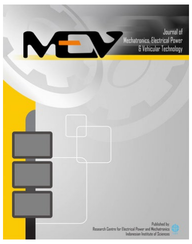
Journal of Mechatronics, Electrical Power, and Vehicular Technology 15 (2024) 186-196



Journal of Mechatronics, Electrical Power, and Vehicular Technology

CrossMark

e-ISSN: 2088-6985
p-ISSN: 2087-3379

mev.brin.go.id

Non-linear model predictive control with single-shooting method for autonomous personal mobility vehicle

Rakha Rahmadani Pratama [a], Catur Hilman Adritya Haryo Bhakti Baskoro [b], Joga Dharma Setiawan [a], Dyah Kusuma Dewi [b], Paryanto Paryanto [a], Mochammad Ariyanto [c], Roni Permana Saputra [b, *]

[a] *Department of Mechanical Engineering, Diponegoro University*
Jl. Prof. Soedarto, No. 13 Tembalang, Kota Semarang, 50275, Indonesia

[b] *Research Center for Smart Mechatronics, National Research and Innovation Agency (BRIN)*
Kawasan Sains dan Teknologi (KST) Samaun Samadikun, Jalan Sangkuriang, Bandung, 40135, Indonesia

[c] *Department of Mechanical Engineering, Osaka University*
1-1 Yamadaoka, Suita, Osaka, 565-0871, Japan

Abstract

The advancement of autonomous vehicle technology has markedly evolved during the last decades. Reliable vehicle control is one of the essential technologies in this domain. This study aims to develop a proposed method for controlling an autonomous personal mobility vehicle called single-passenger electric autonomous transporter (SEATER) using non-linear model predictive control (NMPC). We propose a single-shooting technique to solve the optimal control problem (OCP) via non-linear programming (NLP). The NMPC is applied to a non-holonomic vehicle with a differential drive setup. The vehicle utilizes odometry data as feedback to help guide it to its target position while complying with constraints, such as vehicle constraints and avoiding obstacles. To evaluate the method's performance, we have developed the SEATER model and testing environment in the Gazebo simulation and implemented the NMPC via the robot operating system (ROS) framework. Several simulations have been done in both obstacle-free and obstacle-filled areas. Based on the simulation results, the NMPC approach effectively directed the vehicle to the desired pose while satisfying the set constraints. In addition, the results from this study have also pointed out the reliability and real-time performance of NMPC with a single-shooting method for controlling SEATER in the various tested scenarios.

Keywords: model predictive control; autonomous robot; collision avoidance; robot operating system; non-linear programming.

I. Introduction

In the era of rapid technological advancement, self-driving cars are one of the most significant disruptive technologies that will affect transportation [1]. The industry and research community continue to progress with self-driving car technology to discover new modes of transportation, increase road safety, and automate processes [2]. Developing appropriate control strategies is critical for increasing the potential of autonomous vehicles [3]. Model predictive control

* Corresponding Author. roni.permana.saputra@brin.go.id (R. P. Saputra)

https://doi.org/10.55981/j.mev.2024.1105

Received 28 November 2024; revised 17 December 2024; accepted 18 December 2024; available online 30 December 2024; published 31 December 2024

How to Cite: R. R. Pratama *et al*, “Non-linear model predictive control with single-shooting method for autonomous personal mobility vehicle,” *Journal of Mechatronics, Electrical Power, and Vehicular Technology*, vol. 15, no. 2, pp. 186-196, Dec. 2024.

(MPC) is a popular advanced control method that has been widely used and developed due to its capability of managing multivariable systems while considering physical and environmental restrictions and its use of optimization techniques [4].

Over the last decades, MPC has shown its capability of controlling complex systems [5] like under actuated systems [6], industrial processes [7], smart buildings [8], automation and autonomous systems [9], power electronics [10], mobile manipulation [11], and rescue robotic systems [12]. It can also deal with harsh conditions like bad weather [13] handle non-linear problems while supporting multi-input–multi-output (MIMO) systems [14], and for systems or processes in which safety is critical [15]. By enabling prediction in control, MPC helps in quick and effective decision-making, improving autonomous vehicles' skills to operate in dynamic conditions [16].

The MPC methods have developed rapidly along with the advanced control with various numerical approaches to solve optimization problems [17]. In optimal control, field optimization problems are typically addressed using two main approaches: direct and indirect techniques [18]. The direct method transforms the optimal control problem into non-linear programming (NLP) via suitable discretization techniques [19]. The single-shooting method is one of the simplest direct methods for solving MPC control problems. One example is controlling non-holonomic robots, as presented in [20]. This approach's main benefit is its simplicity and ability to integrate complex system models with reduced complexity [21].

One of the most difficult tasks in the autonomous system field is maintaining the vehicle's motion stability while producing optimal paths [22]. Implementing an MPC controller with the single shooting approach has demonstrated its ability to meet this challenge by providing an optimal solution considering the vehicle's complex dynamics [23]. For example, the work presented in [24] uses an non-linier model predictive control (NMPC) approach for differential-drive wheeled mobile robots. This controller allows the robots to reach the target pose considering dynamic obstacles and physical constraints. The study presented in [25] also demonstrates that the MPC, which uses single-shooting, has a potential to perform well in terms of computational speed and is suitable for real-time applications.

Our prior work presented in [26] introduced NMPC-based visual servoing method for the autonomous docking of the SEATER platform. This method uses real-time visual data to guide the vehicle's movements. Based on the simulation results, the proposed method could control the vehicle to generate optimal docking paths in various scenarios. This paper expands upon prior studies by integrating waypoints into the NMPC trajectory-following framework. In addition, the vehicle model and control system are developed in the Gazebo physics engine to enable realistic simulations that represent the vehicle's dynamic features.

The main objective of this study is to develop a controller for the single-passenger electric autonomous transporter (SEATER) platform using NMPC approach. The NMPC aims to improve how SEATER is controlled, adjust to different situations, and maintain accurate positioning.

More specifically, the contributions of this study are as follows:

- The formulation of an NMPC method to address the autonomous navigation of the SEATER platform, including path following via waypoints.
- The modelling of the SEATER platform within the Gazebo physics engine for simulation tests and to validate the formulated NMPC method.
- Implementing the formulated NMPC controller via the robot operating system (ROS) framework on the SEATER platform.

The next sections are organized as follows: Section II provides the materials and methods, discussing the NMPC problem setup and its implementation on the SEATER platform. Section III presents the experimental results and discusses their evaluations. Finally, Section IV sums up the key findings.

II. Materials and Methods

A. SEATER kinematic model and model discretization

SEATER is an autonomous personal mobility vehicle equipped with a differential drive system and two independent rear wheels. This design enables fine control of speed and direction, resulting in good maneuverability. Controlling the speed of each individual driving wheel allows for a variety of trajectories. SEATER is classified as a non-holonomic system because of its inherent motion constraints. Non-holonomic systems include those that are unable to move in all directions due to constraints such as wheel configuration. The limitations affect the control design for such vehicles, requiring advanced control systems such as MPC for precise maneuvering [17]

To represent the kinematics of SEATER, we use a basic unicycle model, as also presented in [26], wherein the state vector of the vehicle is defined as follows equation (1),

$$\mathbf{x}(t) = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix} \quad (1)$$

The initial two spatial components of the state $\mathbf{x}$ denote the vehicle's position in the plane (x, y), whereas the angle θ indicates the vehicle's heading.

The vehicle is controlled with input is defined as the following equation (2),

$$\mathbf{u}(t) = \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} \quad (2)$$

where v represents forward velocity, and ω denotes angular velocity.

The SEATER dynamics model $\dot{\mathbf{x}}(t)$ is formulated in a continuous-time state space defined by the function f and the control input $\mathbf{u}(t)$ as shown in equation (3) [27].

$$\dot{\mathbf{x}}(t) = f\big(\mathbf{x}(t), \mathbf{u}(t)\big) = \begin{bmatrix} v(t)\cos\theta(t) \\ v(t)\sin\theta(t) \\ \omega(t) \end{bmatrix} \quad (3)$$

This continuous model allows for a detailed representation of the vehicle's behavior, but to implement control algorithms effectively a discrete-time model is often more practical. The Euler method is used to discretize the SEATER model, converting the continuous model into a form suitable for digital computation and control [26]. The reformulated discrete system equations are provided in equation (4).

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k)) \quad (4)$$

Eulerian discretization goes further by numerically solving ordinary differential equations (ODEs), as shown in equation (5).

$$\begin{bmatrix} x(k+1) \\ y(k+1) \\ \theta(k+1) \end{bmatrix} = \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} + \Delta T \begin{bmatrix} v(k)cos\big(\theta(k)\big) \\ v(k)sin\big(\theta(k)\big) \\ \omega(k) \end{bmatrix} \quad (5)$$

Using this approach the next state $(k+1)$ of the robot or vehicle can be predicted based on the current state at the time k. The calculation adds the product of the sampling time ΔT, linear velocity v, and the trigonometric components $\cos(\theta(k))$ and $\sin(\theta(k))$ to the current position, enabling the model to estimate the vehicle's updated state.

B. NMPC cost function

Model Predictive Control (MPC) is a control technique that optimizes a system's future behaviour by addressing an optimization problem at each control step. MPC uses a system dynamics model to predict a series of control inputs across a time interval known as the prediction horizon [28]. The control inputs then optimize the desired objective function under particular constraints, such as obstacle avoidance constraints. NMPC is an effective method for managing complex non-linear systems with multiple objectives and constraints [29].

The optimal control sequence that produces a state $\mathbf{x}(k)$ that is close to the reference value (set point) $\mathbf{x}^r$ for $k = 0, \dots, N-1$ is determined through an optimization process aimed at minimizing the cost function, $l(\mathbf{x}(k), \mathbf{u}(k))$ [15]. This cost function calculates the error between the current state $\mathbf{x}(k)$ and the reference $\mathbf{x}^r$. In addition, the cost function also calculates the error between the value of the applied control, $\mathbf{u}(k)$, and the control target $\mathbf{u}^r$. The square norms of each element are computed and then summed together, as formulated in equation (6). A weighting matrix that includes Q (the state cost weighting matrix) and R (the control effort weighting matrix) is also incorporated. The weighting matrix in the cost function can be fine-tuned to adjust the control performance [30]. In this study, the weighting matrices Q and R were defined as fixed variables, with Q set such that $Q[0,0] = 1$, $Q[1,1] = 5$, $Q[2,2] = 0.1$, and R set such that $R[0,0] = 0.5$, $R[1,1] = 0.05$.

$$l(\mathbf{x}(k), \mathbf{u}(k)) = \|\mathbf{x}(k) - \mathbf{x}^r\|_Q^2 + \|\mathbf{u}(k) - \mathbf{u}^r\|_R^2 \quad (6)$$

The MPC formulation integrates motion planning and trajectory tracking, generating the control input based on the optimal control problem (OCP) problem [31]. The OCP aims to determine the control sequence that minimizes the cost function [32]. In this work, the OCP is formulated and solved as NLP problem. The standard NLP framework for numerical parametric optimization is formulated in equation (7) [33].

$$\begin{aligned} &minimize: \quad J_N(\mathbf{x}, \mathbf{u}) = \textstyle\sum_{k=0}^{N-1} l\big(\mathbf{x}(k), \mathbf{u}(k)\big) \quad (7) \\ &subject\ to: \quad \mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k)), \end{aligned}$$

$$\mathbf{x}(0) = \mathbf{x}_0$$

$$\mathbf{u}(k) \in U, \quad \forall k \in [0, N-1],$$

$$\mathbf{x}(k) \in X, \quad \forall k \in [0, N]$$

According to equation (7), the cost function to be minimized, J, represents the cumulative sum of the stage costs $l(\mathbf{x}(k), \mathbf{u}(k))$ at each time step k. This function quantifies the penalty or cost associated with the state $\mathbf{x}(k)$ and the control input $\mathbf{u}(k)$. The system must comply with the conditions specified by $\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k))$ and start from an initial condition $\mathbf{x}(0)$. Additionally, the control $\mathbf{u}(k)$ and the state $\mathbf{x}(k)$ must stay within their designated sets U and X to guarantee that the solution is valid and meets all required constraints. Although extended prediction horizons may yield better results, they also heighten the demand for computational resources. Therefore, it is essential to establish a balance in selecting a prediction horizon that ensures satisfactory performance while also preserving computational efficiency [34].

C. NMPC shooting method

NMPC's shooting technique is one of the numerical approaches used to solve optimum control issues. This approach involves decomposing non-linear dynamical systems, using ordinary differential equation (ODE) solvers to solve differential equations repeatedly, and adjusting control inputs over a set period of time. This approach increases computational efficiency and enforces constraints while allowing for the modelling of complex systems. Because of these properties, it is effective for addressing optimal control problems in dynamic non-linear systems.

NMPC's single-shooting method focuses on a single path to maximize the MPC objective and discover the optimal solution. This method is simple, relying on a single series of control activities. As a result, this study uses a single-shooting strategy to tackle the NMPC optimization problem. Single-shot optimization uses fewer dimensions than other approaches, which helps to accelerate the solution process [21].

Let w denote the decision variable of the optimization process, representing the control vector comprising a sequence of control values from u_0 to u_{N-1} over the finite horizon N. These control values are the variables optimized in MPC as shown in equation (8).

$$w = [u_0, \dots, u_{N-1}] \tag{8}$$

The robot state trajectory $X_u(.)$ along the horizon N can be expressed as a recursive function of the control trajectory using the robot's dynamic function as shown in equation (9).

$$X_u(.) = F(W, X_0, t_k) \tag{9}$$

$$F(w, x_0, t_0) = x_0$$

where $X_u(.)$ represents the state trajectory, dependent on the control vector W, initial state X_0, and time t_k, $F(W, X_0, t_k)$ describes the system's dynamics, showing how the system's state changes based on the applied control.

$$\min_{w} \Phi\,(F(w, x_0, t_k), w) \tag{10}$$

$$\text{subject to} : (F(w, x_i, t_k), w) \leq 0$$

Equation (10) shows the optimization problem that aims to minimize a cost function while keeping to system constraints. These limits can include physical, safety, and operational requirements the system must satisfy. In this study, to add the obstacle avoidance factor, the obstacle area is described by a circular boundary, as shown in equation (11).

$$\sqrt{(x - x_{ob})^2 + (y - y_{ob})^2} - \mathrm{r} + r_{ob} \geq 0 \tag{11}$$

The safe distance from the obstacles to the vehicle can be determined using equation (11) as a constraint for obstacle avoidance. The vehicle's coordinates are (x, y), while those of the obstacle are (x_{ob}, y_{ob}), with r and r_{ob} representing the vehicle's and the obstacle's sizes, respectively. This constraint guarantees that the system stays away from the obstacle and follows a safe path by setting a safe zone around it.

Algorithm 1 summarizes the NMPC approach used in this research. Initialization involves defining the target state $\mathbf{x}^r$, prediction horizon N, and initial state $\mathbf{x}_0$. The optimization process minimizes costs while satisfying the system's constraints so that the minimum cost value obtained from the cost function $J_N(\hat{\mathbf{x}}, \mathbf{u}^*)$ is V_N. We achieve this by solving an OCP based on the estimated current state subject to the constraint $(F(w, x_i, t_k), w) \leq 0$ at each sample point. The first control input $\mathbf{u}_0$ from the optimized sequence is then applied to the system, and this loop continues at each time step to guide the system towards the desired state effectively.

Algorithm 1: Non-linear model predictive control via single shooting method

MPC init:
Prediction horizon: = N
Define the initial vehicle state:
$\mathbf{x}(0) := \mathbf{x}_0$
Define the target state: $\mathbf{x}^r$
Define the initial control: $\mathbf{u}_0$
Apply $\mathbf{u}_0$ to the system
for every sampling instant k = 1, 2, ... do
Estimate the states $\mathbf{x}(k)$
Solve OCP:
Find the optimal control horizon
w= [u_0, …, $u_{(N-1)}$]
which satisfies
$J_N(\hat{\mathbf{x}}, \mathbf{u}^*) = V_N$
s.t.
$(F(w, x_i, t_k), w) \leq 0$
Apply $\mathbf{u}_0$ to the system

III. Results and Discussions

In order to evaluate the performance of the proposed NMPC implemented on the SEATER platform, various experimental scenarios were designed and carried out utilizing Python and Gazebo simulation environments. The computational hardware utilized for the simulations comprised an

Table 1.
MPC data collection with obstacles-free.

Parameter		Performance			
ΔT	N	Total time iteration(s)	Max time iteration (s)	Euclidean position error (m)	Rotation error (rad)
0.01	5	10.002	0.007	1.086	0.021
	10	9.836	0.009	0.618	0.039
	15	9.031	0.017	0.222	0.004
	20	8.699	0.016	0.150	0.004
	25	8.295	0.016	0.129	0.010
0.05	5	10.165	0.014	0.167	0.018
	10	6.961	0.015	0.079	0.003
	15	6.959	0.020	0.059	0.009
	20	6.963	0.015	0.051	0.008
	25	6.996	0.019	0.043	0.009
0.1	5	6.818	0.014	0.084	0.025
	10	6.206	0.014	0.041	0.016
	15	1.523	0.017	0.039	0.014
	20	0.897	0.018	0.036	0.026
	25	0.802	0.026	0.034	0.028
0.5	5	0.171	0.015	0.026	0.014
	10	0.222	0.026	0.021	0.028
	15	0.361	0.029	0.021	0.004
	20	0.390	0.032	0.025	0.019
	25	0.452	0.041	0.026	0.021

Intel Core i7 processor with eight cores operating at a frequency of 2.30 GHz. To ensure consistent and reproducible testing conditions, the experimental setup examined scenarios featuring flat terrain and fixed obstacles. The NMPC algorithm was created in Python and employed the CasADi framework to address NLP issues through algorithmic differentiation [35].

1) Sampling time and prediction horizon evaluation in obstacle-free condition

The first experiment in this study evaluates the effect of sampling time and prediction horizon value on controller performance. We used Python simulations to implement the NMPC controller to move the vehicle from its starting point (0, 0) to three target points: (1.5, 1.5), (1.5, 0), and (1.5, -1.5). The complete iteration duration was set to 10 seconds, and the maximum iteration time had to be shorter than the sampling time. The performance requirements were a goal Euclidean distance of 0.4 meters and a rotation threshold of 0.4 radians. To make the testing more realistic, 10 % control noise and a localization error of 0.02 m were introduced. The simulations were performed under obstacle-free conditions.

Table 1 shows the performance of NMPC under obstacle-free settings. The results indicate that a shorter sampling time is associated with a rise in total iteration time, maximum duration, and inaccuracy in distance and rotation, all of which can lead to failure. The experiments with the sampling time parameter of 0.5 seconds and prediction horizons 15, 20, and 25 have satisfied the performance requirements, while the experiments with the sampling time parameter of 0.5 seconds have met the requirements in all tested prediction horizons.

2) Sampling time and prediction horizon evaluation in obstacle condition

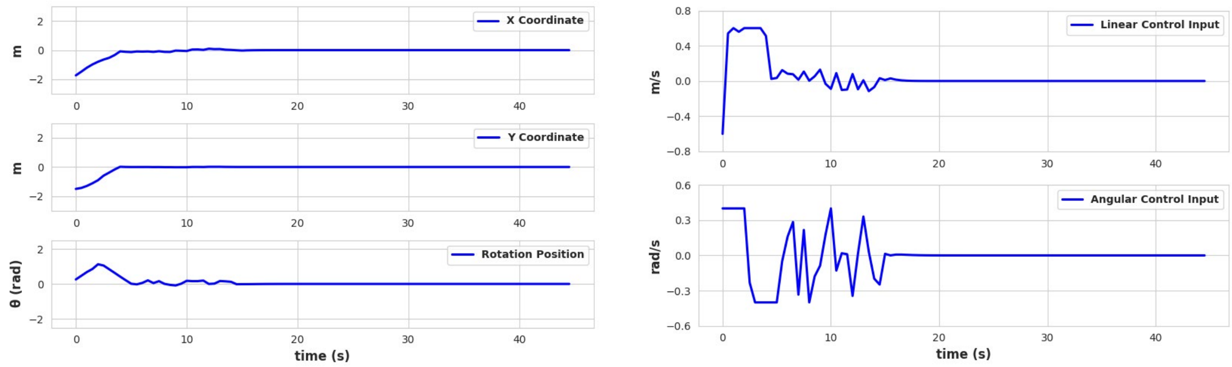
The obstacle condition was then tested using parameters that met these thresholds in the previous experiments, with sampling time 0.1 along with prediction horizons 15, 20, and 25, and with sampling time 0.5 along with prediction horizons 5, 10, 15, 20, and 25. In these experiments, SEATER was modelled with a perimeter diameter of 0.3 m, and obstacles, each with a 0.2 m diameter, were positioned at (0.8, 0.3), (0.8, -0.3), and (1, 0). We also include a safety tolerance of 0.05 m to prevent intersection into the model. The NMPC controller is designed to control the vehicle's movement from the starting point to the target pose, similar to the previous experiments. In these experiments, the vehicle must navigate without colliding with the obstacle during movement.

The performance results of the NMPC controller simulations in the presence of a static obstacle can be found in Table 2. The results indicate that optimal performance was attained with a sampling time of 0.5 seconds and prediction horizons of 20 and 25, with N = 20 identified as the most effective setup.

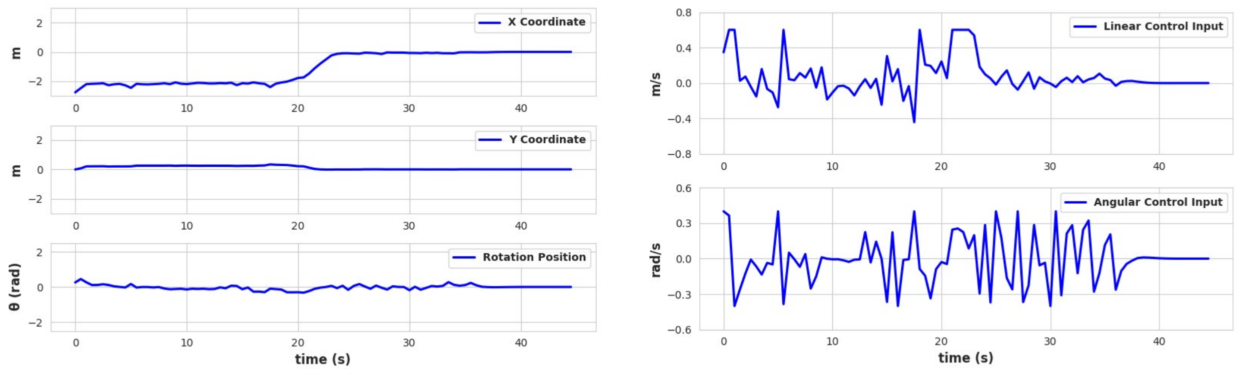
We subsequently assessed the stability of the controller to attain the desired control input value at the terminal state. At the target pose, we intend for the vehicle's control input values for both linear velocity and angular rate to be zero in order to achieve stability at the terminal state. Figure 1 illustrates the vehicle's performance and trajectory within a Python simulation. The simulation operates with specified parameters $\Delta T = 0.5$ and $N = 20$ in conjunction with the presence of a static obstacle. The results show that the vehicle adheres to the speed limit and arrives at the destination with a terminal velocity of zero, indicating its stability.

3) NMPC implementation and evaluation in Gazebo simulations.

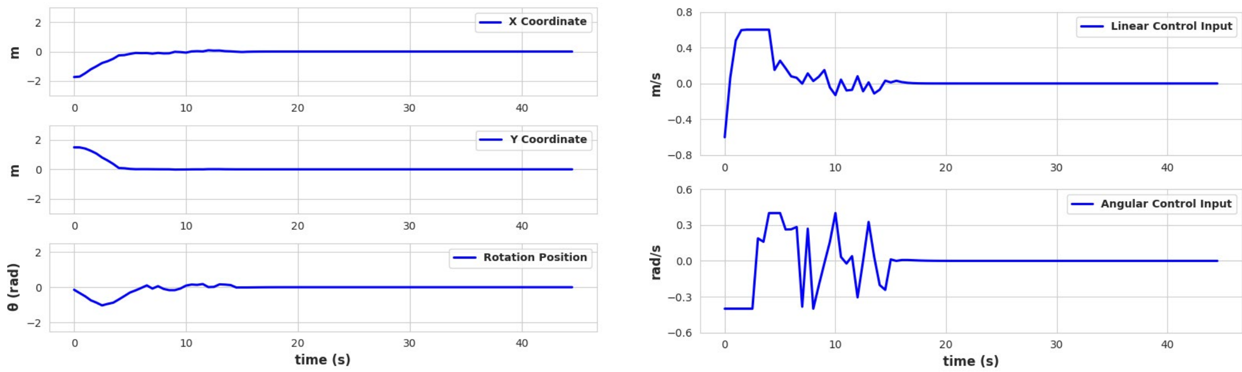
The NMPC controller developed for SEATER was then implemented and evaluated in the Gazebo simulation environment. The experiments were designed to demonstrate the performance of the



(a)



(b)



(c)

Figure 1. MPC performance with static obstacle: (a) turning left; (b) straight; (c) turning right.

Table 2.
MPC data collection with static obstacles.

Parameter		Performance			
ΔT	N	Total time iteration (s)	Max time iteration (s)	Euclidean position error (m)	Rotation error (rad)
0.10	15	9.897	0.030	0.914	0.284
	20	4.912	0.033	0.787	0.007
	25	7.317	0.042	0.784	0.016
0.50	5	3.566	0.023	0.796	0.027
	10	3.426	0.032	0.781	0.014
	15	3.480	0.039	0.765	0.032
	20	0.830	0.055	0.017	0.015
	25	0.820	0.075	0.018	0.021

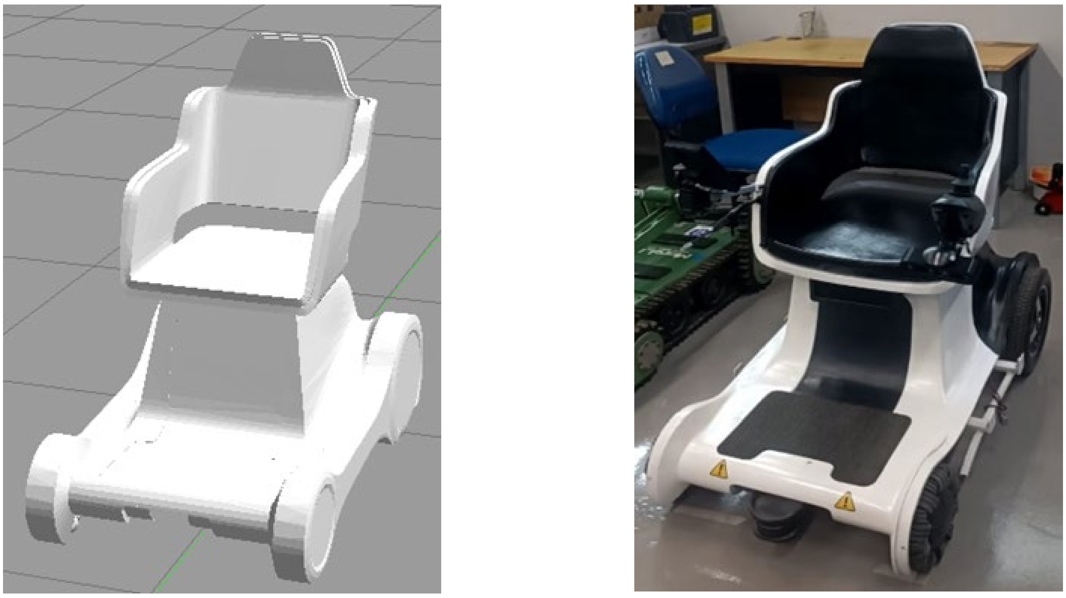
proposed NMPC algorithm in a physics engine simulation environment. This procedure comprises integrating the NMPC system with the robot's dynamic model in the Gazebo environment. Figure 2 illustrate the SEATER model in the Gazebo simulation environment, contrasting with its real-world equivalent, thereby emphasizing the accuracy of the simulated representation.

The experiments were designed to demonstrate the NMPC's ability to control SEATER accurately by adhering to predetermined waypoints, avoiding obstacles, and operating in real-time across a variety of scenarios. The optimal parameters derived from prior Python simulation experiments were applied within the Gazebo simulation environment. As can be seen in Figure 3 the testing was carried out on two different maps: one map was the obstacle-free scenario and the other map containes static obstacles placed at the predetermined position.

The simulations were conducted utilizing the ROS framework in order to ensure easy integration and control. Odometry was employed to acquire the real-time pose of SEATER, which served as the vehicle state input for the NMPC calculation. The control inputs generated from the NMPC optimization process were subsequently implemented to move the vehicle within the simulation environment. To obtain realistic results, a control noise of 15 % and a localization error margin of 0.04 m were used.

Figure 4 compares the SEATER's real trajectory and the predetermined waypoints produced by the Gazebo simulations in two scenarios — one without obstacles and the other with a static obstacle. The trajectory comparisons show how well the NMPC control system performs when completing obstacle avoidance and path-following tasks.

Under obstacle-free conditions — as can be seen in Figure 4(a) SEATER moved from (0, 0, 1.57) through waypoints at (0, 4.1, 1.57), (0, 4.1, 0), (17.2, 4.1, 0), (17.2, 4.1, 1.57), and (17.2, 7.7, 1.57). The coordinates used are (x, y, and θ), where θ is the angular SEATER rotated in radians. It can be seen that SEATER was closely following the waypoint path. The path's smoothness indicates a low tracking error, proving that the

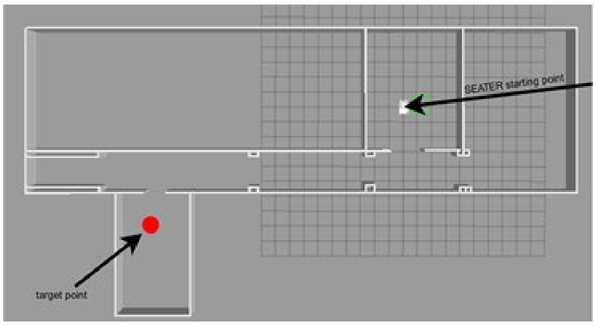


(a) (b)

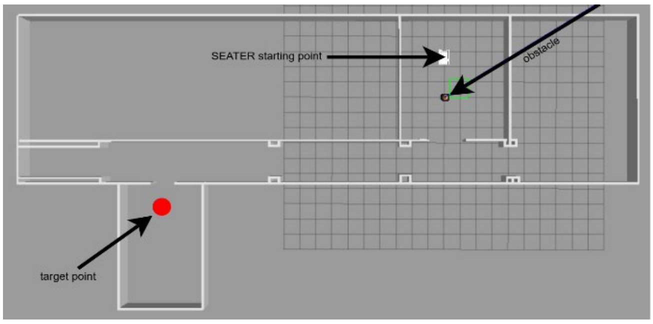
Figure 2. SEATER model: (a) in Gazebo; (b) actual platform.

controller maintains stability and accuracy under ideal circumstances. In the absence of environmental disturbances, the SEATER shows good performance in the desired tasks.

In the presence of an obstacle scenario as shown in Figure 4(b) SEATER started at (0.0, 3.0, -1.57) and followed the same waypoints as the previous experiment (i.e., obstacle-free experiment), with an additional obstacle positioned at (0, 0). The generated path slightly diverges from the direct waypoint path while SEATER tries to avoid the obstacle and maintain a safe distance. This behaviour demonstrates the capability of the NMPC controller to achieve that path-following objective while satisfying the obstacle avoidance constraint.

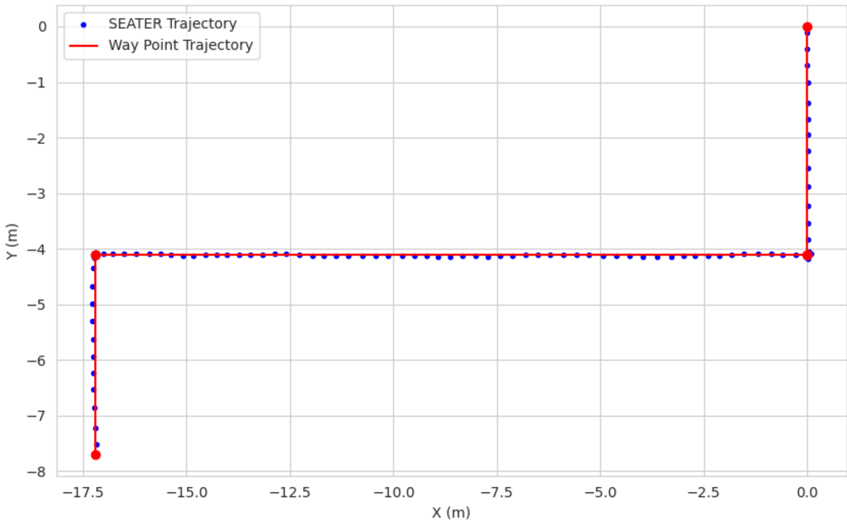


(a)

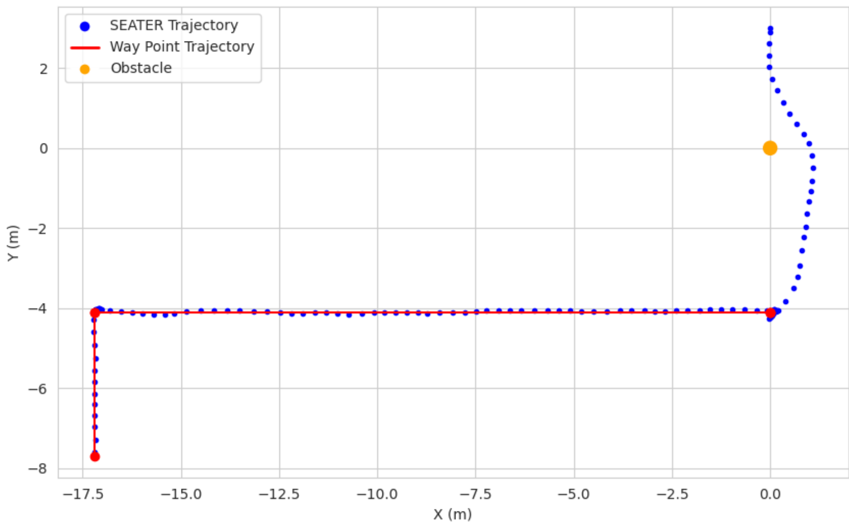


(b)

Figure 3. Simulation map: (a) obstacle-free; (b) static-obstacle.



(a)



(b)

Figure 4. Comparison The SEATER trajectory and the waypoint trajectory. (a) obstacle-free; (b) static-obstacle.

Table 3.
Error trajectory and error final seater position for obstacle-free.

Number	Euclidean position error (m)	Rotation error (rad)	Maximum trajectory error (m)	Average trajectory error (m)
1	0.049	0.105	0.223	0.057
2	0.077	0.075	0.109	0.046
3	0.099	0.136	0.105	0.028
Average	0.075	0.105	0.145	0.044

Table 4.
Error trajectory and error final seater position for static-obstacle.

Number	Euclidean position error (m)	Rotation error (rad)	Maximum trajectory error (m)	Average trajectory error (m)	Minimum Euclidean obstacle (m)
1	0.054	0.142	0.099	0.033	0.758
2	0.108	0.104	0.114	0.024	0.825
3	0.096	0.064	0.149	0.029	0.896
Average	0.086	0.103	0.121	0.028	0.758

Table 3 and Table 4 show the quantified evaluation of how well SEATER followed its path with and without obstacles. To evaluate how well SEATER moved compared to the predetermined path, we used four measurement metrics: position error (m), rotation error (radians), max trajectory error (m), and average trajectory error (m) in obstacle-free scenarios and additional metric minimum Euclidean obstacle (m) in the obstacle presence scenarios.

In general, it can be found from Table 3 that based on three tests in obstacle-free scenarios SEATER produces an average position error of 0.075 meters at the final position and an average rotation error of 0.105 rad. This performance is acceptable for SEATER implementation, with a typical maximum error threshold of 0.25 meters and 0.2 rad for final position and rotation error, respectively. In terms of trajectory tracking performance, it can be seen from Table 3 that SEATER produces a max trajectory error of 0.145 meters and an average trajectory error of 0.044 meters along the trajectory. This trajectory-tracking performance shows the controller's capability to follow a predefined trajectory in a narrow path — such as in corridors — with a maximum trajectory deviation of less than 0.5 meters.

Table 4 gives a quick view of the findings of the experiments with the obstacle-presence scenarios. It can be found that, based on three tests, the SEATER trajectories produce similar performances to the obstacle-free scenarios in terms of position error (m), rotation error (radians), max trajectory error (m), and average trajectory error (m), with average values 0.086 meters, 0.103 rad, 0.121 meters, and 0.028 meters, respectively. The performance measurements demonstrate that the NMPC controller sustains its performance despite a static obstacle. The results from Table 4 also show that SEATER produces the average minimum Euclidean obstacle of 0.758 meters, showing the controller's capability to keep SEATER away from collision with the obstacle.

IV. Conclusion

This study aims to develop a controller for the single-passenger electric autonomous transporter (SEATER) platform. The proposed control method uses NMPC with the single-shooting approach. The NMPC method is implemented and evaluated in Python simulation and Gazebo simulation environment. The experiments in the Python simulation indicate that using a short sampling time and prediction horizon in the NMPC single-shooting approach could limit the control system's capability to reach the desired target. Choosing the right sampling time and prediction horizon is key to balancing performance and computational efficiency. Based on several simulation experiments conducted on the Python simulation environment, the optimal sampling time and prediction horizon parameter values producing the best performance of the NMPC controller can be obtained. The experiments in the Gazebo simulation environment using the optimal sampling time and prediction horizon parameter values demonstrate the capability of the proposed NMPC controller to address trajectory-tracking tasks in a corridor environment with both obstacle-free and obstacle-presence scenarios. The experimental results show that the controller could achieve satisfied performance in terms of position error (m), rotation error (radians), max trajectory error (m), average trajectory error (m), and average minimum distance to obstacle (m). Future research should explore the implementation of the proposed NMPC controller in real-world settings. The results obtained from the simulation experiments carried out in this study serve as a reference for guiding the parameter tuning process, aimed at achieving optimal performance of the proposed NMPC controller.

Acknowledgements

The experiments conducted in this study result from a collaboration between BRIN and Diponegoro University.

Declarations

Author contribution

R.R. Pratama: Writing Original Draft, Investigation and Validation. **C.H.A.H.B. Baskoro**: Supervision, Review & Editing. **J.D. Setiawan**: Supervision, Review & Editing. **D.K. Dewi**: Review & Editing. **Paryanto**: Review & Editing. **M. Ariyanto**: Review & Editing. **R.P. Saputra**: Conceptualization, Supervision, Review & Editing, Funding Acquisition.

Funding statement

This reseach was fully funded by the Research Organization for Electronics and Informatics, National Research and Innovation Agency (BRIN).

Competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Additional information

Reprints and permission: information is available at https://mev.brin.go.id/.

Publisher's Note: National Research and Innovation Agency (BRIN) remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

[1] K. Berntorp, R. Quirynen, T. Uno, and S. Di Cairano, “Trajectory tracking for autonomous vehicles on varying road surfaces by friction-adaptive nonlinear model predictive control,” *Vehicle System Dynamics*, vol. 58, no. 5, pp. 705–725, 2020.

[2] Y. Shi and K. Zhang, “Advanced model predictive control framework for autonomous intelligent mechatronic systems: A tutorial overview and perspectives,” *Annu Rev Control*, vol. 52, pp. 170–196, 2021.

[3] M. Ataei, A. Khajepour, and S. Jeon, “Model Predictive Control for integrated lateral stability, traction/braking control, and rollover prevention of electric vehicles,” *Vehicle System Dynamics*, vol. 58, no. 1, pp. 49–73, Jan. 2020.

[4] T. Ma, “A model‐and data‐driven predictive control approach for tracking of stochastic nonlinear systems using Gaussian processes,” *International Journal of Robust and Nonlinear Control*, vol. 33, no. 15, pp. 9338-9363, 2023.

[5] M. Rick, J. Clemens, L. Sommer, A. Folkers, K. Schill, and C. Büskens, “Autonomous driving based on nonlinear model predictive control and multi-sensor fusion,” *IFAC-PapersOnLine*, vol. 52, no. 8, pp. 182–187, 2019.

[6] H. Ebel, M. Rosenfelder, and P. Eberhard, “A note on the predictive control of non-holonomic systems and underactuated vehicles in the presence of drift,” *PAMM*, vol. 23, no. 4, p. e202300022, Dec. 2023.

[7] K. Huang, K. Wei, F. Li, C. Yang, and W. Gui, “LSTM-MPC: A deep learning based predictive control method for multimode process control,” *IEEE Transactions on Industrial Electronics*, vol. 70, no. 11, pp. 11544–11554, 2022.

[8] D. Mariano-Hernández, L. Hernández-Callejo, A. Zorita-Lamadrid, O. Duque-Pérez, and F. Santos García, “A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis,” *Journal of Building Engineering*, vol. 33, p. 101692, 2021.

[9] P. Hang, X. Xia, G. Chen, and X. Chen, “Active safety control of automated electric vehicles at driving limits: A tube-based MPC approach,” *IEEE Transactions on Transportation Electrification*, vol. 8, no. 1, pp. 1338–1349, 2021.

[10] P. Karamanakos, E. Liegmann, T. Geyer, and R. Kennel, “Model predictive control of power electronic systems: Methods, results, and challenges,” *IEEE Open Journal of Industry Applications*, vol. 1, pp. 95–114, 2020.

[11] R. Sabbagh Novin, A. Yazdani, A. Merryweather, and T. Hermans, “A model predictive approach for online mobile manipulation of non-holonomic objects using learned dynamics,” *Int J Rob Res*, vol. 40, no. 4–5, pp. 815–831, 2021.

[12] R. P. Saputra, N. Rakicevic, D. Chappell, K. Wang, and P. Kormushev, “Hierarchical decomposed-objective model predictive control for autonomous casualty extraction,” *IEEE Access*, vol. 9, pp. 39656–39679, 2021.

[13] S. Blindheim, S. Gros, and T. A. Johansen, “Risk-based model predictive control for autonomous ship emergency management,” *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 14524–14531, 2020.

[14] N. Chowdhri, L. Ferranti, F. S. Iribarren, and B. Shyrokau, “Integrated nonlinear model predictive control for automated driving,” *Control Eng Pract*, vol. 106, p. 104654, 2021.

[15] J Zeng, B. Zhang, and K. Sreenath, “Safety-Critical Model Predictive Control with Discrete-Time Control Barrier Function,” in *2021 American Control Conference (ACC)*, 2021, pp. 3882–3889.

[16] S. Abdallaoui, H. Ikaouassen, A. Kribèche, A. Chaibet, and E. Aglzim, “Advancing autonomous vehicle control

systems: An in-depth overview of decision-making and manoeuvre execution state of the art," *The Journal of Engineering*, vol. 2023, no. 11, p. e12333, 2023

[17] M. Rosenfelder, H. Ebel, J. Krauspenhaar, and P. Eberhard, "Model predictive control of non-holonomic systems: Beyond differential-drive vehicles," *Automatica*, vol. 152, p. 110972, 2023

[18] B. Pan, Y. Wang, and S. Tian, "A High-Precision Single Shooting Method for Solving Hypersensitive Optimal Control Problems," *Math Probl Eng*, vol. 2018, no. 1, p. 7908378, 2018.

[19] J. T. Betts," *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, Second Edition*," Second. Society for Industrial and Applied Mathematics, 2010.

[20] K. Worthmann, M. W. Mehrez, M. Zanon, G. K. I. Mann, R. G. Gosine, and M. Diehl, "Model predictive control of nonholonomic mobile robots without stabilizing constraints and costs," *IEEE transactions on control systems technology*, vol. 24, no. 4, pp. 1394–1406, 2015.

[21] S. Bayat and J. T. Allison, "SS-MPC: A user-friendly software based on single shooting optimization to solve Model Predictive Control problems," *Software Impacts*, vol. 17, p. 100566, 2023.

[22] S. Cheng, L. Li, H.-Q. Guo, Z.-G. Chen, and P. Song, "Longitudinal collision avoidance and lateral stability adaptive control system based on MPC of autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 6, pp. 2376–2385, 2019

[23] H. Wei and Y. Shi, "MPC-based motion planning and control enables smarter and safer autonomous marine vehicles: Perspectives and a tutorial survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 1, pp. 8–24, 2022.

[24] Q. Lu, D. Zhang, W. Ye, J. Fan, S. Liu, and C.-Y. Su, "Targeting posture control with dynamic obstacle avoidance of constrained uncertain wheeled mobile robots including unknown skidding and slipping," *IEEE Trans Syst Man Cybern Syst*, vol. 51, no. 11, pp. 6650–6659, 2020.

[25] J. Wurts, J. L. Stein, and T. Ersal, "Design for Real-Time Nonlinear Model Predictive Control With Application to Collision Imminent Steering," *IEEE Transactions on Control Systems Technology*, vol. 30, no. 6, pp. 2450–2465, 2022.

[26] R. P. Saputra, M. Mirdanies, E. J. Pristianto, and D. Kurniawan, "Autonomous Docking Method via Non-linear Model Predictive Control," in *2023 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*, IEEE, 2023, pp. 331–336.

[27] J. A. Báez-Hernández, M. Velasco-Villa, and S. Mondié, "Non-Linear Prediction-Based Trajectory Tracking for Non-Holonomic Mobile Robots," *IEEE Access*, vol. 11, pp. 124265–124277, 2023.

[28] Y. Al Younes and M. Barczyk, "Nonlinear model predictive horizon for optimal trajectory generation," *Robotics*, vol. 10, no. 3, p. 90, 2021.

[29] S. Khan and J. Guivant, "Fast nonlinear model predictive planner and control for an unmanned ground vehicle in the presence of disturbances and dynamic obstacles," *Sci Rep*, vol. 12, no. 1, p. 12135, 2022.

[30] L. Du, B. Sun, X. Huang, X. Wang, and P. Li, "A Learning-Based Nonlinear Model Predictive Control Approach for Autonomous Driving," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 2792–2797, 2023.

[31] S. Yu, C. Shen, and T. Ersal, "Nonlinear model predictive planning and control for high-speed autonomous vehicles on 3D terrains," *IFAC-PapersOnLine*, vol. 54, no. 20, pp. 412–417, 2021.

[32] M. W. Mehrez, G. K. I. Mann, and R. G. Gosine, "Comparison of stabilizing NMPC designs for wheeled mobile robots: An experimental study," in *2015 Moratuwa Engineering Research Conference (MERCon)*, IEEE, 2015, pp. 130–135.

[33] J. Cenerini, M. W. Mehrez, J. Han, S. Jeon, and W. Melek, "Model Predictive Path Following Control without terminal constraints for holonomic mobile robots," *Control Eng Pract*, vol. 132, p. 105406, 2023.

[34] S. Sivashangaran, D. Patel, and A. Eskandarian, "Nonlinear model predictive control for optimal motion planning in autonomous race cars," *IFAC-PapersOnLine*, vol. 55, no. 37, pp. 645–650, 2022,

[35] A. Florez, A. Astudillo, W. Decré, J. Swevers, and J. Gillis, "IMPACT: A toolchain for nonlinear model predictive control specification, prototyping, and deployment," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 3164–3169, 2023.