BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 74(1), 2026, Article number: e156764
DOI: 10.24425/bpasts.2025.156764

Experimental performance analysis of an AMR controlled by a hybrid MPC-PID method

Orkan Murat CELIK[1], **Murat KOSEOGLU**[2]*, **and Furkan Nur DENIZ**[2]

[1] Department of Electrical and Electronics Engineering, Institute of Science, Inonu University, Malatya, Türkiye
[2] Department of Electrical and Electronics Engineering, Faculty of Engineering, Inonu University, Malatya, Türkiye

Abstract. Recent advances in decision-making algorithms used in mobile robotics require more advanced and adaptive control strategies. Model predictive control (MPC) is one of the prominent strategies to manage diverse kinds of complex dynamic systems. Despite their widespread adoption in industrial robotics owing to their structural simplicity and ease of implementation, proportional-integral-derivative (PID) controllers exhibit notable limitations in effectively addressing process variations and system constraints, particularly those arising from mechanical constraints on joint positions and velocities. As autonomous mobile robots (AMRs) have been increasingly deployed in various and demanding applications, the need for more advanced control algorithms has become critical. In this study, a novel hybrid control framework integrating MPC and PID strategies is proposed and experimentally validated on a real-world differential drive robot, aiming to enhance tracking accuracy and overall operational performance. The system model of the TurtleBot3 robot is identified using the System Identification Toolbox and validated through extensive motion tests on the real robot by using Robot Operating System 2. The proposed control scheme combines the predictive capabilities of MPC with the reactive nature of PID to facilitate improved management of system constraints, aiming to improve the performance of AMR in controlling both linear and angular velocities. Experimental results show that the hybrid MPC-PID controller exhibits better performance by reducing tracking errors while maintaining reliability and robustness characteristics over a conventional PID controller. These results demonstrate that the hybrid MPC-PID approach provides a more effective solution for dynamic control tasks in mobile robotic systems, particularly in scenarios requiring high accuracy and reliability.

Keywords: ROS2; AMR; MPC-PID; System Identification.

1. INTRODUCTION

The use of autonomous mobile robots (AMR) is rapidly growing in both industrial applications and everyday life [1]. In addition, the technology for AMRs is advancing at an accelerating rate. With the expanding range of robotic applications, expectations for improved performance and efficiency continue to rise. The growing diversity of use cases highlights the increasing importance of robust and effective control algorithms for AMRs. Moreover, as robots are deployed in more complex environments, achieving precise task execution and optimizing energy consumption become critical priorities. In recent years, research and development in AMR technology have been the focus of active exploration across various fields, including logistics [2], construction [3], and the automotive industry [4, 5]. For industrial robots, proportional-integral-derivative (PID) controllers have been predominantly used due to their simplicity and ease of implementation [6]. However, classical control methods have notable limitations, particularly in their inability to effectively manage complex dynamic systems with significant process variations or constraints, such as mechanical limits on maximum joint positions and velocities. These limitations generally reduce the efficiency of the classical control algorithms [7]. For instance, the performance of PID controllers depends on fixed constants (K_p, K_i, and K_d) and feedback to adjust the control signal in real-time, often leading to suboptimal performance, particularly in systems with nonlinear behavior or time-varying dynamics. Additionally, PID controllers cannot account for constraints on control inputs and outputs, which may result in instability, undesired overshoot, and undershoot in control output.

Considering the recent advancements, various decision-making algorithms have been developed and implemented in mobile robotics. As decision-making processes in AMRs have improved, there has been an increasing need for more flexible controller designs that operate within actuator limits. Consequently, advanced control strategies have increasingly supplanted classical controllers due to their better ability to handle system complexities and dynamic uncertainties. Among these advanced controllers, model predictive control (MPC) has become one of the most popular choices in robotics.

There is a growing interest in MPCs, since they can provide more precise solutions for AMR motion, particularly in scenarios where classical control methods fall short. Accordingly, various solutions based on MPC have been widely adopted in the robotics community. Gold *et al*. proposed a model predictive interaction control framework specifically designed for industrial robotic systems operating in environments involving physical interaction, highlighting the capability of MPC to manage contact dynamics effectively [7]. A holonomic mobile robot control strategy was introduced, demonstrating that system stability can

*e-mail: murat.koseoglu@inonu.edu.tr

Manuscript submitted 2024-11-27, revised 2025-09-10, initially accepted for publication 2025-10-07, published in January 2026.

© 2026 The Author(s). This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/)

be ensured without relying on terminal constraints or costs, provided that the cost function is carefully designed and the prediction horizon is appropriately chosen in [8]. Based on this, Xie and Fierro developed a first-state contractive MPC scheme for nonholonomic mobile robots, which guarantees stability by imposing a contractive constraint on the initial predicted state, thereby eliminating the requirement for terminal conditions [9]. Ding *et al*. have proposed an optimal velocity MPC framework for a four-wheel independently steered and redundantly actuated omnidirectional mobile robot. Their approach incorporates a fuzzy logic-based steering selector along with a local micro-error-based pose correction mechanism to enhance trajectory tracking accuracy under complex steering constraints [10]. Furthermore, a comprehensive study on dynamic modelling and MPC-driven robust control of a rotorcraft equipped with four tilting rotors, addressing challenges in stability and maneuverability through advanced control strategies, was presented in [11]. Salzmann *et al*. proposed a neural network-based MPC framework that enhances real-time prediction for quadrotors and agile robots [12]. Le Cleac'h *et al*. presented a contact-implicit MPC approach that optimizes the trajectory of robotic systems during contact and noncontact phases, demonstrating its effectiveness in real-time scenarios [13]. Furthermore, a unified MPC framework that integrates whole-body dynamic locomotion and manipulation tasks, offering a versatile solution for multi-limbed robots, was introduced in [14]. In addition, Meduri *et al*. introduced the BiConMP framework, a nonlinear MPC approach designed for whole-body motion planning in legged robots, which enhances performance in diverse terrains and under unpredictable disturbances [15].

MPC utilizes a predictive model of the system to optimize control actions over a prediction horizon. It is particularly effective at managing limitations and constraints by incorporating them directly into the optimization problem. This capability allows MPC to deliver more robust and reliable control performance, even under challenging operating conditions.

The high accuracy of control outputs in MPC-based algorithms primarily stems from the use of a highly precise kinematic model for calculations [16]. Therefore, kinematic models are becoming increasingly crucial in mobile robotics. Unlike some classical control algorithms, MPC is capable of minimizing the impact of uncertainties or disturbances present in a kinematic model [17]. Despite the apparent simplicity of the kinematic models for many common mobile robots, defining reliable control methods for these systems can be challenging due to the presence of nonholonomic (non-integrable) constraints [18].

MPC addresses this challenge by solving an optimization problem to determine the optimal control actions, utilizing the multi-input multi-output (MIMO) model of the mobile robot to predict its future behavior. Conventional controllers, such as PID, often struggle with MIMO mobile robot systems due to the complex interactions between interdependent inputs and outputs [19].

In this study, a novel MPC-PID controller is designed and implemented practically to improve the control efficiency of a differential drive robot. The majority of MPC applications directly utilize the kinematic model, which can be defined using either white-box or black-box methodologies [20]. In modeling dynamic systems, a white-box approach utilizes known physical parameters and system structure to derive models from known parameters and basic principles of the system. When all mechanical and electrical properties are available, an accurate kinematic model can be formulated using established equations, offering high precision and physical interpretability [20, 21]. However, this method becomes impractical when the internal parameters of the system are unknown or not directly measurable.

Conversely, a black-box approach treats the system as an input-output mapping without explicit knowledge of internal dynamics. The kinematic model is derived exclusively from observed input-output data, which makes this method suitable when the system cannot be modelled physically due to its complexity.

In this study, the TurtleBot 3 Burger (TB3) serves as the AMR platform, controlled via an MPC-PID framework. Treated as a black-box system, the TB3 dynamics are identified through system identification in MATLAB, and the resulting model is directly integrated into the MPC for motion control. The MPC plays a significant role in mobile robotics in many aspects. Usage of the controller for the collision-free multi-robot system was presented by Arvindh *et al*. [22] and Krishnan *et al*. [23]. MPC was also used as a formation planner for multi-robot systems by Li *et al*. [24] and Yang *et al*. [25]. PID and MPC were used together following a trajectory by Chiang *et al*. [26]. MPC was also used as an optimization methodology to set PID coefficients [27–31]. Although there are some closed-loop PID-MPC hybrid control applications in chemistry, there is no application that aims at mobile robotics [32,33]. A review of existing studies confirms the versatility of MPC in controlling various robotic parameters for diverse applications. However, prior work lacks a clear methodology for deriving a system model from a black-box approach and applying a hybrid MPC-PID controller to a mobile robot. The main contributions of this study are as follows:

i) Instead of relying on a conventional kinematic model, the system model of the AMR, specifically the TB3, is derived using system identification techniques in MATLAB, based on experimental data from a real TB3 Burger robot. This model is subsequently validated through motion tests to verify its accuracy.
ii) The design and practical implementation of an MPC controller are realized using a hybrid MPC-PID approach to improve the operational efficiency of the robot. This method integrates PID-controlled motors with the MPC controller to achieve enhanced performance. To the best of the authors' knowledge, this hybrid MPC-PID control strategy has not been previously applied to a real AMR based on an integer order transfer function (model) obtained by system identification.

2. METHODOLOGY

2.1. Extracting of system model by system identification

System identification is a process used to develop mathematical models of dynamic systems based on measured and observed data. The methodology relies on analyzing the inputs and out-

puts of the target system. Data is collected from the system inputs and outputs, and then an algorithm is applied to this data to extract an accurate system model [34,35]. Robotic systems are generally characterized as nonlinear systems at the system level due to the presence of nonlinear mechatronic components and algorithms. Unlike linear systems, defining a nonlinear system using design parameters is more challenging. Therefore, system identification is employed to gain a deeper understanding of robotic behavior by modelling its dynamics based on observed data [36, 37]. In addition to providing insights into the overall system model, system identification is also employed to determine partial specifications of the robot. This allows for a more detailed understanding of specific components or subsystems, enhancing the ability to refine and optimize individual aspects of robotic performance [38].

The importance of accurate system definition becomes particularly more evident as the complexity of robotic systems increases. As complexity increases, the difference between the real system and the design-based calculated system models tends to increase. This difference can arise due to numerous factors, including environmental disturbances, component variability, and deviations from design specifications in electronics, mechanics, etc. Consequently, system identification is considered a valuable technique to bridge the difference between the theoretical system model and the actual behavior of the robot, ensuring more precise and reliable control and performance [39]. In this study, given the advantages outlined earlier, the system model extracted through system identification is employed by the control algorithm to achieve more precise and efficient movement. The MATLAB System Identification Toolbox was utilized for the system identification process, providing a robust model for the control strategy.

TB3 is one of the most popular robot platforms used in various research due to extendibility, modularity, and affordability [40]. The TB3 is designed with a differential drive system for motion, consisting of two independent actuators and wheel systems positioned along the same axis. The robot can control each wheel independently for forward or backward movement. By adjusting the velocity and rotational direction of each wheel separately, the robot can perform forward and backward motions, as well as change directions.

Figure 1 shows TB3 and the rotation axes of the robot, respectively. The position of the robot on a Cartesian plane, relative to the global frame, can be defined using the following three variables in the local frame (robot position)

$$x = \left[x, y, \theta\right]^T. \tag{1}$$

In equation (1), the x matrix describes the local frame, x and y are the robot position along the X-axis and Y-axis, respectively. θ is the orientation (or heading) angle of the robot with respect to the global coordinate system, typically measured in radians or degrees.

The control signal for the robot can be expressed as

$$u = \left[v, w\right]^T, \tag{2}$$

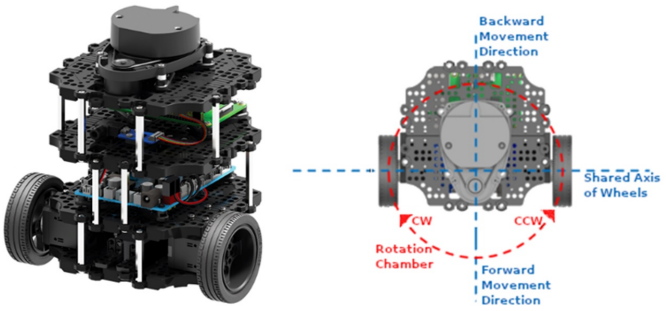

Fig. 1. TB3 Burger [52] and TB3 axis view [53]

where u matrix defines the control signal, v is the linear component of the velocity, and w is the angular velocity. The kinematic model of a differential drive robot can be determined as a matrix based on the robotic position matrix and control signal matrix, as shown below [41]

$$\dot{x} = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ w\end{bmatrix}. \tag{3}$$

Equation (4) shows how the robot position (x, y) and orientation θ change over time based on the linear v and angular w velocities provided by the control inputs. The twist $\dot{x}$ of the robot can be defined in terms of its state variables and control inputs (v, ω) as

$$\dot{x} = f(x,u) = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\cos(\theta).v\\ \sin(\theta).v\\ w\end{bmatrix}, \tag{4}$$

where $\dot{x}$ and $\dot{y}$ represent the linear velocities along the x and y directions, respectively. $\dot{\theta}$ represents the angular velocity (rate of change of orientation). $\cos(\theta).v$ and $\sin(\theta).v$ expresses robotic motion in a global coordinate frame based on its current orientation θ. The kinematic model of the robot can also be expressed in the state-space domain as follows [18, 42–44]:

$$\begin{gathered} x(k+1) = A(k)x(k) + B(k)u(k),\\ A(k) = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}, \quad B(k) = \begin{bmatrix}\cos\left(\theta(k)\right)T_s & 0\\ \sin\left(\theta(k)\right)T_s & 0\\ 0 & T_s\end{bmatrix}. \end{gathered} \tag{5}$$

This equation shows a nonlinear relationship between the states and control signal [45, 46].

From equation (5), the discrete kinematic model of the robot can be written as

$$\begin{aligned} x(k+1) &= x(k) + v(k)\cos\left(\theta(k)\right)T_s,\\ y(k+1) &= y(k) + v(k)\sin\left(\theta(k)\right)T_s,\\ \theta(k+1) &= \theta(k) + w(k)T_s, \end{aligned} \tag{6}$$

where T_s is the sample time, and k describes each time step. If the elapsed time during the process is t ($t = kT_s$), then the

equation can be expressed in state-space form as

$$x(k+1) = x(k) + f\big(x(k), u(k)\big)T_s\,. \tag{7}$$

The robot states and control inputs may be subject to certain constraints. These constraints can arise from environmental factors (such as indoor or outdoor settings) and internal factors, such as actuator limits or delays due to calculation speed [46].

Differential drive robots are generally considered as nonlinear systems [47]. The nonlinearity arises from their kinematic and dynamic models, particularly due to the way robot motion is influenced by its wheel velocities and the coupling between linear and angular velocities. In a differential drive robot, the relationship among the control inputs (wheel velocities), the robot position, and orientation in the Cartesian plane is nonlinear. This is especially evident when trying to control robot orientation or to follow a curved path. Due to nonlinearity, dynamic modelling of the system is not as obvious as kinematic models. The dynamic model of a mobile robot is affected by numerous parameters. So, it is not possible to create an exact dynamic model of a system without extensive calculations. Owing to difficulties in deriving a complete system model through calculations, extracting nonlinear system models based on discrete or continuous states and system inputs requires using data collection techniques guided by the principles of optimal input design for linear dynamic systems [36].

The algorithms needed to define, estimate, and analyze nonlinear models are typically complex and require the support of advanced computer software [48]. Once the extraction process based on the System Identification is completed, the obtained model must be validated by comparing the actual system. The extracted system model should behave like the dynamic system it is derived from, without any discrepancies with the real system [49]. The performance of a nonlinear system is significantly affected by the properties of the excitation signal, even if the maximum input amplitude and power spectrum are kept the same. Thus, a model needs to include a wide range of input signals, which can be assessed during the validation phase [48,50]. The System Identification Toolbox is utilized to validate the behavior of the extracted model by comparing the input and output data with those of real models.

A Simulink model, which is designed to extract data from TB3 using the ROS Toolbox, is shown in Fig. 2. There are two step signal generators that produce step signals to measure the TB3 step response for both linear and angular movements. These step responses are used for the system identification of the robot. The geometry_msgs/Twist block is used to define the message type published to ROS, and the step signals are sent using this message format. The Bus Assignment block merges the messages based on their type and sends a signal to the Publish block. The Publish block then sends the signal to the /cmd_vel

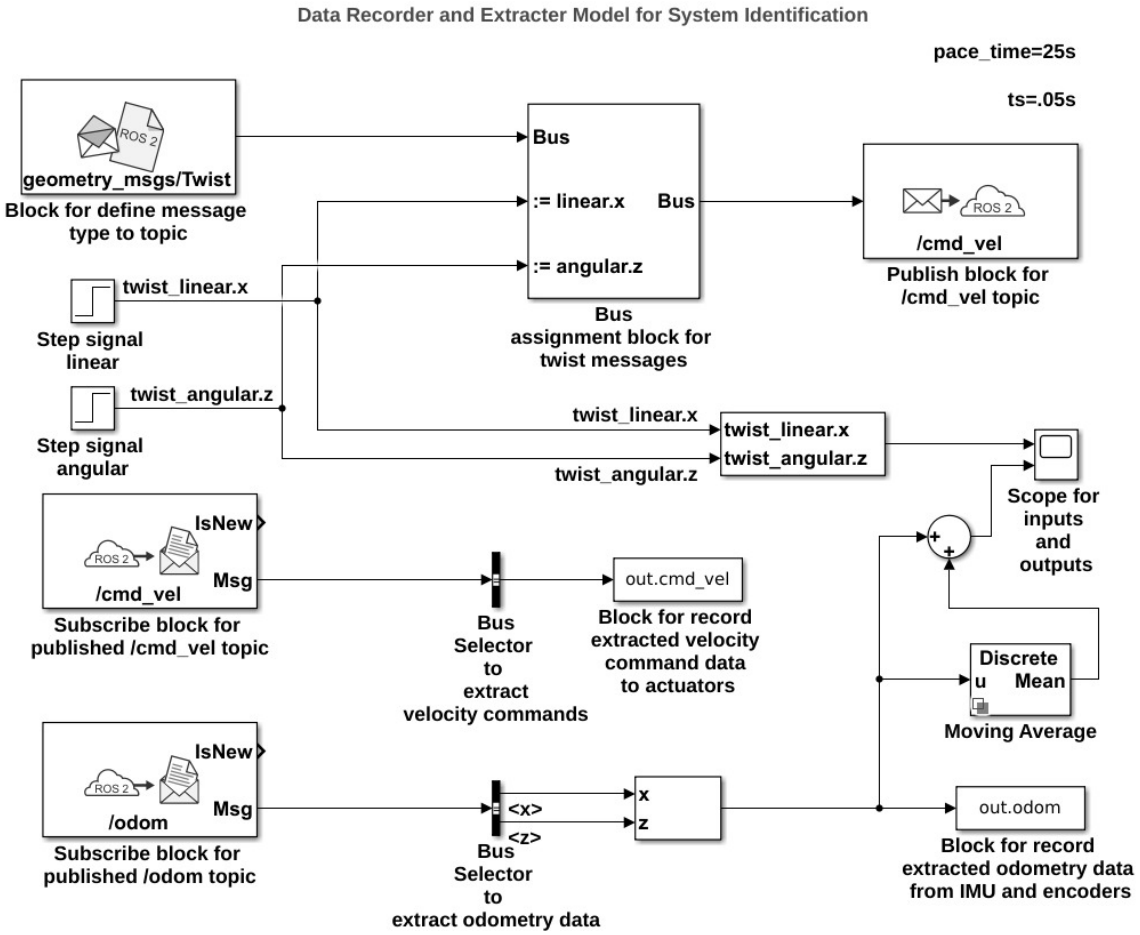

Fig. 2. Simulink data recorder and extractor model for system identification

topic. The /cmd_vel Subscribe block subsequently collects the messages published on this topic. A Bus Selector extracts the velocity data from the published messages and sends it to the workspace for use in the system identification process. Simultaneously, odometry data is collected by the /odom Subscribe block and extracted by another Bus Selector. Finally, the odometry data is sent to the workspace, like the velocity data, for use in the system identification process. The step responses of TB3 for linear and angular movements are shown in Figs. 3 and 4, respectively.

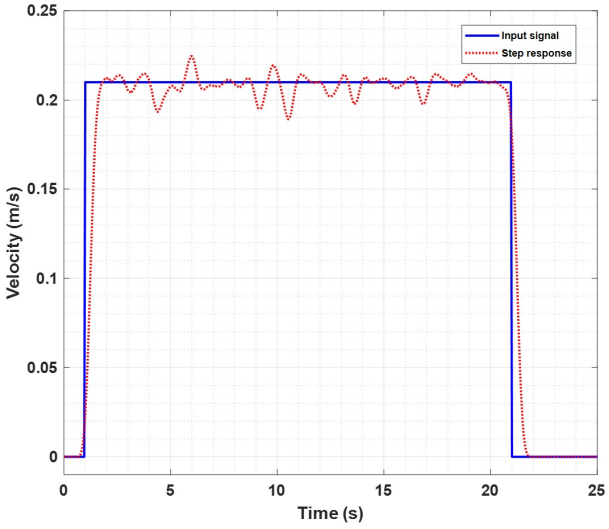

Fig. 3. Step response of TB3 for linear movement

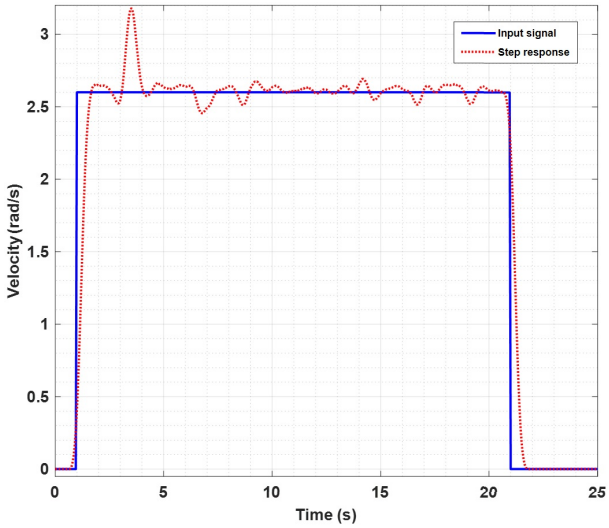

Fig. 4. Step response of TB3 for angular movement

When Fig. 3 is considered, the linear odometry output is shown across the input of the linear velocity command. In general applications, the step response is typically used to observe the initial reaction to a sudden input [51]. Significant and potentially rapid deviations from the long-term steady state can affect the component itself and other parts of the overall system that depend on it. But in this study, a relatively long-term response of the system is gathered to investigate any long-term deviation based on a single-step input. The value for cmd_vel is selected as 0.22 m/s (maximum translational velocity [52]), and the iteration time is selected as 25 s. On the other hand, the output signal is filtered by a Gaussian filter to smooth the noisy output of the odometry signal. The filter is applied to each of 20 samples across the input and output.

Figure 4 reveals the angular odometry output in response to the angular velocity command input. The value for cmd_vel is selected as 2.84 rad/s (maximum translational velocity [53]), and the iteration time is selected as 25 s. On the other hand, the output signal is filtered by a Gaussian filter to smooth the noisy output of the odometry signal. The filter is applied to each of 20 samples across the input and output. For system identification, both motors are assumed to be identical.

In this paper, the proposed method aims to create a comprehensive system model that involves all system dynamics, including the existing controller dynamics of the TB3. As seen in Fig. 5, the motion system of the TB3 is composed of a Raspberry Pi 4, an OpenCR 1.0 Controller, two Dynamixel Servo motors, and two wheels. These components are interconnected, with the motors and wheels rigidly attached. Motors and wheels are arranged as depicted in Fig. 1 to form a differential drive motion system. Raspberry Pi 4 is used as a single-board computer that runs Ubuntu and ROS2 Humble [54–56].

All robotic algorithms, including simultaneous localization and mapping (SLAM), local and global planners, motion controllers, and more, operate in ROS2. The OpenCR 1.0 is a microcontroller-based board specifically designed for robotics applications [53]. It is based on the ARM Cortex-M7 architecture and features various peripherals and interfaces suitable for controlling robotic systems. In TB3, the OpenCR 1.0 serves as a unit that manages and coordinates the operation of the Dynamixel servo motors, processes sensor data, and executes higher-level algorithms that are driven by ROS2 algorithms for navigation, localization, and obstacle avoidance. Dynamixel servo motors are high-performance smart actuators commonly used in robotics applications. In TB3, Dynamixel servo motors are used to drive the wheels and provide mobility to the robot. These servo motors allow precise control of the robot movement, including speed, direction, and rotation. In normal operation of TB3, ROS2 algorithms calculate motion commands and convert them to control signals for Dynamixels. These control signals include target position for the robot, velocity, and torque for each Dynamixel. On TB3, the velocity data for each motor is sent to the Dynamixel motors through OpenCR 1.0. Unlike traditional analog DC motors, Dynamixel motors do not process analog signals directly. Instead, each Dynamixel motor has a built-in PID controller that ensures the motor operates at the target speed. Figure 5 provides a detailed view of this motion system. In this study, the system model includes the entire motion system, including the Dynamixel PID controllers. Thus, system identification is conducted for the complete distributed motion system.

Figure 6 illustrates the sequence of motion control signal flow for TB3 on ROS2. As seen, there are two types of blocks in this figure as round and rectangular. The round-shaped blocks

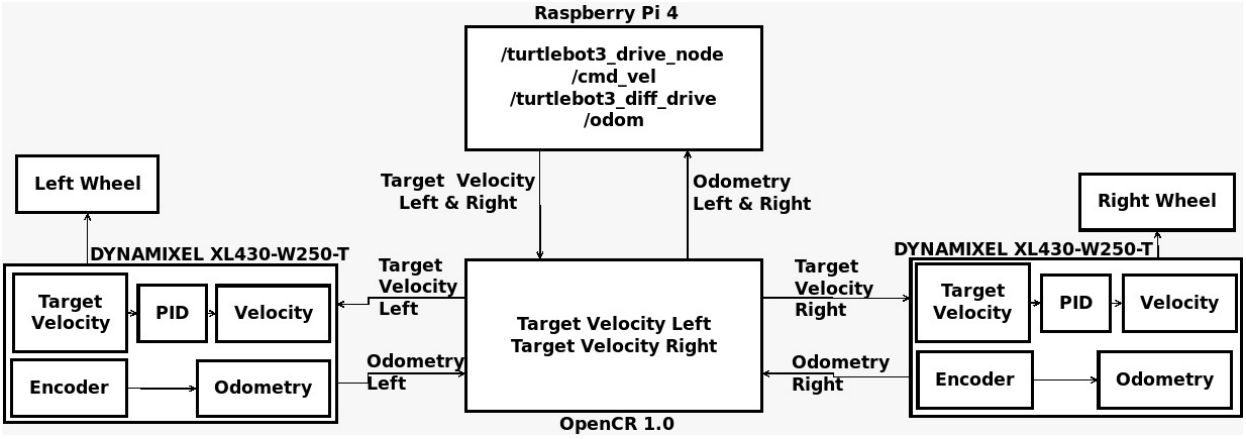

Fig. 5. TB3 motion system breakdown structure and motion system interfaces and signals

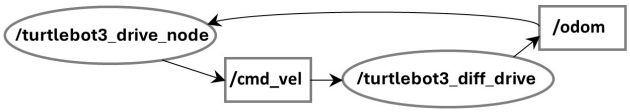

Fig. 6. TB3 Motion control signal graph

represent ROS2 nodes, while rectangular blocks denote ROS2 topics. Nodes are executable units that typically communicate with other nodes via topics, each of which serves as a specific type of communication channel between the nodes [57, 58].

In Fig. 6, the control loop starts with the /cmd_vel topic that is published by the /turtlebot3_drive_node. This node calculates the robot velocity and publishes the data through this topic. In the second phase, the /turtlebot3_diff_drive node subscribes to the velocity data published on the /cmd_vel topic. This node converts the received velocity data into odometry data for the left and right motor-wheel pair and publishes it through the /odom topic. In the final phase, the provided odometry data for each motor-wheel pair through the /odom topic are used as a feedback signal for the /turtlebot3_drive_node, restarting the loop. Although these nodes and topics are run by ROS2 on Raspberry Pi 4, differential drive commands are executed by the OpenCR 1.0. Also, odometry data gathered by the OpenCR 1.0 are published to Raspberry Pi 4, namely, the system is designed as a distributed system.

2.2. Designing a hybrid MPC-PID controller for TB3

The most common application of MPC in robotics relies on kinematic models as the foundation. However, these models often neglect real-world factors such as tolerances, uncertainties, and internal system dynamics. For example, Rosenfelder *et al*. extended kinematic MPC for nonholonomic systems, addressing some of these challenges with a novel approach based on sub-Riemannian geometry, but still faced limitations in accounting for full system dynamics [59]. Similarly, Tang *et al*. proposed an improved kinematic MPC strategy for high-speed path tracking, integrating yaw rate feedback to reduce transient errors, but their model remains limited by its lack of consideration for motor tolerances and uncertainties [60]. Additionally, Wang *et al*. introduced a robust MPC strategy for omnidirectional robots, optimizing trajectory tracking while considering actuator and kinematic constraints, yet their approach also fails to fully integrate the motor tolerances and real-world disturbances that impact performance in practical applications [61]. These studies underline the need for advanced MPC techniques that incorporate these practical factors for more accurate control in dynamic environments.

In this study, rather than a kinematic model, the system model, including the predefined components of the entire motion, is employed for MPC. This approach allows AMR to consider internal tolerances, the motor models, drivers, and disturbances caused by the internal PID controller of the motor.

Due to the usage of MPC and PID control methods in a hybrid form, the method is called a hybrid MPC-PID controller. The details of motion control integrated with the designed controller for one of the motors are shown in Fig. 7.

Firstly, the system model that is created by the system identification method is used to publish /cmd_vel topic with the calculated target speed for the robot by using MPC. The differential drive node then calculates the target velocity for both Dynamixels and sends the data to OpenCR 1.0. Secondly, the OpenCR 1.0 distributes the target velocity data to each Dynamixel. Finally, the Dynamixels use their internal PID controllers to determine the target speed for the wheels. Additionally, Dynamixels collect odometry data through encoders to use as feedback signals for PID controllers and to generate odometry information, which is sent back to the MPC via the OpenCR 1.0 and differential drive nodes. Odometry is used by the MPC to predict the next iteration output and for the calculation of the new velocity.

A Simulink model, which is designed to control the motion of a robot using MPC blocks, is shown in Fig. 8. These blocks are specifically designed to manage both the linear and angular components of the motion for TB3. The MPC.linear and MPC.angular blocks are derived from the same transfer function, which was obtained through a system identification process. To validate the robustness of the system, white noise was injected into the manipulated variable (MV) of the controllers (mv.x and mv.y) and the input of the Bus, where the twist message block is

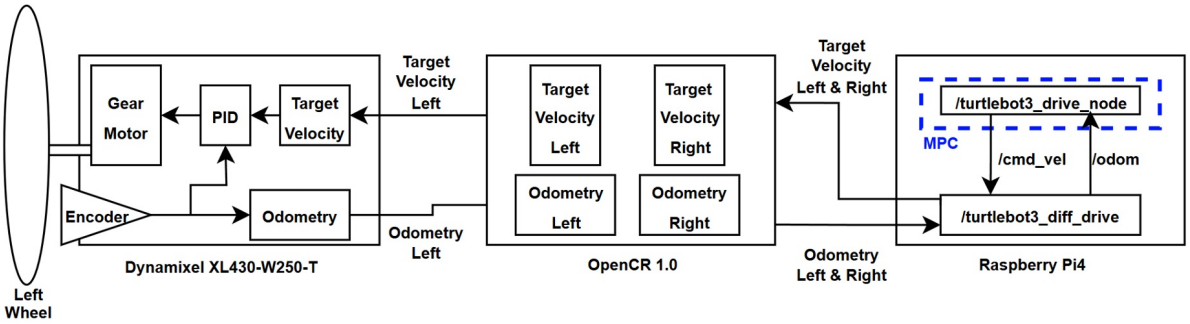

Fig. 7. TB3 motion system with implemented MPC controller

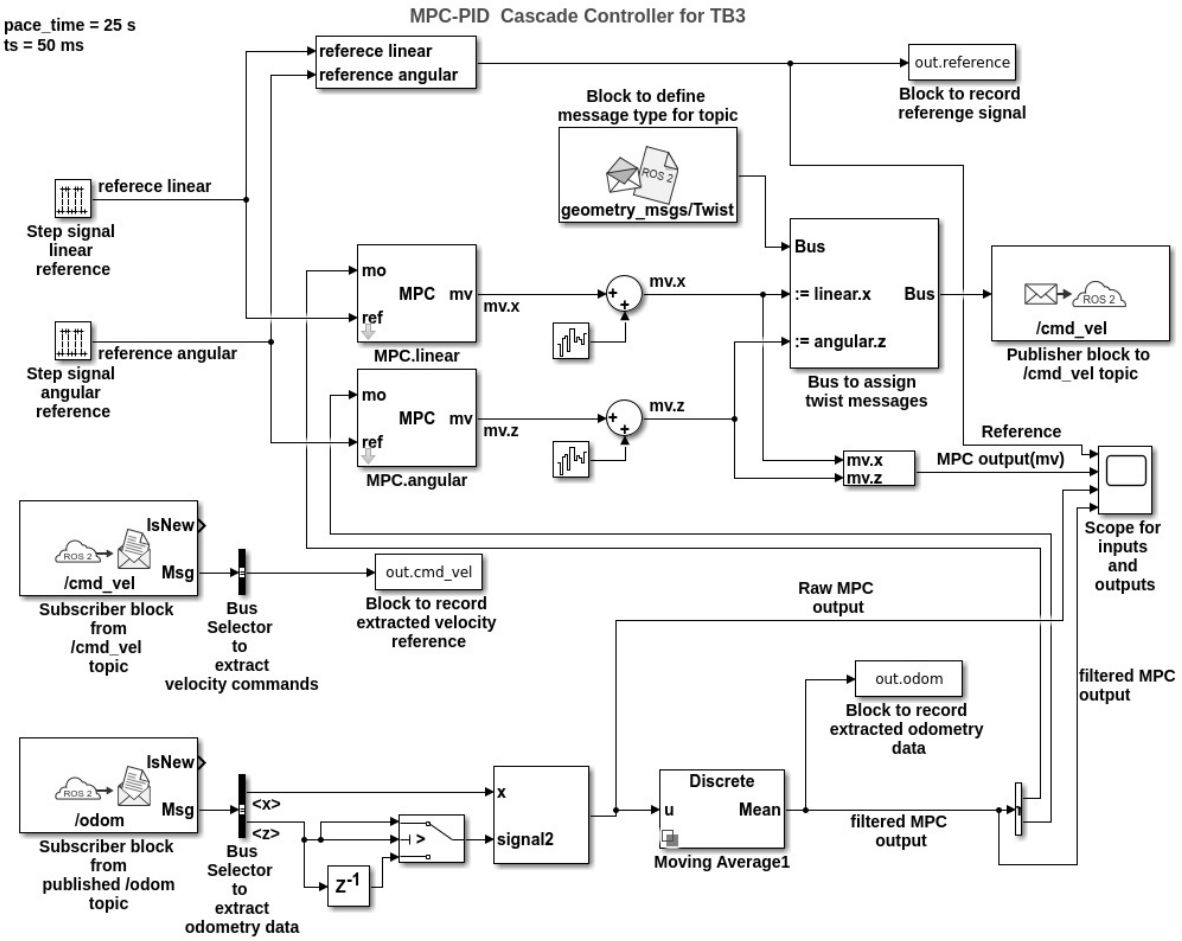

Fig. 8. Simulink model of proposed controller

assigned. MVs are the inputs or control signals that the controller manipulates to achieve desired outcomes. The controller output is transmitted to the TB3 via a Publisher block that publishes to the /cmd_vel topic. To gather published controller messages to the robot, the Subscriber block listens /cmd_vel topic (block). In this study, as previously mentioned, a built-in PID controller that is already integrated with the TB3 is utilized. Consequently, no manipulation or optimization processes were applied to the PID controller. Simultaneously with publishing to the /cmd_vel topic, odometry data is collected by a Subscriber block from the published /odom topic.

The obtained odometry data is processed using a Bus Selector block to extract both the linear and angular components of velocity from the odometry data. The angular component of velocity is initially filtered to remove excessive values and minimize noise. To enhance the quality of feedback signals provided to both MPC blocks, a moving average filter is applied to both the linear and angular components of velocity. This filtering process helps to smooth the measured velocity data and reduce the negative effect of the noise. The processed and filtered data is then transmitted to the controllers as measured output (MO) input, where it is used as a feedback signal for the subsequent

control step in the predictive horizon, thereby improving the accuracy and effectiveness of the control strategy. These are the system variables that the controller tries to regulate or control, based on measurements.

Figure 9 presents a block diagram of the controller developed in this study, outlining its core structure and function. Additionally, Fig. 10 illustrates the process in detail through an operational diagram that emphasizes the interactions between various nodes within the system. Figure 10 highlights the communication pathways, visually mapping out the nodes and the topics they manage. It effectively demonstrates the flow of data and the connections between different components, offering a clear view of how information is exchanged throughout the system.

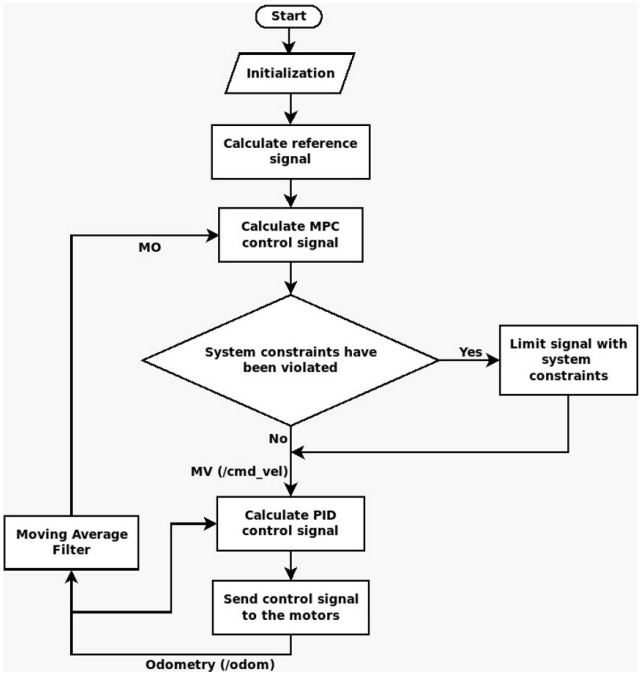

Fig. 9. Flowchart of MPC-PID hybrid controller

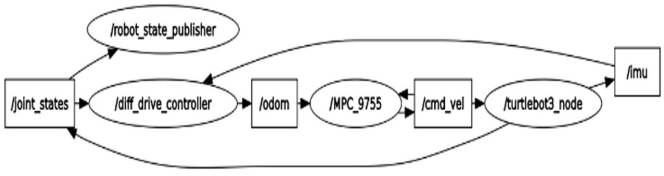

Fig. 10. Nodes and topics which are used in this study

3. RESULTS AND DISCUSSION

This section begins by presenting the results of the system identification process for the TB3 system model, employing various identification methodologies. Initially, the outcomes of each system identification technique are explained briefly to analyze the performance and characteristics of the derived models. The main criteria for selecting the optimal system model are then explained, considering the features such as the model adaptation rate to reference signals, step response – including overshoot and undershoot – and the time required to reach a steady state. This comparative analysis clarifies the selection methodology and the basic aspects to choose the most suitable system model for TB3.

The second part of this section presents the results of applying MPC based on the identified system model under various scenarios and experimental conditions. The performance of the implemented hybrid MPC-PID controller is elaborately evaluated and compared with the standard PID controller traditionally used in TB3. This comparison emphasizes critical performance metrics, such as accuracy, reliability, and responsiveness, across different operational scenarios. The findings offer valuable insights into the effectiveness of the hybrid MPC-PID controller, illustrating its advantages and potential improvements over the conventional PID controller, thereby underscoring its suitability for advanced robotic applications.

3.1. Result of system identification process

In the system identification process, several parameters and settings were established to ensure robust and accurate modelling. The iteration number was set to 1000, with a tolerance of 0.01, and the outlier threshold was set to 0. The transfer function structure derived from the system was configured with three poles and one zero, incorporating feedthrough in the process.

Various search methods were employed sequentially to identify the system dynamics accurately, including adaptive Gauss-Newton (GNA), Levenberg-Marquardt (LM), trust-region reflective Newton (lsqnonlin), gradient descent (grad), sequential quadratic programming (fmincon_sqp), and interior-point (fmincon_interior-point). Each method was applied systematically to capture the best-fitting system model.

The outputs of the identified system models are presented in Figs. 11 and 12, showing both the linear and angular responses, respectively. These figures provide a detailed comparison of the identified system models, highlighting the effectiveness and precision of each identification method in the modelling process. The step responses of seven system models, which were gener-

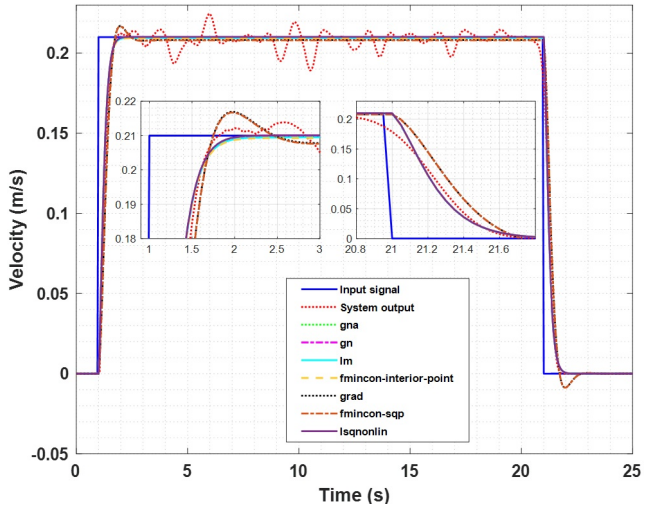

Fig. 11. Linear motion step responses for identified system models

ated by different identification methods, are presented in Figs. 11 and 12. Depending on the identification method, each extracted system model exhibits unique characteristics and performance attributes. As seen in Table 1, fit to estimation rates and root mean squared error (RMSE) values of identified systems are given.

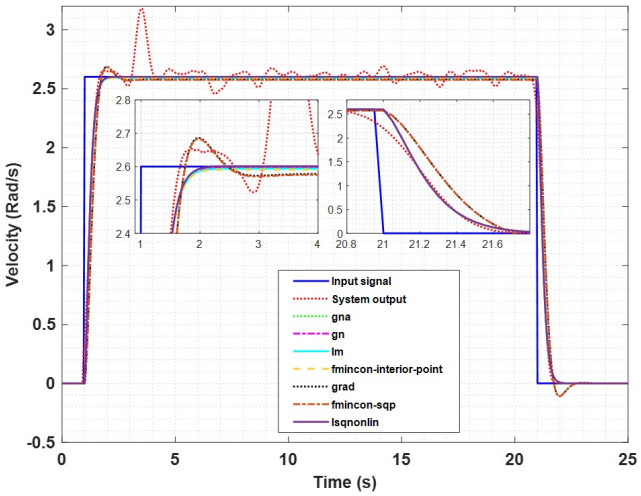

Fig. 12. Angular motion step responses for identified system models

Table 1
Comparison of system identification methods for the step response

Methods	Fit to estimation %		RMSE	
	Linear	Angular	Linear	Angular
Gna	89.25	91.93	0.0254	0.3146
Gn	89.25	91.93	0.0254	0.3146
Lm	89.25	91.93	0.0254	0.3146
Fmincon-int.-point	89.24	91.93	0.0254	0.3144
Grad	88.96	88.91	0.0297	0.3683
Fmincon–sqp	88.96	88.91	0.0297	0.3683
Isqnonlin	89.24	92.63	0.0254	0.3145

Evaluation and selection criteria of extracted system models depend on a systematic approach. Firstly, it is an important criterion that the selected system model follows the reference signal with a high adaptation rate [62, 63]. Secondly, the step response characteristics of the selected model must exhibit minimal overshoot and undershoot, thereby enhancing the accuracy and precision of the control system. Lastly, a critical consideration is the time required for the step response of the chosen model to reach the reference signal. This comparison is very important to select an optimal model that meets the required criteria for an effective MPC implementation [64].

To determine the most suitable model for MPC, each system identification approach must be analyzed separately.

A. Gradient descent method (grad): As seen in Figs. 11 and 12, the extracted system model exhibits one of the highest overshoot responses when an input signal is applied. It reaches the reference input after 2 s and maintains stable behavior until the input is removed. Upon removal, the system indicates an undershoot and takes more than 1.6 s to return to the reference value ratio for both linear and angular components of movement.
B. Trust-region reflective Newton method (lsqnonlin): As seen in Figs. 11 and 12, it is evident that the model demonstrates no overshoot and the lowest undershoot. It achieves a reference signal in 1 s, both at the rising and falling edges of the reference signal.
C. Levenberg-Marquardt method (lm): As seen in Figs. 11 and 12, the responses of the identified model for both linear and angular components are almost identical with the lsqnonlin responses.
D. Gauss-Newton (gn) and adaptive Gauss-Newton (gna) methods: As shown in Figs. 11 and 12, the identified model by the gn/gna is similar to lsqnonlin at the rising edge of the reference signal, but it has a late response at the falling edge of the reference signal.
E. Sequential quadratic programming method (fmincon_sqp): As seen in Figs. 11 and 12, the responses of the identified model for both linear and angular components are almost identical to the grad responses.
F. Interior-point method (fmincon_interior-point): As seen in Figs. 11 and 12, the responses of the identified model for both linear and angular components are almost identical to the lsqnonlin responses. But fmincon_interior-point has 0.1 s more latency than the following reference signal.

Based on the above explanations, it is seen that each method has some cons and pros in the selection of the system model for utilization in MPC [64]. In Figs. 11 and 12, system models derived from the lsqnonlin, and lm methods exhibit a better match with actual system characteristics, compared to the other models. When Table 1 is considered for linear motion, the fit to the estimation rates is nearly identical for both lsqnonlin and lm models, as are the RMSE values.

However, the fit to the estimation rate of the lsqnonlin model for angular motion is higher compared to that of the lm-based model, even though the RMSE angular values are almost identical. MPC relies on precise and reliable system models to predict future states and optimize control actions effectively [10,62,63]. Thus, the model based on the lsqnonlin method was preferred in MPC by considering the importance of a precise representation of system dynamics. The transfer function of the acquired system model is expressed as

$$TF = \frac{0.0921z^{-1}}{1-1.391z^{-1}+0.4873z^{-2}}. \tag{8}$$

To evaluate the robustness and stability of this transfer function, parametric sensitivity analysis and frequency response analysis were performed. In the time-domain sensitivity analysis, a square-wave input signal with an amplitude of ±0.2 and a frequency of 0.05 Hz was applied to the system. The nominal discrete-time transfer function is defined by the coefficients: $b_1 = 0.0921$, $a_1 = -1.391$ and $a_2 = 0.4873$. Each of these coefficients was perturbed by ±5% to evaluate the system sensitivity

to parameter variations. The corresponding output deviations were calculated as $\Delta y(t) = y_{\text{perturb}}(t) - y_{\text{nominal}}(t)$, and are presented in Fig. 13. The results show that the gain parameter b_1 has a negligible effect on the system response, indicating low sensitivity to this parameter. The parameter a_2 exhibited a limited effect on the system transient behavior.

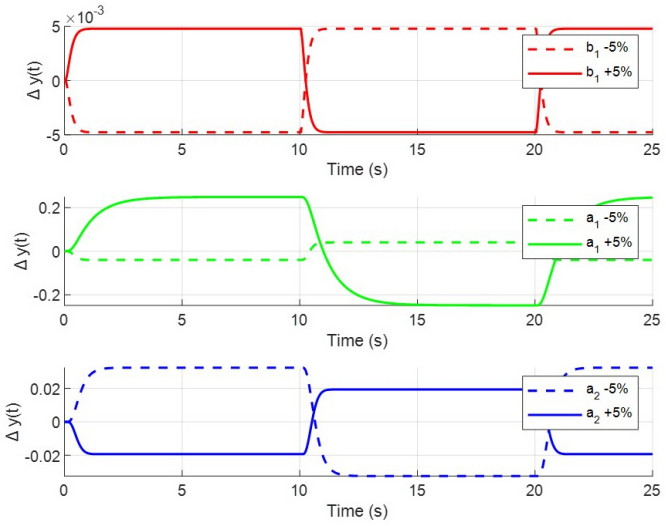

Fig. 13. Deviations in output responses for ±5% variations in the system parameters

In contrast, the parameter a_1 demonstrated asymmetric sensitivity, while perturbation of −5% caused only a lower deviation, a +5% variation in a_1 resulted in a higher change in the system response.

In the frequency-domain analysis, magnitude and phase responses of the perturbed transfer functions were evaluated and illustrated in Fig. 14. The frequency response results are consistent with the time-domain analysis and the critical role of the parameter a_1, particularly in affecting the system gain and phase margins.

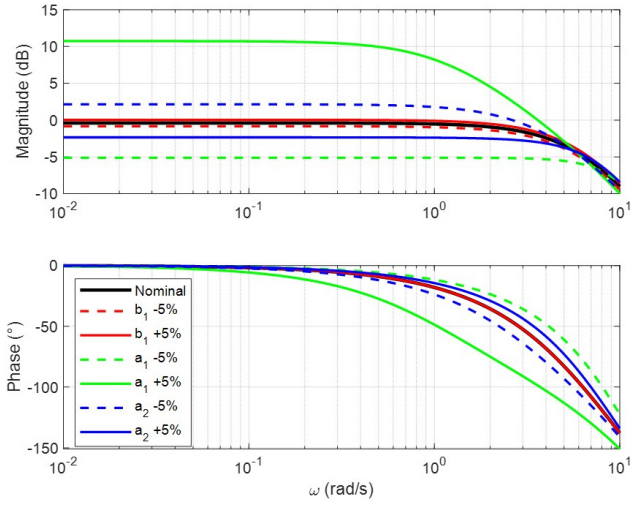

Fig. 14. Magnitude and phase plots of the system with ±5% perturbation

Variations in a_2 produced limited changes, while variations in b_1 affected the system gain without changing its dynamic characteristics.

When both the time and frequency domain parametric sensitivity analyses are considered together, it can be concluded that the transfer function of the acquired system model exhibits sufficient robustness in the presence of ±5% parameter uncertainties. Especially, the system preserves closed-loop stability even under perturbations in the most sensitive parameter a_1.

3.2. Hybrid MPC-PID controller implementation results

The proposed MPC-PID hybrid controller introduced in this study is built upon an identified system model. The design of the MPC component was accomplished based on the transfer function given in equation (8) by using MATLAB and Simulink. To investigate the combined effects of both the prediction horizon (P) and the control horizon (M) on controller performance, an extensive set of simulations was conducted. A sample time of 0.05 s was selected for the controller to comply with the sample rate of the robot. Figure 15 presents a surface plot that illustrates the variation of the integral of squared error (ISE) as a function of the prediction horizon (P) and control horizon (M). This analysis is essential for capturing the inherent trade-offs in MPC design between control accuracy and computational burden, which generally increases with P and M.

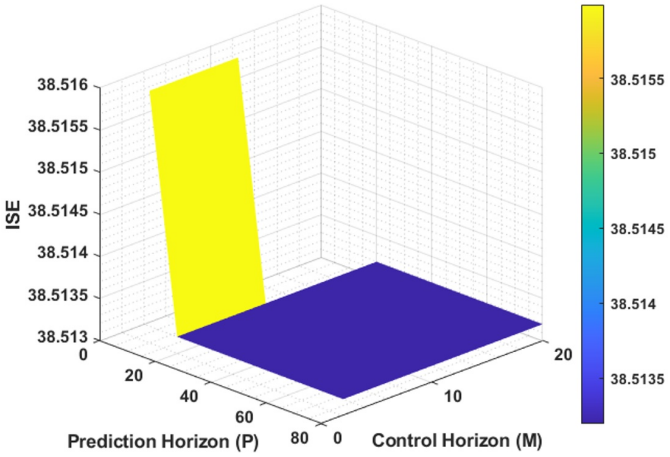

Fig. 15. ISE across prediction and control horizons

The plot reveals that the ISE remains relatively fixed and low across a broad range of P and M combinations, indicating a region of robust controller performance. However, a noticeable degradation in control quality is observed at P = 15, where a sharp increase in the ISE emerges. This localized increase suggests that the controller predictive capability becomes insufficient at this horizon length, resulting in compromised performance. Therefore, to ensure reliable control, the prediction horizon should be chosen as $P \geq 20$. Regarding the control horizon, the surface plot suggests that $M \approx 4$ consistently yields the lowest ISE values within the tested parameter space. In the experiments, according to data gathered from the system, when the different values for P and M were considered, the configuration

with P = 20 and M = 4 not only provides a good control performance but also achieves one of the lowest computation times observed among all test combinations. This makes it an attractive candidate for real-time implementation on embedded hardware, where computational efficiency is a key constraint. While the control performance shows modest improvement beyond P = 20, the computational cost increases noticeably for larger P values. Thus, P = 20 and M = 4 represent an optimal point that balances accuracy and efficiency, and was consequently adopted in both simulation and physical experiments throughout this study. A sample time of 0.05 s was selected for the controller, with a prediction horizon of 20 and a control horizon of 4. The prediction horizon represents the number of future control intervals that the MPC controller must consider when optimizing its MVs, using the internal plant model to forecast system behavior over that period. The control horizon refers to the number of MV adjustments that are optimized during each control interval, with its value ranging between 1 and the prediction horizon [46]. The structure of the controller was maintained consistently for both linear and angular controllers. However, the constraints of the controller were adapted to align with the operational limitations of the TB3 platform:

For the linear controller:

- The MV constraint was set with a minimum of −0.22 cm/s and a maximum of 0.22 cm/s.
- The MO constraint was defined as a minimum of −0.30 cm/s and a maximum of 0.30 cm/s.
- The equality constraint relaxation (ECR) values for both MV and MO were established with a minimum of 0 and a maximum of 0.1. ECR allows small deviations from equality constraints, making it easier for the controller to find a solution when the system is over-constrained.
- The weight assigned to MV was 0.2, while the weight for the MO was set at 0.6.

For the angular controller:

- The MV constraint was set with a minimum of −2.84 rad/s and a maximum of 2.84 rad/s.
- The MO constraint was defined as a minimum of −3 rad/s and a maximum of 3 rad/s.
- The ECR values for both MV and MO were established with a minimum of 0 and a maximum of 0.1.
- The weight assigned to MV was 0.2, while the weight for the MO was set at 0.6.

These constraints and weights were carefully selected to ensure that the controller performance was compatible with the physical and operational capabilities of the TB3, thus optimizing control dynamics. Once the constraints specified for the controller were defined, it was subjected to rigorous testing on the TB3 platform. The testing process involves using various set points for the reference signal to evaluate the robustness and performance of the hybrid MPC-PID controller compared to the standard PID controller. This approach aims to demonstrate the enhanced capabilities and advantages of the hybrid MPC-PID controller in handling different operational conditions.

The capability of the proposed controller was tested by evaluating the performance of TB3 for different linear and angular velocity values. Also, mean squared error (MSE) and ISE values were calculated [65].

i) Firstly, the controller was tested with two different linear reference signal inputs (1 cm/s; 20 cm/s) via two separate experiments. The angular component of the target velocity was aimed to be 0 rad/s to keep the robot on a straight path with no rotational motion.

In Figs. 16a and b, the input and output signals were shown for the PID controller and MPC-PID controller, respectively, for the reference input signal of 1 cm/s. To compare the output signals for the PID controller and MPC-PID controller, squared error (SE) and ISE graphics were presented in Figs. 16c and d, respectively. As seen in the graphics, the MPC-PID controller has lower error values when compared to the PID controller, in general. The MSE value for the MPC-PID controller was calculated as 3.36E-02, while that value for the PID controller was 3.95E-02. Also, the ISE value for the MPC-PID controller was calculated as 8.41E-01, while that value for the PID controller was 9.89E-01.

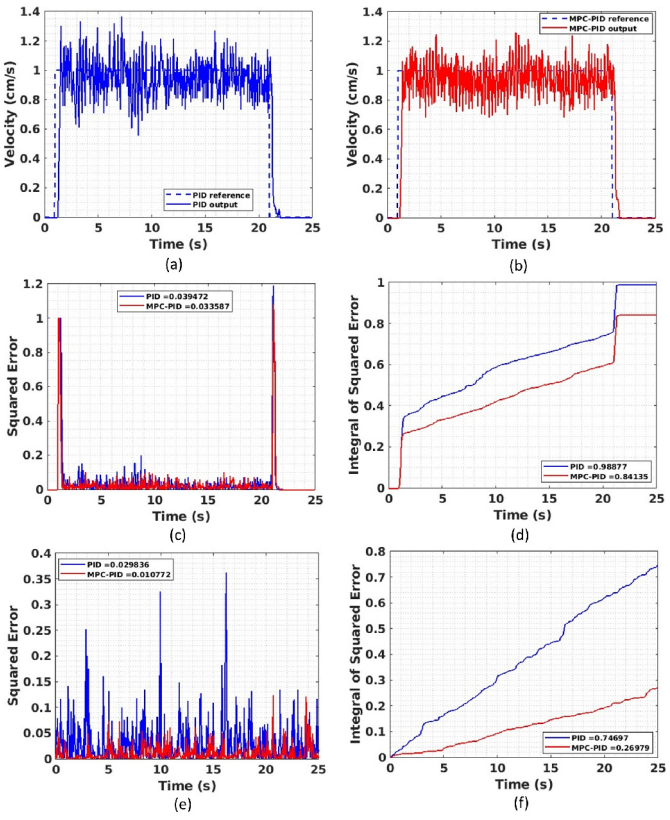

Fig. 16. Input and output signals for the system based on PID (a) and MPC-PID (b) controller for linear velocity of 1 cm/s. Comparative squared error (c) and ISE (d) graphs for linear motion. Comparative squared error (e) and ISE (f) graphs for angular motion when angular velocity is set to 0

On the other hand, the angular component of the target velocity, which was set to 0, was measured for PID and MPC-PID

controllers. Some deviations in the angular velocity value were observed for both controllers, as expected in a practical application. The calculated SE and ISE values for these deviations are shown in Figs. 16e and f. As seen in these figures, the error values are lower for the MPC-PID controller compared to the PID controller, in general. The MSE value for the MPC-PID controller was calculated as 1.08E-02, whereas it was 2.98E-02 for the PID controller. Also, the ISE value for the MPC-PID controller was calculated as 2.70E-01, whereas it was 7.47E-01 for the PID controller.

In the following experiment, the linear input signal was increased to 20 cm/s to compare the performance of controllers in high velocity values. Figures 17a and b show the input and output signals for PID and MPC-PID controllers, respectively. The squared error SE and integrated squared error ISE are presented in Figs. 17c and d for better analysis of the output performance of PID and MPC-PID controllers, respectively. As seen, the MPC-PID controller has lower error values compared to the PID controller, in general. As in the previous experiment, the angular component of the target velocity, which was intended to be 0 to maintain a straight path, was measured for both the PID and MPC-PID controllers. As expected in practical applications, some deviations from the desired angular velocity were observed for both controllers. The calculated SE and ISE values for these deviations are presented in Figs. 17e and f. These figures indicate that, in general, the error values were lower for the MPC-PID controller compared to the PID controller.

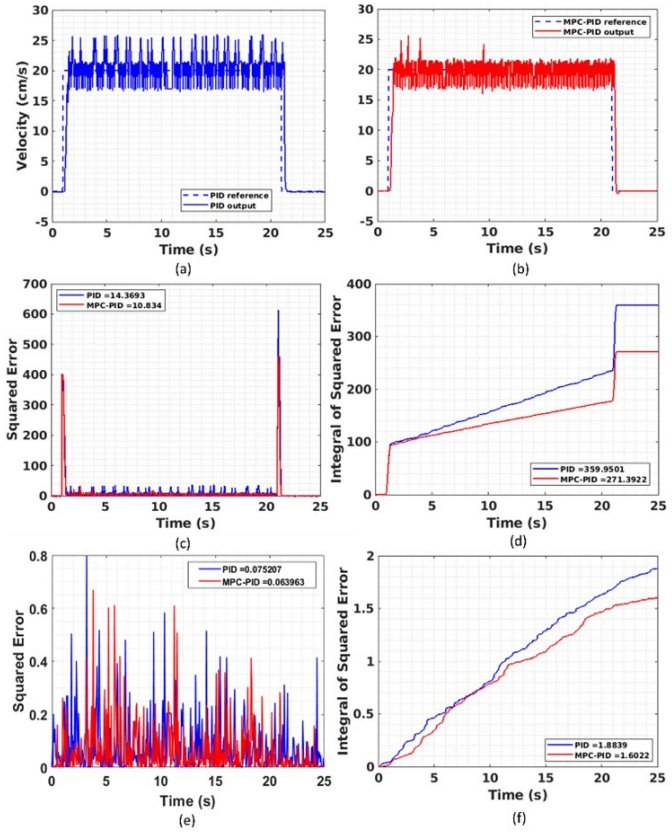

Fig. 17. Input and output signals for the system based on PID (a) and MPC-PID (b) controller for linear velocity of 20 cm/s. Comparative squared error (c) and ISE (d) graphs for linear motion. Comparative squared error (e) and ISE (f) graphs for angular motion when angular velocity is 0

ii) Secondly, the controller was tested for two different angular reference input signals of 0.1 rad/s and 2.6 rad/s through two separate experiments.

In Figs. 18a and b, the input and output signals were shown for the PID controller and MPC-PID controller, respectively, for the reference input signal of 0.1 rad/s. To compare the output signals of PID and MPC-PID controllers, SE and ISE graphics were presented in Figs. 18c and d, respectively. As seen in these figures, the MPC-PID controller has lower error values compared to the PID controller. The MSE value for the MPC-PID controller was calculated as 3.32E-04, while that value for the PID controller was 3.82E-04. Also, the ISE value for the MPC-PID controller was calculated as 8.31E-03, while that value for the PID controller was 9.57E-03.

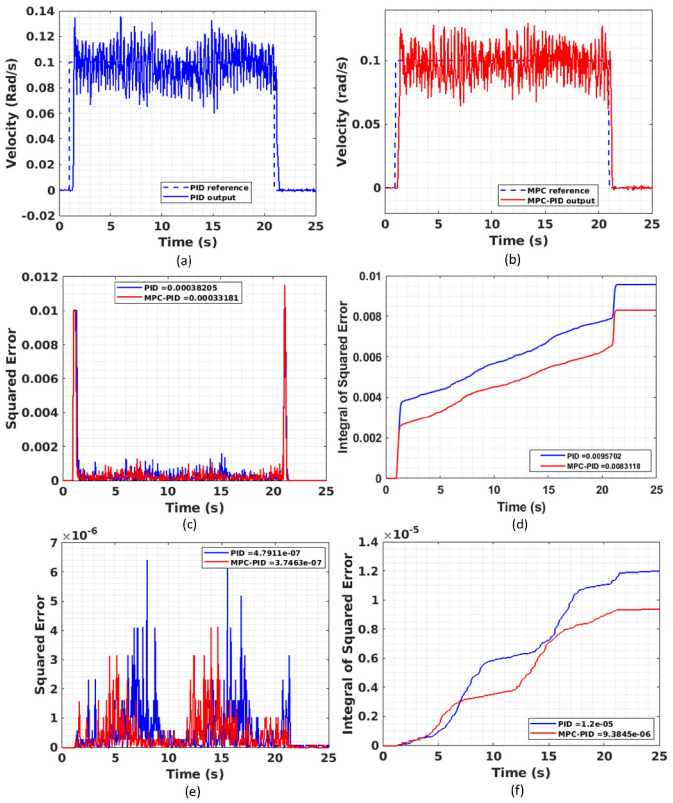

Fig. 18. Input and output signals for the system based on PID (a) and MPC-PID (b) controller for angular velocity of 0.1 rad/s. Comparative squared error (c) and ISE (d) graphs for angular motion. Comparative squared error (e) and ISE (f) graphs for linear motion when linear velocity is 0

On the other hand, the linear component of the target velocity, which was set to 0, was measured for PID and MPC-PID controllers. Some deviations in the linear velocity value were observed for both controllers, as expected in a practical application. The calculated SE and ISE values for these deviations are shown in Figs. 18e and f. As seen, the error values were lower for the MPC-PID controller compared to the PID controller, in gen-

eral. The MSE value for the MPC-PID controller was calculated as 3.75E-07, whereas it was 4.79E-07 for the PID controller. Also, the ISE value for the MPC-PID controller was calculated as 9.38E-06, whereas it was 1.20E-05 for the PID controller.

In the following experiment, the angular input signal was increased to 2.6 rad/s to compare the controller performance in high velocity values. Figures 19a and b show the input and output signals for PID and MPC-PID controllers, respectively. For a better analysis of the output performance of PID and MPC-PID controllers, SE and ISE are presented in Figs. 19c and d, respectively. As seen in the graphics of the SE and ISE, the MPC-PID controller has lower error values when compared to the PID controller, in general. Specifically, the MSE value for the MPC-PID controller was calculated as 1.68E-01, whereas it was 1.78E-01 for the PID controller. Similarly, the ISE value for the MPC-PID controller was calculated as 4.1992, while that value for the PID controller was 4.4486. Also, the linear component of the target velocity, which was intended to be 0, was measured for both the PID and MPC-PID controllers. As expected in practical applications, some deviations from the desired linear velocity were observed for both controllers. The calculated SE and ISE values for these deviations are presented in Figs. 19e and f. As seen in these figures, in general, the error values were lower for the hybrid MPC-PID controller compared to the PID controller.

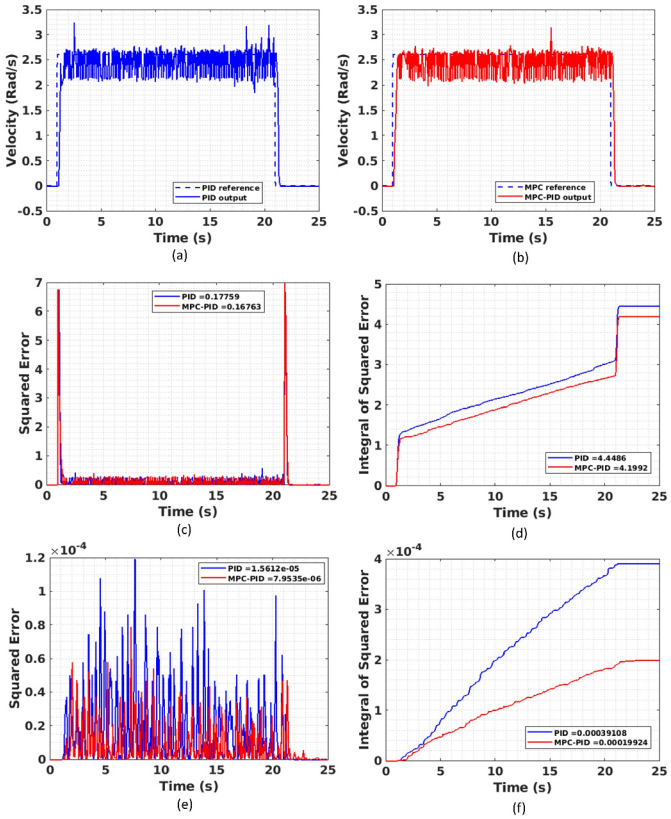

Fig. 19. Input and output signals for the system based on PID (a) and MPC-PID (b) controller for angular velocity of 2.6 rad/s. Comparative squared error (c) and ISE (d) graphs for angular motion. Comparative squared error (e) and ISE (f) graphs for linear motion when linear velocity is 0

Furthermore, the experimental results underscore the superior tracking capabilities of the hybrid MPC-PID controller compared to the conventional PID approach. Across a variety of reference inputs – specifically linear velocities of 1 cm/s, 10 cm/s, and 20 cm/s, and angular velocities of 0.1 rad/s, 1.3 rad/s, and 2.6 rad/s – the hybrid controller consistently achieved lower tracking errors. This improvement is evidenced by reduced RMSE and ISE values, as presented in Table 2, thereby validating the effectiveness of the hybrid strategy in handling both trajectory tracking and robustness under uncertain conditions.

Table 2

Comparison of PID and MPC-PID control error values for different linear and angular velocity values

Controller	Velocity		RMSE		ISE	
	Linear (cm/s)	Angular (rad/s)	Linear	Angular	Linear	Angular
PID	1	0	1.99E-1	1.73E-1	9.89E-1	7.47E-1
MPC-PID			1.83E-1	1.04E-1	8.41E-1	2.70E-1
PID	10	0	1.69E-2	2.03E-3	7.18E-3	1.03E-4
MPC-PID			1.57E-2	1.76E-3	6.15E-3	7.75E-5
PID	20	0	3.79E+0	2.74E-1	3.60E+2	1.88E+0
MPC-PID			3.29E+0	2.53E-1	2.71E+2	1.60E+0
PID	0	0.1	6.92E-4	1.95E-2	1.20E-5	9.57E-3
MPC-PID			6.12E-4	1.82E-2	9.38E-6	8.31E-3
PID	0	1.3	1.72E-3	2.11E-1	7.38E-5	1.11E+0
MPC-PID			1.56E-3	1.93E-1	6.10E-5	9.30E-1
PID	0	2.6	3.95E-3	4.21E-1	3.91E-4	4.45E+0
MPC-PID			2.82E-3	4.09E-1	1.99E-4	4.20E+0

To evaluate the robustness of the proposed MPC controller, a band-limited white noise signal was injected into the controller output (manipulated variable), as described in the methodology. The noise power was set to 3% of the reference signal amplitude to simulate realistic external disturbances typically encountered in practical scenarios. Despite the presence of this noise, the system consistently maintained its intended behavior with minimal deviation from the desired trajectory. The robustness was quantitatively confirmed by ISE and MSE values, indicating that the controller performance remained largely unaffected by the disturbance.

However, comparing the computational burden of both hybrid MPC-PID and PID controllers, it seems that it is not reasonable enough to use MPC directly on robots with limited computational power. The hybrid MPC-PID controller employed in this study requires solving a constrained optimization problem at every control step. Its computational complexity is influenced by several factors, including the number of system states and control inputs, the lengths of the prediction (P) and control (M) horizons, and the structure of the underlying quadratic programming (QP) solver. For an MPC with a system state dimension n, the maximum time for the computational duration of solving the

QP is $O(P^3n^3)$, considering matrix operations. This exponential growth underscores why even modest increases in model size or horizon lengths can make real-time execution infeasible on resource-limited platforms.

In this implementation, the prediction and the control horizon were set to $P = 20$ and $M = 20$, respectively, with a system state dimension $n = 2$ (for linear velocity and angular velocity).

Due to these computational demands, all MPC-related calculations were performed on a client PC that works with TB3 simultaneously during experiments. The onboard processor of the TB3 Burger (Raspberry Pi 4 with 2GB RAM) lacked the processing capacity to run the MPC in real time alongside essential tasks such as navigation and sensor fusion.

Due to its simplicity, the built-in PID controller runs directly on the Dynamixel actuators, requiring only basic arithmetic operations at each control cycle. This results in a constant-time computational complexity of $O(1)$, meaning the execution time remains fixed regardless of the state dimension or complexity of the system. As a result, each PID controller, executed for one of two Dynamixel actuators, introduces minimal computational load and does not need any additional processing resources from the Raspberry Pi.

4. CONCLUSIONS

The first part of the study details the system identification process, where various methodologies were employed to derive models for TB3. Key factors such as the model response to reference signals, step response behavior (e.g., overshoot and undershoot), and the time required to achieve steady state were considered in selecting the most suitable model. The system model selection was performed to ensure that the selected model effectively captures the dynamics of TB3. Then, the identified system model was tested for PID and MPC-PID controllers under various scenarios. In all scenarios, the MPC-PID controller demonstrated improved performance in terms of accuracy and responsiveness. The proposed controller consistently exhibited lower error metrics, such as MSE, RMSE, and ISE, in both linear and angular velocity control. Its ability to track reference signals more accurately, particularly at different velocity ranges, has underscored its effectiveness in reducing errors and improving system reliability. In conclusion, the MPC-PID controller offers significant improvements in the minimization of the error, reliability, and robustness over the conventional PID controller. Its superior performance in both linear and angular velocity control, especially under dynamic conditions, makes it a considerable solution for AMR control. The integration of predictive control mechanisms like MPC into traditional control architectures promises new solutions to improve the performance of robotic systems in complex and demanding environments.

In addition, the hybrid MPC-PID structure can provide a feasible solution for embedded platforms with limited resources. When the prediction and control horizons are set to minimal values, the controller can operate in real time without the need for significant hardware upgrades.

In future work, to design more robust and efficient controllers for robotic systems, advanced control strategies such as linear quadratic regulator, adaptive PID, and sliding mode control will be investigated within hybrid control frameworks [66]. This strategy may be useful in environments where different use cases for the robots emerge.

ACKNOWLEDGEMENTS

This work was supported by Inonu University, Scientific Research Projects Coordination Unit (BAP) [Grant number FDK-2022-2777].

REFERENCES

[1] D. Chikurtev, "Mobile Robot Simulation and Navigation in ROS and Gazebo," in *2020 International Conference Automatics and Informatics (ICAI)*, Varna, Bulgaria: IEEE, Oct. 2020, pp. 1–6. doi: 10.1109/ICAI50593.2020.9311330.

[2] J. Pizoń, Ł. Wójcik, A. Gola, Ł. Kański, and I. Nielsen, "Autonomous Mobile Robots in Automotive Remanufacturing: A Case Study for Intra-Logistics Support," *Adv. Sci. Technol. Res. J.*, vol. 18, no. 1, pp. 213–230, Feb. 2024, doi: 10.12913/22998624/177398.

[3] L. Yang and H. Cai, "Enhanced visual SLAM for construction robots by efficient integration of dynamic object segmentation and scene semantics," *Adv. Eng. Inf.*, vol. 59, p. 102313, Jan. 2024, doi: 10.1016/j.aei.2023.102313.

[4] K. Sugimoto, S. Ishihara, and M. Itoh, "Mobile Robot Navigation in Warehouses by MPC Handling Multiple Travel Strategies Considering Independent Safety LiDAR," in *2024 IEEE/SICE International Symposium on System Integration (SII)*, Jan. 2024, pp. 1085–1092. doi: 10.1109/SII58957.2024.10417666.

[5] R. Benotsmane, L. Dudás, and G. Kovács, "Survey on New Trends of Robotic Tools in the Automotive Industry," in *Vehicle and Automotive Engineering 3*, K. Jármai and K. Voith, Eds, in Lecture Notes in Mechanical Engineering. Singapore: Springer, 2021, pp. 443–457. doi: 10.1007/978-981-15-9529-5_38.

[6] M.I. Azeez, A.M.M. Abdelhaleem, S. Elnaggar, K.A.F. Moustafa, and K.R. Atia, "Optimization of PID trajectory tracking controller for a 3-DOF robotic manipulator using enhanced Artificial Bee Colony algorithm," *Sci. Rep.*, vol. 13, no. 1, p. 11164, July 2023, doi: 10.1038/s41598-023-37895-3.

[7] T. Gold, A. Völz, and K. Graichen, "Model Predictive Interaction Control for Industrial Robots," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9891–9898, Jan. 2020, doi: 10.1016/j.ifacol.2020.12.2696.

[8] M.W. Mehrez, K. Worthmann, J.P.V. Cenerini, M. Osman, W.W. Melek, and S. Jeon, "Model Predictive Control without terminal constraints or costs for holonomic mobile robots," *Rob. Auton. Syst.*, vol. 127, p. 103468, May 2020, doi: 10.1016/j.robot.2020.103468.

[9] F. Xie and R. Fierro, "First-state contractive model predictive control of nonholonomic mobile robots," in *American Control Conference*, June 2008, pp. 3494–3499. doi: 10.1109/ACC.2008.4587034.

[10] T. Ding, Y. Zhang, G. Ma, Z. Cao, X. Zhao, and B. Tao, "Trajectory tracking of redundantly actuated mobile robot by MPC velocity control under steering strategy constraint," *Mechatronics*, vol. 84, p. 102779, June 2022, doi: 10.1016/j.mechatronics.2022.102779.

[11] Y. Tang, X. Chu, J. Huang, and K.W. Samuel Au, "Learning-Based MPC With Safety Filter for Constrained Deformable Lin-

ear Object Manipulation," *IEEE Rob. Autom. Lett.*, vol. 9, no. 3, pp. 2877–2884, Mar. 2024, doi: 10.1109/LRA.2024.3362643.

[12] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-Time Neural MPC: Deep Learning Model Predictive Control for Quadrotors and Agile Robotic Platforms," *IEEE Rob. Autom. Lett.*, vol. 8, no. 4, pp. 2397–2404, Apr. 2023, doi: 10.1109/LRA.2023.3246839.

[13] S. Le Cleac'h *et al.*, "Fast Contact-Implicit Model Predictive Control," *IEEE Trans. Rob.*, vol. 40, pp. 1617–1629, 2024, doi: 10.1109/TRO.2024.3351554.

[14] J.-P. Sleiman, F. Farshidian, M.V. Minniti, and M. Hutter, "A Unified MPC Framework for Whole-Body Dynamic Locomotion and Manipulation," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4688–4695, July 2021, doi: 10.1109/LRA.2021.3068908.

[15] A. Meduri, P. Shah, J. Viereck, M. Khadiv, I. Havoutis, and L. Righetti, "BiConMP: A Nonlinear Model Predictive Control Framework for Whole Body Motion Planning," *IEEE Trans. Rob.*, vol. 39, no. 2, pp. 905–922, Apr. 2023, doi: 10.1109/TRO.2022.3228390.

[16] A. Dalla Libera, N. Castaman, S. Ghidoni, and R. Carli, "Autonomous Learning of the Robot Kinematic Model," *IEEE Trans. Rob.*, vol. 37, no. 3, pp. 877–892, June 2021, doi: 10.1109/TRO.2020.3038690.

[17] J. Li, J. Sun, L. Liu, and J. Xu, "Model predictive control for the tracking of autonomous mobile robot combined with a local path planning," *Meas. Control*, vol. 54, no. 9–10, pp. 1319–1325, Nov. 2021, doi: 10.1177/00202940211043070.

[18] F. Kühne, J.F. Manoel, G.R.G. Silva, and W.F. Lages, "Mobile Robot Trajectory Tracking Using Model Predictive Control," 2005. [Online]. Available: https://api.semanticscholar.org/CorpusID:17764198

[19] O.Y. Ismael, M. Almaged, and A.I. Abdulla, "Nonlinear Model Predictive Control-based Collision Avoidance for Mobile Robot," *J. Rob. Control*, vol. 5, no. 1, p. 1, Jan. 2024, doi: 10.18196/jrc.v5i1.20615.

[20] Z. Sabir, S.B. Said, Q. Al-Mdallal, and S.A. Bhat, "A reliable stochastic computational procedure to solve the mathematical robotic model," *Expert Syst. Appl.*, vol. 238, p. 122224, Mar. 2024, doi: 10.1016/j.eswa.2023.122224.

[21] S. Zhan and A. Chong, "Data requirements and performance evaluation of model predictive control in buildings: A modeling perspective," *Renewable Sustainable Energy Rev.*, vol. 142, p. 110835, May 2021, doi: 10.1016/j.rser.2021.110835.

[22] V. Arvindh, Govind Aadithya R, S. Krishnan, and Sivanathan K, "Collision-Free Multi Robot Trajectory Optimization in Unknown Environments using Decentralized Trajectory Planning," Dec. 03, 2018, *arXiv*: arXiv:1812.00868. doi: 10.48550/arXiv.1812.00868.

[23] Govind Aadithya R, S. Krishnan, V. Arvindh, and Sivanathan K, "Online Decentralized Receding Horizon Trajectory Optimization for Multi-Robot systems," Dec. 28, 2018, *arXiv*: arXiv:1812.11135. Accessed: Apr. 22, 2024. [Online]. Available: http://arxiv.org/abs/1812.11135

[24] E. Li, L. Bi, and W. Chi, "Brain-Controlled Leader-Follower Robot Formation Based on Model Predictive Control," in *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, Boston, MA, USA: IEEE, July 2020, pp. 290–295. doi: 10.1109/AIM43001.2020.9158815.

[25] Z. Yang, L. Bi, W. Chi, H. Shi, and C. Guan, "Brain-Controlled Multi-Robot at Servo-Control Level Based on Nonlinear Model Predictive Control," *Complex Syst. Model. Simul.*, vol. 2, no. 4, pp. 307–321, Dec. 2022, doi: 10.23919/CSMS.2022.0019.

[26] P.-L. Chiang, Y.-C. Wu, C. Ssu-Chien, and C.-Y. Yang, "ROS-based Mobile robot PID and MPC control," in *2023 8th International Conference on Information Technology Research (ICITR)*, Colombo, Sri Lanka: IEEE, Dec. 2023, pp. 1–6. doi: 10.1109/ICITR61062.2023.10382922.

[27] A.A. Abdelrauf, W.W. Saad, A. Hebala, and M. Galea, "Model Predictive Control Based PID Controller for PMSM for Propulsion Systems," in *2018 IEEE International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles & International Transportation Electrification Conference (ESARS-ITEC)*, Nov. 2018, pp. 1–7. doi: 10.1109/ESARS-ITEC.2018.8607585.

[28] A. Aboelhassan, M. Abdelgeliel, E.E. Zakzouk, and M. Galea, "Design and Implementation of Model Predictive Control Based PID Controller for Industrial Applications," *Energies*, vol. 13, no. 24, p. 6594, Jan. 2020, doi: 10.3390/en13246594.

[29] A.A. Abdelrauf, M. Abdel-Geliel, and E. Zakzouk, "Adaptive PID controller based on model predictive control," in *2016 European Control Conference (ECC)*, June 2016, pp. 746–751. doi: 10.1109/ECC.2016.7810378.

[30] S. Rajasekaran and D.T. Kannadasan, "An Improved PID controller Design based on Model Predictive Control for a Shell and Tube Heat Exchanger," *Aust. J. Basic Appl. Sci.*, vol. 7, no. 7, pp. 679–685, 2013.

[31] R.M. Miller, S.L. Shah, R.K. Wood, and E.K. Kwok, "Predictive PID," *ISA Trans.*, vol. 38, no. 1, pp. 11–23, Jan. 1999, doi: 10.1016/S0019-0578(98)90041-6.

[32] F. Norlund, T. Hägglund, and K. Soltesz, "Seamless PID–MPC Hybrid Control⁎," *IFAC-PapersOnLine*, vol. 58, no. 7, pp. 19–24, Jan. 2024, doi: 10.1016/j.ifacol.2024.08.004.

[33] M. Sen, R. Singh, and R. Ramachandran, "A Hybrid MPC-PID Control System Design for the Continuous Purification and Processing of Active Pharmaceutical Ingredients," *Processes*, vol. 2, no. 2, p. 392, June 2014, doi: 10.3390/pr2020392.

[34] D. Saputra, A. Ma"arif, H. Maghfiroh, P. Chotikunnan, and S.N. Rahmadhia, "Design and Application of PLC-based Speed Control for DC Motor Using PID with Identification System and MATLAB Tuner," *Int. J. Rob. Control Syst.*, vol. 3, no. 2, pp. 233–244, Apr. 2023, doi: 10.31763/ijrcs.v3i2.775.

[35] A. Miller, "Model predictive ship trajectory tracking system based on line of sight method," *Bull. Pol. Acad. Sci. Tech. Sci.*, vol. 71, no. 5. p. e145763, 2023. doi: 10.24425/bpasts.2023.145763.

[36] H. Mania, M.I. Jordan, and B. Recht, "Active Learning for Nonlinear System Identification with Guarantees," June 18, 2020, *arXiv*: arXiv:2006.10277. doi: 10.48550/arXiv.2006.10277.

[37] W. Zhao, J.P. Queralta, and T. Westerlund, "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec. 2020, pp. 737–744. doi: 10.1109/SSCI47803.2020.9308468.

[38] T. Zioud, J. Escareno, and O. Labbani-Igbida, "Dynamic Modeling and MPC-driven Robust Control of a Rotorcraft Having Four Tilting-Rotors," preprint, Feb. 2024. doi: 10.36227/techrxiv.170794025.52424117/v1.

[39] S. Hofer *et al*., "Sim2Real in Robotics and Automation: Applications and Challenges," *IEEE Trans. Automat. Sci. Eng*., vol. 18, no. 2, pp. 398–400, Apr. 2021, doi: 10.1109/TASE.2021.3064065.

[40] A. Pandey, K. Prasad, S. Zade, A. Babbar, G.K. Singh, and N. Sharma, "Implementation of simultaneous localization and mapping for TurtleBot under the ROS design framework," *Int. J. Interact. Des. Manuf*., vol. 18, pp. 3799–3812, Mar. 2024, doi: 10.1007/s12008-024-01781-7.

[41] J. Keighobadi, M.S. Sadeghi, and K.A. Fazeli, "Dynamic Sliding Mode Controller for Trajectory Tracking of Nonholonomic Mobile Robots," *IFAC Proc. Vol*., vol. 44, no. 1, pp. 962–967, Jan. 2011, doi: 10.3182/20110828-6-IT-1002.03048.

[42] A.-M.D. Tran and T.-V. Vu, "A study on general state model of differential drive wheeled mobile robots," *J. Adv. Eng. Comput*., vol. 7, no. 3, pp. 174–186, Sept. 2023, doi: 10.55579/jaec.202373.417.

[43] D. Chwa, "Tracking Control of Differential-Drive Wheeled Mobile Robots Using a Backstepping-Like Feedback Linearization," *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans*, vol. 40, no. 6, pp. 1285–1295, Nov. 2010, doi: 10.1109/TSMCA.2010.2052605.

[44] F. Kuhne, W.F. Lages, and J.G. da Silva Jr, "Model predictive control of a mobile robot using linearization," in *Proceedings of mechatronics and robotics*, Citeseer, 2004, pp. 525–530.

[45] K.M. Lynch and F.C. Park, *Modern robotics*. Cambridge University Press, 2017.

[46] A. Salimi Lafmejani and S. Berman, "Nonlinear MPC for collision-free and deadlock-free navigation of multiple nonholonomic mobile robots," *Rob. Auton. Syst*., vol. 141, p. 103774, July 2021, doi: 10.1016/j.robot.2021.103774.

[47] M. Tang, K. Tang, Y. Zhang, J. Qiu, and X. Chen, "Motion/force coordinated trajectory tracking control of nonholonomic wheeled mobile robot via LMPC-AISMC strategy," *Sci. Rep*., vol. 14, no. 1, p. 18504, Aug. 2024, doi: 10.1038/s41598-024-68757-1.

[48] J. Schoukens and L. Ljung, "Nonlinear System Identification: A User-Oriented Road Map," *IEEE Control Syst*., vol. 39, no. 6, pp. 28–99, Dec. 2019, doi: 10.1109/MCS.2019.2938121.

[49] Z. Yu, F. Ge, and S.F. Wong, "Design of a new Robot Operating System-MATLAB-based autonomous robot system and trajectory tracking experiment," *Int. J. Adv. Rob. Syst*., vol. 21, no. 2, Mar. 2024, doi: 10.1177/17298806241229257.

[50] M. Enqvist, J. Schoukens, and R. Pintelon, "Detection of Unmodeled Nonlinearities Using Correlation Methods," in *Finds and Results from the Swedish Cyprus Expedition: A Gender Perspective*, Medelhavsmuseet, May 2007.

[51] C. Ionescu and D. Copot, "Hands-on MPC Tuning for Industrial Applications," *Bull. Pol. Acad. Sci. Tech. Sci*., vol. 67, no. 5. pp. 925–945, 2019. doi: 10.24425/bpasts.2019.130877.

[52] "TurtleBot3," robotis.com. [Online]. Available: https://emanual.robotis.com/docs/en/platform/turtlebot3/features/ (Access: May 01, 2024).

[53] "Open Source Robots – TurtleBot 3 – ROBOTIS." [Online]. Available: https://www.robotis.us/turtlebot-3/ (Accessed: Apr. 30, 2024).

[54] V. Mayoral-Vilches, S.M. Neuman, B. Plancher, and V.J. Reddi, "RobotCore: An Open Architecture for Hardware Acceleration in ROS 2," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2022, pp. 9692–9699. doi: 10.1109/IROS47612.2022.9982082.

[55] "ROS 2 Documentation – ROS 2 Documentation: Humble documentation." [Online]. Available: https://docs.ros.org/en/humble/index.html (Accessed: Sept. 15, 2024)

[56] T. Zemčík and L. Kratochvíla, "Mobile robot perception system in ROS 2 on embedded computing platforms," *IFAC-PapersOnLine*, vol. 58, no. 9, pp. 305–310, Jan. 2024, doi: 10.1016/j.ifacol.2024.07.414.

[57] F. Duan, W. Li, and Y. Tan, "ROS Debugging," in *Intelligent Robot: Implementation and Applications*, F. Duan, W. Li, and Y. Tan, Eds, Singapore: Springer Nature, 2023, pp. 71–92. doi: 10.1007/978-981-19-8253-8_4.

[58] K. Zheng, "ROS Navigation Tuning Guide," in *Robot Operating System (ROS): The Complete Reference* (Vol. 6), A. Koubaa, Ed., Cham: Springer International Publishing, 2021, pp. 197–226. doi: 10.1007/978-3-030-75472-3_6.

[59] 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, June 2023, doi: 10.1016/j.automatica.2023.110972.

[60] L. Tang, F. Yan, B. Zou, K. Wang, and C. Lv, "An Improved Kinematic Model Predictive Control for High-Speed Path Tracking of Autonomous Vehicles," *IEEE Access*, vol. 8, pp. 51400–51413, 2020, doi: 10.1109/ACCESS.2020.2980188.

[61] D. Wang, W. Wei, Y. Yeboah, Y. Li, and Y. Gao, "A Robust Model Predictive Control Strategy for Trajectory Tracking of Omnidirectional Mobile Robots," *J. Intell. Robot Syst*., vol. 98, no. 2, pp. 439–453, May 2020, doi: 10.1007/s10846-019-01083-1.

[62] J. Berberich, J. Köhler, M.A. Müller, and F. Allgöwer, "Linear Tracking MPC for Nonlinear Systems – Part II: The Data-Driven Case," *IEEE Trans. Autom. Control*, vol. 67, no. 9, pp. 4406–4421, Sept. 2022, doi: 10.1109/TAC.2022.3166851.

[63] K. Zhang, Q. Sun, and Y. Shi, "Trajectory Tracking Control of Autonomous Ground Vehicles Using Adaptive Learning MPC," *IEEE Trans. Neural Networks Learn. Syst*., vol. 32, no. 12, pp. 5554–5564, Dec. 2021, doi: 10.1109/TNNLS.2020.3048305.

[64] E. Madsen, O.S. Rosenlund, D. Brandt, and X. Zhang, "Comprehensive modeling and identification of nonlinear joint dynamics for collaborative industrial robot manipulators," *Control Eng. Pract*. vol. 101, p. 104462, Aug. 2020, doi: 10.1016/j.conengprac.2020.104462.

[65] J. Dong, J. Xu, Q. Zhou, J. Zhu, and L. Yu, "Dynamic Identification of Industrial Robot Based on Nonlinear Friction Model and LS-SOS Algorithm," *IEEE Trans. Instrum. Meas*., vol. 70, pp. 1–12, 2021, doi: 10.1109/TIM.2021.3124039.

[66] Z. Tang and T. Zhang, "A hybrid control combining an advanced sliding mode control law with a proportional-derivative control by applying on vibration suppression," *Mech. Syst Signal Process*., vol. 204, p. 110771, Dec. 2023, doi: 10.1016/j.ymssp.2023.110771.