \\ + \left(\chi_{k+N} - \zeta_{k+N}\right)^T \mathbf{P}\left(\chi_{k+N} - \zeta_{k+N}\right) \end{array}\right) \end{aligned} \tag{12}$$

Where: $\chi = \begin{Bmatrix} \chi_k, \chi_{k+1}, \\ ..., \chi_{k+N} \end{Bmatrix}$ and $u = \begin{Bmatrix} u_k, u_{k+1}, \\ ..., u_{k+N-1} \end{Bmatrix}$ are the predicted state and control sequences, respectively, over the prediction horizon N, χ_k is the robot's state at time k, u_k is the control signal at time k. $\zeta_{k+j} \in \chi_r$ is the local reference state at prediction step j, extracted from the global trajectory χ_r. χ_{k+N} and ζ_{k+N} represent the states at the end of the horizon (terminal states). The values $\mathbf{Q} \in \mathbb{R}^{3\times3}, \mathbf{R} \in \mathbb{R}^{2\times2}, \mathbf{P} \in \mathbb{R}^{3\times3}$ are positive definite weighting matrices, and N is the prediction horizon of the NMPC controller.

The equality and inequality constraints are established as follows:

$$\begin{cases} \chi_{k+j+1} = f\left(\chi_{k+j}, u_{k+j}\right); \forall j \in \left[0, N-1\right] \\ \chi_{\min} \le \chi_{k+j} \le \chi_{\max}; \forall j \in \left[0, N\right] \\ u_{\min} \le u_{k+j} \le u_{\max}; \forall j \in \left[0, N-1\right] \end{cases} \tag{13}$$

Where: $\chi_{\min}$ and $\chi_{\max}$ are the lower and upper bounds of the state variables, $u_{\min}$ and $u_{\max}$ are the lower and upper bounds of the control signals, respectively.

Similar to other studies on NMPC [8-10], the general solution to the optimal control problem in (12) subject to constraints (13) is an optimal control sequence of the following form:

$$u^* = \left\{u_k^*, u_{k+1}^*, ..., u_{k+N-1}^*\right\} \tag{14}$$

Regarding NMPC, according to the receding horizon control principle, only the first element $u_k^* = \left[v_k^*, \omega_k^*\right]^T$ is extracted to be applied for robot control. For practical hardware implementation, these high-level control commands must be converted into target angular velocities for each DC motor. Based on the kinematic model (1), the optimal setpoints for the angular velocities of the right and left wheels are determined as follows:

$$\begin{cases} \omega_R^*\left(k\right) = \dfrac{1}{r}\left(v_k^* + R\omega_k^*\right) \\ \omega_L^*\left(k\right) = \dfrac{1}{r}\left(v_k^* - R\omega_k^*\right) \end{cases} \tag{15}$$

Where: $\omega_L^*\left(k\right)$ and $\omega_R^*\left(k\right)$ are the optimal angular velocities of the left and right wheels at sampling time k, respectively. This conversion ensures consistency between the high-level control signals and the execution signals on the hardware. Furthermore, thanks to the constraints established in (13), the resulting angular velocities always remain within the safe operating limits of the DC motor's torque and power, consistent with the analysis in **Remark 1**.

In real-world environments, besides static obstacles already known and modeled in the global map, mobile robots frequently encounter sudden obstacles during operation. Unlike traditional NMPC approaches [8–10], where safety requirements are typically integrated directly as hard constraints in the optimization problem, this study applies a real-time trajectory replanning mechanism based on the improved A* algorithm. This approach simplifies the NMPC optimization problem while effectively leveraging the advantages of the improved A* algorithm in reducing the number of expanded nodes and ensuring global safety.

At each control sampling period Δ_T, the robot continuously monitors data from its sensors. Assuming the robot is equipped with a sensor having a range R_{sensor}, the condition to trigger trajectory replanning is when the distance from the robot to a sudden obstacle is less than or equal to the sensor's safety threshold:

$$\sqrt{\left(x_o - x_n\right)^2 + \left(y_o - y_n\right)^2} \le R_{sensor} \tag{16}$$

Where: $\left(x_o, y_o\right)$ are the coordinates of the obstacle, and $\left(x_n, y_n\right)$ are the coordinates of the current node n.

When condition (16) is satisfied, the improved A* algorithm is reactivated using the robot's current state as the new start node to replan a global trajectory suitable for the updated environment. The new trajectory is then smoothed via Cubic Spline and provided to the NMPC control layer as a new reference trajectory. Thanks to this mechanism, the system can adapt flexibly to local environmental changes while maintaining safety and control efficiency without significantly increasing the computational complexity of the NMPC problem.

4. Simulation results and analysis

In this study, the effectiveness of the proposed method is validated through simulations conducted using MATLAB R2023b. The hardware system employed for the experiments

is a personal computer equipped with an Intel® Core™ i7-11850H processor (8 cores, 2.50 GHz) and 32 GB of RAM. We established two primary experimental scenarios: (4.1) The robot moves in a static environment with known fixed obstacles, and (4.2) The robot operates in a dynamic environment with sudden obstacles, requiring trajectory replanning capabilities. To evaluate performance, the proposed method is compared with classical motion planning algorithms, specifically Dijkstra [1] and conventional A* [2]. For the NMPC local optimization method, we utilize the CasADi library [23] combined with the IPOPT solver [24] to search for optimal solutions that satisfy system constraints. The control parameters of the proposed method were selected using a trial-and-error approach and are listed in detail in **Table 1**. The constraints on control signals and the operational workspace are established as follows: $v \in [-0.5, 0.5]$ (m/s), $\omega \in \left[\frac{-\pi}{4}, \frac{\pi}{4}\right]$ (rad/s), $x \in [-2, 2]$ (m), and $y \in [-2, 2]$ (m). The sampling time for each control cycle is set to $\Delta_T = 0.1$ (s), the robot's starting point coordinates are $(x, y) = (-1.5, -1.5)$ (m), the goal coordinates are $(x_G, y_G) = (0.5, 1.0)$ (m). The total number of simulation steps is $I_M = 350$ (steps), and the grid resolution is 0.1 (m). The robot parameters are selected as follows [11]: $r = 0.07$ (m) and $R = 0.19$ (m).

Table 1: Control parameters of the proposed method

Control parameters	Value
Q	$diag([3500, 3500, 500])$
R	$diag([1.5, 1.5])$
P	$diag([3500, 3500, 1000])$
α	1.5
β	1.0
R_{sensor}	0.5
N	20
δ	0.5
η	0.3
ε	10^{-6}

Remark 3: The selection of control parameters is conducted using a trial-and-error approach combined with sensitivity analysis to balance trajectory tracking performance, safety, and real-time computational feasibility. Specifically, parameters α and β in the improved A* algorithm are tuned to balance the influence of heuristic and orientation components within the cost function (8). The parameter δ determines the distance threshold for switching between heuristic functions (7). Meanwhile, the constant η, in coordination with the sensing range R_{sensor}, ensures that the robot consistently maintains a safety buffer and executes timely replanning upon the emergence of unexpected obstacles. The constant ε is chosen to be sufficiently small to avoid division-by-zero singularities in the risk cost function (9). Furthermore, the weighting matrices $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{P}$ are established to prioritize position tracking accuracy while penalizing control effort, thereby reducing abrupt variations in control signals. This indirectly contributes to ensuring that the acceleration and torque limits of the propulsion system are respected. Finally, the prediction horizon N is selected to achieve a trade-off between control performance and real-time computational overhead. After fine-tuning, these parameters are kept constant across all simulation scenarios to ensure consistency during the comparative analysis.

In this study, three key criteria are employed to evaluate the effectiveness of motion planning methods: the number of expanded nodes, the total path length, and the capability to avoid sudden obstacles. Specifically, the number of expanded nodes serves as an indicator of the planning algorithm's computational cost. An efficient algorithm must achieve a balance between minimizing node expansion and ensuring the discovery of a feasible trajectory, particularly in complex environments populated with numerous obstacles. A minimized total path length indicates a shorter and more optimal trajectory. Finally, the capability to avoid sudden obstacles is assessed by observing the robot's ability to perform trajectory replanning and maintain safe motion when environmental changes occur during operation. In summary, these three criteria provide a comprehensive reflection of the algorithm's computational efficiency, path optimality, and operational safety.

The total path length is calculated as follows:

$$L = \sum_{n=1}^{M} \sqrt{(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2} \tag{17}$$

Where: L denotes the total path length, and M represents the total number of points along the trajectory.

The average computational time per step is calculated as follows:

$$T_{\text{average}} = \frac{1}{I_M} \sum_{k=1}^{I_M} (toc - tic)_k \tag{18}$$

Where: I_M is the total number of simulation steps, and $(toc - tic)_k$ represents the execution time of the solver at the k-th step. This interval is determined by the difference between the completion time (toc) and the start time (tic) of the algorithm at that specific step within the MATLAB environment.

In the simulation scenarios, the green circle denotes the robot's starting position, the green star indicates the goal position, and the red circle represents the robot itself.

4.1 Static obstacle scenario

The global trajectory planning results presented in Fig. 4 reveal a distinct disparity in trajectory quality among the algorithms. Classical methods such as Dijkstra (black dashed line) and conventional A* (pink dashed line) tend to find the mathematically shortest path, resulting in trajectories that closely hug the sharp corners of obstacles. Although optimal in terms of distance, these trajectories contain numerous jagged, orthogonal segments characteristic of a discrete grid. Consequently, they pose a high collision risk as they fail to account for the robot's physical dimensions, uncertainties, and tracking control errors under real-world operating conditions.

In contrast, the proposed improved A* algorithm comprehensively overcomes this drawback by integrating a Safety Buffer around obstacles. As a result, the global reference trajectory (red dashed line in Fig. 4) generated by the Global Planner not only maintains a necessary safe distance from obstacles but also exhibits higher smoothness, featuring soft rounded segments at turning points, and is directly oriented towards the goal rather than adhering to the discrete diagonals of the grid map.

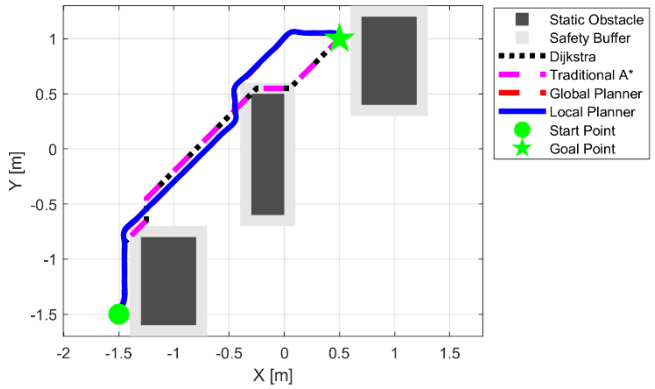

Figure 4: Motion trajectory planning results in the known static obstacle scenario.

To provide a quantitative analysis of global planning performance among the methods, **Table 2** details information regarding the number of expanded nodes and total path length. First, regarding the number of expanded nodes, a significant difference is observable. The Dijkstra algorithm requires expanding up to 1128 nodes, while the Conventional A* slightly reduces this to 1041 nodes by utilizing a heuristic function to guide the search toward the goal. Conversely, the improved A* algorithm in the Global Planner requires expanding only 35 nodes, representing a reduction of approximately 96.64% compared to Conventional A* and 96.90% compared to Dijkstra. This result demonstrates the superior efficiency of the adaptive heuristic strategy in minimizing the number of expanded nodes, serving as a crucial premise for trajectory replanning when encountering sudden obstacles. Regarding path length, both Dijkstra and Conventional A* generate trajectories with a length of 3.49 m, reflecting the general characteristic of classical pathfinding algorithms to optimize distance in a purely geometric sense on a grid map. However, these trajectories often graze obstacle boundaries and contain many sharp orthogonal turns, significantly reducing feasibility for real robots. Meanwhile, the Global Planner generates a trajectory with a length of 3.65 m, which is only about 4.6% longer than the other two methods. Although not absolutely optimal in length, this increase is intentional and justified to trade off for maintaining a safe distance from obstacles and creating a smoother, less jagged trajectory. Notably, this trajectory is more suitable for the Cubic Spline smoothing phase and the NMPC controller at the local layer, significantly reducing the requirement for sharp steering and the risk of violating kinematic constraints.

Table 2: Global trajectory planning performance in the known static obstacle scenario

Evaluation Criteria	Dijkstra	A*	Global Planner (improved A*)
Expanded nodes	1128	1041	**35**
Path length	3.49 (m)	3.49 (m)	3.65 (m)

In summary, the results in **Table 2** prove that the improved A* algorithm at the global planning layer not only excels in computational efficiency by drastically reducing the number of expanded nodes but also generates a safer and more feasible trajectory for mobile robots in real-world environments. The minor trade-off in path length is entirely acceptable, especially when considering the overall benefits regarding stability, trajectory smoothness, and effective integration with the local NMPC controller.

Based on the global reference trajectory that has been optimized and smoothed by Cubic Spline, the Local Planner utilizing NMPC plays a decisive role in ensuring kinematic feasibility for the robot while strictly satisfying system constraints. Observing the local trajectory represented by the blue solid line in Fig. 4 and Fig. 5, it is evident that the robot accurately tracks the global reference trajectory generated by the improved A* algorithm while maintaining constraint conditions. For instance, it adheres to the established movement range limits of $x \in [-2, 2]$ (m) and $y \in [-2, 2]$ (m). The simulation states at (a), (b), and (c) in Fig. 5, corresponding to steps 70 (7.0 s), 182 (18.2 s), and 259 (25.9 s), also confirm that the robot always maintains a safe corridor, avoiding violation of the light gray Safety Buffer zone, even when executing complex steering maneuvers at locations with high curvature.

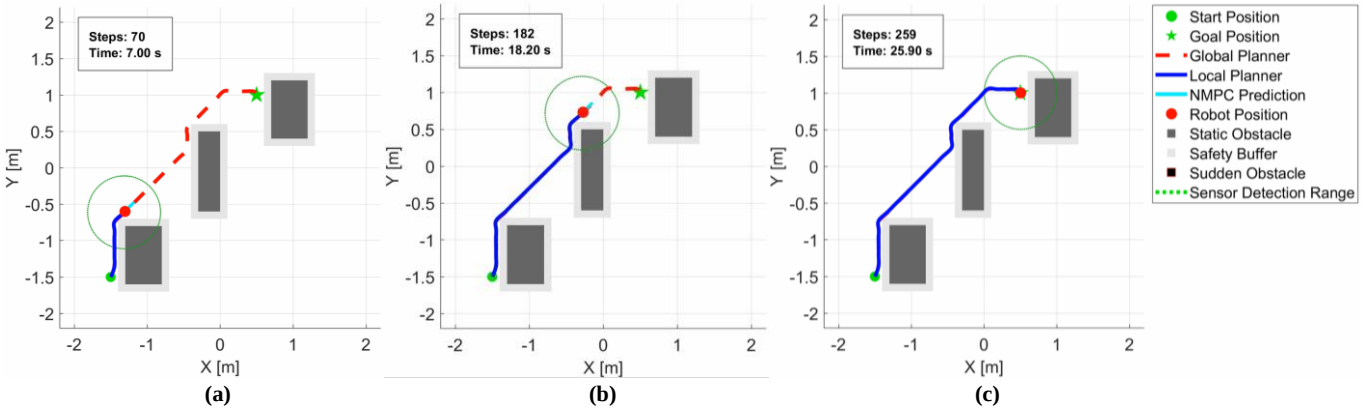

Figure 5: Process execution of the proposed method in the known static obstacle scenario: (a) Step 70, (b) Step 182, and (c) Step 259. The sequence proceeds from (a) to (c).

The performance of the control system is demonstrated through the tracking error graph in Fig. 6. Throughout the operation, position errors along the x and y axes are consistently controlled at very low levels, with a maximum oscillation amplitude ranging only from 0.005 m to 0.008 m. Specifically, the Euclidean distance error remains stably below 0.01 m and converges to a constant value of 0.005 m as the robot approaches the goal, affirming the high reliability and precision of the proposed method in a static environment. Furthermore, NMPC ensures the robot strictly adheres to actual velocity constraints, which can be observed more closely in Fig. 7. It can be seen that the control inputs always remain within the established limits of $v \in [-0.5, 0.5]$ (m/s) and $\omega \in \left[\frac{-\pi}{4}, \frac{\pi}{4}\right]$ (rad/s), and converge to zero immediately as the robot reaches the target position without generating redundant motion. Furthermore, the optimal control input signals can be converted into the optimal wheel angular velocities as per (15) for practical hardware implementation. Thanks to the continuity of the control signals and the adherence to the established constraints, these values can be directly utilized as setpoints for the low-level DC motor controllers. This ensures both the feasibility and stability of the system in real-world scenarios.

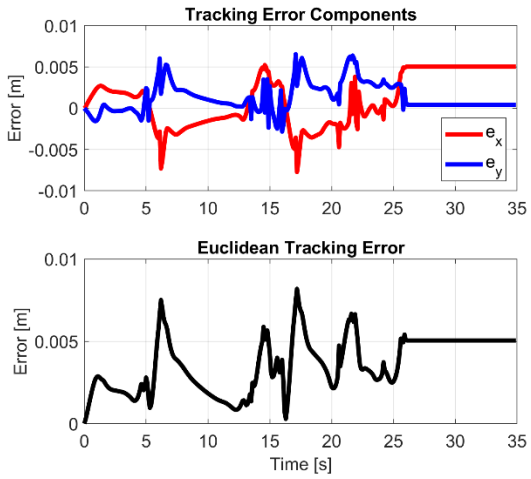

Figure 6: Tracking error of the proposed method in the known static obstacle scenario.

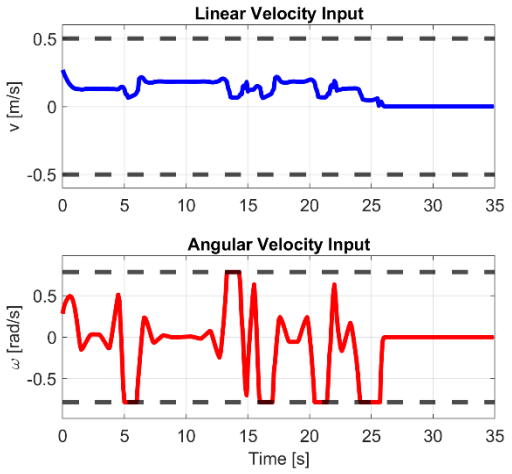

Figure 7: Control inputs of the proposed method in the known static obstacle scenario.

4.2 Sudden Obstacle Scenario

In this scenario, the robot operates in an environment where unexpected obstacles appear, in addition to the static obstacles already known on the global map. This situation is significantly more challenging than the static scenario, as the global planning and local planning systems must detect and replan trajectories in real-time to ensure safety against unforeseen obstacles along the movement path. Fig. 8 illustrates the overall trajectory in the sudden obstacle scenario. It can be observed that when a dynamic obstacle (black square with red border) enters the sensor's field of view, the initial global reference trajectory (red dashed line), originally constructed for static obstacles, no longer ensures the robot's safety. At this moment, the trajectory replanning mechanism based on improved A* is activated, allowing the robot to rapidly construct a new global trajectory to avoid the sudden obstacle while still heading toward the final goal. Meanwhile, Dijkstra and conventional A* lack a replanning mechanism, thus maintaining the initial path plan and causing a collision with the sudden obstacle.

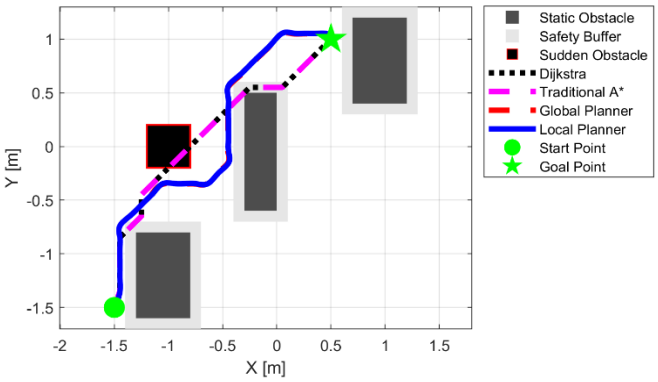

Figure 8: Motion trajectory planning results in the sudden obstacle scenario.

Alongside the role of the Global Planner, the NMPC controller at the local planning layer plays a pivotal role in ensuring the robot tracks the global reference trajectory accurately and stably. The global trajectory generated by the improved A* algorithm, after being smoothed by Cubic Spline, is fed into the NMPC as a fixed reference state sequence at each sampling cycle. NMPC not only optimizes tracking error but also simultaneously considers dynamic constraints and control input limits, thereby ensuring the global trajectory can be realized on a physical robot. Simulation results in Fig. 8 show that although the global trajectory may change abruptly when the replanning mechanism is triggered, NMPC ensures the transition process between trajectories occurs smoothly and continuously.

The image sequence in Fig. 9(a)–(f) clearly demonstrates the operational progress of the proposed method at different time steps. As soon as the sudden obstacle enters the sensor's detection zone (green dashed circle), the Global Planner updates and generates a new obstacle-avoidance trajectory. Simultaneously, the NMPC controller undertakes the generation of the local trajectory (blue line) to transition the robot smoothly from the old trajectory to the new one without oscillation or instability. Notably, the trajectory replanning process does not require stopping the robot; instead, it occurs continuously while the robot is in motion, demonstrating the

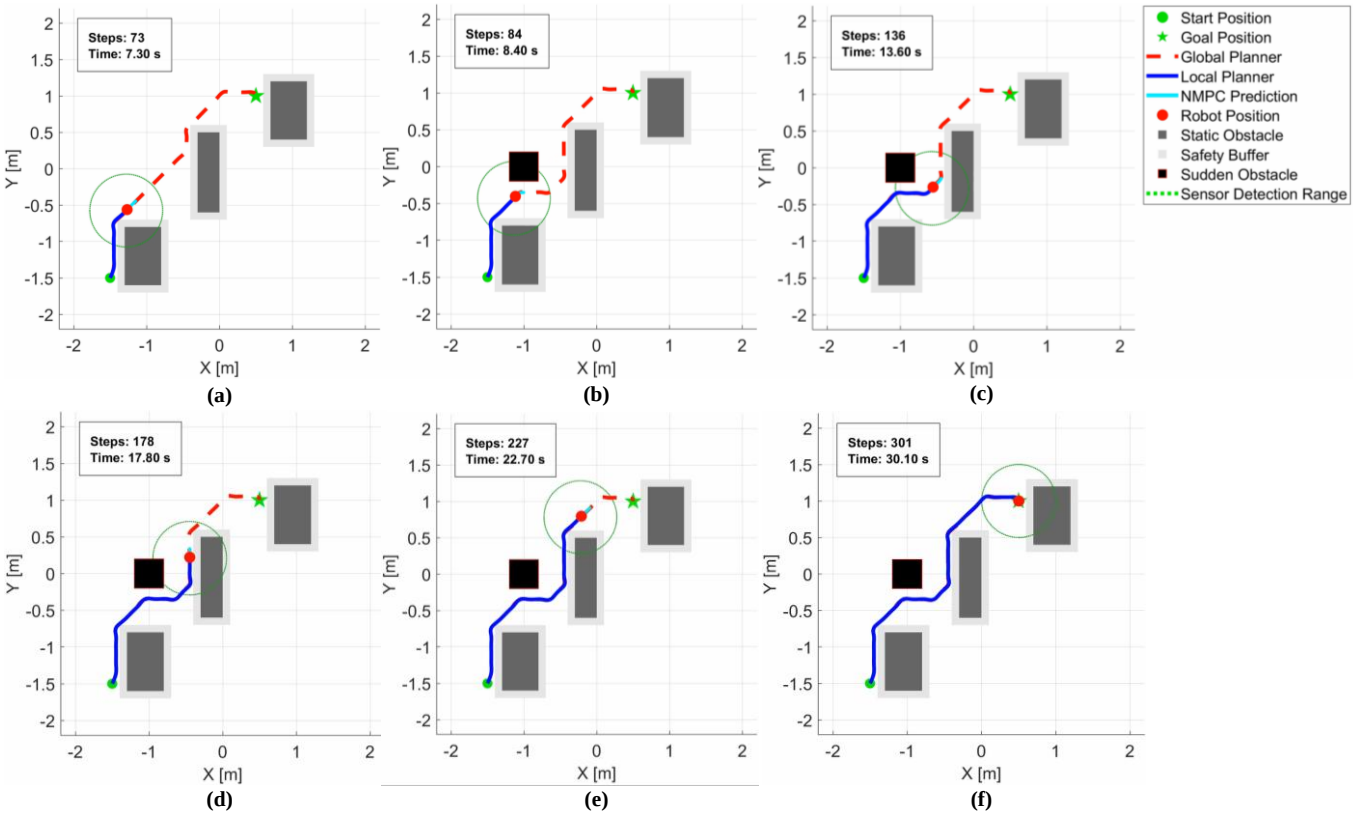

Figure 9: Process execution of the proposed method in the sudden obstacle scenario: (a) Step 73, (b) Step 84, (c) Step 136, (d) Step 178, (e) Step 227, and (f) Step 301. The sequence proceeds from (a) to (f).

high efficiency and practical applicability of the proposed architecture. Closer observation at Fig. 9(b) and Fig. 9(c), corresponding to steps 84 (8.4 s) and 136 (13.6 s), reveals that the trajectory replanning and obstacle avoidance process is performed safely and smoothly.

Table 3: Global trajectory planning performance in the sudden obstacle scenario

Evaluation Criteria	Dijkstra	A*	Global Planner (improved A*)
Expanded nodes	1128	1041	**64**
Path length	3.49 (m)	3.49 (m)	3.89 (m)

As shown in **Table 3**, although it must replan the global trajectory to avoid obstacles, the improved A* algorithm with the adaptive heuristic strategy still yields a very low number of expanded nodes. In contrast, Dijkstra and Conventional A* maintain the fixed global path as in the static scenario, leading to collisions. Specifically, the number of expanded nodes for Dijkstra and Conventional A* are 1128 and 1041, respectively, with the same search path length of 3.49 m. Meanwhile, the Improved A* algorithm at the global planning layer requires expanding only 64 nodes to find a safe obstacle-avoidance trajectory, an improvement of 94.33% compared to Dijkstra and 93.85% compared to Conventional A*. Compared to the static scenario in **Table 2** (35 nodes), the number of expanded nodes increases due to the requirement for trajectory replanning in a dynamic environment; however, this increase is reasonable and acceptable, traded off for the ability to ensure safety in more complex real-world conditions. In this case, the path length of the Global Planner is 3.89 m, longer than Dijkstra and A* due to the detour required to avoid collision.

Fig. 10 presents the tracking error components along the x and y axes, as well as the composite Euclidean error in the sudden obstacle scenario. It can be observed that even when a trajectory replanning event occurs, the error amplitude remains strictly controlled. Specifically, the error along each axis oscillates with an amplitude of approximately 0.008 m, while the maximum Euclidean error does not exceed 0.01 m. After each obstacle avoidance phase and re-entry into the new reference trajectory, the error rapidly decays and converges back to a stable level. This proves that the NMPC controller is capable of handling sudden fluctuations in the reference trajectory effectively while maintaining high precision even in a dynamic environment. The Euclidean error converges to a constant value smaller than 0.005 m at the end of the motion, demonstrating the precision and safety of the proposed hybrid motion planning framework. Furthermore, similar to the static scenario, the control inputs observed in Fig. 11 always remain within the established limits of $v \in [-0.5, 0.5]$ (m/s) and $\omega \in \left[\frac{-\pi}{4}, \frac{\pi}{4}\right]$ (rad/s), and converge to zero immediately as the robot reaches the target position without generating redundant motion. These control input signals can be converted into optimal wheel angular velocities via (15) for direct hardware implementation, ensuring the feasibility and stability of the system in practical scenarios.

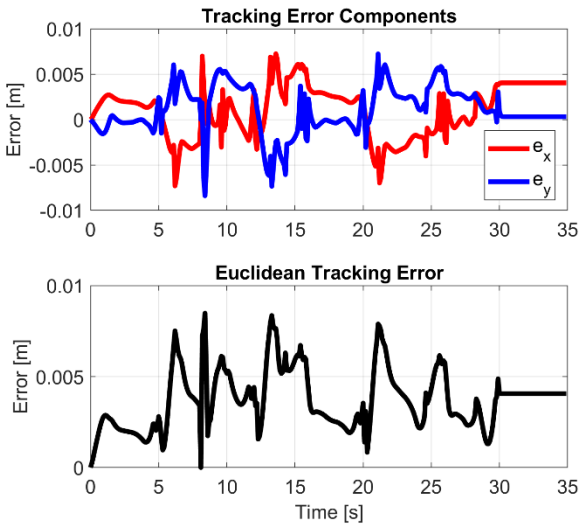

Figure 10: Tracking error of the proposed method in the in the sudden obstacle scenario

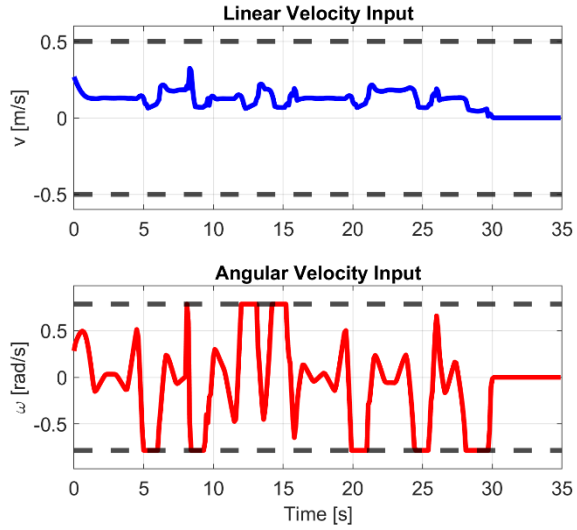

Figure 11: Control inputs of the proposed method in the sudden obstacle scenario.

To further evaluate the real-time local planning capability of the NMPC, Fig. 12 and Fig. 13 provide quantitative results for the execution time and the average execution time per step in both simulation scenarios. Specifically, the average execution time per step is calculated using equation (18) to ensure consistency and objectivity throughout the evaluation process. Based on the obtained results in Fig. 12, it is observed that the execution time at all instances remains below the sampling time of 0.1 s. This confirms that the NMPC optimization problem is consistently solved within a single sampling period, thereby ensuring the feasibility of real-time control implementation. Quantitatively, Fig.13 shows that the average execution time per step in the static obstacle scenario is 0.0299 s, while in the dynamic obstacle scenario, it is 0.0311 s. The slight increase in computational time for the dynamic scenario can be attributed to the system's need to continuously update environmental information and process reference trajectory modifications, which heightens the complexity of the optimization problem. Nevertheless, this difference is marginal and remains well within the sampling interval limit, indicating that the proposed method maintains stable computational performance even in complex environments with random obstacles. These results demonstrate that the NMPC-based local planning layer in the hybrid framework not only ensures trajectory tracking accuracy but also satisfies real-time computational requirements, making it suitable for mobile robot systems with limited hardware resources. Furthermore, with the advancement of modern embedded computing platforms and energy storage technologies, contemporary mobile robots possess greater processing power and longer operational endurance. Consequently, the computational overhead of the proposed method is well within the capabilities of existing hardware systems, further reinforcing its feasibility for practical deployment.

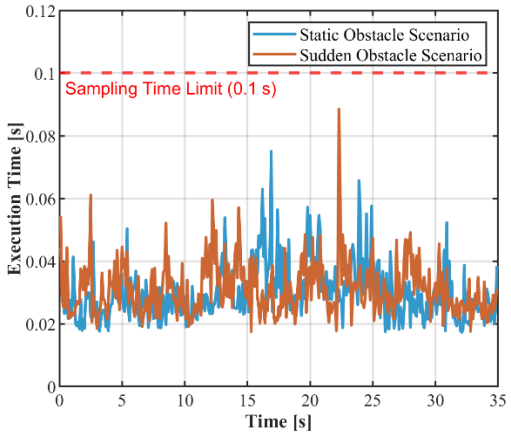

Figure 12: Execution time in the two simulation scenarios.

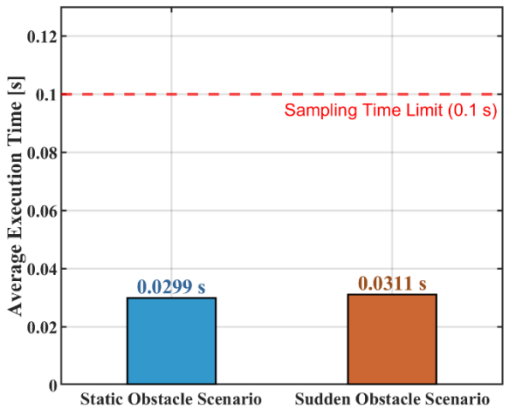

Figure 13: Average execution time per step in the two simulation scenarios.

From the simulation results and quantitative analysis, it can be affirmed that the architecture combining the improved A* algorithm for the global planning layer and NMPC for the local control layer not only operates effectively in static environments but also demonstrates robustness and a high level of safety in dynamic environments with sudden obstacles. The proposed system ensures safe collision avoidance, high trajectory tracking accuracy, and maintains rapid response capabilities to environmental changes, thereby indicating high suitability for motion planning problems for mobile robots under complex real-world conditions. In the static obstacle scenario, the number of expanded nodes of the Global Planner using improved A* is reduced by 96.64% compared to Conventional A* and 96.90% compared to Dijkstra. For the dynamic

environment scenario with sudden obstacles, the improvement level remains high, with the number of expanded nodes reduced by 94.33% compared to Dijkstra and 93.85% compared to Conventional A*. These results demonstrate the distinct effectiveness of the adaptive heuristic strategy in reducing computational complexity while still ensuring safe trajectory replanning capabilities. Furthermore, from a real-time perspective, the significant reduction in the number of expanded nodes at the global planning layer shortens the path-finding duration and minimizes latency during the replanning process when environmental changes occur. Simultaneously, at the local layer, the NMPC controller consistently ensures that the execution time remains below the sampling period, allowing the system to continuously update and apply control signals in real-time. This synergy establishes a hybrid motion planning mechanism in which the global layer provides oriented trajectories with low computational cost and rapid response to fluctuations, while the local layer ensures trajectory tracking capability. Consequently, the entire system maintains stable real-time performance, even in complex dynamic environmental scenarios.

5. Conclusions and future works

This paper proposed and evaluated a hybrid motion planning framework for three-wheeled mobile robots based on the combination of an improved A* algorithm for global planning and an NMPC controller for local planning. Through the utilization of an adaptive heuristic strategy, the improved A* algorithm demonstrated the capability to drastically reduce the number of expanded nodes during the search process, while generating a safer and more suitable global trajectory for practical implementation compared to classical algorithms such as Dijkstra and conventional A*. At the local layer, the NMPC controller proved its capability to track the global trajectory with high precision, ensuring smooth motion and strict compliance with kinematic, non-holonomic, workspace, and control limit constraints. Notably, the trajectory replanning mechanism based on improved A* allowed the system to adapt effectively to sudden obstacles, while maintaining low computational cost due to the minimal number of expanded nodes required. Simulation results in both static and dynamic environments indicated that the proposed method not only excels in computational efficiency but also ensures a higher level of safety and robustness compared to Dijkstra and Conventional A* methods. In the future, subsequent research directions will focus on extending this planning framework to environments with more complex dynamic obstacles, integrating model uncertainties and sensor noise, directly considering acceleration or motor torque constraints, as well as implementing and verifying it on a real robot platform to comprehensively evaluate its applicability in real-world conditions. Additionally, integrating physics-informed deep learning models, such as Physics-Informed Neural Networks (PINNs), for model identification and dynamics prediction is considered a promising extension, contributing to enhancing the accuracy and adaptability of the control architecture.

References

[1]. Wijaya, R. S., Mahendra, A., Prayoga, S., & Wibisana, A. (2024, *December) Path Planning Application using Dijkstra Algorithm on Service Robot.* In 7th International Conference on Applied Engineering (ICAE 2024) (pp. 262-271). Atlantis Press.

[2]. Wang, B. (2021, August) *Path planning of mobile robot based on A* algorithm*. In 2021 IEEE International Conference on Electronic Technology, Communication and Information (ICETCI) (pp. 524-528). IEEE.

[3]. Tran, N. H., Dao, Q. T., & Ngoc-Tam, B. U. I. (2025) *Trajectory and parameter optimization in robust tracking control of a quadrotor.* IEEE Access.

[4]. Liu, H., Cao, J., & Wang, Z. (2025) *Research on path planning of mobile robot in complex environment.* Discover Applied Sciences, 7(4), 1-18.

[5]. Fu, X., Huang, Z., Zhang, G., Wang, W., & Wang, J. (2025) *Research on path planning of mobile robots based on improved A* algorithm.* PeerJ Computer Science, 11, e2691.

[6]. Li, X., & Tong, Y. (2023) *Path planning of a mobile robot based on the improved RRT algorithm.* Applied Sciences, 14(1), 25.

[7]. Alshammrei, S., Boubaker, S., & Kolsi, L. (2022) *Improved Dijkstra algorithm for mobile robot path planning and obstacle avoidance.* Comput. Mater. Contin, 72(3), 5939-5954.

[8]. Ismael, O. Y., Almaged, M., & Abdulla, A. I. (2024) *Nonlinear model predictive control-based collision avoidance for mobile robot.* Journal of Robotics and Control (JRC), 5(1), 142-151.

[9]. Sani, M., Hably, A., Robu, B., Dumon, J., & Meslem, N. (2023, June) *Real-time Dynamic Obstacle Avoidance For A Non-holonomic Mobile Robot.* In 2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE) (pp. 1-6). IEEE.

[10]. Do Nguyen, Q.-D., Dinh, X.-M., Pham, T.-L., Duong, T.-T., Le, X.-H., & Pham, H.-T. (2025) *Intelligent motion planning of three-wheeled mobile robots via nonlinear model predictive control.* Journal of Measurement, Control, and Automation, 29(4), 95–106. https://doi.org/10.64032/mca.v29i4.350

[11]. Dang, S. T., Dinh, X. M., Kim, T. D., Xuan, H. L., & Ha, M. H. (2023) *Adaptive backstepping hierarchical sliding mode control for 3-wheeled mobile robots based on RBF neural networks.* Electronics, 12(11), 2345.

[12]. Li, B., Ni, K., Zhou, F., Li, Y., Huang, W., Jiang, H., & Liu, F. (2025) *Research on robot path planning based on fused Dijkstra and TEB algorithms.* Journal of Mechanical Science and Technology, 39(8), 4651-4660.

[13]. Dang, S. T., Dinh, X. M., Than Ngoc, T. A., Nguyen, V. T., Kim, D. T., & Le, X. H. (2025) *Adaptive Sliding Mode Control for Robust Trajectory Tracking of Car-Like Mobile Robots in Autonomous Navigation Systems.* Journal of Applied and Computational Mechanics, e19517.

[14]. Rösmann, C., Feiten, W., Wösch, T., Hoffmann, F., & Bertram, T. (2012, May) *Trajectory modification considering dynamic constraints of autonomous robots.* In ROBOTIK 2012; 7th German Conference on Robotics (pp. 1-6). VDE.

[15]. Chen, L., Liu, R., Jia, D., Xian, S., & Ma, G. (2025, January) *Improvement of the TEB Algorithm for Local Path Planning of Car-like Mobile Robots Based on Fuzzy Logic Control.* In Actuators (Vol. 14, No. 1, p. 12). MDPI.

[16]. Bui, H. L., Nguyen, T. D., & Mac, T. T. (2026) *A novel approach to design multi-input Hedge-Algebras-based controllers and applications in motion control of autonomous vehicles.* Intelligent Service Robotics, 19(1), 1.

[17]. Quang, H. D., Tran, T. L., Manh, T. N., Manh, C. N., Nhu, T. N., & Duy, N. B. (2022) *Design a nonlinear MPC controller for autonomous mobile robot navigation system based on ROS.* International Journal of Mechanical Engineering and Robotics Research, 11(6).

[18]. DANG, T. V. (2024) *Optimization Hybrid Path Planning Based on A-star Algorithm Combining with DWA.* MM Science Journal, 10, 7551-7555.

[19]. De Carvalho Filho, J. G. N., Carvalho, E. Á. N., Molina, L., & Freire, E. O. (2019) *The impact of parametric uncertainties on mobile robots velocities and pose estimation.* IEEE Access, 7, 69070-69086.

[20]. Wang, F., Sun, W., Yan, P., Wei, H., & Lu, H. (2024) *Research on path planning for robots with improved A* algorithm under bidirectional JPS strategy.* Applied Sciences, 14(13), 5622.

[21]. De Boor, C., & De Boor, C. (1978) *A practical guide to splines (Vol. 27, p. 325).* New York: springer.

[22]. Bock, H. G., & Plitt, K. J. (1984) *A multiple shooting algorithm for direct solution of optimal control problems.* IFAC Proceedings Volumes, 17(2), 1603-1608.

[23]. Andersson, J. A., Gillis, J., Horn, G., Rawlings, J. B., & Diehl, M. (2019) *CasADi: a software framework for nonlinear optimization and optimal control*. Mathematical Programming Computation, 11(1), 1-36.

[24]. Wächter, A., & Biegler, L. T. (2006) *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming.* Mathematical programming, 106(1), 25-57.